Na prawach rękopisu

Wydział Budownictwa Lądowego i Wodnego
Politechniki Wrocławskiej

Auto-adaptacyjny system identyfikacji eksploatacyjnych obciążeń drogowych obiektów mostowych z wykorzystaniem uczenia maszynowego

Raport serii PRE nr 4/2025

Praca doktorska

Słowa kluczowe:
most drogowy,
mostowe systemy monitoringu obciążeń eksploatacyjnych
Systemy Bridge Weigh-in-Motion (B-WIM)
uczenie maszynowe
identyfikacja obciążeń

Promotor: prof. dr hab. inż. Jan Bień
Promotor pomocniczy: dr inż. Tomasz Kamiński

Wrocław, maj 2025 r.

Autor:

mgr inż. Aleksander Mróz……

Politechnika Wrocławska
Wydział Budownictwa Lądowego i Wodnego
Katedra Dróg, Mostów, Kolei i Lotnisk
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław
tel. 71 320 23 45 tel./fax. 71 320 36 45

Lista odbiorców:

Recenzenci	3 egz.
Promotor	1 egz.
Promotor pomocniczy	1 egz.
Autor	1 egz.
Archiwum W-2	1 egz.
Razem	7 egz.

Serdecznie dziękuję mojemu Promotorowi, Panu prof. dr hab. inż. Janowi Bieniowi, za cenne uwagi, które znacząco wpłynęły na ostateczny kształt pracy, a także za redaktorską czujność i życzliwe podejście.

Dziękuję również Promotorowi pomocniczemu, Panu dr. inż. Tomaszowi Kamińskiemu, za wspólne poniedziałkowe zmagania z materiałem, za czas, cierpliwość i cenne rozmowy, które niejednokrotnie pomogły mi ruszyć z miejsca.

Dziękuję mojej Mamie i Babci za nieustające wsparcie i obecność. Ze szczególnym wzruszeniem dziękuję również mojemu Dziadkowi, który choć dziś już nieobecny, przez wiele lat był dla mnie wzorem cierpliwości, pasji i pomysłowości. Wspólne godziny spędzone na majsterkowaniu i rozmowach miały ogromny wpływ na to, kim dziś jestem.

Spis treści

Wykaz ważniejszych oznaczeń stosowanych w rozprawie

Mechanika klasyczna:

Symbole łacińskie:

c – prędkość przemieszczania się siły P,

EI – sztywność belki wyrażona jako iloczyn modułu Younga oraz momentu bezwładności,

$G_d(\mathrm{n})$ – współczynnik nierówności zależny od przestrzennej częstotliwości nierówności nawierzchni,

L – rozpiętość teoretyczna belki,

M – masa resorowanego punktu masowego,

m – masa nieresorowanego punktu masowego,

n – przestrzenna częstotliwość nierówności nawierzchni drogowej,

P – siła wymuszająca,

$r(x)$ – funkcja opisująca nierówności nawierzchni drogi,

t – czas,

v – pionowe przemieszczenia belki,

w – pionowe przemieszczenia punktu masowego,

WE – wskaźnik oceny skuteczności zawieszenia pojazdu.

Symbole greckie:

α – współczynnik proporcjonalności masy α,

β – współczynnik proporcjonalności sztywności β,

$\delta(x)$ – funkcja Diracka,

ε – bezwymiarowa funkcja czasu, która opisuje postęp przejazdu pojazdu po obiekcie mostowym,

μ_b – masa własna pręta,

μ – wartość średnia,

η – tłumienie zawieszenia,

ω_b – częstość kołowa tłumienia belki,

σ – odchylenie standardowe od wartości średniej.

Uczenie maszynowe:

Symbole łacińskie:

B – wektor obciążeń,

W – macierz wag,

$w_{i,j}$ – waga połączenia pomiędzy i-tym neuronem wejściowym a j-tym neuronem wyjściowym,

X – macierz cech wejściowych,

x_i – wartość wejściowa bieżącego przykładu uczącego,

y_j – oczekiwany wynik j-tego neuronu wyjściowego,

$\widehat{y_j}$ – wynik j-tego neuronu wyjściowego dla bieżącego przykładu uczącego,

Y – odpowiedź oczekiwana od modelu uczenia maszynowego,

$\widehat{\mathrm{Y}}$ – odpowiedź przewidziana przez model uczenia maszynowego.

Symbole greckie:

$\emptyset$ – funkcja aktywacji,

η_{lr} – współczynnik uczenia.

Terminologia

Pojęcia anglojęzyczne:

Bridge Weigh-In-Motion (B-WIM): Szczególna odmiana systemu WIM, która wykorzystuje odpowiedź konstrukcji obiektu mostowego, mierzoną za pomocą szeregu czujników, do określenia co najmniej jednego z następujących parametrów: nacisku całkowitego pojazdu, nacisku na oś, nacisku na grupę osi, nacisków na koła, równoważnego standardowego obciążenia osi oraz liczby pojazdów. Dodatkowo system może zbierać informacje o pojazdach, takie jak prędkość przejazdu, klasa pojazdu i konfiguracja osi.

Dropout: Technika regularyzacji w sieciach neuronowych, która polega na losowym dezaktywowaniu części neuronów podczas procesu trenowania. Ta metoda pomaga uniknąć nadmiernego dopasowania, zmuszając model do generowania bardziej rozproszonych reprezentacji danych. Dropout to strategia, która pozwala sieciom neuronowym na bardziej efektywne i zrównoważone uczenie.

Frequency-Time Domain Method (FTDM): Metoda identyfikacji nacisków osi pojazdu w dziedzinie częstotliwości, oparta na analizie transformacji Fouriera. Wykorzystuje funkcje odpowiedzi częstotliwościowej dla różnych form modalnych do wyznaczenia sił wymuszających na podstawie odwrotnej transformaty Fouriera

Integer (int): Typ danych w Pythonie reprezentujący liczby całkowite. Integer może mieć dowolną długość, ograniczoną jedynie ilością dostępnej pamięci. Przykłady: 10, -5, 0.

Interpretive Method (IM): Ogólna metoda identyfikacji sił wymuszających w systemach Moving Force Identification (MFI), wykorzystująca modele matematyczne do analizy odpowiedzi konstrukcji mostowej na ruchome obciążenia. Opiera się na rozwiązaniu zagadnienia odwrotnego, gdzie siły wymuszające są wyznaczane na podstawie pomierzonych przemieszczeń, prędkości i przyspieszeń konstrukcji. Metoda ta umożliwia określenie nacisków osi pojazdów, uwzględniając zarówno statyczne, jak i dynamiczne aspekty oddziaływania.

Moving Force Identification (MFI): Metoda analizy sił wymuszających działających na konstrukcję mostową, stosowana w systemach identyfikacji obciążeń. MFI wykorzystuje pełną historię zmierzonych przemieszczeń, prędkości oraz przyspieszeń konstrukcji podczas przejazdu pojazdu, aby wyznaczyć statyczne i dynamiczne naciski na osie. Proces polega na rozwiązywaniu zagadnienia odwrotnego, gdzie na podstawie znanej odpowiedzi konstrukcji obliczane są siły wymuszające.

String (str): Typ danych w Pythonie przechowujący sekwencję znaków. Stringi mogą zawierać litery, cyfry, symbole i spacje. Są niemutowalne, co oznacza, że nie można ich zmieniać po utworzeniu. Tworzy się je za pomocą pojedynczych (') lub podwójnych (") cudzysłowów, np. "tekst".

Time Domain Method (TDM): Metoda identyfikacji nacisków osi pojazdu w dziedzinie czasu, oparta na modalnej superpozycji oraz całce splotu. Wykorzystuje równania opisujące drgania belki, aby obliczyć przemieszczenia, prędkości i momenty zginające. Metoda zakłada dyskretny czas i rozwiązuje układ równań liniowych z zastosowaniem metod numerycznych, takich jak metoda najmniejszych kwadratów.

Weigh-in-Motion (WIM): System wag wraz z oprzyrządowaniem pomocniczym wbudowanym w nawierzchnię drogi. System określa co najmniej jeden z następujących elementów zgodnie z wymaganiami tj. nacisk całkowity pojazdu, nacisk na oś, nacisk na grupę osi, naciski na koła, równoważne standardowe obciążenie osi, liczba pojazdów. Dodatkowo system może również zbierać informacje o pojazdach takie jak: prędkość przejazdu, klasa pojazdu, konfiguracja osi. Cele systemu WIM są identyczne z celami B-WIM.

Pojęcia polskojęzyczne:

Algorytm, ang. *algorithm*: To skończony zbiór jasno określonych kroków lub instrukcji, które mają na celu rozwiązanie określonego problemu lub wykonanie zadania. Algorytm można traktować jako instrukcję, która opisuje krok po kroku, jak przetwarzać dane wejściowe, aby uzyskać oczekiwane wyniki. Algorytmy mogą być stosowane w różnych dziedzinach, takich jak matematyka, informatyka, inżynieria czy logistyka i muszą charakteryzować się efektywnością oraz poprawnością, aby osiągnąć zamierzony cel.

Atrybut, ang. *attribute*: Cechy lub właściwości obiektu definiowane przez jego klasę. Atrybuty przechowują dane powiązane z obiektem i mogą być odczytywane lub modyfikowane w trakcie działania programu.

Auto-adaptacyjny, ang. *auto-adaptive*: To przymiotnik określający zdolność systemu, układu lub organizmu do samodzielnego przystosowania się do zmieniających się warunków lub sytuacji, bez konieczności zewnętrznej interwencji. Odnosi się do mechanizmów, które automatycznie dostosowują swoje działanie w celu optymalizacji, utrzymania równowagi lub osiągnięcia określonego celu.

Autodekoder, ang. *autoencoders*: To rodzaj sztucznych sieci neuronowych, stosowanych do nauki efektywnego kodowania danych nieoznakowanych (uczenie nienadzorowane). Autodekoder uczy się dwóch funkcji: kodowania, która przekształca dane wejściowe do wektora zakodowanych cech szczególnych, oraz funkcji dekodowania, która odtwarza dane wejściowe z zakodowanej reprezentacji. Poprzez wymuszenie redukcji sieć neuronowa musi nauczyć się wyciągać cechy szczególne z danych wejściowych.

Biblioteka, ang. *library*: Zbiór gotowych funkcji, klas i modułów, które można wykorzystać w programie, aby uprościć lub przyspieszyć rozwój oprogramowania. Biblioteki zawierają zestawy narzędzi do konkretnych zadań, takich jak analiza danych, przetwarzanie obrazów czy operacje matematyczne. Przykładem popularnej biblioteki w Pythonie jest NumPy.

Całkowity nacisk pojazdu, ang. *total ground pressure*: Suma sił, jakie koła pojazdu wywierają na podłoże w wyniku działania grawitacji. Całkowity nacisk pojazdu jest równoważny masie całkowitej pojazdu pomnożonej przez przyspieszenie ziemskie.

Funkcja, ang. *function*: Podstawowy element programu, który grupuje zestaw instrukcji w celu wykonania określonego zadania. Funkcja może przyjmować argumenty (dane wejściowe) i zwracać wartość (wynik), co pozwala na ponowne użycie kodu w różnych częściach programu. Funkcje są

kluczowym narzędziem w wielu paradygmatach programowania, w tym proceduralnym i funkcyjnym. W przeciwieństwie do procedury, funkcja zawsze zwraca wartość, co umożliwia jej wykorzystanie w wyrażeniach i dalszych obliczeniach.

Funkcja aktywacji, ang. *activation function*: Funkcja definiująca wartość wyjścia sztucznego neuronu przy danym wejściu lub zestawie danych wejściowych. Jest to naśladownictwo stymulacji neuronu biologicznego.

Funkcja wpływu mierzonej wielkości statycznej, ang. *influence function*: Wykres mierzonej przez czujnik wielkości statycznej uzyskanej w zależności od lokalizacji pojazdu kalibracyjnego o znanych parametrach.

Głęboka sieć konwolucyjna, ang. *Deep Convolutional Neural Network* (DCNN): Rodzaj sztucznej sieci neuronowej, która jest szczególnie skuteczna w analizie obrazów i danych przestrzennych. Składa się z wielu warstw konwolucyjnych, które przetwarzają dane w sposób hierarchiczny. Warstwy konwolucyjne w tej sieci filtrują dane wejściowe, wykorzystując specjalne filtry, które automatycznie uczą się rozpoznawać istotne cechy, takie jak krawędzie, tekstury, a w głębszych warstwach bardziej złożone struktury, np. części obiektów. Sieć składa się również z warstw takich jak warstwy uogólniające i w pełni połączone warstwy, które przekształcają i redukują wymiarowość danych, jednocześnie zachowując istotne cechy.

Głębokie uczenie, ang. *deep learning*: Poddziedzina uczenia maszynowego skupiająca się na konstruowaniu i trenowaniu głębokich sieci neuronowych. Głębokie sieci neuronowe, zbudowane z wielu warstw neuronów, mają zdolność do uczenia się złożonych reprezentacji danych, co pozwala na efektywne rozwiązywanie bardziej skomplikowanych problemów.

Grupa osi, ang. *axle group*: Co najmniej dwie osie składowe, które oddziałują na siebie ze względu na szczególną konstrukcję zawieszenia.

Hiperparametry, ang. *hyperparameter*: Wartości konfiguracyjne, które określają sposób działania modelu uczenia maszynowego. Przykłady hiperparametrów to np. liczba warstw w sieci neuronowej, rozmiar wsadu, czy współczynnik prędkości uczenia. Dostosowanie hiperparametrów może znacząco wpłynąć na wydajność modelu, kontrolując kluczowe aspekty procesu uczenia.

Klasa, ang. *class*: Szablon definiujący strukturę i zachowanie obiektów w programowaniu obiektowym. Klasa łączy dane (atrybuty) oraz funkcje (metody), które operują na tych danych. Jest podstawowym

elementem organizacji kodu w paradygmacie obiektowym. Klasy umożliwiają tworzenie wielu obiektów o tej samej strukturze, ale z różnymi stanami.

Konstruktor, ang. *constructor*: Specjalna metoda w klasie, używana do inicjalizacji nowych obiektów. W Pythonie konstruktor jest definiowany za pomocą metody __init__. Konstruktor pozwala przypisywać wartości początkowe atrybutom obiektu w momencie jego tworzenia.

Lista, ang. *list*: struktura danych umożliwiająca przechowywanie sekwencji elementów dowolnego typu. w Pythonie listy są dynamiczne, co oznacza, że ich rozmiar może zmieniać się w trakcie działania programu, i umożliwiają operacje takie jak dodawanie, usuwanie, sortowanie czy indeksowanie.

Metoda Elementów Skończonych (MES), ang. *Finite Element Method* (FEM): Numeryczna metoda rozwiązywania równań różniczkowych w inżynierii i modelowaniu matematycznym. Polega na podziale analizowanego obszaru na mniejsze elementy skończone, dla których obliczane są przybliżone rozwiązania. MES znajduje zastosowanie w analizie konstrukcji, przepływu ciepła, dynamice płynów i wielu innych dziedzinach, umożliwiając modelowanie złożonych problemów z wysoką precyzją.

Metoda pobierająca, ang. *getter*: Metoda w klasie służąca do pobierania wartości atrybutu obiektu. Gettery są używane do zapewnienia kontrolowanego dostępu do danych, np. do obliczeń lub walidacji przed ich zwróceniem.

Metoda ustawiająca, ang. *setter*: Metoda w klasie umożliwiająca ustawienie lub modyfikację wartości atrybutu obiektu. Settery są często używane do kontroli dostępu i weryfikacji danych przed przypisaniem wartości.

Nacisk grupy osi pojazdu, ang. *axle group load*: Suma sił, jakie koła grupy osi (np. podwójnej, potrójnej) wywierają na podłoże.

Nacisk osi pojazdu, ang. *axle load*: Siła, jaką koła jednej osi wywierają na podłoże, wynikająca z rozkładu masy pojazdu i ładunku przypadającej na tę oś.

Nadmierne dopasowanie, *ang. overfitting*: Problem w uczeniu maszynowym, gdy model zbyt dokładnie dopasowuje się do danych treningowych, tracąc zdolność do generalizacji na nowe dane. Może wynikać z nadmiernej złożoności modelu, niewystarczającej liczby danych treningowych lub zbyt długiego treningu.

Obiekt, ang. *object*: Instancja klasy, reprezentująca konkretny byt w programie. Obiekt łączy dane (atrybuty) oraz funkcje (metody) zdefiniowane w klasie, które działają na tych danych. Obiekty są podstawowymi elementami interakcji w programowaniu obiektowym.

Parametry, ang. *parameters*: Numeryczne wartości używane przez modele sztucznej inteligencji do określania, jak przetwarzać dane wejściowe i wykonywać zadania. W sieciach neuronowych obejmują one wagi i składowe progowe, czyli wartości, które muszą być przekroczone przez sumę ważonych wejść, aby neuron został aktywowany. Parametry są dostosowywane w trakcie uczenia maszynowego, aby optymalizować wydajność modelu, ucząc się z dostępnych danych.

Procedura ang. *procedure*: specjalny rodzaj podprogramu, który wykonuje określone operacje, ale nie zwraca wartości. Procedury mogą przyjmować argumenty i wykonywać działania, np. modyfikować zmienne globalne, wyświetlać dane lub zapisywać wyniki do pliku, jednak nie zwracają wyniku, który można by przypisać do zmiennej czy użyć w obliczeniach. W niektórych językach programowania procedury i funkcje są odrębnymi konstruktami, natomiast w innych (np. Pythonie) każda funkcja technicznie może działać jak procedura, jeśli nie używa instrukcji return lub zwraca None.

Program, ang. *program*: Zbiór instrukcji zapisanych w określonym języku programowania, które są wykonywane przez komputer w celu rozwiązania konkretnego problemu lub realizacji określonych zadań. Programy mogą mieć różną postać, od prostych skryptów do złożonych systemów aplikacyjnych i działają w środowisku systemu operacyjnego lub specjalnych interpreterów.

Programowanie obiektowe, ang. *Object-Oriented Programming* (OOP): Paradygmat programowania oparty na koncepcji obiektów, które łączą dane (właściwości) oraz operacje na tych danych (metody) w jedną całość. Programowanie obiektowe umożliwia organizację kodu w sposób bardziej modularny i zrozumiały, wspierając zasady takie jak dziedziczenie, enkapsulacja oraz polimorfizm. Popularne języki programowania obiektowego to Java, Python, C++ czy C#.

Słownik, ang. *dictionary* (dict): Struktura danych w Pythonie przechowująca pary klucz-wartość. Słowniki umożliwiają szybki dostęp do wartości na podstawie unikalnych kluczy. Klucze muszą być niemutowalne (np. liczby, stringi), natomiast wartości mogą być dowolnego typu. Tworzy się je za pomocą nawiasów klamrowych, np. {"klucz1": "wartość1", "klucz2": "wartość2"}.

Spadek gradientu, ang. *gradient descent*: Algorytm optymalizacji stosowany w uczeniu maszynowym do dopasowywania parametrów modelu na podstawie gradientów funkcji kosztu. Model stara się minimalizować funkcję kosztu, dążąc do optymalizacji swojej wydajności. Spadek gradientu to

mechanizm, który pozwala maszynom na ciągłe ulepszanie się poprzez dostosowywanie parametrów w odpowiedzi na obliczone błędy.

Splotowa sieć neuronowa, ang. *Convolutional Neural Network* (CNN): Rodzaj sieci neuronowej zaprojektowanej specjalnie do przetwarzania danych o stałej siatce, jak np. obrazy. Splotowe sieci neuronowe wykorzystują filtry do analizy i wykrywania cech na różnych poziomach abstrakcji, co umożliwia efektywne rozpoznawanie wzorców i cech wizualnych w danych.

System, ang. *system*: To zestaw wzajemnie powiązanych elementów lub komponentów, które współpracują ze sobą w celu osiągnięcia określonego celu. System może obejmować zarówno zasoby materialne (np. urządzenia fizyczne), jak i niematerialne (np. procedury, procesy, oprogramowanie). Systemy są złożone z różnych podsystemów, które razem realizują funkcje istotne dla działania całości. Systemy mogą mieć charakter techniczny, organizacyjny, biologiczny, społeczny lub inny, a ich celem jest przetwarzanie informacji, kontrolowanie procesów lub dostarczanie usług.

System Identyfikacji Obciążeń (SIO), ang. *Load Identification System* (LIS): Podsystem systemu B-WIM, którego celem jest analiza sił wymuszających działających na konstrukcję mostową na podstawie odpowiedzi konstrukcji rejestrowanej przez czujniki oraz danych o pojeździe z SIP. Realizuje zadanie odwrotne do analizy statycznej, wyznaczając naciski na osie lub grupy osi.

System Identyfikacji Pojazdów (SIP), ang. *Vehicle Identification System* (VIS): Podsystem systemu B-WIM odpowiedzialny za detekcję pojazdów na obiekcie mostowym, określanie ich geometrii (liczba osi, rozstaw osi) oraz parametrów ruchu (prędkość, podłużna i poprzeczna lokalizacja w czasie). Uzyskane dane są przekazywane do Systemu Identyfikacji Obciążeń, gdzie stanowią podstawę dalszych analiz. Metody stosowane w SIP obejmują m.in. detekcję mechaniczną, ang. *Free Axle Detector* (FAD), Nothing On the Road (NOR) oraz systemy wizyjne.

Sztuczna Inteligencja (SI), ang. *Artificial Intelligence* (AI): Technologia, która umożliwia komputerom i maszynom symulowanie ludzkiej inteligencji i zdolności rozwiązywania problemów. Sztuczna inteligencja jako dziedzina naukowa obejmuje różnorodne podejścia i techniki, takie jak uczenie maszynowe, przetwarzanie języka naturalnego, systemy eksperckie, przetwarzanie obrazu, przetwarzanie mowy, planowanie, robotykę.

Sztuczne sieci neuronowe, ang. *Artificial Neural Network* (ANN): Model uczenia maszynowego, który podejmuje decyzje na wzór ludzkiego mózgu, korzystając z procesów naśladujących sposób, w jaki neurony biologiczne współpracują ze sobą w celu identyfikowania zjawisk, ważenia opcji i wyciągania wniosków. Sieci neuronowe opierają się na danych szkoleniowych, aby uczyć się

i poprawiać swoją dokładność w procesie uczenia. Po dostrojeniu pod kątem dokładności stają się potężnymi narzędziami w informatyce i sztucznej inteligencji.

Tablica wielowymiarowa, ang. *n-dimensional array (ndarray)*: Podstawowy obiekt w bibliotece NumPy, reprezentujący wielowymiarową tablicę liczb o stałym typie danych. ndarray umożliwia wykonywanie szybkich operacji matematycznych, takich jak algebra liniowa, analiza danych czy przetwarzanie obrazów. Jest zoptymalizowana pod kątem wydajności i obsługuje operacje wektorowe.

Uczenie maszynowe, ang. *Machine Learning* (ML): Dziedzina sztucznej inteligencji i informatyki, która koncentruje się na wykorzystaniu danych i algorytmów, aby wzorując się na procesach decyzyjnych ludzkiego mózgu w trakcie działania samodzielnie poprawiać swoją wydajność.

Uczenie nadzorowane, ang. *supervised learning:* Rodzaj uczenia maszynowego, w którym model uczy się na oznaczonych danych, czyli takich, które zawierają zarówno dane wejściowe, jak i odpowiadające im prawidłowe odpowiedzi (etykiety). Celem modelu jest nauczenie się przewidywania odpowiednich etykiet dla nowych, nieznanych danych. Uczenie nadzorowane znajduje zastosowanie m.in. w klasyfikacji i regresji.

Uczenie nienadzorowane, ang. *unsupervised learning*: Rodzaj uczenia maszynowego, w którym model uczy się identyfikować wzorce i struktury w danych bez etykiet (prawidłowych odpowiedzi). Jest wykorzystywane głównie w grupowaniu, redukcji wymiarów i detekcji anomalii.

Widzenie komputerowe, ang. *computer vision*: Dziedzina informatyki skoncentrowana na rozwijaniu algorytmów i systemów, które umożliwiają komputerom rozumienie i interpretowanie świata wizualnego. Technologie te znajdują zastosowanie w rozpoznawaniu obrazów, analizie wideo i robotyce. Widzenie komputerowe to zdolność maszyn do analizowania i interpretowania obrazów, co pozwala im na interakcję z otoczeniem w sposób zbliżony do ludzkiego postrzegania.

Współczynnik uczenia, ang. *learning rate*: Hiperparametr kontrolujący szybkość, z jaką model uczenia maszynowego aktualizuje swoje parametry w trakcie treningu. Zbyt wysoki współczynnik uczenia może spowodować niestabilne uczenie, podczas gdy zbyt niski może znacznie spowolnić proces uczenia się. Współczynnik ten jest decydujący dla tempa, w jakim model adaptuje się do błędów i poprawia swoje działanie.

Zbiór testowy, ang. *test set*: Podzbiór danych, który służy do oceny końcowej wydajności modelu po zakończeniu procesu treningu i walidacji. Zbiór testowy zawiera dane, które nie były wykorzystywane

ani do trenowania, ani do walidacji modelu, co pozwala na ocenę jego rzeczywistej zdolności do generalizacji i przewidywania wyników dla nowych, nieznanych danych.

Zbiór treningowy, ang. *training set*: Podzbiór danych używany do trenowania modelu uczenia maszynowego. Zbiór treningowy zawiera dane wejściowe wraz z odpowiadającymi im etykietami (w przypadku uczenia nadzorowanego), na podstawie których model uczy się wzorców i zależności. To na tym zbiorze model uczy się dostosowywać swoje parametry w celu minimalizacji błędów.

Zbiór walidacyjny, ang. *validation set*: Podzbiór danych używany do oceny wydajności modelu uczenia maszynowego podczas treningu. Zbiór walidacyjny jest wykorzystywany do dostosowania hiperparametrów i zapobiegania nadmiernemu dopasowaniu. To dane, które pozwalają maszynom ocenić swoją wydajność w kontrolowanych warunkach, dostosowując się do zmieniających się sytuacji.

1 Wprowadzenie

1.1 Geneza podjęcia tematu

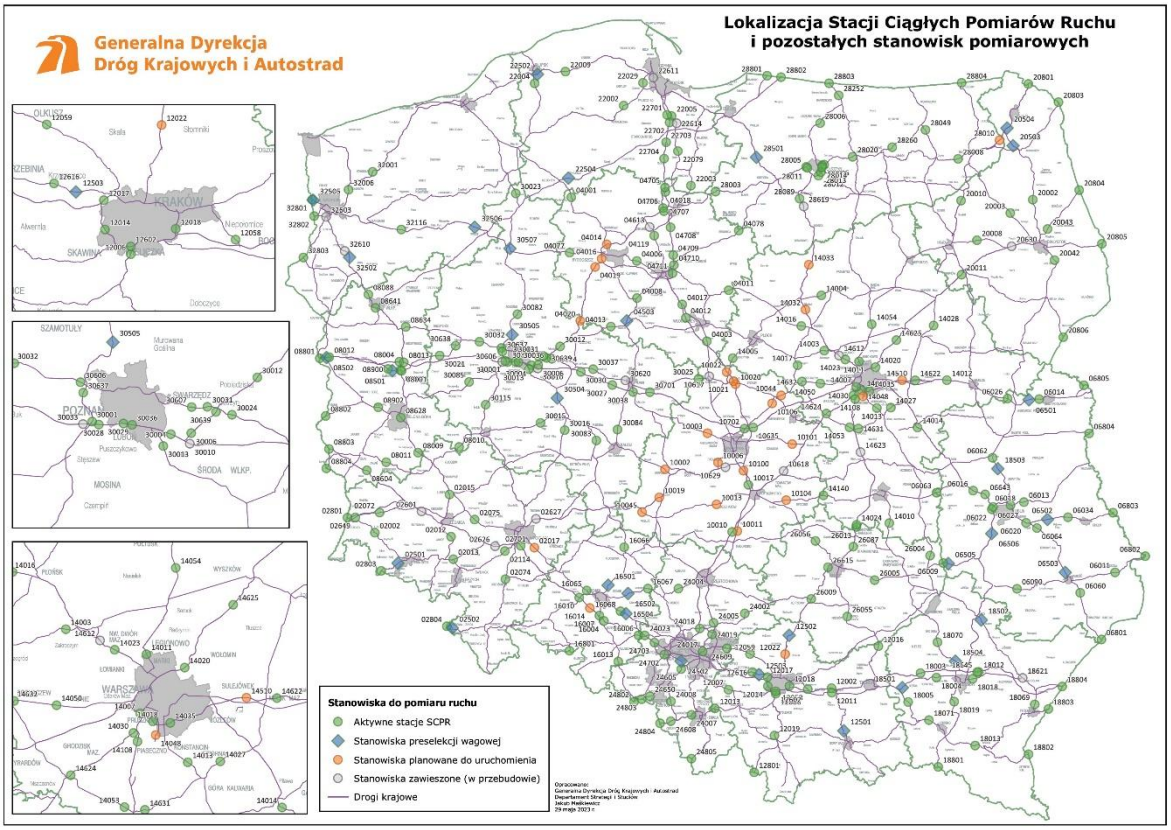
Podjęte zagadnienie określania rzeczywistej struktury obciążeń jest ważne nie tylko w odniesieniu do pojazdów przeciążonych, ale także np. na potrzeby analiz zmęczeniowych. Ruch pojazdów przeciążonych jest znaczącym problemem dla organów odpowiedzialnych za utrzymanie infrastruktury drogowej, ponieważ nadmierne obciążenia osi pojazdów ciężarowych powodują uszkodzenia dróg i mostów oraz wymuszają prowadzenie częstszych remontów, które są kosztowne. Identyfikacja pojazdów przeciążonych jest tematem intensywnie analizowanym, który jednak nie doczekał się dotąd kompleksowego rozwiązania. Z badań przeprowadzonych przez Politechnikę Gdańską [1] wynika, że udział pojazdów przeciążonych w ruchu wynosi od 14 % do 23 %, natomiast w publikacji [2] wykazano, że udział pojazdów przeciążonych w ruchu drogowym wynosi średnio 18.5 %. W badaniach zaprezentowanych w publikacji [3] udział pojazdów przeciążonych w ruchu drogowym wynosił natomiast 30 %. Wyniki analiz różnią się, chociażby z uwagi na metodykę oraz czas prowadzonych pomiarów jednak wszystkie badania jasno i wyraźnie przedstawiają fakt występowania znaczącego udziału ruchu pojazdów przeciążonych w ruchu drogowym. Skutkiem ruchu pojazdów przeciążonych jest przyspieszenie procesów degradacji wszystkich elementów infrastruktury transportowej, w tym obiektów mostowych, a także powstawanie uszkodzeń (deformacje, zarysowania i pęknięcia, destrukcja i ubytki materiału itp.). Zgodnie z [1] pojazdy przeciążone powodują od 35% do 70% szkód konstrukcji nawierzchni dróg. Dodatkowo uszkodzeniu może ulegać niewidoczna dla oka sieć podziemna jak np. sieć wodociągowa czy elektryczna. Problem ruchu pojazdów przeciążonych dotyczy praktycznie wszystkich klas technicznych dróg. Na uszkodzenia są narażone nie tylko autostrady oraz drogi szybkiego ruchu, ale głównie drogi miejskie i zlokalizowane na terenach zurbanizowanych. Problem był przedmiotem kontroli przeprowadzonej przez Najwyższą Izbę Kontroli, której wyniki zostały opublikowane w [4]. Głównym celem kontroli była odpowiedź na pytanie: „Czy organy administracji publicznej skutecznie eliminują ruch pojazdów przeciążonych na obszarach zurbanizowanych?”.

NIK skontrolowała dziewięć wojewódzkich inspektoratów transportu drogowego oraz dziesięć podmiotów zarządzających drogami. W ocenie ogólnej raportu [4], kontrolerzy NIK stwierdzili między innymi:

- Organy administracji publicznej, odpowiedzialne na terenie miast objętych kontrolą za eliminowanie z ruchu pojazdów przeciążonych nie wywiązały się w pełni ze swoich obowiązków, a tym samym nieskutecznie przeciwdziałały poruszaniu się pojazdów przeciążonych po drogach przebiegających przez obszary zurbanizowane.
- Systemy preselekcji wagowej nie miały istotnego wpływu na eliminację pojazdów przeciążonych na terenie dużych miast, mimo poniesienia znacznych wydatków na ich wybudowanie.

- Inspekcja Transportu Drogowego nie wypracowała rozwiązań w zakresie eliminacji z ruchu pojazdów przeciążonych na terenie kontrolowanych miast.
- Przyjęte i realizowane w miastach plany rozbudowy sieci dróg uwzględniały zadania inwestycyjne, mające wpływ na ograniczenie oddziaływania na infrastrukturę drogową ruchu pojazdów przeciążonych.

Podstawowym narzędziem do automatycznego mierzenia nacisków kół pojazdów na elementy infrastruktury transportowej są systemy WIM. Są to wbudowane w nawierzchnię drogową poprzeczne wagi, wraz z odpowiednim systemem monitoringu, które mierzą naciski poszczególnych osi przejeżdżających pojazdów. Następnie na podstawie odczytów, odpowiedni organ administracji publicznej podejmuje dalsze działania. Systemy te z uwagi na ograniczoną dokładność pomiaru nacisków pojazdów poruszających się z typowymi prędkościami po drogach służą za system preselekcji wagowej. Na rys. 1.1 przedstawiono lokalizację stanowisk preselekcji wagowej oraz pozostałych stanowisk pomiarowych Generalnej Dyrekcji Dróg Krajowych i Autostrad. Zauważyć należy małą liczbę stanowisk preselekcji wagowej.



Rys. 1.1 Lokalizacja Stacji Ciągłych Pomiarów Ruchu i pozostałych stanowisk pomiarowych [W 1.1]

Rozpatrując problem pojazdów przeciążonych należy również zwrócić uwagę na samą strukturę ruchu na drogach oraz jej zmiany na przestrzeni lat. Dokumentami, które przedstawiają takie dane są między innymi raporty z Generalnego Pomiaru Ruchu (GPR) [5] wykonywane od 2000 roku systematycznie co 5 lat. Wyjątkiem tutaj był rok 2020/2021 w którym, z uwagi na pandemię Covid-19, „lockdown” oraz

związane z tym zaburzenia w transporcie i logistyce, generalny pomiar ruchu został przedłużony i obejmował 2 lata.

Na podstawie danych uzyskanych z pomiarów, określono następujące parametry:

- Średni Dobowy Ruch Roczny (ŚDRR) i rodzajową strukturę ruchu w punktach pomiarowych,
- obciążenie ruchem sieci dróg krajowych i w poszczególnych województwach z uwzględnieniem podziału funkcjonalnego dróg oraz podziału na klasy techniczne.

Poza określeniem wymienionych wielkości ruchu drogowego, wykonano obliczenia analityczne opisujące:

- długości dróg w przedziałach obciążeń ŚDRR,
- zmiany wielkości ruchu drogowego,
- charakter ruchu.

W tab. 1.1 zaprezentowano dane odnoszące się do obciążenia ruchem pojazdów silnikowych oraz zmian w ruchu na sieci dróg krajowych w GPR 2015 oraz GPR 2020/21, uwzględniając funkcjonalny podział dróg.

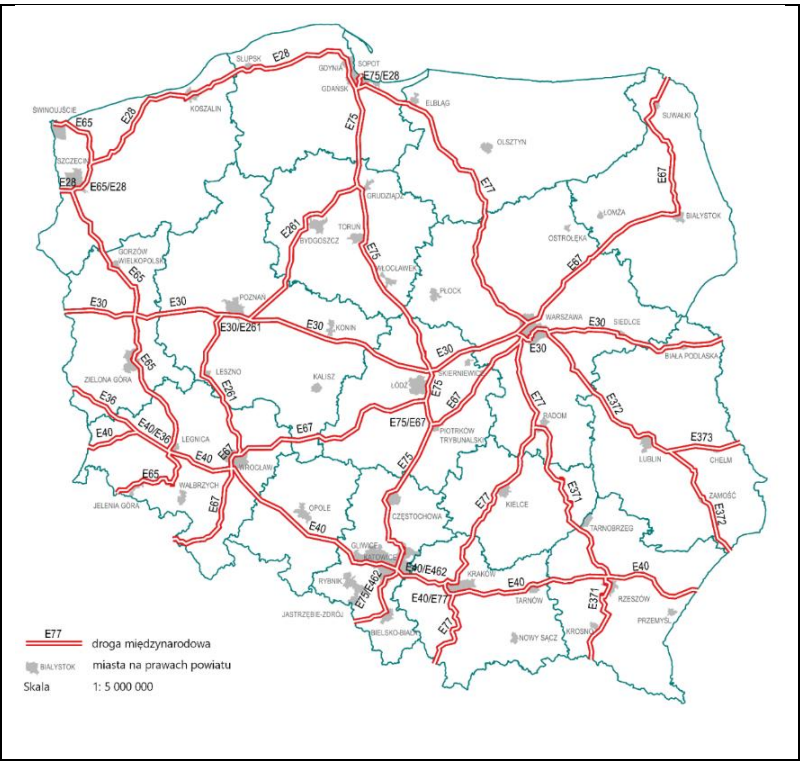
Tab. 1.1 Średni dobowy ruch roczny oraz wskaźnik zmian ruchu pojazdów silnikowych obliczony w GPR 2015 i GPR 2020/21 z uwzględnieniem podziału funkcjonalnego dróg [5]

Klasa drogi	Średni dobowy ruch roczny (ŚDRR) (poj./dobę)		Wskaźnik zmian ruchu w latach	
	Pojazdy silnikowe		Pojazdy silnikowe	
	GPR 2015	GPR 2020/21	2010-2015	2015-2020/21
Krajowe	11 178	13 574	1,14	1,21
w tym:				
Międzynarodowe	20 067	25 485	1,17	1,27
Pozostałe	7 614	8 746	1,12	1,15

Obciążenie ruchem na sieci dróg krajowych nie było jednolite i zależało od znaczenia poszczególnych odcinków. Istotne jest podkreślenie, że między GPR 2015, a GPR 2020/21 odnotowano wzrost ruchu o średnio 21%. W ramach dróg krajowych można wyróżnić drogi międzynarodowe, na których dynamika wzrostu osiągnęła 27%, oraz pozostałe drogi, gdzie wzrost był na poziomie 15%. Struktura i natężenie ruchu nie jest stała dla wszystkich dróg i zależy od lokalizacji i połączenia z ważnymi ośrodkami przemysłowymi, co zostało zaprezentowane w tab. 1.2.

Tab. 1.2 Średni dobowy ruch roczny pojazdów silnikowych na poszczególnych drogach międzynarodowych [5]

Numer drogi E	ŚDRR (2020/21) (poj./dobę)
E28	22 245
E30	28 496
E36	16 902
E40	**37 875**
E65	18 700
E67	28 517
E75	29 777
E77	30 450
E261	23 458
E371	**10 235**
E372	19 007
E373	13 188



Najbardziej obciążoną drogą była międzynarodowa droga E40 biegnąca szlakiem autostrady A4 na linii wschód-zachód oraz łącząca główne ośrodki przemysłowe południowej Polski z Ukrainą oraz Niemcami. Dużym obciążeniem charakteryzuje się też droga E77 biegnąca na linii północ-południe i przebiegająca w pobliżu stolicy kraju. Są to główne drogi odpowiadające za eksport i import dóbr z krajami ościennymi oraz łączące najważniejsze ośrodki przemysłowe w Polsce. Na sieć drogową składają się również drogi o niższych klasach technicznych. Analizując obciążenie dróg według klas technicznych, w tab. 1.3 przedstawiono odpowiednie dane z sesji pomiarowych GPR 2015 i GPR 2020/21.

Tab. 1.3 Średni dobowy ruch roczny na sieci dróg krajowych w GPR 2015 i GPR 2020/21, z uwzględnieniem podziału dróg na klasy techniczne [5]

Klasy techniczne dróg	GPR 2015		GPR 2020/2021	
	Długość (km)	SDRR (poj./dobę)	Długość (km)	ŚDRR (poj./dobę)
A – autostrady	1 556	26 509	1 712	33 749
S – ekspresowe	1 484	21 232	2 567	25 167
GP – główne ruchu przyśpieszonego	10 536	9 995	9 644	10 353
G – główne	4 446	5 260	4 333	5 900
Drogi krajowe	18 022	11 178	18 256	13 574

W sesji pomiarowej GPR 2020/21 największe obciążenie odnotowano na autostradach i drogach ekspresowych - odpowiednio ponad 3-krotnie i 2-krotnie wyższe niż na drogach głównego ruchu przyspieszonego. Najniższe obciążenie miały drogi klasy G, z wartością ŚDRR wynoszącą 5900 poj./dobę.

W kontekście prezentowanej rozprawy doktorskiej warto zwrócić uwagę na analizę struktury rodzajowej ruchu przedstawionej w tab. 1.4 i ukazującej udział różnych kategorii pojazdów w podziale na klasy techniczne dróg.

Tab. 1.4 Udział poszczególnych kategorii pojazdów silnikowych w GPR 2020/21 na drogach krajowych według klas technicznych [5]

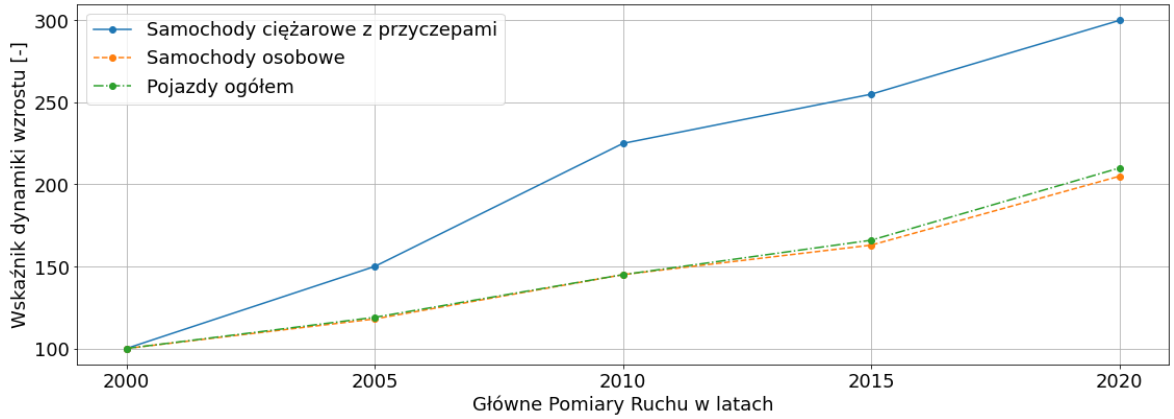
Kategorie pojazdów	Udział poszczególnych kategorii pojazdów silnikowych w SDRR 2020/21 na drogach krajowych według klas technicznych							
	A		S		GP		G	
	SDRR (poj./dobę)	[%]	SDRR (poj./dobę)	[%]	SDRR (poj./dobę)	[%]	SDRR (poj./dobę)	[%]
Motocykle	48	0,1	67	0,3	47	0,5	36	0,6
Samochody osobowe	22 241	66,0	18 393	73,0	7 644	73,8	4 615	78,2
Lekkie samochody ciężarowe (dostawcze)	3 677	10,9	2 541	10,1	1 022	9,9	580	9,8
Samochody ciężarowe bez przyczep	673	2,0	525	2,1	253	2,4	134	2,3
Samochody ciężarowe z przyczepami	7 032	20,8	3 589	14,3	1 338	12,9	499	8,5
Autobusy	78	0,2	52	0,2	38	0,4	22	0,4
Ciągniki rolnicze	-	-	-	-	11	0,1	14	0,2
Pojazdy silnikowe ogółem	33 749	100	25 167	100	10 353	100	5 900	100

ŚDRR dla klas technicznych pojazdów silnikowych zależy od klasy technicznej dróg. Najwięcej samochodów ciężarowych i lekkich pojazdów ciężarowych porusza się po autostradach (20,8% i 10,9%), a najmniej po drogach klasy G (8,5% i 9,8%). Samochody osobowe i motocykle dominują na drogach klasy G, ale są rzadsze na autostradach. Od 2015 roku rozwinęła się w Polsce sieć dróg o nowe odcinki autostrad i dróg ekspresowych. W GPR 2020/21 sieć była dłuższa o ok. 250 km w porównaniu do sieci dróg z GPR 2015. Te zmiany wpłynęły na ruch, co utrudnia bezpośrednie porównanie wyników różnych sesji GPR. Dlatego do obliczeń używa się średniej ważonej ŚDRR, z długościami odcinków jako wagami. Istotne informacje zawierają też wskaźniki zmian ruchu według kategorii pojazdów. Dane te zostały zaprezentowane w tab. 1.5.

Tab. 1.5 Wskaźnik zmian ruchu pomiędzy GPR 2010, a GPR 2015 oraz GPR 2015 a GPR 2020/21, w podziale na kategorie pojazdów [5]

Kategorie pojazdów	Wskaźnik zmian ruchu w latach 2010 – 2015	Wskaźnik zmian ruchu w latach 2015 – 2020/21
Motocykle	1,15	1,04
Samochody osobowe	1,17	1,22
Lekkie samochody ciężarowe (dostawcze)	1,05	1,42
Samochody ciężarowe bez przyczep	0,87	0,83
Samochody ciężarowe z przyczepami	1,18	1,18
Autobusy	0,90	0,49
Ciągniki rolnicze	0,81	1,0

Wskaźniki zmian ruchu dla lekkich pojazdów ciężarowych pokazują 42 % wzrost prawdopodobnie z powodu zmian w handlu spowodowanych pandemią Covid-19 oraz wzrostem popularności zakupów przez Internet. Dodatkowo notuje się wzrost o 18 % intensywności ruchu pojazdów ciężarowych z przyczepami. Na rys. 1.2 zaprezentowano dynamikę wzrostu ruchu dla różnych kategorii pojazdów na przestrzeni ostatnich 20 lat. Przyjęto jako poziom odniesienia rok 2000 oraz wartość wskaźnika ruchu na poziomie 100.



Rys. 1.2 Wskaźnik dynamiki wzrostu natężenia ruchu [5]

Na podstawie analizy wykresu można wyciągnąć następujące wnioski:

- Tendencja wzrostowa utrzymuje się na przestrzeni ostatnich 2 dekad.
- Widać wyraźną różnicę pomiędzy tempem wzrostu wskaźnika dla pojazdów ciężarowych z przyczepami, a dla pojazdów osobowych.
- Intensywność ruchu pojazdów ciężarowych wzrosła 3 krotnie w stosunku do roku 2000. Wartość ta mocno koreluje ze wzrostem PKB, które w 2000 r. wynosiło 172.2 miliarda $ oraz 599.4 miliarda $ w 2020 r.

Transportowa infrastruktura lądowa, obejmująca drogi, koleje oraz obiekty mostowe, stanowi jeden z kluczowych elementów systemu każdego państwa. Wzrost obciążenia ruchem pojazdów ciężarowych powoduje uszkodzenia nie tylko nawierzchni dróg, ale również wszelkiego rodzaju obiektów inżynierskich.

W kontekście utrzymania odpowiedniego stanu infrastruktury lądowej istotne jest stworzenie skutecznych systemów monitorowania obciążeń eksploatacyjnych. Próby opracowania takich systemów podejmowano wielokrotnie, a obecnie znane są dwie główne kategorie systemów monitorowania obciążeń użytkowych, znane w literaturze w terminologii angielskiej:

- Weigh in Motion (WIM),
- Bridge Weigh in Motion (B-WIM).

Terminy te reprezentują różne rozwiązania, których celem jest określanie nacisków użytkowych pojazdów za pomocą systemów pomiarowych wbudowanych w nawierzchnię drogi (WIM) lub wykorzystujących odpowiedź konstrukcji mostowych na obciążenie (B-WIM).
Nadal istnieje potrzeba rozwiązania istotnych wyzwań, wśród których wyróżnić można:

- eliminację pojazdów przeciążonych z ruchu, która zwiększyłaby międzyremontowy okres użytkowania dróg oraz mostów i zmniejszyłaby koszty utrzymania i napraw,
- monitoring obciążeń eksploatacyjnych dróg i mostów, który pozwoliłby na sprawne zarządzanie obiektami mostowymi.

Dlatego ważne jest, aby stworzyć jak najlepszy system, który mógłby z wysoką dokładnością określać rzeczywiste obciążenia eksploatacyjne na obiektach mostowych, co jest głównym celem niniejszej rozprawy.

W podrozdziale 1.2 zaprezentowano cele i zakres rozprawy, a w podrozdziale 1.3 przedstawiono przegląd treści pracy.

1.2 Cele i zakres rozprawy

Podstawowym celem pracy jest opracowanie auto-adaptacyjnego systemu pozwalającego na identyfikację obciążeń eksploatacyjnych drogowych obiektów mostowych na podstawie dynamicznej odpowiedzi konstrukcji mostowej z wykorzystaniem uczenia maszynowego. Istotną cechą systemu jest auto-adaptacja, czyli zdolność do samodzielnego dostosowywania parametrów modelu do zmieniających się warunków eksploatacyjnych oraz nowych danych pomiarowych, bez konieczności zewnętrznej interwencji użytkownika. W kontekście prezentowanej pracy oznacza to zdolność systemu do dynamicznej aktualizacji i optymalizacji algorytmów identyfikacji obciążeń w odpowiedzi na zmieniające się charakterystyki ruchu pojazdów, zmienne warunki środowiskowe. Mechanizm auto-adaptacyjności bazuje na wykorzystaniu metod uczenia maszynowego, które pozwalają na iteracyjne dostosowanie modelu do nowych danych w sposób analogiczny do procesu uczenia i adaptacji w systemach biologicznych. Oznacza to, że system nie tylko poprawia swoją dokładność w czasie, ale także może identyfikować i kompensować nieoczekiwane zakłócenia czy anomalie w danych wejściowych.
Opisany w podrozdziale 1.1 udział pojazdów przeciążonych w ruchu drogowym oraz charakter wskaźnika dynamiki wzrostu pokazują jak ważnym zadaniem jest stworzenie systemu, umożliwiającego określenie oraz weryfikowanie rzeczywistych nacisków pojazdów na nawierzchnię drogową oraz obiekty mostowe.

W ramach rozprawy przyjęto następujące cele cząstkowe, definiujące zakres prac:

- Przeprowadzenie przeglądu literatury, na podstawie którego możliwe jest przedstawienie aktualnego stanu wiedzy na temat systemów identyfikacji obciążeń eksploatacyjnych drogowych obiektów mostowych.
- Przeprowadzenie przeglądu literatury, na podstawie którego będzie możliwe przedstawienie aktualnego stanu wiedzy na temat metod uczenia maszynowego wykorzystywanych w pracy w aspekcie przydatności w systemach identyfikacji obciążeń obiektów mostowych.
- Określenie architektury proponowanego systemu identyfikacji parametrów obciążeń eksploatacyjnych.
- Określenie metodyki implementacji, weryfikacji i walidacji proponowanego systemu,
- Określenie dokładności oszacowań nacisku całkowitego pojazdów obciążających oraz określenie czynników wpływających na dokładność oszacowania.
- Dokonanie analizy stabilności proponowanego auto-adaptacyjnego systemu oraz określenie kluczowych czynników wpływających na jego działanie.

Efektem końcowym, będzie opracowanie systemu umożliwiającego identyfikację oraz monitoring obciążeń eksploatacyjnych drogowych konstrukcji mostowych. Na podstawie pracy systemu będzie można między innymi:

- Aktualizować modele obciążeń norm projektowych dotyczących konstrukcji mostów oraz dróg.
- Sterować ruchem w celu ograniczenia maksymalnego obciążenia na obiekcie mostowym, w przypadku wykrycia kumulacji pojazdów przeciążonych na obiekcie. Poprzez elektroniczne znaki drogowe będzie można doraźnie wyłączać poszczególne pasy na obiektach mostowych oraz wprowadzać ograniczenia prędkości w celu redukcji uszkodzeń nawierzchni oraz konstrukcji mostowych.
- Aktualizować i monitorować zmiany oraz tendencje w strukturze ruchu w celu optymalnego projektowania nowych oraz modernizowania istniejących odcinków dróg na podstawie określonych nacisków np. poprzez podniesienie bądź obniżenie wymagań stawianych monitorowanym drogom oraz konstrukcjom mostowym.
- Monitorować obciążenia w aspekcie procesów zmęczeniowych konstrukcji, co dla wielu obiektów może stanowić dobrą i wiarygodną bazę do określania dalszej przydatności użytkowej.

1.3 Przegląd treści pracy

Niniejsza dysertacja podzielona została na 7 rozdziałów.

- Pierwszy rozdział stanowi wprowadzenie. Omówiono problem wpływu obciążeń pojazdami na stan infrastruktury transportowej. Przedstawiono w nim motywację do podjęcia tematu pracy, cele naukowe oraz zakres rozprawy. Rozdział zakończony jest syntetycznym przeglądem treści rozprawy.
- Rozdział drugi obejmuje analizę literatury przedmiotu w zakresie systemów identyfikacji obciążeń eksploatacyjnych drogowej infrastruktury transportowej. Podzielono go na tematyczne podrozdziały, omawiające idee drogowych i mostowych systemów identyfikacji obciążeń, systemów detekcji i identyfikacji pojazdów, a także omówiono algorytmy identyfikacji obciążeń. Przedstawiono stosowane techniki pomiarowe, w tym monitoring sensoryczny obiektów mostowych oraz charakterystykę stosowanych czujników. Na końcu przedstawiono podsumowanie omawianych systemów, a także zasygnalizowano technologie, które zastosowano w pracy doktorskiej do realizacji założonych celów.
- W rozdziale trzecim przedstawiono koncepcję systemu identyfikacji obciążeń z wykorzystaniem uczenia maszynowego. Omówiono podstawowe informacje o uczeniu maszynowym, w tym sieci neuronowe i uczenie głębokie, neuronowe sieci splotowe oraz autodekodery. Następnie zaprezentowano schemat budowy systemu identyfikacji obciążeń obiektów mostowych i metodykę implementacji systemu.

- Rozdział czwarty zawiera opis symulatora dynamicznej odpowiedzi konstrukcji mostowej. Szczegółowo przedstawiono rozważane modele obliczeniowe, walidację modeli obliczeniowych oraz analizę wpływu wybranych parametrów na wyniki symulacji, w tym wpływ prędkości przejazdu, tłumienia konstrukcji, masy i rozpiętości przęseł, sztywności oraz tłumienia zawieszenia pojazdu, a także nierówności nawierzchni. Opisano także generator populacji pojazdów i przedstawiono weryfikację wyników jego działania.
- Rozdział piąty zawiera opis autorskiego auto-adaptacyjnego systemu identyfikacji obciążeń obiektów mostowych pojazdami, dla którego przedstawiono założenia oraz szczegółową strukturę. Opisano zastosowaną metodykę uczenia maszynowego, analizę błędów określenia nacisku całkowitego pojazdu oraz nacisków na poszczególne osie, a także wpływ prędkości i masy pojazdu na błąd odtworzenia.
- Rozdział szósty przedstawia analizę stabilności proponowanego systemu. Omówiono wpływ różnych parametrów na działanie systemu, takich jak rozpiętość teoretyczna obiektu mostowego, masa własna obiektu mostowego, nierówności nawierzchni, liczba danych uczących, szum pomiarowy odpowiedzi konstrukcji obiektu mostowego, niepewność określenia prędkości pojazdu, niepewność określenia konfiguracji osi pojazdu, szum pomiarowy przy określaniu funkcji wpływu oraz błąd określenia funkcji wpływu. Każda z tych analiz zawiera szczegółowe informacje, wybrane efekty działania sieci oraz dokładność określenia nacisku całkowitego pojazdu.
- Rozdział siódmy zawiera podsumowanie, wnioski oraz proponowane kierunki dalszych badań.

Dodatkowe części dysertacji to:

- Wykaz ważniejszych oznaczeń stosowanych w rozprawie oraz sekcja terminologiczna – zamieszczone na początku pracy.
- Piśmiennictwo, gdzie zebrano bibliografię wykorzystaną w pracy – przedstawione po rozdziale 7.
- Streszczenie w j. polskim oraz w j. angielskim,
- Załączniki A, B oraz C, które zawierają szczegółowe informacje o implementacji poszczególnych elementów systemu.

2 Systemy monitorowania obciążeń eksploatacyjnych infrastruktury drogowej – przegląd wiedzy

2.1 Rozwój drogowych systemów monitorowania obciążeń eksploatacyjnych

Według definicji przyjętej w [6], sformułowanej przez American Society for Testing and Materials drogowe systemy monitorowania obciążeń eksploatacyjnych znane w literaturze jako ang. *Weigh-In-Motion* (WIM) bazują na procesie pomiaru dynamicznych nacisków kół poruszającego się pojazdu i szacowania odpowiadających im nacisków statycznych. Pierwsza koncepcja identyfikacji parametrów pojazdów za pomocą czujników wbudowanych w nawierzchnię drogową została opublikowana w 1952 roku w artykule *„Weighing Vehicles in Motion”* [7] autorstwa O.K. Normann’a oraz R.C. Hopkins’a, gdzie opisana została koncepcja elektronicznego urządzenia, będącego w stanie mierzyć naciski poszczególnych osi, odległości między osiami oraz wykrywać prędkość poruszających się pojazdów. Opisywane urządzenie składało się z wąskiej, swobodnej platformy wbudowanej w nawierzchnię pasa ruchu i podpartej słupkami, do których przymocowane były grupy tensometrów. Nacisk przyłożony do platformy przez przejeżdżający po niej pojazd powodował zmiany w natężeniu prądu przepływającego przez tensometry, a zmiany te za pomocą sprzętu elektronicznego były odwzorowywane na przykład jako wzór światła na oscyloskopie, co zostało przedstawione na rys. 2.1.

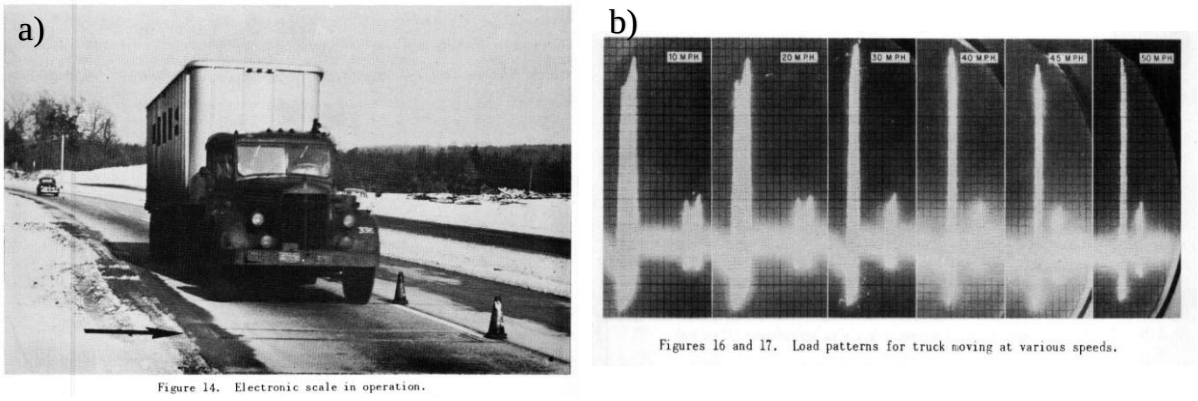


Figure 14. Electronic scale in operation.

Figures 16 and 17. Load patterns for truck moving at various speeds.

Rys. 2.1 a) Przejazd ciągnika siodłowego wraz z naczepą po urządzeniu pomiarowym wbudowanym w nawierzchnię; b) Odpowiedź urządzenia pomiarowego na przejazd pojazdu z różną prędkością [7]

Technologia nie spełniała oczekiwań, gdyż wyniki były obarczone bardzo dużym błędem pomiarowym. Prędkość najazdu miała decydujący wpływ na rezultat odczytów. Przy prędkości 10 mil/godz błąd oszacowania masy pojazdu o 3 osiach wynosił 33.4%, a przy prędkości 40 mil/godz błąd wyniósł ponad 100%. Główną wadą systemów WIM jest duża niedokładność pomiaru. W początkowym okresie rozwoju systemów WIM uważano, że jedyna przyczyna dużej niedokładności uzyskiwanych wyników ważenia leży po stronie pojazdu i wynika ze zmienności nacisku kół na podłoże. Nie jest to pełna prawda, ponieważ dokładność systemu ważenia pojazdów w ruchu zależy też od budowy samego systemu pomiarowego. Technologia systemów WIM była rozwijana przez wiele lat, a aktualny stan

wiedzy o niej został zaprezentowany m.in. w serii artykułów ([8], [9], [10], [11]), które obejmują przegląd i klasyfikację systemów.

W obecnie stosowanych rozwiązaniach rzeczywistym wynikiem uzyskiwanym z bezpośrednich pomiarów systemem WIM jest chwilowa wartość nacisku kół pojazdu na podłoże w momencie przejazdu przez urządzenie pomiarowe. Rozkładając chwilowy nacisk kół poruszających się pojazdów, możemy wymienić dwie składowe:

- składowa statyczna,
- składowa dynamiczna.

Siła statyczna jest efektem oddziaływania grawitacji, a zarazem to do niej odnoszą się rozporządzenia, które określają dopuszczalne naciski (np. [12], [13]). Składowa dynamiczna jest efektem sił bezwładności, które oddziałują na poruszający się pojazd oraz wielu dodatkowych czynników zewnętrznych takich jak np.:

- prędkość ruchu pojazdu,
- stan nawierzchni drogowej,
- typ i ogólny stan techniczny zawieszenia.

Zaznaczyć trzeba, że w skrajnych przypadkach wartości składowych dynamicznych, mogą osiągnąć aż 40% składowej statycznej.
Głównymi zaletami systemów WIM są:

- brak konieczności zatrzymywania kontrolowanych pojazdów,
- wykonywanie pomiarów w szerokim zakresie prędkości pojazdów,
- automatyczne działanie,
- mały koszt pojedynczego pomiaru,
- stosunkowo mały koszt budowy stanowiska pomiarowego (w porównaniu z wagą statyczną).

W artykule [11] oceniono w sposób ilościowy wpływ każdego z czynników z osobna na wyniki otrzymywane za pomocą systemu WIM, a wybrane rezultaty przedstawiono w tab. 2.1 oraz w tab. 2.2.

Tab. 2.1 Główne czynniki wpływające na dokładność systemów WIM [11]

Czynniki związane z pojazdem	Czynniki związane z systemem
Pionowe wahania pojazdu mające wpływ na amplitudę składowej dynamicznej nacisku osi (zależne od: jakości drogi, prędkości pojazdu, parametrów jego zawieszenia, masy całkowitej oraz rodzaju ogumienia i ciśnienia w oponach)	Kategoria nawierzchni rozumiana jako jej jakość (zależna od: geometrii drogi, jakości materiałów użytych do budowy drogi, etc.)
	Liczba zastosowanych czujników nacisku (zależna od: pożądanej dokładności wyników ważenia, dostępnych środków finansowych)
Zmiana prędkości pojazdu na stanowisku ważącym (zależna od: czynnika ludzkiego, ukształtowania terenu, oznakowania)	Rodzaj zastosowanych czujników nacisku oraz ich klasa (zależne od: przeznaczenia systemu ważącego (administracyjny, preselekcyjny), dostępnych środków finansowych)
	Zmiana temperatury nawierzchni oraz jej gradient wzdłuż stanowiska WIM (zależne od: warunków klimatycznych, zacienienia drogi)

Tab. 2.2 Dodatkowe czynniki wpływające na dokładność systemów WIM [11]

Czynniki związane z pojazdem	Czynniki związane z systemem
Efekt windy powietrznej (np. odciążanie przodu pojazdu) – tylko przy ocenie nacisków osi (zależny od: prędkości wiatru, prędkości pojazdu i jego sylwetki)	Przestrzenna powtarzalność nacisku osi (zależna od: profilu jezdni na ścieżce przejazdu pojazdów przez stanowisko WIM)
	Czas relaksacji nawierzchni drogi i czujników nacisku (zależne od właściwości materiałów użytych do budowy drogi)
Konfiguracja osi – problem przy wyznaczaniu nacisków osi składowych osi wielokrotnej (zależne od: czasu relaksacji nawierzchni drogi i czujników nacisku)	Niepewność pomiaru prędkości pojazdu (zależna od: metody pomiaru prędkości, dokładności montażu czujników i ich właściwości)
	Dokładność kalibracji systemu

Badania zaprezentowane w artykule [11] pozwoliły na ilościową ocenę wpływu różnych czynników na dokładność ważenia. Poniżej przedstawiono najważniejsze wnioski:

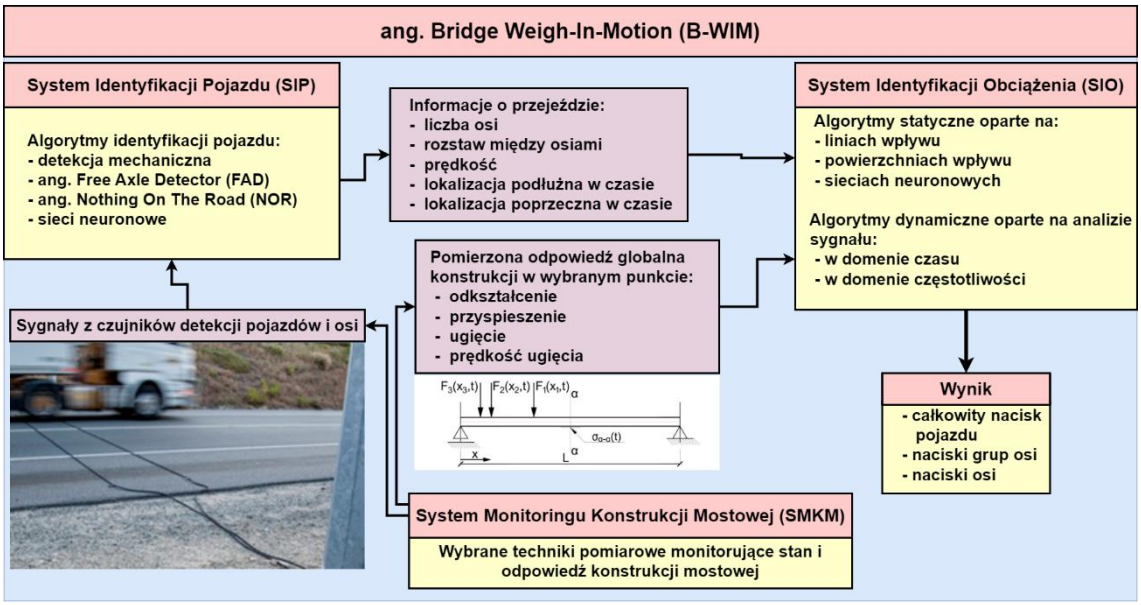
- Pionowe wahania pojazdu i zmiany temperatury nawierzchni (dla czujników polimerowych) mają największy niekorzystny wpływ na dokładność.
- Amplituda dynamicznej składowej nacisku osi zależy głównie od rodzaju nawierzchni i prędkości pojazdu.
- Zmniejszenie wpływu pionowych wahań można osiągnąć przez odpowiednie umiejscowienie stanowiska pomiarowego systemu WIM, ograniczenie prędkości do 60 km/h i użycie więcej niż dwóch czujników (a najlepiej 16).
- Aby zminimalizować wpływ niejednorodności czułości czujników, należy używać czujników o najlepszych parametrach, preferencyjnie – kwarcowych.
- W systemach z czujnikami polimerowymi, konieczne jest wyposażenie ich w algorytmy kompensujące zmiany temperatury, takie jak korekcja temperaturowa czy auto kalibracja.

Choć teoria jest obiecująca, w praktyce spełnienie wszystkich tych wymagań jest trudne. Popularne systemy dwuczujnikowe służą głównie do wstępnej selekcji pojazdów, a nie do dokładnej identyfikacji masy.

2.2 Rozwój mostowych systemów monitorowania obciążeń

Alternatywą dla wag wbudowanych w nawierzchnię drogową jest mostowy system monitorowania obciążenia eksploatacyjnego znany w literaturze jako system ang. *Bridge Weigh-in-Motion* (B-WIM), który wykorzystuje obiekt mostowy do identyfikacji parametrów, głównie masy, poruszających się po obiekcie pojazdów na podstawie historii efektów oddziaływań.

Pierwsza koncepcja systemu została zaproponowana przez F. Moses'a w artykule [14] z 1979 roku. Kolejne systemy AXWAY [15] oraz CULWAY [16], bazujące na koncepcji Moses'a, zostały zaproponowane i przetestowane z początkiem lat 80-tych w Australii. Już wtedy zauważono problem z uwzględnieniem dynamicznej natury zjawiska, dlatego próbowano wykorzystać przepusty jako obiekty pomiarowe z uwagi na zdolność gruntu pokrywającego przepust do tłumienia drgań [16]. W Europie również trwały intensywne prace nad systemami WIM i B-WIM na przełomie XX i XXI w. w ramach projektów „COST 323" [17] oraz „WAVE" [18], które wniosły wiele udoskonaleń do całego systemu oraz doprowadziły ostatecznie do powstania pierwszego komercyjnego systemu B-WIM o nazwie SiWIM. Systemy B-WIM wydają się posiadać duży potencjał i przewagę nad innymi rozwiązaniami, ponieważ są bardziej niezawodne. Wykorzystywane urządzenia pomiarowe są montowane na dolnej powierzchni dźwigara mostowego lub innego elementu konstrukcyjnego przęsła i dzięki temu nie są narażone na bezpośrednią ekspozycję na warunki pogodowe oraz na kontakt z pojazdami [19]. Montaż systemów B-WIM jest bezpieczny i nie wymaga zamykania ruchu drogowego w trakcie prowadzenia prac. Ponadto systemy B-WIM dokładniej szacują całkowity nacisk pojazdu od systemów WIM. Systemy B-WIM charakteryzują się możliwością zastosowania wielu metod w celu określania nacisków pojazdu. Konstrukcję większości systemów można przedstawić za pomocą schematu blokowego (rys. 2.2). Na kompletny system składa się System Identyfikacji Pojazdu (SIP) oraz System Identyfikacji Obciążeń (SIO). Są to oddzielne i najczęściej niezależne od siebie systemy, wykorzystujące dane z monitoringu sensorycznego obiektów mostowych oraz różne algorytmy i metody do opracowania informacji. Ich sposób funkcjonowania został opisany w kolejnych podrozdziałach. Efektem końcowym działania całego systemu jest określona całkowita masa przejeżdżającego pojazdu oraz ewentualnie jej rozkład na grupy osi, czy poszczególne osie.



Rys. 2.2 Schemat działania mostowych systemów monitorowania obciążenia eksploatacyjnego

2.3 Systemy Identyfikacji Pojazdów

2.3.1 Wprowadzenie

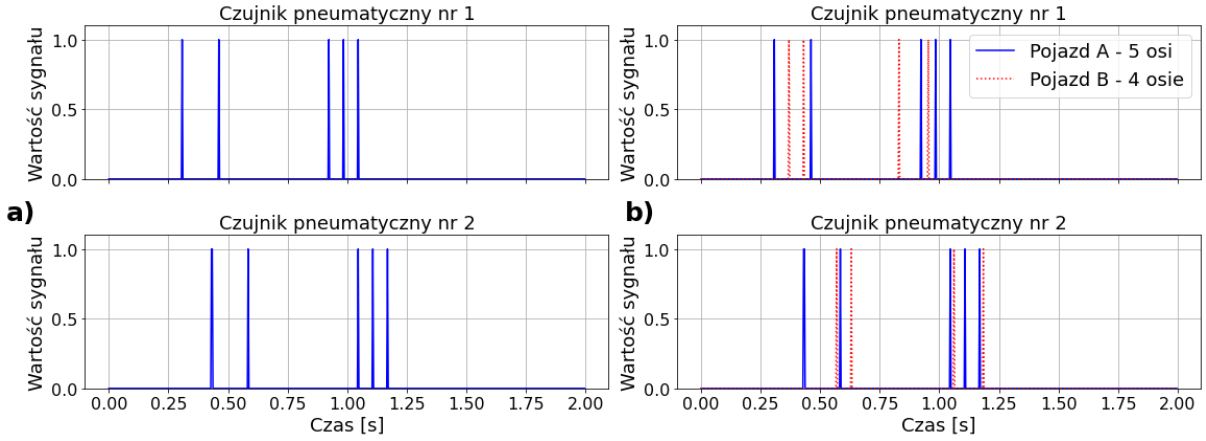
Zgodnie z przedstawionym na rys. 2.2 schematem jednym z podstawowych elementów składowych B-WIM jest System Identyfikacji Pojazdu (SIP), którego zadaniem jest wykrycie obecności pojazdu na obiekcie, określenie jego geometrii (liczby i rozstawu osi) oraz parametrów ruchu (prędkości oraz podłużnej i poprzecznej lokalizacji w czasie). Jest to niezbędne, ponieważ te podstawowe informacje o przejeździe są dalej przekazywane jako kluczowe dane do Systemu Identyfikacji Obciążeń. Z uwagi na różne wymagania stawiane takim systemom możemy wyróżnić kilka podstawowych Systemów Identyfikacji Pojazdów, którymi są:

- systemy detekcji mechanicznej,
- ang. *Free Axle Detector* (FAD),
- ang. *Nothing On the Road* (NOR),
- systemy bazujące na systemach wizyjnych.

2.3.2 Mechaniczna detekcja pojazdu

Podstawowe rozwiązanie, tzw. system detekcji mechanicznej, składa się z pary czujników pneumatycznych ułożonych równolegle względem siebie na całej szerokości jezdni tuż przed obiektem mostowym (rys. 2.3).

Rys. 2.3 System identyfikacji pojazdu w postaci pary czujników pneumatycznych zamontowanych w nawierzchni drogowej [20]



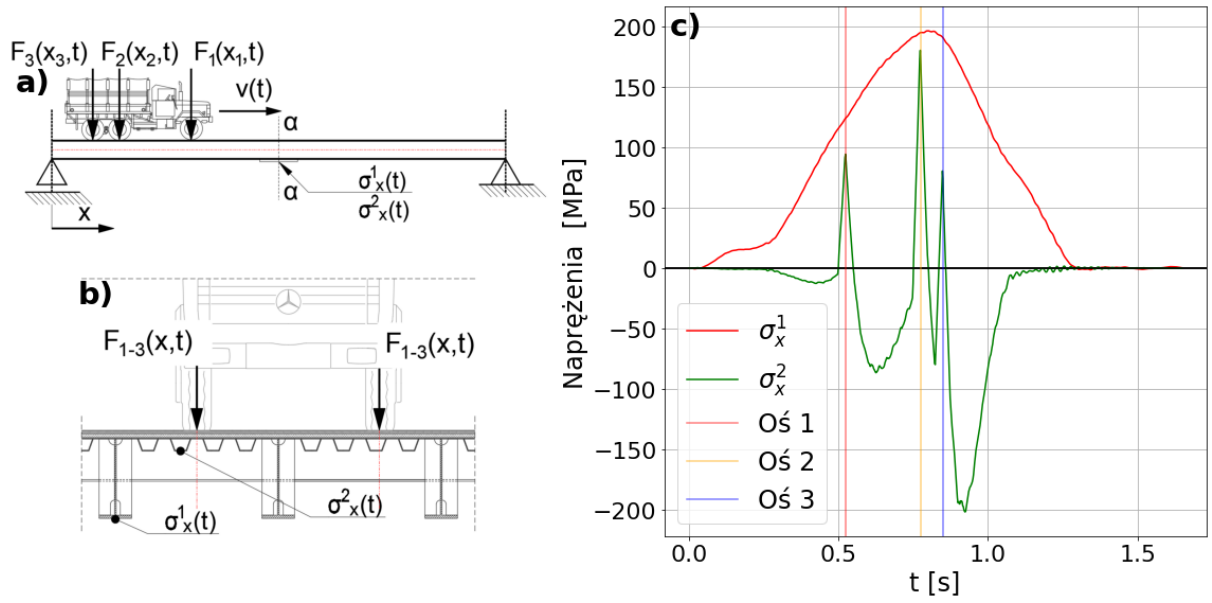
Rys. 2.4 Zapis sygnałów w trakcie przykładowych przejazdów przez czujniki pneumatyczne: a) pojazdu A o 5 osiach (wykresy po lewej), b) pojazdu A o 5 osiach i wymijającego go pojazdu B o 4 osiach (wykresy po prawej)

Przejazd kół pojazdu generuje zmianę ciśnienia wewnątrz układu pomiarowego, która jest wielkością mierzalną. Porównując różnice czasowe pomiędzy wywołanymi impulsami, określa się prędkość, liczbę oraz konfigurację osi i na tej podstawie aproksymuje się podłużną lokalizację w czasie przejazdu pojazdu po obiekcie (rys. 2.4) [21]. Jest to najprostsze i stąd wykorzystywane w wielu badaniach rozwiązanie, które zostało też zastosowane we wspomnianym algorytmie Moses'a, czy też w systemach AXWAY oraz CULWAY. Rozwiązanie takie niesie ze sobą pewne komplikacje związane z jakością danych oraz niezawodnością. Wbudowanie czujników pneumatycznych wiąże się z naruszeniem nawierzchni drogowej i narażeniem jej na szybsze uszkodzenie. Dodatkowo nie uwzględnia się lokalizacji poprzecznej, co generuje błąd w końcowej estymacji obciążeń, zgodnie z publikacją [22]. Występuje tu też problem w identyfikacji pojazdów w przypadku intensywnego ruchu, co zostało przedstawione na rys. 2.4a. Przy niewielkim natężeniu ruchu, można z łatwością odseparować sygnały wygenerowane przez różne pojazdy. W przypadku, kiedy pojazdy poruszają się w niewielkich odległościach albo ruch odbywa się na dwóch pasach, istnieje problem powiązania pomierzonych

sygnałów z odpowiednimi pojazdami. W związku z tym powyższe rozwiązanie sprawdza się zwłaszcza w przypadku wykonywania krótkookresowych badań na obiektach o niewielkim natężeniu ruchu.

2.3.3 Detekcja i identyfikacja pojazdów na podstawie odpowiedzi konstrukcji mostowej

W celu zapewnienia większej niezawodności oraz trwałości układu pomiarowego w projekcie WAVE [23] zaproponowano nowy system pod nazwą ang. *Free Axle Detector* (FAD). W założeniu czujniki pomiarowe montowane są w wybranych miejscach na dolnej powierzchni dźwigara mostowego lub innego elementu konstrukcyjnego przęsła w celu wykrywania przejazdu pojazdów. Najczęściej czujniki montuje się w miejscach dających wyraźny skok wartości mierzonej wielkości, np. nad punktami podparcia konstrukcji, z uwagi na charakterystyczną i przewidywalną trajektorię naprężeń głównych w strefie podporowej [24], bądź też na żebrach pomostu ortotropowego [23]. Ideę systemu przedstawiono na rys. 2.5, na którym pokazano różnicę jakościową między globalną i lokalną odpowiedzią konstrukcji. Jako teoretyczny model ilustrujący ten problem przyjęto jednoprzęsłowy układ składający się z dźwigarów głównych, poprzecznic i pomostu ortotropowego oraz zadano na nim przejazd trzech sił F_i reprezentujących naciski osi pojazdu. Rys. 2.5c) przedstawia zmianę naprężeń $\sigma_x(t)$ dla wybranych punktów w przekroju α-α: dźwigara głównego (1) i żebra pomostu (2).



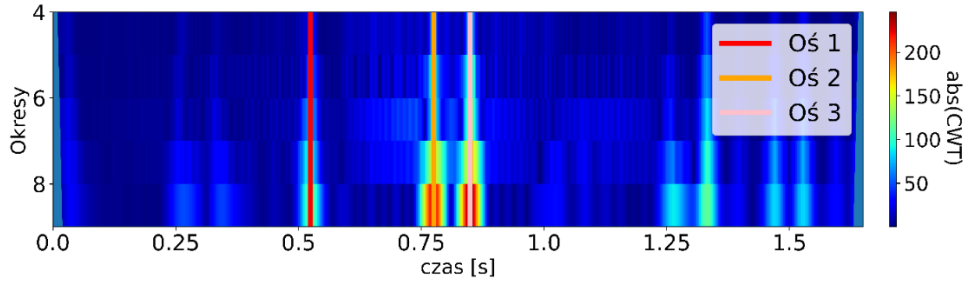
Rys. 2.5 Schemat ideowy systemu typu FAD.
a) Przyjęty w analizie schemat statyczny konstrukcji i modelowe obciążenie,
b) przekrój poprzeczny α-α,
c) wykresy naprężeń σx(t) w dźwigarze głównym i w żebrze pomostu w przekroju α-α

Na wykresie naprężeń w żebrze (w odróżnieniu od dźwigara głównego) zauważalne są charakterystyczne wierzchołki odpowiadające przejazdom osi, które pozwalają zidentyfikować ich liczbę i rozstaw. Z uwagi na niewielki zakres odkształceń pionowych elementów konstrukcyjnych, bądź też duży szum pomiarowy, nie możemy wykorzystać informacji z czujników FAD do określania

nacisków pojazdów [25]. Z tego względu rola systemów FAD ogranicza się do samej identyfikacji pojazdu oraz parametrów jego ruchu. Dodatkowo, co też określono w trakcie projektu WAVE [23], systemy te najlepiej spisują się dla krótkich obiektów mostowych, o rozpiętości 6-12 m, o konstrukcji z pomostem ortotropowym, klasycznej konstrukcji zespolonej ze stosunkowo cienką płytą lub typowej żelbetowej konstrukcji belkowo-płytowej. Montaż czujników na dolnej powierzchni dźwigara mostowego lub elementu konstrukcyjnego przęsła wyklucza możliwość bezpośredniego kontaktu z kołami pojazdu i zwiększa trwałość zarówno systemu pomiarowego, jak i samej nawierzchni drogowej. W praktyce do wykrywania osi oraz określania prędkości ruchu wykorzystuje się analizę sygnałów, na przykład przez wyznaczanie korelacji funkcji sygnału z dwóch czujników oddalonych od siebie o znaną odległość. Dodatkowo detekcja osi za pomocą systemu FAD pozwala na określanie nie tylko pozycji podłużnej, ale także poprzecznej przy odpowiednio zwiększonej liczbie czujników.

Inny system detekcji osi znany pod nazwą ang. *Nothing-On-the-Road* (NOR) jest uogólnieniem FAD i wykorzystuje nie tylko odkształcenia lokalne np. środników nad podporami [26], ale także sygnały otrzymywane w ramach globalnej odpowiedzi konstrukcji. System NOR zakłada, że sygnały z czujników mierzących globalną odpowiedź konstrukcji mogą posłużyć do identyfikacji pojazdu. Jest to zagadnienie skomplikowane z uwagi na fakt, że czujniki te są zazwyczaj zlokalizowane w środku rozpiętości konstrukcji, a ich sygnał charakteryzuje się łagodnym, ciągłym przebiegiem wartości. W celu uzyskania wyników stosuje się zazwyczaj narzędzia matematyczne z dziedziny analizy sygnałów. W badaniach opisanych m.in. w publikacjach [27], [28], [29] których przedmiotem jest analiza sygnałów naprężeniowych z wykorzystaniem systemu NOR, zastosowano podejście oparte na ciągłej transformacie falkowej [30], która w przeciwieństwie do analizy Fouriera pozwala zachować czasową informację o sygnale.

Stosowana procedura prowadzi do uzyskania z globalnej odpowiedzi konstrukcji informacji o konfiguracji osi. Przykład zastosowania ciągłej transformaty falkowej do sygnału naprężeń uzyskanego za pomocą tego samego modelu co na rys. 2.5, lecz tym razem dotyczących dźwigara głównego, przedstawiono na rys. 2.6.



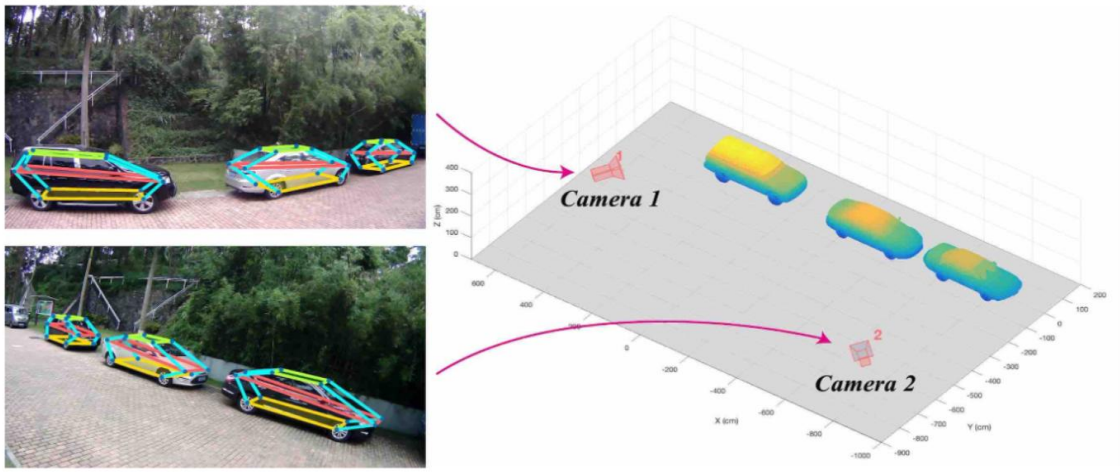
Rys. 2.6 Przykładowy spektrogram wynikowy ciągłej transformaty falkowej sygnału odpowiedzi konstrukcji z momentami przejazdu kolejnych osi zaznaczonymi liniami

Zauważalne są wyraźne odpowiedzi odpowiadające wystąpieniom osi pojazdu, które pokrywają się z wynikami z rys. 2.5c (zaznaczonymi pionowymi liniami).

Systemy detekcji mechanicznej, FAD i NOR łączy wspólna cecha, którą jest dyskretna forma analizy. Oba systemy określają konfigurację osi pojazdu oraz prędkość na podstawie analizy sygnałów np.: porównując przesunięcie w czasie wybranych wartości. Następnie zakładają, że pojazd porusza się ze stałą prędkością po całym obiekcie. Nie jest znana dokładna funkcji lokalizacji osi na obiekcie w czasie, lecz przyjmowana jest jej aproksymacja.

2.3.4 Detekcja i identyfikacja pojazdów na podstawie widzenia komputerowego

Najnowsze systemy służące do identyfikacji pojazdów wykorzystują informacje nie tylko z czujników pomiarowych, ale np. także z kamer.



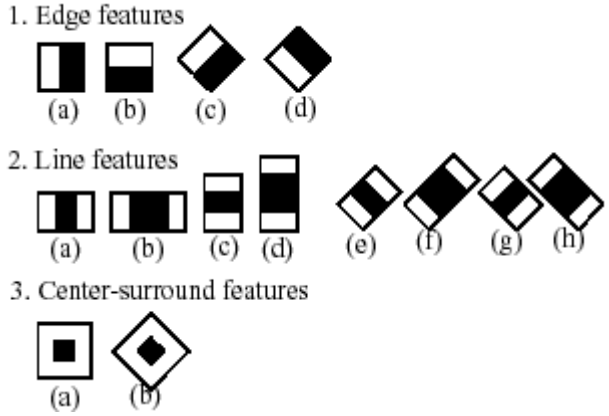
Rys. 2.7 Estymacja konfiguracji pojazdu na podstawie systemów wizyjnych [31]

Z uwagi na intensywny rozwój wiedzy z zakresu sztucznej inteligencji oraz uczenia maszynowego można wykorzystać systemy wizyjne oparte na widzeniu komputerowym do klasyfikowania oraz śledzenia lokalizacji pojazdów na obiekcie mostowym z wykorzystaniem sztucznych sieci neuronowych. W artykule [31] opisano proces technologiczny, który ma na celu określenie pozycji i orientacji pojazdów w przestrzeni za pomocą systemów wizyjnych. Metoda ta wykorzystuje kamery umieszczone w różnych lokalizacjach. Dane obrazowe są analizowane przy użyciu algorytmów przetwarzania obrazów i uczenia maszynowego, które potrafią wykrywać cechy charakterystyczne pojazdów i na ich podstawie szacować położenie (np. przód, tył, boki pojazdu) oraz orientację (kierunek, w którym pojazd jest zwrócony). Dzięki tej metodzie możliwe jest monitorowanie ruchu pojazdów, ich automatyczne rozpoznawanie i śledzenie na drogach.

Możemy wyszczególnić kilka algorytmów stosowanych w zagadnieniu wizyjnej detekcji obiektów, np. pojazdów:

- kaskady Haar'a, ang. *Haar Cascade* [32],
- histogram zorientowanych gradientów, ang. *Histogram of Oriented Gradients* (HOG) [33],
- ang. *Region Convolutional Neural Networks* (R-CNN) [34],
- ang. *You Only Look Once* (YOLO) [35],
- ang. *Single Shot multibox Detector* (SSD) [36],
- ang. *Region-based Fully Convolutional Network* (R-FCN) [37].

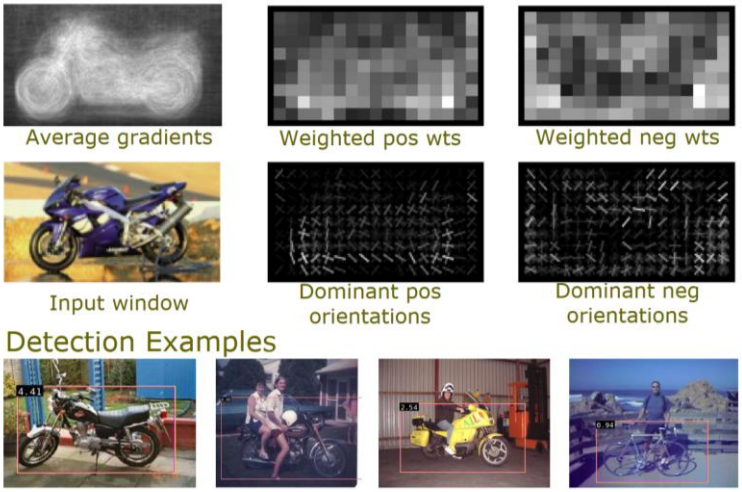
Kaskady Haar'a (ang. *Haar Cascade*) to jedna z najbardziej efektywnych metod detekcji twarzy, bazująca na analizie obrazu poprzez porównanie z zestawem cech przypominających te widoczne na rys. 2.8 . Algorytm został opracowany przez Paul Viola i Michela J. Jones'a i opisany w artykule [32].



Rys. 2.8 Zestaw cech kaskady Haar'a [38]

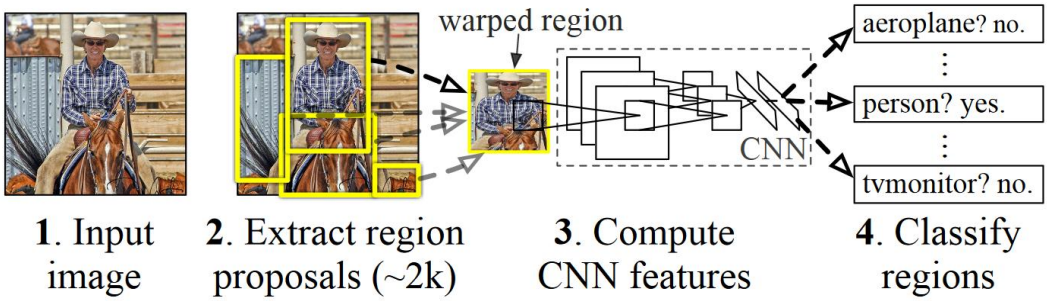
Chociaż kaskady Haara są bardzo efektywne w wykrywaniu twarzy, ich stosowanie do lokalizacji pojazdów i precyzyjnych elementów pojazdów, takich jak osie czy koła, może być mniej skuteczne. Kaskady Haara najlepiej radzą sobie z obiektami o wyraźnych, regularnych granicach i charakterystycznych cechach, takimi jak ludzka twarz. Pojazdy i ich komponenty, takie jak koła, mają bardziej złożone kształty i mogą różnić się w zależności od modelu, kąta widzenia czy warunków oświetleniowych, co utrudnia ich efektywną detekcję przy użyciu tego podejścia.

Histogram Zorientowanych Gradientów (ang. *Histogram of Oriented Gradients,* HOG) to klasyczny algorytm do detekcji obiektów, wprowadzony przez Navneed Dalala i Billa Triggsa w 2005 roku [33]. Metoda ta bazuje na analizie gradientów obrazu w celu wykrywania charakterystycznych krawędzi i wzorów, które mogą reprezentować interesujące, obiekty, takie jak pojazdy lub ludzie. W ramach warsztatów Pascal Visual Object Classes w 2006 r. Dalal i Triggs przedstawili wyniki zastosowania deskryptorów histogramu zorientowanych gradientów do innych obiektów na obrazach, takich jak samochody, autobusy i rowery, a także popularnych zwierząt, takich jak psy, koty i krowy. Dołączyli do wyników optymalne parametry dla formułowania i normalizacji bloków w każdym przypadku. Na rys. 2.9 przedstawiono przykład wykrywania motocykli.



Rys. 2.9 Zastosowanie HOG do wykrywania motocykli [39]

R-CNN (ang. *Regions with Convolutional Neural Networks*) to rodzina algorytmów służących do detekcji obiektów, opublikowana w artykule [34]. Pierwsza wersja, opracowana w 2014 roku, stanowiła istotny krok naprzód w dziedzinie wykrywania obiektów, łącząc klasyczne techniki wizji komputerowej z możliwościami głębokiego uczenia. Algorytm używa zewnętrznego algorytmu, takiego jak Selective Search, aby znaleźć propozycje regionów, które mogą zawierać obiekty. Przeskalowane regiony są podawane jako wejście do wstępnie wytrenowanej konwolucyjnej sieci neuronowej (CNN). Sieć generuje wektory cech dla każdego regionu.



Rys. 2.10 Schemat lokalizowania oraz klasyfikowania wybranych elementów na obrazie [34]

R-CNN zapewnia dużą precyzję w detekcji obiektów, dzięki użyciu głębokiego uczenia do ekstrakcji cech. Przemyślana struktura kroków przetwarzania pozwala na uzyskanie lepszych wyników niż wcześniejsze metody. Jest stosunkowo wolny, ponieważ każdy proponowany region jest analizowany oddzielnie. Wymaga znacznych zasobów obliczeniowych do trenowania oraz stosunkowo dużej liczebności danych treningowych.

Dalsze wersje tej metody, takie jak Fast R-CNN, Faster R-CNN i Mask R-CNN, skupiały się na poprawie szybkości i efektywności, eliminując konieczność stosowania zewnętrznych algorytmów generowania propozycji regionów oraz integrując klasyfikację i regresję w jednym modelu.

YOLO (ang. *You Only Look Once*) to innowacyjny algorytm do detekcji obiektów, który zyskał popularność dzięki swojej zdolności do przetwarzania obrazów w czasie rzeczywistym. Został wprowadzony przez Josepha Redmona i jego zespół oraz opublikowany w artykule [35]. Jego główną zaletą jest podejście do wykrywania obiektów jako do problemu regresji, co odróżnia go od innych metod, takich jak R-CNN, które polegają na wieloetapowym procesie analizy. W YOLO cały obraz jest przetwarzany jednorazowo przez sieć neuronową, co pozwala na bezpośrednie przewidywanie klas obiektów oraz ich lokalizacji w postaci pól ograniczających. Obraz wejściowy jest dzielony na siatkę kwadratowych komórek, z których każda jest odpowiedzialna za wykrywanie obiektów w swoich granicach. Dla każdej komórki algorytm przewiduje kilka pól ograniczających i przypisuje im poziomy zaufania, które wskazują, na ile prawdopodobne jest, że dana przewidywana ramka zawiera obiekt. Przewidywane pola ograniczające są następnie przetwarzane przy użyciu metody ang. *Non-Maximum Suppression* (NMS) w celu eliminacji redundantnych detekcji, co pomaga zredukować nakładanie się i podwojenie wykrytych obiektów. Mimo swojej wydajności, YOLO posiada kilka ograniczeń. Jego główną wadą jest niższa dokładność w porównaniu z innymi, wolniejszymi metodami, które oferują bardziej precyzyjne detekcje. Ponadto, YOLO może mieć trudności z dokładnym lokalizowaniem małych obiektów i obiektów w gęsto zaludnionych scenach, gdzie obiekty często się nakładają. z drugiej strony, sieć YOLO jest cały czas rozwijana. Obecna 8 wersja sieci cechuje się zupełnie inną architekturą i wydajnością od swoich wersji poprzednich [40]. Zalety YOLO obejmują bardzo szybkie tempo przetwarzania, które pozwala na działanie nawet na sprzęcie konsumenckim z wysoką wydajnością. Jego struktura modelu jest również prostsza, co ułatwia optymalizację i implementację. YOLO jest więc często wybierane w zastosowaniach, gdzie szybkość jest kluczowym czynnikiem, takich jak monitorowanie w czasie rzeczywistym czy systemy autonomiczne.

SSD (ang. *Single Shot multibox Detector*), to kolejny popularny algorytm do detekcji obiektów, który, podobnie jak YOLO, wykonuje detekcję w jednym kroku, ale z pewnymi kluczowymi różnicami, które poprawiają jego zdolność do wykrywania obiektów o różnych rozmiarach. Został opracowany i opublikowany przez Wei Liu i współpracowników w 2016 roku w artykule [36].
Algorytm SSD rozpoczyna od podstawowego obrazu wejściowego, który przetwarza obraz za pomocą głębokiej sieci konwolucyjnej. Ta sieć służy do wyodrębniania map cech na różnych poziomach głębokości i rozdzielczości. Dzięki temu SSD jest w stanie efektywnie wykrywać obiekty o różnych rozmiarach, co jest jedną z jego największych zalet. Po wyodrębnieniu cech, SSD stosuje zestaw sześciokątnych (lub innego kształtu) filtrów na każdą z map cech w celu przewidzenia klas i lokalizacji

pól ograniczających dla obiektów. Każda lokalizacja na mapie cech może sugerować obiekty na różnych skalach i proporcjach, co jest zdefiniowane przez zestaw tzw. ang. *anchor boxes*.
SSD stosuje również technikę nazywaną po angielsku *hard negative mining* do równoważenia klasyfikacji między obiektami a tłem. Metoda ta polega na wybieraniu próbek, które zostały błędnie sklasyfikowane jako obiekty z wysoką pewnością, co pozwala na skuteczniejsze uczenie i lepszą generalizację.

R-FCN (ang. *Region-based Fully Convolutional Network*)**,** to algorytm detekcji obiektów oparty na sieciach neuronowych, który został zaprojektowany, aby efektywnie wykorzystać zalety sieci konwolucyjnych przy jednoczesnym zachowaniu wysokiej precyzji lokalizacji obiektów. Został opracowany i opublikowany przez Jifenga Daia i współpracowników w 2016 roku jako metoda, która skupia się na zwiększeniu wydajności i prędkości detekcji obiektów, szczególnie w kontekście zastosowań w czasie rzeczywistym [37] .
R-FCN różni się od innych popularnych algorytmów detekcji, takich jak Faster R-CNN, tym, że jest w pełni konwolucyjną siecią, co oznacza, że wszystkie warstwy, od wejściowych po wyjściowe, są warstwami konwolucyjnymi. Dzięki temu osiąga dużą efektywność obliczeniową, eliminując potrzebę stosowania w pełni połączonych warstw, które są typowe dla wielu innych modeli detekcji. Dzięki wykorzystaniu w pełni konwolucyjnych warstw i unikaniu redundantnych obliczeń, R-FCN jest szybszy w porównaniu z metodami, które używają oddzielnych sieci dla każdej propozycji regionu. Warstwy pozycyjnie wrażliwe pozwalają na dokładne określenie granic obiektów. Podobnie jak inne algorytmy oparte na głębokich sieciach konwolucyjnych, R-FCN może mieć trudności z wykrywaniem bardzo małych obiektów. R-FCN stanowi wartościową opcję wśród algorytmów detekcji obiektów, oferując dobre połączenie prędkości i precyzji, co sprawia, że jest chętnie wykorzystywany w aplikacjach, gdzie obie te cechy są kluczowe.

2.4 Systemy Identyfikacji Obciążeń

2.4.1 Wprowadzenie

Zgodnie ze schematem przedstawionym na rys. 2.2, drugim elementem systemu B-WIM jest System Identyfikacji Obciążeń (SIO), który uzyskuje informacje z Systemu Identyfikacji Pojazdu (SIP) o pojawieniu się na obiekcie mostowym pojazdu oraz podstawowe parametry jego ruchu. Dodatkowo SIO otrzymuje sygnał odpowiedzi konstrukcji mostowej zmierzonej czujnikami. Zadaniem SIO jest rozwiązanie zagadnienia odwrotnego do analizy statycznej układu, czyli znana jest odpowiedź konstrukcji, a wielkościami poszukiwanymi są wartości sił wymuszających.
Algorytmy rozwiązujące ten problem można podzielić na 2 główne rodzaje:

1. **Algorytmy quasi-statyczne:** Algorytmy zakładające, że chwilowo wartość nacisków na oś może się wahać, ale w dłuższym okresie, wartość średnia będzie odpowiadać wartości

statycznej. Te algorytmy najczęściej bazują na funkcji wpływu wybranej wielkości statycznej lub funkcjach pokrewnych.

2. **Algorytmy dynamiczne**: Algorytmy, które poprzez wykorzystanie równań ruchu dla ciągłych lub dyskretnych modeli matematycznych odwzorowują oraz wyznaczają siły wymuszające.

2.4.2 Algorytmy quasi-statyczne

Reprezentantem pierwszej kategorii jest algorytm Moses'a [14], który został stworzony dla belkowo-płytowych obiektów mostowych. Dla tego typu konstrukcji zmierzony, całkowity moment zginający M^m_k w przęśle wywołany przejazdem pojazdu dla kroku czasowego k może być wyrażony jako suma indywidualnych momentów zginających w każdym dźwigarze, określona wzorem:

$$M_k^m = \sum_i^G EW_i \varepsilon_i, \tag{2.1}$$

gdzie:

- G – sumaryczna liczba dźwigarów,
- E – moduł sprężystości,
- W_i – wskaźnik wytrzymałości na zginanie przekroju i-tego dźwigara,
- ε_i – odkształcenia pomierzone na i-tym dźwigarze.

Teoretyczny moment zginający M^t_k w wybranym kroku czasowym k można uzyskać za pomocą linii wpływu wg wzorów:

$$M_k^t = \sum_{j=1}^N A_j I_{j,(k-C_j)} \tag{2.2}$$

$$C_j = \frac{D_j f}{v} \tag{2.3}$$

gdzie:

- N – liczba osi pojazdu,
- A_j – nacisk j-tej osi,
- $I_{j,(k-C_j)}$ – wartość rzędnej funkcji wpływu momentu zginającego w miejscu j-tej osi pojazdu,
- D_j – odległość między pierwszą a j-tą osią,
- C_j – liczba wykonanych pomiarów (skanów),
- f – częstotliwość próbkowania wartości,
- v – prędkość przejazdu.

Cała metoda sprowadza się do określenia wartości niewiadomych A_j np. poprzez minimalizację funkcji błędu E zdefiniowanej jako błąd kwadratowy pomiędzy wartościami uzyskanymi ze wzoru (2.2) i (2.3):

$$E = \sum_{k=1}^{T} (M_k^t - M_k^m)^2 = \sum_{k=1}^{T} \left(\sum_{i=1}^{N} A_j I_{j,(k-c_j)} - M_k^m \right)^2 \quad (2.4)$$

Podejście Moses'a jest ideowo najprostsze jednak zawiera w sobie kilka znaczących wad. Przede wszystkim nie jest uwzględniona dynamika zjawiska, rozdział poprzeczny obciążenia oraz zagadnienie dobrego uwarunkowania układu równań, które polega na znalezieniu takiego zestawu równań $Ax = b$, który ma stabilne rozwiązanie i nie jest wrażliwy na małe zmiany w macierzy A lub wektorze b.

Powstało wiele podejść próbujących rozwiązać wspomniane problemy. W artykule [21] zaproponowano rozwiązanie 2D bazujące na powierzchni wpływu odkształceń. Z jednej strony metoda znacząco podnosi dokładność rozwiązania poprzez uwzględnienie rozdziału poprzecznego. z drugiej strony wymaga dokładnego modelu MES obiektu oraz skomplikowanej jego kalibracji w celu uzyskania poprawnych wartości powierzchni wpływu.

Kolejny pomysł rozwoju algorytmu Moses'a został zaproponowany w artykule [41] poprzez podzielenie sensorów w grupy przyporządkowane do każdego pasa ruchu. Zamiast wyznaczać sumę efektów pomierzonych we wszystkich dźwigarach obiektu, wartości te dodawane są do siebie w obrębie swoich grup. Dzięki temu uzyskuje się dodatkowe informacje, na podstawie których uściślany jest rozdział poprzeczny obciążenia.

W artykule [42] przedstawiono zmodyfikowany algorytm dwuwymiarowy (2D) z wykorzystaniem koncepcji Moses'a, gdzie do uzyskania linii wpływu wykorzystano odkształcenia pochodzące od pojazdów kalibracyjnych o znanych naciskach oraz rozstawach osi. Biorąc pod uwagę poprzeczny rozdział obciążeń wynikający z przestrzennego zachowania przęsła, możliwe jest określanie linii wpływu dla każdego z dźwigarów bez wymagania, aby dźwigary te posiadały identyczne właściwości materiałowe i geometryczne.

Ostatni problem metod statycznych związany z wrażliwością rozwiązania na uwarunkowanie układu równań może być rozwiązany przez wykorzystanie np. metody regularyzacji Tichonowa, która sprowadza się do dodania członu regularyzacyjnego. Zastosowanie tej techniki daje znaczącą poprawę stabilności i poprawności rozwiązania zagadnienia [43].
Inne podejście do metody wyznaczania masy pojazdu zaprezentowali Ojio i Yamada [44] wykazując, masę pojazdu można przedstawić jako iloczyn (2.5):

$$MC = A \frac{MC_{kal}}{A_{kal}} \quad (2.5)$$

gdzie:

- MC – nieznana masa pojazdu,

- MC_{kal} – znana masa pojazdu kalibracyjnego,
- A_{kal} – pole powierzchni pod krzywą odpowiedzi dla pojazdu kalibracyjnego,
- A – pole powierzchni pod krzywą odpowiedzi dla nieznanego pojazdu.

Metody wykorzystujące linie lub powierzchnie wpływu bardzo dobrze sprawdzają się w określaniu masy całkowitej pojazdu. Wyniki dotyczące nacisków poszczególnych osi są obarczone jednak bardzo dużym błędem.

2.4.3 Algorytmy dynamiczne

Drugim, obok algorytmów quasi-statycznych, możliwym podejściem do wyznaczania nacisków są algorytmy dynamiczne, znane jako Moving Force Identification (MFI). Na podstawie pełnej historii pomierzonych wielkości otrzymanych w trakcie przejazdu obciążenia pozwalają dokładnie określać wartości nacisku na poszczególne osie. Metody MFI dzielą się na metody rozwiązujące zagadnienie w dziedzinie czasu oraz w dziedzinie częstotliwości.

W pracy wyszczególniono podstawowe metody:

- Interpretive Method I (IM I) [45],
- Interpretive Method II (IM II) [46],
- Time Domain Method (TDM) [47],
- Frequency-Time Domain Method (FTDM) [48].

Pierwsza metoda MFI – Interpretive Method I (IM I) [45] – została zaproponowana w 1988 r., gdzie belkowy obiekt mostowy jest odwzorowany jako układ skupionych mas, połączonych nieważkim sprężystym prętem. Proces identyfikacji sił wymuszających jest zagadnieniem odwrotnym, polegającym na analizie zjawiska opisanego wzorami (2.6) oraz (2.7):

$$\{V\} = [V_A]\{P\} - [V_I][\Delta m]\{\ddot{V}\} - [V_I][C]\dot{V} \quad (2.6)$$

$$\{M\} = [M_A]\{P\} - [M_I][\Delta m]\{\ddot{V}\} - [M_I][C]\{\dot{V}\} \quad (2.7)$$

gdzie:

- $\{V\}, \{\dot{V}\}, \{\ddot{V}\}$ – wektory przemieszczeń, prędkości i przyspieszeń punktów masowych,
- $\{M\}$ – wektor momentów zginających,
- $\{P\}$ – wektor nacisków na osie,
- $[\Delta m]$ – diagonalna macierz mas skupionych,
- $[C]$ – macierz tłumienia,
- $[V_A]$ oraz $[M_A]$ – macierze przemieszczeń węzłów oraz momentów zginających, gdzie każda *i*-ta kolumna prezentuje odpowiedź wywołaną obciążeniem działającym od *i*-tego koła,

- $[V_I]$ oraz $[M_I]$ – macierze sztywności.

Jeśli założymy, że $\{V\}$ oraz $\{M\}$ są znane, to jednocześnie wyznaczyć możemy pochodne wartości $\{V\}$. W takim wypadku równania (2.6) i (2.7) stają się układem równań liniowych, z których wyznaczyć można wektor nacisków $\{P\}$, chociażby za pomocą metody najmniejszych kwadratów. W przeciwieństwie do algorytmu Moses'a w równaniach uwzględnione są przyspieszenia oraz prędkości poszczególnych punktów masowych.

W artykule [46] została zaprezentowana druga metoda z rodziny Moving Force Identification – Interptetive Method II (IM II), która wykorzystuje teorię belki Eulera opisaną wzorem:

$$\mu\frac{\partial^2 v(x,t)}{\partial t^2} + 2\mu\omega_b\frac{\partial v(x,t)}{\partial t} + EI\frac{\partial^4 v(x,t)}{\partial x^4} = \delta(x-ct)P(t) \qquad (2.8)$$

gdzie:

- x – współrzędna analizowanego punktu,
- t – czas,
- $v\,(x,\,t)$ – pionowe przemieszczenie punktu x w chwili czasowej t,
- EI – sztywność belki wyrażona jako iloczyn modułu Younga oraz momentu bezwładności,
- μ –równomiernie rozłożona masa własna pręta,
- ω_b – częstotliwość kołowa tłumienia belki,
- $\delta\,(x - ct)$ – funkcja Dirack'a,
- $P(t)$ – funkcja wymuszająca,
- l – rozpiętość teoretyczna belki,
- c – prędkość przemieszczania się siły $P(t)$.

Założono warunki brzegowe oraz warunki początkowe funkcji (2.8) jako:

$$v(0,t)\ oraz\ v(l,t) = 0 \qquad \left.\frac{\partial^2 v(x,t)}{\partial x^2}\right|_{x=0} = 0 \qquad \left.\frac{\partial^2 v(x,t)}{\partial x^2}\right|_{x=l} = 0$$

$$v(x,0) = 0 \qquad \left.\frac{\partial v(x,t)}{\partial t}\right|_{t=0} = 0$$

Jeśli przyjmiemy, że i-ta modalna forma belki jest wyrażona jako $\sin(\frac{i\pi x}{l})$, wtedy rozwiązanie wyrażenia (2.8) przyjmie formę:

$$v = \sum_{i=1}^{\infty} \sin\frac{i\pi x}{L} V_i(t) \qquad (2.9)$$

gdzie:

- $V_i(t)$ – modalne przemieszczenia.

Po podstawieniu równania (2.9) do (2.8), równanie (2.8) jest mnożone przez funkcję kształtu modalnego dla j-tej formy $\sin(\frac{j\pi x}{l})$(j=1,2…). Rozwiązanie jest całkowane po x w przedziale 0 do l. Wykorzystując własność funkcji Diracka uzyskane zostanie rozwiązanie w postaci (2.10):

$$\ddot{V}_j(t) + 2\omega_b\dot{V}_j(t) + \omega_j^2 V_j(t) = \frac{2P(t)}{\mu l}\sin\frac{j\pi ct}{l} \ (\mathrm{j} = 1,2,\dots,\infty) \qquad (2.10)$$

gdzie:

- $V_j(t), \dot{V}_j(t), \ddot{V}_j(t)$ – funkcja przemieszczenia pręta dla *j*-tej postaci oraz jej pochodne.

Zakładając, że belka obciążona jest *k* ruchomymi siłami, można stworzyć analogiczny układ wielu równań (2.10), gdzie niewiadomymi są szukane siły wymuszające P_k.

$$\begin{Bmatrix}\ddot{V}_1\\ \ddot{V}_2\\ \vdots\\ \ddot{V}_j\end{Bmatrix} + \begin{Bmatrix}2\omega_b\dot{V}_1\\ 2\omega_b\dot{V}_2\\ \vdots\\ 2\omega_b\dot{V}_j\end{Bmatrix} + \begin{Bmatrix}\omega_1^2 V_1\\ \omega_2^2 V_2\\ \vdots\\ \omega_j^2 V_j\end{Bmatrix} =$$

$$= \frac{2}{\mu L}\begin{bmatrix}\sin\frac{\pi(ct-\hat{x}_1)}{L} & \sin\frac{\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{\pi(ct-\hat{x}_k)}{L}\\ \sin\frac{2\pi(ct-\hat{x}_1)}{L} & \sin\frac{2\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{2\pi(ct-\hat{x}_k)}{L}\\ \vdots & \vdots & \vdots & \vdots\\ \sin\frac{j\pi(ct-\hat{x}_1)}{L} & \sin\frac{j\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{j\pi(ct-\hat{x}_k)}{L}\end{bmatrix}\begin{Bmatrix}P_1\\ P_2\\ \vdots\\ P_k\end{Bmatrix} \qquad (2.11)$$

gdzie:

- $\hat{x}_k$ – dystans pomiędzy k-tą siłą, a siłą pierwszą. Przy takim założeniu $\hat{x}_1 = 0$.

Niewiadome wartości sił można wyznaczyć rozwiązując układ, wykorzystując np. metodę najmniejszych kwadratów.

Time Domain Method (TDM) zaproponowana w pracy [47], bazuje na teorii belki Eulera i wykorzystuje do identyfikacji pojazdu modalną superpozycję oraz całkę splotu. Wychodząc z rozwiązania (2.10) i przyjmując, że $\omega_j^2 = \frac{j^4\pi^4}{L^4}\frac{EI}{\mu}$ – częstotliwość własna dla *j*-tej formy, oraz $2\omega_b = 2\varepsilon_j\omega_j$, równanie (2.10) można przedstawić w postaci:

$$\ddot{V}_j(t) + 2\varepsilon_j\omega_j\dot{V}_j(t) + \omega_j^2 q_j(t) = \frac{2P(t)}{\mu l}\sin\frac{j\pi ct}{l} \ (\mathrm{j} = 1,2,\dots,\infty) \qquad (2.12)$$

Powyższe równanie można rozwiązać w domenie czasu wykorzystując całkę splotu:

$$V_j(t) = \frac{2}{\mu L}\int_0^t h_j(t-\tau)p_j(\tau)d\tau \qquad (2.13)$$

gdzie:

$$h_j(t) = \left(\frac{1}{\omega_j'}\right) e^{-\xi_j \omega_j t} \sin\left(\omega_j' t\right), t \geq 0 \quad (2.14)$$

$$\omega_j' = \omega_j \sqrt{1 - \xi_j^2} \quad (2.15)$$

$$p_j(\tau) = P(\tau) \sin \frac{j\pi c\tau}{L} \quad (2.16)$$

Podstawiając równanie (2.13) do (2.9), wartość pionowego przemieszczenia belki w punkcie x w czasie t można wyrazić jako:

$$v(x,t) = \sum_{\mathrm{j}=1}^{\infty} \frac{2}{\mu L \omega_j'} \sin\frac{j\pi x}{L} \int_0^t e^{-\xi_j \omega_j (t-\tau)} \sin \omega_j'(t-\tau) \sin\frac{j\pi c\tau}{L} P(\tau) d\tau \quad (2.17)$$

Uwzględniając, że moment zginający wyraża się jako: $M(x,t) = -EI \frac{\partial^2 v(x,t)}{\partial x^2}$

$$M(x,t) = \frac{2EI\pi^2}{\mu L^3} \sum_{\mathrm{j}=1}^{\infty} \frac{j^2}{\omega_j'} \sin\frac{j\pi x}{L} \int_0^t e^{-\xi_j \omega_j (t-\tau)} \sin \omega_j'(t-\tau) \sin\frac{j\pi c\tau}{L} P(\tau) d\tau \quad (2.18)$$

zakładając dyskretną formę czasu, czyli założenie stałych kroków czasowych Δt, po przekształceniu równania (2.17) na formę dyskretną, moment zginający dla i-tego kroku czasowego możemy zapisać:

$$M(x,i) = \frac{2EI\pi^2}{\mu L^3} \sum_{j=1}^{\infty} \frac{j^2}{\omega_j'} \sin\frac{j\pi x}{L} \sum_{t_0=0}^{i} e^{-\xi_j \omega_j \Delta t (i-t_0)} \sin \omega_j' \Delta t (i - t_0) \sin\frac{j\pi c \Delta t t_0}{L} P(t_0) \Delta t, \; \mathrm{i}=0,1,2\ldots,\mathrm{N} \quad (2.19)$$

gdzie:

- Δt - odstęp czasowy między kolejnymi krokami czasowymi,
- t_0 - indeks kroku czasowego w przeszłości,
- i - bieżący krok czasowy,
- N+1 - liczba kroków czasowych, dla których znany jest moment.

Zakładając:

$$C_{xj} = \frac{2EI\pi^2}{\mu L^3} \frac{j^2}{\omega_j'} \sin\frac{j\pi x}{L} \Delta \mathrm{t}$$

$$E_j^k = e^{-\xi_j \omega_j \Delta \mathrm{tk}},$$

$$S_1(k) = \sin\left(\omega_j' \Delta \mathrm{tk}\right),$$

$$S_2(k) = \sin\left(\frac{j\pi c \Delta t k}{L}\right),$$

Da się ustawić równanie (2.19) w formę macierzową (2.20):

$$\begin{Bmatrix} M(0) \\ M(1) \\ M(2) \\ \vdots \\ M(N) \end{Bmatrix} = \sum_{j=0}^{\infty} C_{xj} \times \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & E_j^1 S_1(1) S_2(1) & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & E_j^{N-1} S_1(N-1) S_2(1) & E_j^{N-2} S_1(N-2) S_2(1) & \cdots & E_j^{N-N_B} S_1(N-N_B) S_2(N_B) \end{bmatrix} \times \begin{Bmatrix} P(0) \\ P(1) \\ P(2) \\ \vdots \\ P(N_B) \end{Bmatrix}, \quad (2.20)$$

gdzie:

- $N_B = \frac{l}{c\Delta t}$.

Zakładając, że siła $f(0) = 0$ oraz $f(N_B) = 0$, macierz da się zapisać w uproszczeniu jako

$$\underset{(N-1)\times(N_B-1)}{B} \; \underset{(N-1)\times 1}{P} = \underset{(N-1)\times 1}{m} \quad (2.21)$$

W zależności od wymiarów macierzy B oraz liczby kroków czasowych w stosunku do nieznanych sił, można wyznaczyć bezpośrednio z równania wektor sił wykorzystując metodę najmniejszych kwadratów.

Frequency-Time Domain Method (FTDM) została zaproponowana w artykule [48]. Wychodząc z równania (2.12), poprzez przeprowadzenie transformaty Fouriera (2.24), można wyrazić rozwiązanie przemieszczenia pionowego $v(x,t)$ w postaci (2.22):

$$V(x,\omega) = \sum_{\mathrm{j}=1}^{\infty} \frac{2}{\mu L} \sin\left(\frac{\mathrm{j}\pi \mathrm{ct}}{\mathrm{L}}\right) H_j(\omega) P_j(\omega) \quad (2.22)$$

gdzie:

- $H_j(\omega)$ – funkcja częstotliwości odpowiedzi dla *j*-tej formy.

$$H_j(\omega) = \frac{1}{\omega_j^2 - \omega^2 + i2\xi_j\omega_j\omega} \quad (2.23)$$

$$P_j(\omega) = \int_{-\infty}^{\infty} P(t) e^{-i\omega t} dt \quad (2.24)$$

Rzeczywiste, jak i urojone części $P_j(\omega)$ można uzyskać, rozwiązując układ równań w dziedzinie częstotliwości. Na tej podstawie historia nacisku osi $P(t)$, może zostać wyznaczona poprzez odwrotną transformatę Fourier'a.

Wszystkie metody bazujące na podejściu dynamicznym przy szczególnych założeniach i po pewnych przekształceniach sprowadzają się do rozwiązania układu równań liniowych w postaci ogólnej $Ax = B$. Ponadto podjęto wiele wysiłków, aby poprawić dokładność i rozszerzyć zastosowanie istniejących algorytmów MFI. Zhu i Law rozszerzyli metodę IM II na most ciągły modelowany jako wieloprzęsłowa belka Timoshenki o niejednorodnym przekroju [49]. Celem było rozwinięcie metody dla mostów

wieloprzęsłowych, uwzględniając zmienne właściwości dynamiczne i wpływy, które wcześniej były pomijane w klasycznych modelach opartych na belkach Eulera-Bernoulliego. Zastosowano zasadę Hamiltona do analizy wibracji wieloprzęsłowej belki Timoshenko poddanej działaniu poruszających się sił. Identyfikacja sił została opracowana w dziedzinie czasu przy użyciu superpozycji modalnej oraz technik optymalizacyjnych. Zastosowano metodę najmniejszych kwadratów z tłumieniem (ang. *Damped Least Squares*, DLS), aby poprawić stabilność wyników. Wykorzystano symulacje komputerowe oraz testy laboratoryjne do oceny skuteczności metody w identyfikacji sił. W symulacjach błędy wynosiły od 9.92% do 14.04% w przypadku identyfikacji jednej siły. Dla dwóch sił błędy oszacowania wynosiły od ~25% przy zastosowaniu metody DLS oraz nawet 114% przy zastosowaniu tradycyjnej metody najmniejszych kwadratów (ang. *Least Squares,* LS).

Chan i współautorzy m.in. w publikacjach [50],[51] zastosowali IM I i TDM w identyfikacji obciążeń na betonowych mostach sprężonych. Celem pierwszego artykułu było opracowanie i ocena metody identyfikacji sił osiowych z uwzględnieniem efektów sprężania. Na podstawie analiz ustalono, że efekty sprężania muszą być uwzględniane w procesie identyfikacji i kalibracji. Ich pominięcie prowadzi do znaczących błędów w wynikach. Dodatkowo zauważono, że szumy pomiarowe znacząco wpływają na dokładność identyfikacji sił. Na poziomie 1% powodowały błędy sięgające 181 % dla oszacowania siły stałej oraz 330 % dla oszacowania siły zmiennej. Zastosowanie filtra low-pass znacząco zmniejszyło błędy identyfikacji. W praktycznych zastosowaniach metoda wymaga precyzyjnych danych wejściowych, wysokiej jakości kalibracji oraz filtracji danych pomiarowych, aby osiągnąć dokładne i stabilne wyniki identyfikacji sił. Celem drugiego artykułu [51] było walidowanie metody identyfikacji sił osiowych przy zastosowaniu istniejącego mostu sprężonego w warunkach rzeczywistych. Zgodnie z treścią artykułu (p. 9.3.2 [51]), nie były dostępne informacje dotyczące obciążeń osi pojazdów rzeczywistych i nie przeprowadzono porównań z rzeczywistymi obciążeniami osi. Dokładność dynamicznych obciążeń osi badano wyłącznie na podstawie błędów odtworzenia. W artykule wykazano tylko 5 przejazdów pojazdu kalibracyjnego, dla których tylko 2 były udane i na ich podstawie wykazywano zgodność oszacowań.

Zhu i Law zaproponowali wykorzystanie metody TDM dla obiektu mostowego, który był modelowany jako płyta ortotropowa [52]. Wykorzystano sprzężony modelu most-pojazd, uwzględniający właściwości modalne mostu i fizyczne parametry pojazdu. Dynamiczne obciążenia osi identyfikowane są poprzez analizę odkształceń i ugięć mostu, z użyciem metody najmniejszych kwadratów i regularyzacji Tikhonova. Metoda jest dokładna i odporna na czynniki, takie jak prędkość pojazdu czy trasa, ale wymaga uwzględnienia sił inercji mostu. Przy założeniu 1% poziomu szumu, dla teoretycznego modelu, dokładność oszacowania wyniosła odpowiednio 2.72% oraz 3.14% dla pierwszej i drugiej osi.

Asnachinda i współautorzy przedstawili metodę identyfikacji wielu pojazdów na moście ciągłym uwzględniając metodę elementów skończonych, w której most jest modelowany jako ciągła belka Eulera-Bernoulliego [53]. Celem artykułu było opracowanie i ocena metody identyfikacji dynamicznych obciążeń osi pojazdów na mostach wieloprzęsłowych. Zastosowano podejście optymalizacyjne oparte na metodzie najmniejszych kwadratów z regularyzacją, aby poprawić dokładność identyfikacji w warunkach symulacyjnych i eksperymentalnych. Dla idealnych warunków i odpowiedniego parametru regularyzacji błędy identyfikacji dynamicznych obciążeń wynosiły poniżej 10-13%.

Dowling zastosował metodę optymalizacji z wykorzystaniem krzyżowej entropii, do kalibracji algorytmów identyfikacji sił dynamicznych osi pojazdów [54]. Celem było wyeliminowanie błędów związanych z niedokładnymi modelami MES mostów, poprzez dostosowanie macierzy sztywności i masy na podstawie danych z pomiarów odkształceń. Wykorzystano regularyzację Tikhonova i metodę krzywej L do stabilizacji wyników. Kalibracja algorytmu MFI za pomocą krzyżowej entropii skutecznie zmniejszyła błędy w przewidywaniu sił dynamicznych osi, osiągając różnice poniżej 1% w modelach z niewielkim szumem (≤ 5%). Proponowane podejście umożliwia pełną automatyzację kalibracji systemów WIM, co czyni je bardziej praktycznymi w rzeczywistych zastosowaniach

Law i współautorzy przedstawili metodę MFI opartą na MES i ulepszonej technice kondensacji systemu polegającej na zmniejszeniu liczby stopni swobody, przy jednoczesnym zachowaniu najważniejszych właściwości dynamicznych [55]. Most został zamodelowany jako dyskretna belka prostoliniowa. Wykorzystano równania ruchu systemu pojazd-most, aby odtworzyć siły dynamiczne na podstawie odkształceń zmierzonych w mostach. Regularyzacja Tikhonova i metoda krzywej L były stosowane do rozwiązywania źle uwarunkowanego układu równań. W artykule przeanalizowano dyskretyzacji konstrukcji i częstotliwości próbkowania, liczby czujników i poziomu szumu oraz prędkości pojazdu i nierówności nawierzchni drogi. Wszelkie analizy wykonano na modelu matematycznym. Symulacje wykazały wysoką zgodność między wprowadzonymi i zidentyfikowanymi obciążeniami osiowymi.

Wielu badaczy zauważyło, że metody wyznaczania wartości sił wymuszających są wrażliwe na szum pomiarowy i wykazują duże fluktuacje, dlatego konieczne jest zastosowanie regularyzacji. W badaniach przyjmuj się więc metodę regularyzacji Tichonowa, która jest skuteczna w zmniejszaniu wpływu szumów na dokładność identyfikacji. (m.in. [52], [55], [56], [57], [58], [59]). Pomimo że metody MFI mają potencjał do bycia bardzo dokładnymi wciąż istnieje wiele wyzwań w implementacji algorytmów MFI w nowoczesnych komercyjnych systemach B-WIM. Po pierwsze, MFI jest kosztowne obliczeniowo, co może utrudnić identyfikację nacisków osi w czasie rzeczywistym. Ponadto większość badań nad MFI nadal opiera się na uproszczonych modelach mostów.

2.4.4 Algorytmy wsparte uczeniem maszynowym

Najnowszą, szybko rozwijającą się, kategorią algorytmów SIO są algorytmy wykorzystujące technologie sztucznej inteligencji, uczenia maszynowego oraz sztucznych sieci neuronowych (SSN). Podejście do zagadnienia z uwzględnieniem sztucznych sieci neuronowych zostało zaproponowane w [60], gdzie w ramach procedury analizy próbowano wyodrębnić informacje dotyczące nacisków osi pojazdów ciężarowych z danych odkształceń z mostowego systemu identyfikacji obciążeń w dziedzinie czasu. Jako jedną z kilku możliwych technik rozpoznawania wzorców zastosowano gęstą sztuczną sieć neuronową opartą na procesie uczenia nadzorowanego. Celem podstawowym była identyfikację masy całkowitej pojazdów oraz nacisków na osie pojazdu na podstawie danych z pomiarów odkształceń. Dane wejściowe obejmowały odkształcenia belek głównych lub poprzecznic, prędkość pojazdu oraz odległości między osiami. Przeprowadzono badania na dwóch mostach w Korei: sprężonym moście belkowym (Geumdang Bridge) oraz moście wantowym z pomostem stalowo-betonowym (Seohae Bridge). Dane treningowe i testowe uzyskano z systemów WIM oraz z pojazdów testowych o znanych masach. Osiągnięta dokładność dla:

- błędów identyfikacji masy całkowitej pojazdów:
  - most Geumdang: $\pm 10\%$,
  - most Seohae: $\pm 15\%$,
- błędów identyfikacji obciążeń osiowych dla pojazdów testowych:
  - pojedyncze osie: $\pm 15\%$,
  - grupy osi: $\pm 10\%$.

Proponowana metoda okazała się skuteczna w poprawie dokładności systemów B-WIM, szczególnie w długich mostach, gdzie klasyczne metody są ograniczone. Algorytmy wykorzystujące metody uczenia maszynowego oferują większą elastyczność w analizie dynamicznych obciążeń i mogą być łączone z tradycyjnymi technikami w celu poprawy ich efektywności.

Inny algorytm oparty o sztuczne sieci neuronowe został zaprezentowany w artykule o systemie BwimNet [61]. W pracy zaproponowano zmodyfikowaną architekturę autodekodera z dołączoną warstwą odtwarzania sygnału, której zadaniem jest identyfikowanie parametrów poruszających się pojazdów, w tym ich masy, jedynie za pomocą dynamicznej odpowiedzi obiektu mostowego. Wykorzystano nienadzorowaną metodę uczenia maszynowego bazującą na sieciach neuronowych, która wyodrębnia cechy wyższego rzędu z oryginalnych danych. Metoda miała umożliwić precyzyjną identyfikację właściwości pojazdów (np. prędkości, rozstawu osi, obciążeń osiowych) wyłącznie na podstawie dynamicznych odpowiedzi mostu, eliminując potrzebę stosowania dodatkowych czujników. Wykorzystano jednak uczenie nadzorowane jako podstawową formę uczenia sieci neuronowej. Dane treningowe były generowane w symulacjach i obejmowały dynamiczne odpowiedzi mostu, takie jak odkształcenia w środku przęsła, w odpowiedzi na różne klasy pojazdów. Dodano biały szum Gaussowski (5%) do danych, aby uwzględnić rzeczywiste warunki pomiarowe i zwiększyć odporność

modelu na zakłócenia. Nowa metoda eliminuje wady poprzednich systemów B-WIM takich jak konieczność stosowania dodatkowych czujników wykrywających osie pojazdów oraz drogich systemów akwizycji danych. Kolejną zaletą jest to, że nowa metoda przewyższa tradycyjne algorytmy B-WIM pod względem stabilności i niezawodności. Autorzy wykonali obszerne symulacje i badania parametrów w celu oceny wydajności proponowanej metody. Wyniki, przedstawione w tab. 2.3 pokazują, że metoda ta pozwala osiągnąć dobrą dokładność identyfikacji właściwości pojazdów, nawet w sytuacjach, gdy w danych treningowych występują znaczące błędy początkowe.

Tab. 2.3 Względne błędy oszacowania nacisków osi dla dobrych warunków nawierzchni drogowej dla systemu BwimNet wg [61]

Pojazd	Metoda	Błąd względny [%]					
		Oś 1	Oś 2	Oś 3	Oś 4	Oś 5	Masa całkowita
2 osie	Moses	1.72	2.17	-	-	-	0.40
	BwimNet	2.27	3.51	-	-	-	0.61
3 osie	Moses	4.16	3.26	1.96	-	-	0.32
	BwimNet	6.64	6.29	4.56	-	-	1.48
5 osi	Moses	4.70	7.46	48.03	79.04	40.25	0.27
	BwimNet	7.55	5.76	10.62	11.86	5.76	1.30

W artykule [62] przedstawiono wykorzystanie sieci neuronowej do identyfikacji ciężkich pojazdów i obciążeń osiowych na mostach przy użyciu jednego czujnika odkształceń. Głównym celem było obniżenie kosztów instalacji i utrzymania systemów Bridge Weigh-in-Motion (BWIM) oraz zwiększenie ich dokładności w identyfikacji pojazdów w różnych warunkach. Zastosowano głęboką sieć neuronową z warstwami splotowymi. Sieć trenowano na danych z czujnika odkształceń oraz kamer monitorujących ruch. Dane zbierano z 74-metrowego stalowego mostu, używając pojedynczego czujnika odkształceń i dwóch kamer do identyfikacji pojazdów oraz weryfikacji ich właściwości (np. masy). Model uczono za pomocą rzeczywistych danych od 40 613 pojazdów, w tym 5 923 pojazdów o znanych masach osiowych. Model skutecznie identyfikował prędkość pojazdów i odstępy między osiami. Sieć neuronowa osiągnęła lepsze wyniki w identyfikacji obciążeń osiowych niż tradycyjne podejścia wykorzystujące linie wpływu, eliminując konieczność ich dokładnego wyznaczania. Model radził sobie z zakłóceniami związanymi z przyspieszaniem pojazdów i zmiennymi pozycjami w pasach ruchu, co jest trudne dla konwencjonalnych systemów BWIM. Proponowany system oparty na CNN wykazuje dużą elastyczność i dokładność, jednocześnie redukując koszty instalacji i utrzymania. Dzięki użyciu pojedynczego czujnika i zdolności do adaptacji w różnych warunkach, metoda ta może stanowić nowy standard w systemach BWIM, szczególnie na mostach o zróżnicowanych typach konstrukcji i długich przęsłach.

Cechą wspólną przytoczonych artykułów jest wykorzystanie metod opierających się na uczeniu nadzorowanym. Koszt ich stosowania jest jednak duży, ponieważ do poprawnego działania sieci potrzebują ogromnej liczby danych uczących, czyli precyzyjnie zidentyfikowanych przejazdów pojazdów.

W rozprawie postanowiono stworzyć wariant sieci wykorzystujący uczenie nienadzorowane, który nie wymaga tysięcy poprawnie oznaczonych i zważonych pojazdów jako danych treningowych. Stanowi to rozwinięcie wspomnianych metod, ponieważ umożliwia w doskonalenie się systemu, bez ingerencji człowieka.

2.5 System Monitoringu Konstrukcji Mostowej

2.5.1 Monitoring sensoryczny obiektów mostowych

Mostowe systemy identyfikacji obciążeń typu B-WIM to nie tylko algorytmy oraz techniki analizy sygnału, ale również sposób zbierania danych (patrz rys. 2.2). Konieczne jest dobranie odpowiedniej techniki pomiarowej oraz umiejętne jej zastosowanie w celu osiągnięcia oczekiwanych rezultatów.
Systemy monitoringu sensorycznego na obiektach mostowych są kluczowym narzędziem zbierającym informacje o odpowiedzi konstrukcji mostowej na obciążenia, a centralnym składnikiem takich systemów jest zintegrowany system pomiarowy. Jego główne funkcje to nie tylko pomiar fizycznych wielkości obciążenia obiektu mostowego, ale również wstępne przetwarzanie tych danych, ich wizualizacja w czasie rzeczywistym oraz archiwizacja.

Systemy pomiarowe wykorzystywane do monitorowania obiektów mostowych, zgodnie z zaproponowanym podziałem [63], zbudowane są z pięciu zasadniczych grup komponentów, z których każda spełnia kluczową rolę w zapewnieniu ciągłości i dokładności zbieranych danych:

- **Czujniki pomiarowe** stanowią podstawę systemów pomiarowych i odpowiedzialne są za rejestrację zmian różnorodnych wielkości fizycznych. Wśród nich znajdują się czujniki odkształceń, naprężeń, przemieszczeń liniowych i kątowych, przyspieszeń, prędkości drgań, temperatury konstrukcji i otoczenia, prędkości wiatru oraz wartości obciążeń w funkcji czasu.
- **Urządzenia pomiarowe**, których zadaniem jest odczyt, transformacja i udostępnianie sygnału z czujników, służą jako pośrednik między czujnikami a resztą systemu, konwertując surowe dane w formaty przydatne do dalszej analizy.
- **Elementy do transmisji danych** zapewniają efektywną komunikację pomiędzy czujnikami a urządzeniami pomiarowymi oraz dalsze przesyłanie danych do centrum kontrolnego. Te komponenty są kluczowe dla utrzymania spójności i szybkości przepływu informacji w całym systemie monitoringu.
- **Stacja robocza** to komputer dedykowany do zarządzania procesem pomiarów oraz archiwizacji danych. Pełni funkcję centralnego punktu, który nie tylko rejestruje wyniki pomiarów, ale również koordynuje pracę całego systemu.
- **Specjalistyczne oprogramowanie** to zainstalowane na stacji roboczej, oprogramowanie odpowiedzialne za komunikację z urządzeniami pomiarowymi, kontrolę procesu pomiarowego, zapis danych oraz ich przetwarzanie. Dodatkowo, oprogramowanie to może informować użytkowników o potencjalnych zagrożeniach bezpieczeństwa.

Czujnik to urządzenie, które wykrywa i przetwarza analizowaną wielkość fizyczną lub chemiczną na inną formę, zwykle na wielkość elektryczną, taką jak napięcie elektryczne, natężenie prądu lub częstotliwość prądu. Ogólnie rzecz biorąc, czujnik to urządzenie konwertujące jedną formę energii na

inną [64]. W kontekście monitoringu mostów, zazwyczaj mierzone i przetwarzane sygnały są reprezentowane jako różne formy energii:

- **Energia mechaniczna** jest związana z ruchem i zmianą położenia elementów obiektów względem określonego układu odniesienia, stanowi ona sumę energii potencjalnej i kinetycznej systemu.
- **Energia termiczna** odnosi się do energii kinetycznej atomów i molekuł, której miarą jest temperatura materiału.
- **Energia elektryczna** jest związana z polem elektrycznym, natężeniem prądu, czy napięciem.
- **Energia magnetyczna** jest związana z indukcją magnetyczną i natężeniem pola magnetycznego.
- **Energia promieniowania** jest związana z falami elektromagnetycznymi, szczególnie w zakresie radiowym, mikrofalowym, światła widzialnego lub światła spolaryzowanego.

2.5.2 Klasyfikacja technik pomiarowych oraz ich zastosowań

Struktura czujników oraz innych urządzeń pomiarowych to fizyczna realizacja różnorodnych metod pomiarowych, umożliwiająca detekcję i rejestrowanie nawet subtelnych zmian w mierzonych parametrach. Dlatego też różne techniki pomiarowe oraz rodzaje czujników charakteryzują się specyficznymi właściwościami, które wpływają na ich efektywność pomiarową i użyteczność w danych zastosowaniach.

Różne techniki pomiarowe znajdują zastosowanie w zależności od specyfiki sytuacji oraz potrzeb związanych z analizą konkretnych parametrów konstrukcji mostowej. Przyjęta zgodnie z publikacją [63] i zaprezentowana w tab. 2.4 klasyfikacja, określa przydatność wykorzystania różnych technik pomiarowych w kontekście identyfikacji specyficznych parametrów konstrukcji mostowej.

Tab. 2.4 Obszary przydatności wybranych technik i wybranych czujników w identyfikowaniu wybranych parametrów odpowiedzi konstrukcji mostowej [63]

Techniki pomiarowe	Identyfikowane parametry						
	Przemieszczenia liniowe	Przemieszczenia kątowe	Odkształcenia	Prędkości drgań	Przyspieszenia drgań	Siły	Rozwój zarysowań
Elektryczne indukcyjne	●	●		●	●	●	●
Elektryczne pojemnościowe	●	●		●	●	●	
Elektryczne rezystancyjne			●	○	○	●	●
Emisji akustycznej							●
Geodezyjne	●	○					
Hydrauliczne	●						
Laserowe	●			●	●		
Mechaniczno-elektryczne	●		○				●
Mikroelektro-mechaniczne MEMS		●	○	○	●		
Piezoelektryczne			○	●	●	●	
Radarowe	●			●	●		
Rejestracji i przetwarzania obrazu	●	○	○	○	○		
Satelitarne	○						
Światłowodowe	●	●	●	○	●	●	●
Ultradźwiękowe	●						●

Oznaczenia:
● – pełna przydatność rozwiązania technicznego,
○ – ograniczona przydatność rozwiązania technicznego.

Czujniki stosowane w monitorowaniu mostów zwykle konwertują energię mechaniczną, termiczną lub promieniowania na określoną formę energii elektrycznej, na przykład wartość odkształcenia jest transformowana na wartość analogowego napięcia elektrycznego. Sygnały analogowe są preferowane jako wielkości wyjściowe w czujnikach, ponieważ łatwo je przesyłać i przetwarzać przez kolejne elementy systemu pomiarowego. Parametry sygnałów elektrycznych generowanych przez czujniki zazwyczaj są wprost proporcjonalne do mierzonej wielkości w określonym zakresie pomiarowym. W systemie pomiarowym urządzenie odbiera sygnały z czujników za pomocą kanałów pomiarowych. Otrzymany sygnał jest następnie konwertowany do postaci cyfrowej i przesyłany do komputera sterującego pomiarem. Przekształcenie sygnału do postaci cyfrowej pozwala na łatwiejsze przetwarzanie danych, ich analizę i archiwizację, co jest kluczowe w celu zapewnienia precyzyjnego monitoringu stanu technicznego obiektów mostowych.

Projektowanie systemów monitorowania mostów i ocena danych pomiarowych z nich pochodzących zależą od dokładnego rozważenia technicznych specyfikacji aparatury pomiarowej, szczególnie czujników. Wybór odpowiednich urządzeń pomiarowych dla danej aplikacji opiera się na szczegółowej analizie ich kluczowych charakterystyk technicznych, które istotnie wpływają na jakość uzyskanych danych:

- **Zakres pomiarowy** (ang. *measurement range*) jest to zakres, w którym urządzenie pomiarowe może dokonywać pomiarów z błędem nieprzekraczającym ustalonej granicy. Ten przedział jest określony przez specyfikacje metrologiczne deklarowane przez producenta.
- **Dokładność pomiaru** (ang. *measurement accuracy*) definiuje zdolność urządzenia do precyzyjnego odtworzenia rzeczywistych wartości mierzonych. Dokładność określa maksymalny błąd pomiarowy, symbolizowany jako Δx.
- **Niepewność pomiarów** (ang. *measurement uncertainty*) to przedział $\pm\Delta x$ wokół średniej wartości z serii pomiarów, wykonanych w jednakowych warunkach, który z 95-procentowym prawdopodobieństwem zawiera prawdziwą wartość mierzonej cechy. Uwzględnia wpływ wszystkich możliwych źródeł błędów na wyniki pomiarów oraz określa zakres zmienności wyników tych pomiarów.
- **Rozdzielczość urządzenia** (ang. *measurement resolution*) to najmniejsza zmiana mierzonej wielkości, którą jest w stanie zarejestrować czujnik.
- **Czułość urządzenia** (ang. *sensitivity*) to proporcja pomiędzy zmianą sygnału wyjściowego czujnika a zmianą mierzonej wielkości. Jest bezpośrednio związana z rozdzielczością i może być również wyrażona jako pochodna funkcji przejściowej względem zmieniającego się sygnału fizycznego.
- **Stabilność urządzenia** (ang. *stability*) to zdolność przyrządu do utrzymania niezmiennych przez dłuższy czas właściwości metrologicznych.
- **Powtarzalność urządzenia** (ang. *repeatability*) to zdolność do wielokrotnego generowania identycznych sygnałów wyjściowych na odpowiedź tych samych sygnałów wejściowych w niezmienionych warunkach środowiskowych.
- **Pełzanie mierzonej wielkości** (ang. *drift*) to stopniowa zmiana sygnału wyjściowego czujnika, pojawiająca się nawet gdy nie zachodzą żadne zmiany wielkości mierzonej ani warunków środowiskowych. Jest to związane ze zmianami charakterystyk metrologicznych czujnika wynikającymi z różnych czynników, takich jak niestabilność mechaniczna lub elektroniczna, zmiany temperatury, starzenie się elementów, zakłócenia elektryczne, czy zmiany właściwości elektrycznych czy elektromechanicznych komponentów.

2.5.3 Charakterystyka czujników wykorzystywanych w B-WIM

Biorąc pod uwagę przedstawione argumenty oraz omówione parametry, istotne jest postawienie pytania, które czynniki i w jakim zakresie będą miały kluczowe znaczenie dla mostowych systemów typu B-WIM. Należy zwrócić uwagę, że system B-WIM składa się z dwóch zasadniczych podsystemów tj. Systemu Identyfikacji Pojazdów (SIP) oraz Systemu Identyfikacji Obciążeń (SIO), które korzystają z wybranych technik Systemu Monitoringu Konstrukcji Mostowych (SMKM). Każdy z tych podsystemów wymaga odmiennego poziomu dokładności oraz jest dostosowany do specyfiki swojej pracy, co ma istotny wpływ na dobór odpowiednich czujników do efektywnego funkcjonowania.
W systemach typu B-WIM parametry pojazdu, w tym przede wszystkim masa pojazdu, są ustalane poprzez analizę odpowiedzi konstrukcji mostowej na obciążenia użytkowe. Mierzone są takie wielkości jak odkształcenia, przemieszczenia, przyspieszenia i prędkości drgań konkretnych elementów konstrukcyjnych.

Czujniki odkształceń są najczęściej instalowane na dźwigarach głównych, poprzecznicach, podłużnicach, płycie pomostowej oraz elementach cięgnowych. Dodatkowo przeprowadza się pomiary przyspieszeń drgań, które mogą znacznie wpływać na dokładność pomiaru masy pojazdu ze względu na stan i charakterystyki techniczne zawieszenia pojazdu. Czujniki przyspieszeń zazwyczaj umieszcza się na wybranych elementach przęseł (m.in. [24], [53], [65], [66], [67]). Podkreślić warto, że opracowanie efektywnego systemu identyfikacji obciążeń wymaga dostosowania do specyfiki każdego obiektu mostowego, biorąc pod uwagę jego charakterystyki konstrukcyjne, stan techniczny oraz warunki eksploatacji. Czujniki powinny umożliwiać precyzyjne rejestrowanie dynamicznej reakcji badanych elementów na obciążenia pojazdami. Dzięki starannej kalibracji systemu pomiarowego oraz zautomatyzowanej analizie sygnałów, możliwe jest dokładne określenie całkowitej masy pojazdu, a niekiedy także sił nacisku osi poruszającego się pojazdu.

Czujniki należy dobrać nie tylko dla systemu identyfikacji obciążeń, lecz także dla systemów identyfikacji pojazdów. Stawia się im mniejsze wymagania z uwagi na specyfikę pracy. Najbardziej podstawowe systemy identyfikacji pojazdu wykorzystują mechaniczną detekcję, która najczęściej polega na instalacji czujników w nawierzchni drogowej, które rejestrują przejazd osi pojazdu po obiekcie mostowym. Przykładem najprostszych zastosowań są tuby pneumatyczne, w których zmienia się ciśnienie powietrza pod wpływem przejeżdżających osi. Alternatywą są bardziej zaawansowane systemy, takie jak tradycyjne systemy WIM oparte na czujnikach piezoelektrycznych, pojemnościowych, elektrooporowych oraz światłowodowych. Czujniki te mają za zadanie wstępną identyfikację pojazdu, tj. określenie konfiguracji osi, liczby osi oraz jego prędkości. Montaż czujników w nawierzchni drogowej niesie ryzyko ich uszkodzenia z powodu kontaktu z kołami pojazdów, a także zwiększa ryzyko degradacji nawierzchni pod wpływem czynników atmosferycznych ze względu na naruszenie jej struktury (m.in. [11], [68], [69]).

Druga rodzina technik obejmuje systemy oparte na lokalnej odpowiedzi konstrukcji mostowej, gdzie czujniki umieszczane są w specyficznych miejscach, takich jak na środnikach, żebrach płyty pomostowej czy poprzecznicach. Wybór lokalizacji czujników jest kluczowy, aby wychwycić szczytowe impulsy od przejeżdżających osi, co umożliwia charakteryzację liczby osi oraz konfiguracji pojazdu. Niestety, sygnały z tych czujników często charakteryzują się niewielkimi zakresami zmian i są podatne na wpływ zewnętrznych czynników na sygnał, co sprawia, że ich wykorzystanie do dalszej analizy, na przykład określania masy pojazdu, jest ograniczone (m.in.[29], [67]).
Systemy bazujące na globalnej odpowiedzi konstrukcji mostowej, takie jak ang. *Nothing On the Road*, wykorzystują analizę falkową do odfiltrowania i ustalenia liczby osi oraz konfiguracji pojazdu. Te same czujniki mogą również służyć do określania masy pojazdu na podstawie globalnych odkształceń wybranego pasa ruchu. Integracja tych danych z algorytmem identyfikacji obciążeń pozwala na szczegółową analizę obciążeń na most (m.in. [28], [67]).

Kolejną technologią są systemy wizyjne, które polegają na zastosowaniu kamer. Główną zaletą systemów wizyjnych jest ich lokalizacja poza obiektem mostowym oraz stosunkowo niski koszt, przy jednoczesnym dostarczaniu danych o wysokiej jakości i częstotliwości. Technologie wizyjne, wykorzystujące techniki oparte na uczeniu maszynowym, umożliwiają nie tylko lokalizację pojazdu, ale także identyfikację jego konfiguracji osi oraz klasyfikację całego pojazdu [31]. W odróżnieniu od tradycyjnych systemów detekcji mechanicznej, FAD oraz NOR, kamery precyzyjniej określają sylwetkę pojazdu, a ponadto dostarczają ciągłej informacji o lokalizacji pojazdu na obiekcie w czasie ruchu. Kamery są w stanie zaktualizować fakt zmiany prędkości, czy też pasa ruchu pojazdu na obiekcie, co nie jest takie oczywiste dla tradycyjnych systemów. Systemy wizyjne są podatne na wpływy atmosferyczne, takie jak bardzo intensywny deszcz i mgła uniemożliwiająca poprawną klasyfikację pojazdów. Omawiany system jest systemem monitorowania obciążeń eksploatacyjnych i jego krótkotrwałe wyłączenie z uwagi na pogodę, nie wpływa na ruch oraz bezpieczeństwo ruchu. Systemy te wymagają niewielkich nakładów finansowych związanych z montażem kamery, która na bieżąco dostarczałaby wspomnianych informacji do algorytmu.

Tab. 2.5 przedstawia proponowaną klasyfikację użyteczności różnych technik pomiarowych stosowanych w systemach identyfikacji pojazdów oraz w systemach określania nacisków wywieranych przez te pojazdy.

Tab. 2.5 Obszary przydatności wybranych technik i wybranych czujników pomiarowych dla systemów identyfikacji pojazdów oraz systemów identyfikacji obciążeń (*opracowano na podstawie m.in.* [63],[64],[70])

Techniki pomiarowe	Mostowy system identyfikacji obciążeń typu B-WIM					
	System identyfikacji pojazdów				System identyfikacji obciążeń	
	Detekcja mechaniczna	Free Axle Detector (FAD)	Nothing On the Road (NOR)	Systemy wsparte uczeniem maszynowym	Algorytmy statyczne	Algorytmy dynamiczne
Elektryczne indukcyjne		○	○	●	●	○
Elektryczne pojemnościowe	●	○	○			○
Elektryczne rezystancyjne	●	●	●	●	●	●
Hydrauliczne	●					
Laserowe	●				○	○
Mikroelektro-mechaniczne MEMS				●		○
Piezoelektryczne	●	●	●	●	○	●
Radarowe				●		
Rejestracji i przetwarzania obrazu				●		
Światłowodowe	●			●	●	○

Oznaczenia:
● – pełna przydatność rozwiązania technicznego,
○ – ograniczona przydatność rozwiązania technicznego.

W omawianym systemie identyfikacji obciążeń eksploatacyjnych obiektów mostowych, najlepsze wyniki w zakresie identyfikacji pojazdów zapewniają systemy wizyjne, które dostarczają danych o wysokiej jakości i częstotliwości, a także umożliwiają precyzyjną lokalizację i klasyfikację pojazdu. W przypadku systemu identyfikacji obciążeń rekomendowane jest stosowanie czujników mierzących odpowiedź konstrukcji, które charakteryzują się dużą niezawodnością, stabilnością oraz wysoką częstotliwością pomiarów. Warto przy tym zwrócić uwagę na dobór lokalizacji czujników, aby zapewnić wyraźną i mierzalną odpowiedź konstrukcji, unikając sytuacji, w których odczyty mogłyby być zbliżone do rozdzielczości czujnika, co wiązałoby się z dużą niepewnością pomiarów. W kontekście praktycznych zastosowań oraz uniknięcia ograniczenia skrajni pod obiektem, zaleca się stosowanie czujników odkształceń zlokalizowanych na dźwigarze głównym. Zaleca się wykorzystanie światłowodowych czujników wykorzystujących siatę Bragga w światłowodzie (ang. *Fiber Bragg*

Grating, FBG) do długoterminowego monitorowania odpowiedzi konstrukcji mostowej w systemach B-WIM. Ze względu na ich odporność na zakłócenia elektromagnetyczne, dobrą trwałość oraz możliwość łatwego montażu w wielu punktach konstrukcji, FBG stanowią doskonałą alternatywę dla konwencjonalnych tensometrów foliowych. Tensometry foliowe, choć ekonomiczne i odpowiednie do badań krótkoterminowych, nie spełniają wymagań dotyczących trwałości i odporności na zmiany środowiskowe niezbędnych do zastosowań długoterminowych.

2.6 Podsumowanie

2.6.1 Podsumowanie przeglądu Systemów Identyfikacji Pojazdów

Identyfikacja pojazdu oraz śledzenie położenia w trakcie przejazdu pojazdu po obiekcie mostowym jest kluczowym zagadnieniem dla poprawnego działania całego systemu. Dobór techniki identyfikacji pojazdu jest sprawą indywidualną z uwagi na wymagania, jakie stawiamy systemom B-WIM.
W tab. 2.6 podsumowano podstawowe właściwości przedstawionych systemów identyfikacji pojazdu obejmujące kryteria klasyfikacji, stosowane techniki pomiaru, potencjalne wyniki, cechy szczególne oraz odniesienia literaturowe.

Tab. 2.6 Klasyfikacja algorytmów identyfikacji pojazdu systemów B-WIM

Algorytm	Typowa lokalizacja czujników	Wyniki				Cechy		Odniesienia do literatury
		Konfiguracja osi	Prędkość	Podłużna lokalizacja	Poprzeczna lokalizacja	Rodzaj informacji (dyskretny czy ciągły)	Docelowe przeznaczenie	
System detekcji mechanicznej	W nawierzchni przed obiektem	●	●	○		Dyskretny	Krótko-okresowy monitoring na obiektach o niewielkim natężeniu ruchu, dla systemów o niewielkiej dokładności	[14], [15], [17], [21]
Free Axle Detector (FAD)	Nad punktami podparć lub w punkach dających wyraźną odpowiedź	●	●	○	○	Dyskretny	Trwały monitoring na obiektach o średnim natężeniu ruchu, dla systemów o dobrej dokładności	[23], [26], [71]
Nothing On the Road (NOR)	Najczęściej w środku rozpiętości przęsła	●	●	○	○	Dyskretny	Trwały monitoring na obiektach o średnim natężeniu ruchu, dla systemów o dobrej dokładności	[26], [27], [29], [72]
Systemy bazujące na sieciach neuronowych	Każda z powyższych	●	●	●	○	Ciągły	Trwały monitoring na obiektach o średnim/dużym natężeniu ruchu, dla systemów o dobrej i bardzo dobrej dokładności	[60], [61], [62], [73], [74], [75]
Objaśnienie symboli: przydatność (●) – duża, (○) – średnia, ( ) – niska								

Podstawowe informacje jakie powinien dostarczać system identyfikacji pojazdów to:

- konfiguracja osi,
- prędkość,
- podłużna lokalizacja w czasie,
- poprzeczna lokalizacja w czasie.

Do badań krótkookresowych, na niewielkich obiektach o niewielkim natężeniu ruchu możemy wykorzystać system detekcji mechanicznej. System ten niezawodnie wykrywa osie oraz określa ich prędkość pod warunkiem, że pojazdy poruszają się w znacznych odległościach od siebie. Jest on jednak podatny na uszkodzenia mechaniczne i cechuje się niską trwałością. Jeżeli planowany jest monitoring długookresowy na obiekcie o umiarkowanym natężeniu ruchu można wykorzystać systemy FAD albo NOR. Umiejscowienie czujników pod obiektem mostowym sprawia, że są one dużo bardziej trwałe, jednak kosztem konieczności stosowania trudniejszych i mniej pewnych metod interpretacji wyników.

Kolejną kluczową cechą systemów, jest to czy te dane są dostarczane w formie dyskretnej, czy też ciągłej. W przeciwieństwie do starszych algorytmów SIP te, bazujące na kamerach, pozwalają śledzić i aktualizować parametry ruchu w czasie rzeczywistym. Systemy korzystające ze wsparcia sieci neuronowych pozwolą identyfikować oraz śledzić pojazdy nawet przy bardzo dużej intensywności ruchu oraz dostarczając ciągłej informacji o lokalizacji pojazdu.

Dziedzina technik detekcji obiektów oraz pojazdów korzysta z szerokiej gamy metod, różniących się podejściem oraz zastosowaniem. Tradycyjne metody, takie jak kaskady Haar'a, nadal znajdują zastosowanie w określonych zadaniach. Jednakże najnowsze i najbardziej wszechstronne rozwiązania opierają się na technikach uczenia maszynowego, w szczególności głębokich sieciach neuronowych, takich jak R-CNN, YOLO, SSD czy R-FCN. Metody te pozwalają na precyzyjną, szybką i skuteczną identyfikację oraz lokalizację pojazdów nawet w złożonych warunkach, co czyni je szczególnie przydatnymi w kontekście dynamicznego środowiska drogowego i ciągłego monitorowania infrastruktury. Dzięki temu nowoczesne systemy wizyjne oparte o widzenie komputerowe stają się coraz bardziej niezawodne i wydajne, umożliwiając wykrywanie, klasyfikację i śledzenie pojazdów z niespotykaną dotąd precyzją. Z uwagi na trwałość oraz możliwość dostarczania informacji o parametrach przejazdu w czasie rzeczywistym, w systemie zaproponowanym w niniejszej rozprawie postanowiono zastosować wizyjną identyfikację pojazdów przy użyciu kamer.

2.6.2 Podsumowanie Systemów Identyfikacji Obciążeń

Tab. 2.7 przedstawia zebrane podsumowanie właściwości algorytmów identyfikujących naciski pojazdów systemów B-WIM obejmujące kryteria klasyfikacji, stosowane techniki pomiaru, potencjalne wyniki, cechy szczególne oraz odniesienia literaturowe.

Tab. 2.7 Klasyfikacja algorytmów identyfikacji obciążeń systemów B-WIM

Algorytmy		Wyniki			Cechy				Odniesienie do literatury
		Masa całkowita pojazdu	Masa grup osi	Naciski osi	Skuteczność przy wielu pojazdach	Wykorzystywany model reprezentatywny	Łatwość implementacji	Inne	
Oparte na funkcji wpływu	liniowa funkcja wpływu	●	○			Brak	●	Wymaga kalibracji funkcji wpływu	[14], [15], [21], [22], [76]
	powierzchniowa funkcja wpływu	●	○		○	Dyskretny	●	Wymaga kalibracji funkcji wpływu	[21], [56], [77]
Moving Force Identification	w domenie czasu	●	●	●	●	Dyskretny / ciągły		Wymaga dokładnego modelu obliczeniowego oraz jego kalibracji	[45], [46], [47]
	w domenie częstotliwości	●	●	●	●				[28], [48]
Bazujące na sieciach neuronowych		●	●	●	●	Brak	○	Wymaga danych uczących albo określenia funkcji celu	[60], [61]
Objaśnienie symboli: przydatność (●) duża, (○) – średnia, ( ) – niska									

Metody bazujące na koncepcji Moses'a oraz wykorzystujące funkcje wpływu w postaci liniowych lub powierzchniowych funkcji wpływu cieszą się popularnością z uwagi na łatwość implementacji oraz brak konieczności tworzenia modeli reprezentatywnych konstrukcji mostowej.

Te metody dobrze radzą sobie z wyznaczaniem masy całkowitej pojazdu, jednak mają problem z wyznaczeniem nacisków na poszczególne osie oraz grupy osi, z uwagi na nieuwzględnienie dynamicznej natury przejazdu. Dodatkowo metody te wymagają stworzenia oraz aktualizowania rzeczywistej funkcji wpływ mierzonej wielkości.

Metody typu Moving Force Identification (MFI) uwzględniają dynamiczną naturę przejazdu pojazdu po obiekcie mostowym. Opierają się na modelu reprezentatywnym, który może być modelem matematycznym ciągłym lub dyskretnym. Dzięki rozwiązaniu zagadnienia odwrotnego metody te uzyskują dokładniejsze wyniki oszacowań nacisków osi pojazdu. Istotnym wyzwaniem jest jednak implementacja tego typu metod na rzeczywistych obiektach mostowych, które nie są wyidealizowanymi modelami matematycznymi. Dodatkowo obiekty te ulegają różnym procesom degradacji, które są trudne w monitorowaniu oraz utrudniają w znaczącym stopniu aktualizację modelu reprezentacyjnego.

Najbardziej obiecujące w przyszłym zastosowaniu wydają się, bardzo szybko rozwijające się, metody bazujące na uczeniu maszynowym, które wykorzystują mieszane podejście do zagadnienia poprzez wykorzystanie wielu informacji jednocześnie. Najważniejszą cechą jest zdolność tych algorytmów do samodzielnego doskonalenia się i poprawiania swojego działania (możliwości auto-adaptacyjne). Dodatkowo stworzona przez nie logika zapisana w strukturze wewnętrznej sieci jest elastyczna i uwzględnia wiele parametrów jednocześnie. Możliwe jest wykorzystanie zarówno metod opartych na funkcjach wpływu oraz metod uwzględniających dynamiczną naturę zjawisk.
Algorytmy te są jednak bardzo trudne w stworzeniu, z uwagi na konieczność rozważnego zbudowania struktury sieci. W przeciwieństwie do tradycyjnych algorytmów oraz wzorów, przy tworzeniu sieci neuronowych nie da się stworzyć jawnie opisanej i zrozumiałem procedury, w której każdy etap jest możliwy do odtworzenia i przeanalizowania. W przypadku algorytmów uczenia maszynowego, nie ma możliwości bezpośredniej i jawnej kontroli nad procesem. W tych algorytmach zdefiniowany jest przede wszystkim cel, struktura sieci, sposób samodoskonalenia oraz metody weryfikacji. To właśnie stanowi największe wyzwanie, ale również atut uczenia maszynowego, ponieważ poprawnie stworzony algorytm ma możliwość do nieszablonowego podejścia i rozwiązania stawianego mu problemu.

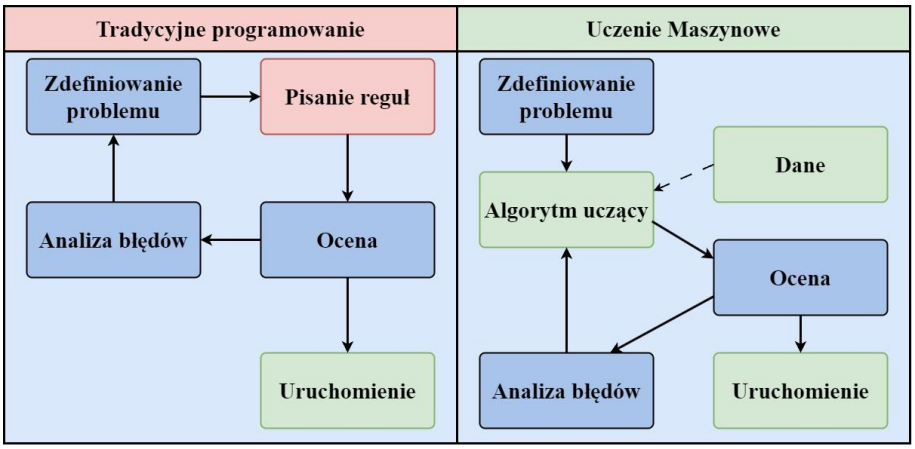
Z uwagi na elastyczność i możliwości samodoskonalenia się algorytmów zawierających metody uczenia maszynowego, w niniejszej rozprawie podjęto się stworzenia systemu wykorzystującego wspomniane techniki sztucznej inteligencji.

3 Systemowa identyfikacja obciążeń z wykorzystaniem uczenia maszynowego

3.1 Uczenie maszynowe – przegląd wiedzy

3.1.1 Przegląd algorytmów uczenia maszynowego

Uczenie maszynowe jest dziedziną informatyki zajmującą się metodami, które pozwalają komputerom na samodzielne uczenie się na podstawie dostępnych danych, bez konieczności tradycyjnego, ręcznego programowania wszystkich reguł. Arthur Samuel w 1959 roku zdefiniował uczenie maszynowe jako zdolność komputera do nauki bez potrzeby bezpośredniego programowania, podkreślając jego zdolność do adaptacji i samodoskonalenia [78]. Tom Mitchell dostarczył bardziej techniczną definicję, wskazując, że *„program komputerowy można uznać za uczący się, gdy jego wydajność w wykonywaniu określonego zadania T, mierzona za pomocą określonej miary wydajności P, poprawia się w miarę zdobywania nowego doświadczenia E”* [79]. Jako przykład ilustrujący te definicje można wskazać filtr antyspamowy, który korzystając z algorytmów uczenia maszynowego, identyfikuje niechciane wiadomości na podstawie zbioru danych treningowych, składających się z wiadomości oznaczonych przez użytkowników jako spam. W tym kontekście zadaniem T jest klasyfikacja wiadomości jako spam, doświadczeniem E są oznaczenia użytkowników, a miarą wydajności P jest dokładność, czyli stosunek poprawnie zidentyfikowanych wiadomości do wszystkich prób klasyfikacji. Różnica pomiędzy tradycyjnym programowaniem, a uczeniem maszynowym została przedstawiona na rys. 3.1.



Rys. 3.1 *Schemat porównawczy tworzenia algorytmu opartego o tradycyjne programowanie oraz uczenie maszynowe [80]*

Uczenie maszynowe to zatem dziedzina sztucznej inteligencji, która umożliwia komputerom uczenie się na bazie doświadczenia, poprawianie swoich działań i podejmowanie decyzji bez konieczności tworzenia złożonych programów zawierających reguły uwzględniające wszystkie przypadki, co praktycznie jest niemożliwe do wykonania. W uczeniu maszynowym algorytmy wykorzystują dane wejściowe do automatycznego wykrywania wzorców, które następnie mogą być użyte do przewidywań lub innych form analizy nieznanych danych. Uczenie maszynowe jest bardzo szeroką dziedziną wiedzy,

na którą składa się wiele, czasami bardzo różnych od siebie rodzin algorytmów oraz algorytmów przedstawionych w tab. 3.1.

Tab. 3.1 Podstawowe rodziny algorytmów uczenia maszynowego oraz wybrane algorytmy (opracowane na podstawie m.in. [80],[81],[82])

Rodzina algorytmów	Opis	Wybrane algorytmy
Metody liniowe	Rodzina algorytmów uczenia maszynowego, które zakładają liniową zależność między zmiennymi wejściowymi (cechami) a zmienną wyjściową (celową). Przewidują wartości ciągłe na podstawie wag przypisanych do cech, natomiast w zadaniach klasyfikacji szacują prawdopodobieństwa lub przypisanie do klas na podstawie funkcji liniowej. Są one proste w implementacji i interpretacji, a dzięki wprowadzeniu regularyzacji mogą radzić sobie z problemami nadmiernego dopasowania.	Regresja liniowa
		Regresja logistyczna
		Regresja grzbietowa
		Regresja Lasso
Drzewa decyzyjne i ich warianty	Rodzina algorytmów uczenia maszynowego stosowanych zarówno w zadaniach klasyfikacji, jak i regresji. Opierają się na hierarchicznej strukturze przypominającej drzewo, gdzie każdy węzeł reprezentuje kryterium podziału danych (zależne od wybranej cechy), a liście wskazują przewidywaną klasę lub wartość. Drzewa decyzyjne są intuicyjne i łatwe do interpretacji, jednak często wymagają dodatkowych metod, takich jak przycinanie lub regularyzacja, aby uniknąć nadmiernego dopasowania.	Drzewa decyzyjne (Decision Trees)
		Lasy losowe (Random Forests)
Maszyny wektorów nośnych	Rodzina algorytmów uczenia maszynowego stosowanych głównie do zadań klasyfikacji i regresji. Algorytmy te działają na zasadzie znalezienia hiperpowierzchni (np. linii w 2D lub płaszczyzny w 3D), która maksymalizuje margines między różnymi klasami w danych treningowych. SVM są szczególnie skuteczne w przypadku danych o wysokiej wymiarowości i niekoniecznie liniowo separowalnych, dzięki zastosowaniu funkcji jądra, która umożliwia mapowanie danych do przestrzeni o wyższej liczbie wymiarów. Ich precyzja i możliwość kontrolowania złożoności sprawiają, że są powszechnie stosowane w różnorodnych dziedzinach.	Support Vector Machines, SVM
Metody oparte na prawdo-podobieństwie	Metody oparte na prawdopodobieństwie to rodzina algorytmów uczenia maszynowego, które wykorzystują modele probabilistyczne do przewidywania wyników lub przypisywania obserwacji do klas. Bazują one na założeniach statystycznych dotyczących danych i prawdopodobieństwach warunkowych, co pozwala na efektywne wnioskowanie nawet przy niewielkich zestawach danych. Algorytmy te są szczególnie skuteczne w przypadku problemów, gdzie można przyjąć określone założenia dotyczące rozkładu danych, takich jak niezależność cech. Metody te są proste w implementacji, szybkie w działaniu i łatwe do interpretacji.	Naiwny klasyfikator Bayesa (Naive Bayes)
		Modele ukryte Markowa (Hidden Markov Models)

Algorytmy redukcji wymiarowości	Algorytmy redukcji wymiarowości to techniki stosowane w celu zmniejszenia liczby cech w danych, przy jednoczesnym zachowaniu jak największej ilości istotnych informacji. Pomagają w radzeniu sobie z problemem "przekleństwa wymiarowości," poprawiają wydajność modeli i ułatwiają wizualizację danych. Metody te można podzielić na dwie główne grupy: techniki selekcji cech (wybierające najważniejsze cechy) i techniki ekstrakcji cech (tworzące nowe cechy poprzez transformację istniejących). Algorytmy redukcji wymiarowości są szczególnie przydatne w analizie danych o dużej liczbie zmiennych, takich jak dane obrazowe czy genetyczne.	Principal Component Analysis, PCA
		Analiza dyskryminacyjna (LDA/QDA)
Algorytmy grupowania (Clustering)	Algorytmy grupowania to metody uczenia nienadzorowanego, które mają na celu podział danych na grupy (klastry) w taki sposób, aby obiekty w ramach jednej grupy były do siebie jak najbardziej podobne, a obiekty w różnych grupach – jak najbardziej różne. Grupowanie jest szeroko stosowane w eksploracji danych, segmentacji klientów, analizie obrazu oraz wykrywaniu anomalii. Algorytmy te różnią się podejściem do tworzenia klastrów, od metod opartych na centroidach po techniki hierarchiczne i gęstościowe.	K-średnich (K-means)
		Klasteryzacja hierarchiczna
		DBSCAN
Sztuczne Sieci Neuronowe	Sztuczne sieci neuronowe (ang. *Artificial Neural Networks,* ANN) to rodzina algorytmów uczenia maszynowego inspirowanych działaniem ludzkiego mózgu. Sieci te składają się z neuronów (węzłów) połączonych za pomocą wag, które są dostosowywane w procesie uczenia. Neurony są zorganizowane w warstwy – wejściową, ukryte i wyjściową – co pozwala na modelowanie złożonych nieliniowych zależności w danych. ANN znajdują zastosowanie w szerokim spektrum dziedzin, takich jak rozpoznawanie obrazów, analiza tekstu, przetwarzanie języka naturalnego czy prognozowanie czasowe.	Feedforward Neural Network
		Recurrent Neural Network (RNN)
		Convolutional Neural Network (CNN)
		Autodekoder
		Generative Adversarial Network (GAN)

Uczenie maszynowe szczególnie sprawdza się w obszarach o wysokiej złożoności problemów, przekraczającej możliwości tradycyjnych metod analitycznych lub w sytuacjach, gdzie brak jest gotowych algorytmów. Jest to dziedzina idealna dla zadań wymagających ciągłej adaptacji algorytmów, w obliczu dynamicznie zmieniających się danych i warunków oraz dla problemów, które charakteryzują się potrzebą analizy ogromnych zbiorów danych. Kluczowe zastosowania uczenia maszynowego obejmują:

1. **Problemy wymagające dynamicznego dostosowywania:** Uczenie maszynowe jest nieocenione, gdy konieczne jest ciągłe dostosowywanie algorytmów do nowych danych, co jest niemożliwe do osiągnięcia przez statyczne, z góry zdefiniowane reguły.
2. **Złożone problemy analityczne:** Tam, gdzie tradycyjne metody zawodzą z powodu złożoności zadania, uczenie maszynowe oferuje narzędzia do konstruktywnego podejścia i rozwiązania problemu poprzez uczenie się z dostępnych danych.

3. **Wsparcie w analizie danych:** Dzięki zdolności do przetwarzania i analizy dużych zbiorów danych, uczenie maszynowe staje się niezastąpionym narzędziem wspierającym ludzi w wydobywaniu wartościowych informacji ze skomplikowanych danych.

Przedstawione wybrane algorytmy oraz rodziny algorytmów specjalizują się wykonywaniu pewnych typów zadań. Tworząc i budując system w pierwszej kolejności należy określić wyzwania oraz cele, które należy osiągnąć, a następnie dobrać odpowiedni algorytm uczenia maszynowego, który wykona to zadanie. W tab. 3.2 przedstawiono podstawowe typy zadań wraz z proponowanymi algorytmami.

Tab. 3.2 Podstawowe typy zadań oraz proponowane algorytmy (opracowano na podstawie m.in. [80],[81],[82])

Typ zadania	Opis	Możliwe do zastosowania algorytmy ML
Regresja (●)	Zadanie polegające na przewidywaniu wartości ciągłych na podstawie istniejących danych. W kontekście tej pracy może być wykorzystana do określenia masy pojazdu na podstawie dostarczonych danych.	*Metody liniowe:* • *Regresja Liniowa* • *Regresja Wielomianowa* ***Sieci neuronowe:*** • ***Perceptrony wielowarstwowe (MLP)***
Klasyfikacja (○)	Proces przypisywania obiektów do określonych kategorii. W kontekście tej pracy może być wykorzystana do identyfikowania sylwetek pojazdów na podstawie dostarczonych danych np. z systemów wizyjnych.	*Metody oparte na prawdopodobieństwie:* • *Naiwny Klasyfikator Bayesa* *Maszyny Wektorów Nośnych:* • *SVM* ***Sieci neuronowe:*** • ***Sieci Konwolucyjne (CNN) dla obrazów*** • *Model YOLO*
Klastrowanie	Grupowanie danych w podobne kategorie bez wcześniejszego przypisywania etykiet. Algorytmy grupowania mogą być stosowane np. do analizy grup obiektów budowlanych pod kątem ryzyka, wieku konstrukcji czy zarządzania stanem technicznym.	*Algorytmy grupowania:* • *K-średnich (K-means)* • *DBSCAN* *Sieci neuronowe:* • *Self-Organizing Maps (SOM)*
Detekcja anomalii	Zadanie polegające na identyfikowaniu nietypowych wzorców lub odchyleń od normy. Algorytmy detekcji anomalii mogą pomagać w monitorowaniu stabilności konstrukcji oraz wykrywaniu nieprawidłowości, takich jak nadmierne odkształcenia czy uszkodzenia.	*Metody oparte na prawdopodobieństwie:* • *Naiwny Bayes* *Maszyny Wektorów Nośnych:* • *One-Class SVM* *Sieci neuronowe:* • *Autodekodery*

Odszumianie danych (●)	Proces usuwania szumów z danych w celu poprawy ich jakości.	***Sieci neuronowe:*** • ***Autodekodery (szczególnie warianty odszumiające)***
Redukcja wymiarowości (●)	Proces zmniejszania liczby zmiennych w dużych zbiorach danych, przy jednoczesnym zachowaniu kluczowych informacji.	*Algorytmy redukcji:* • *PCA, LDA* ***Sieci neuronowe:*** • ***Autodekodery***
Analiza szeregów czasowych	Przewidywanie przyszłych wartości na podstawie danych historycznych. Znajduje zastosowanie w prognozowaniu np. zużycia konstrukcji, ocenie efektywności remontów oraz przewidywaniu zmian w obciążeniu mostów.	*Sieci neuronowe:* • *Recurrent Neural Networks (RNN),* • *Long Short-Term Memory (LSTM)*
Uczenie wzmacniane	Uczenie się podejmowania optymalnych decyzji poprzez interakcję z otoczeniem. W budownictwie mogą być stosowane np. do optymalizacji harmonogramów pracy na budowie, zarządzania ruchem maszyn budowlanych czy efektywnym zarządzanie infrastrukturą lądową	*Sieci neuronowe:* • *Deep Q-Network (DQN),* • *Actor-Critic Methods (A3C)*
(●) - Typ zadania wykorzystany i zaimplementowany w Systemie Identyfikacji Nacisków. (○) - Typ zadania możliwy do zastosowania w Systemie Identyfikacji Pojazdów.		

Podstawowe zadania, które najtrafniej opisują wyzwania stojące przed systemem identyfikacji nacisków to regresja, odszumianie danych oraz redukcja wymiarowości i mają zastosowanie w Systemie Identyfikacji Nacisków. W niniejszej rozprawie do realizacji tych zadań wybrano algorytmy uczenia maszynowego z rodziny sztucznych sieci neuronowych, takie jak:

1. **ang. *Feedforward Neural Network,* FNN:** Zadaniem tej rodziny sieci neuronowych jest określenie całkowitego nacisku pojazdu, grup osi i osi, czyli regresja.
2. **ang. *Convolutional Neural Network,* CNN:** Zadaniem tej rodziny sieci jest ekstrakcja cech, czyli redukcja wymiarowości.
3. **Autodekoder:** Podstawowym zadaniem autodekoderów jest odszumianie oraz ekstrakcja cech.

Dodatkowo opisywanym, lecz wykraczającym poza zakres niniejszej pracy, jest zadanie klasyfikacji, które może być wykorzystane w Systemie Identyfikacji Pojazdu do klasyfikacji (rozpoznanie marki pojazdu, typu pojazdu) na podstawie np. obrazu z kamer. Zadanie to jest tematem szeroko opracowywanym chociażby dla autonomicznych pojazdów np. marki Tesla, które poruszają się po drogach z wykorzystaniem AI oraz systemu kamer i czujników.

3.1.2 Podstawowe klasyfikacje metod uczenia maszynowego

W kontekście rozwoju i zastosowań uczenia maszynowego, istotne jest zdefiniowanie jego podstawowych kategorii, które pozwalają na klasyfikację metod w zależności od specyficznych cech i wymagań. Podział ten uwzględnia zarówno poziom nadzoru ludzkiego w procesie uczenia, możliwość adaptacji algorytmów w czasie rzeczywistym, jak i fundamentalne strategie działania tych systemów.
Wg m.in. publikacji [80],[81],[82],[83] można wyszczególnić następujące kategorie zależne od poziomu nadzoru ludzkiego:

- **Uczenie nadzorowane** wymaga danych etykietowanych, gdzie każdemu wejściu przypisana jest odpowiedź, co umożliwia algorytmowi naukę mapowania wejść na wyjścia. Filtr spamu to klasyczny przykład, gdzie e-maile są klasyfikowane na podstawie wcześniej oznakowanych danych.
- **Uczenie nienadzorowane** operuje na nieoznakowanych danych, próbując zidentyfikować ukryte wzorce bez zewnętrznych instrukcji. W odróżnieniu od uczenia nadzorowanego, gdzie algorytm uczy się na podstawie danych zawierających zarówno wejścia, jak i odpowiednie etykiety (wyniki), uczenie nienadzorowane próbuje znaleźć ukryte wzorce lub struktury w danych bez wcześniej zdefiniowanych etykiet czy instrukcji. Algorytm próbuje samodzielnie zidentyfikować powtarzające się wzorce, grupować podobne do siebie elementy lub redukować złożoność danych, aby ułatwić ich interpretację.
- **Uczenie półnadzorowane** wykorzystuje zarówno oznakowane, jak i nieoznakowane dane, co pozwala na efektywniejsze uczenie się w przypadkach, gdy etykietowane dane są ograniczone lub drogie w pozyskaniu.
- **Uczenie przez wzmacnianie** polega na interakcji agenta z otoczeniem w celu maksymalizacji sumy nagród. Agent uczy się serii decyzji, co jest szczególnie przydatne w zadaniach wymagających strategii lub polityk decyzyjnych. Przykładem jest popularna gra „Snake” na telefonie, gdzie sterujemy wirtualnym wężem, a naszym głównym celem jest zjadanie jabłek i unikanie uderzeń w przeszkody oraz w samego siebie. Im wąż zje więcej jabłek, tym dłuższy się staje. Budując sieć ciężko przygotować dane uczące, bo jak w j. programistycznym opisać zwycięstwo. Tworzy się więc sztuczną sieć neuronową, która steruje agentem, czyli w przypadku tej gry wężem. SSN podejmuje decyzje co do ruchu węża, natomiast otoczenie, podaje skutek podjętej decyzji. Tworzona jest więc funkcja nagrody za zjedzenie jabłek oraz funkcja kary za efekty niepożądane i na tej podstawie wzmacniana jest, bądź osłabiana sił połączeń w SSN, które są odpowiedzialne za analizę i podejmowanie właściwych decyzji.

W publikacjach np. [80],[81],[82] przedstawiono następujące kategorie zależne od możliwości uczenia się w czasie rzeczywistym:

- **Uczenie wsadowe** polega na trenowaniu systemu offline, wykorzystując wszystkie dostępne dane naraz. Jeżeli pojawią się nowe dane, trzeba nowy system uczyć na powiększonym zbiorze

danych, a starty system należy wyłączyć. Proces ten jest czasochłonny i zasobożerny, co sprawia, że aktualizacja systemu wymaga ponownego trenowania z użyciem pełnego zbioru danych. Choć proste i skuteczne, uczenie wsadowe może być niepraktyczne przy dużych zbiorach danych ze względu na wysokie koszty i potrzebę dynamiczniejszych aktualizacji.

- **Uczenie przyrostowe** pozwala na aktualizację modelu w miarę pojawiania się nowych danych, co jest korzystne w dynamicznie zmieniających się środowiskach. Uczenie przyrostowe polega na ciągłym i sekwencyjnym trenowaniu systemu za pomocą nowych danych, co umożliwia mu szybką adaptację do zmieniających się warunków przy niskich kosztach. Jest szczególnie przydatne w dynamicznych środowiskach i przy ograniczonych zasobach obliczeniowych. Jednak systemy te mogą doświadczać spadku wydajności przy otrzymywaniu błędnych danych, co wymaga stałej obserwacji i interwencji, aby utrzymać ich skuteczność.

W publikacji [80] wyszczególniono także, następujące kategorie zależne od sposobu pracy:

- **Uczenie z przykładów** w kontekście uczenia nienadzorowanego polega na analizie i porównywaniu dostępnych danych bez etykiet, aby znaleźć w nich naturalne grupy, wzorce lub anomalie. Algorytmy starają się „zrozumieć" dane, bazując na ich wewnętrznej strukturze i cechach szczególnych, a następnie porównać nowe dane i określić współczynnik podobieństwa do znanych mu przykładów.
- **Uczenie z modeli** koncentruje się na tworzeniu abstrakcyjnych reprezentacji danych, które pomagają zrozumieć ich strukturę i zależności. To podejście pozwala nie tylko na identyfikację podobieństw lub różnic w danych, ale także na głębsze zrozumienie ich struktury i potencjalnych przyczyn zaobserwowanych wzorców.

Podsumowując, uczenie maszynowe oferuje różnorodne metody adaptacji algorytmów do danych i problemów, które należy rozwiązać. Kategorie uczenia maszynowego, w tym uczenie nadzorowane, nienadzorowane, półnadzorowane i przez wzmacnianie, dostosowują się do różnych poziomów dostępności danych i ich etykietowania, oferując elastyczność w podejściu do analizy i modelowania. Uczenie przyrostowe i wsadowe odpowiadają na potrzeby dynamicznie zmieniających się środowisk oraz sytuacji, w których dostępne są pełne zestawy danych. Sposoby pracy, takie jak uczenie z przykładów i uczenie z modeli, pozwalają algorytmom na efektywne odkrywanie ukrytych wzorców i struktur w danych.

W szczególności warto podkreślić przydatność poszczególnych technik uczenia maszynowego w kontekście analizowanego obszaru. Dzięki swojej elastyczności, metody te pozwalają na efektywne odkrywanie i eksploatację ukrytych zależności oraz struktur w danych, co jest nieocenione zarówno w kontekście eksploracji wiedzy, jak i podejmowania trafnych decyzji w skomplikowanych problemach analitycznych

3.1.3 Sieci neuronowe i uczenie głębokie

Pomysł na sztuczne sieci neuronowe został przedstawiony w 1943 roku przez Warrena McCullocha i Waltera Pittsa w publikacji pt. *„A Logical Calculus of Ideas Immanent in Nervous Activity”* [84]. Publikacja była przełomem w podejściu do matematycznego modelowania funkcjonowania neuronów w mózgu. Zaproponowano uproszczony model neuronu, który stał się fundamentem dla architektury pierwszych sztucznych sieci neuronowych. Początkowy optymizm, sugerujący, że wkrótce możliwe będzie tworzenie zaawansowanej sztucznej inteligencji zdolnej do ludzkiego dialogu, napotkał na przeszkody techniczne i teoretyczne, które zostały zawarte w raporcie znanym w środowisku informatycznym jako „*Lighthill report*” [85] z 1973 r. Raport przedstawiał bardzo pesymistyczną prognozę dla wielu kluczowych aspektów badań, takich jak np. zaawansowana automatyka oraz wiedza dot. ośrodkowego układu nerwowego opartego na komputerach, stwierdzając, że *"w żadnej części tej dziedziny, dotychczasowe odkrycia nie wywarły znaczącego wpływu, który był obiecywany. "*. Skutkiem publikacji była redukcja funduszy na rozwój projektów związanych ze sztuczną inteligencją co skutkowało tzw. "*zimą AI*" w latach 70 i wczesnych 80.

Zainteresowanie sztucznymi sieciami neuronowymi odżyło w latach 80. dzięki nowym architekturom sieciowym i lepszym technikom uczenia wpisującym się w ramy koneksjonizmu ([86], [87]). Jednak rozwój innych modeli uczenia maszynowego, takich jak maszyny wektorów nośnych, które oferowały lepsze wyniki i solidniejsze podstawy teoretyczne, ponownie zmniejszył zainteresowanie sztucznymi sieciami neuronowymi (SSN) w latach 90. Współczesny renesans sztucznych sieci neuronowych jest napędzany przez kilka czynników:

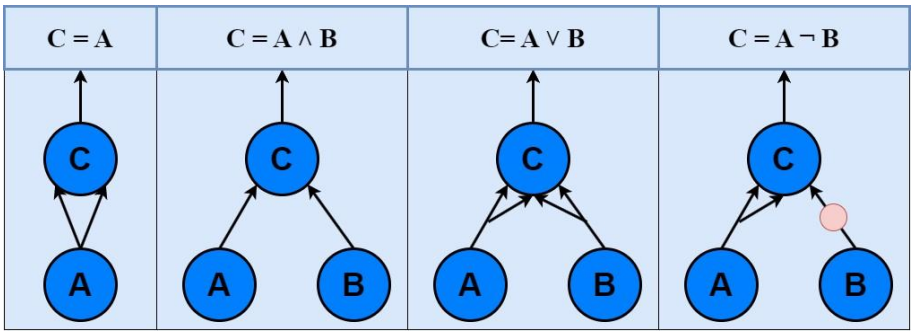
1. **Dostępność dużych zbiorów danych:** W dobie powszechnego internetu dostępne są niespotykane wcześniej ilości danych, które mogą być wykorzystane do trenowania sztucznych sieci neuronowych, co jest szczególnie istotne przy rozwiązywaniu złożonych problemów.
2. **Znaczący wzrost mocy obliczeniowej:** Postęp technologiczny, w tym prawo Moore'a oraz rozwój branży gier komputerowych, zwiększył dostępność zaawansowanych procesorów graficznych (ang. *Graphics Processing Unit,* GPU), które umożliwiają efektywne trenowanie rozbudowanych sieci neuronowych. O ile prawo Moore’a może przestać obowiązywać w stosunku do procesu produkcyjnego tranzystorów i zmniejszaniu się tranzystorów (obecny proces technologiczny to 7 i 5 nm), o tyle w dalszym ciągu prawdziwe jest prawo dla stosunku mocy obliczeniowej do ceny. Nie da się w nieskończoność zmniejszać budowy tranzystorów z uwagi na fizyczne rozmiary atomu, można jednak tworzyć układy wieloprocesorowe oraz redukować koszty produkcji.
3. **Udoskonalenie algorytmów uczących:** Choć współczesne algorytmy niewiele różnią się od tych z lat 90 nawet drobne modyfikacje przyczyniły się do znaczących usprawnień w procesie uczenia. Przykładem takiej modyfikacji jest automatyczne różniczkowanie symboliczne, które

ułatwia stosowanie wielu algorytmów opartych na gradientach, takich jak stochastyczny spadek gradientu (m.in. [88], [89]).

4. **Przełamanie barier teoretycznych:** Początkowe obawy dotyczące ograniczeń sztucznych sieci neuronowych, takie jak ryzyko utknięcia w lokalnych minimach, okazały się w praktyce mniej problematyczne niż przypuszczano [90].
5. **Rozwój rynkowy i finansowy:** Sztuczne sieci neuronowe zyskały na popularności, co przyciąga uwagę mediów, inwestorów i prowadzi do dalszego rozwoju tej technologii.

Fundamentem sieci neuronowych są sztuczne neurony, których działanie można przedstawić poprzez analogię do neuronów biologicznych. Neuron biologiczny, będący podstawową jednostką strukturalną i funkcjonalną układu nerwowego żywych organizmów, wykazuje skomplikowaną strukturę. Składa się z ciała komórkowego, które obejmuje jądro i organella komórkowe, licznych dendrytów przyjmujących sygnały z innych neuronów, oraz pojedynczego aksonu przekazującego impulsy elektryczne do innych komórek nerwowych. Na końcach aksonów znajdują się telodendrony z synapsami, które są odpowiedzialne za transmisję sygnałów do dendrytów lub ciał komórkowych sąsiednich neuronów za pomocą neuroprzekaźników. Neurony generują i przekazują potencjały czynnościowe, aktywujące synapsy i umożliwiające komunikację międzykomórkową.

Pomimo pozornej prostoty pojedynczego neuronu, tworzą one złożoną sieć o znacznym potencjale obliczeniowym składającą się z miliardów komórek nerwowych. Badania nad strukturą sieci neuronowych w mózgu, szczególnie w korze mózgowej, ujawniają warstwową organizację tych komórek, co ujawnia złożoność i zaawansowanie biologicznych systemów nerwowych [91].
W przeciwieństwie do neuronów biologicznych stoją modele sztucznych neuronów, inspirowane pracami McCullocha i Pittsa z 1943 roku [84]. Sztuczny neuron, będący elementarnym składnikiem sztucznych sieci neuronowych, aktywuje się po przekroczeniu określonego progu przez sumę sygnałów wejściowych. Ten model, choć prostszy od biologicznego odpowiednika, umożliwia tworzenie sieci zdolnych do wykonania zadań logicznych i obliczeniowych. Implementacja różnorodnych operacji logicznych w sztucznych sieciach neuronowych pozwala na tworzenie zaawansowanych systemów przetwarzania informacji, analogicznych do działania ludzkiego mózgu.

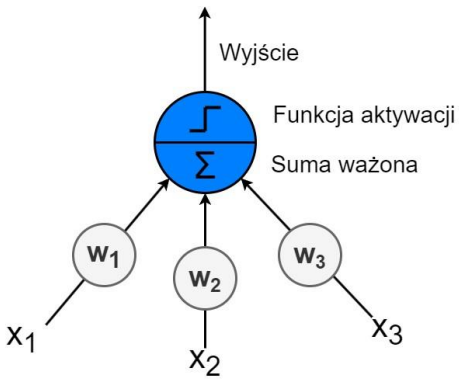


Rys. 3.2 Sztuczne sieci neuronowe i proste operacje logiczne [80]

Analiza operacji logicznych realizowanych przez sztuczne sieci neuronowe pozwala na zrozumienie ich potencjalnych zastosowań w przetwarzaniu informacji. Omówiono następujące konfiguracje sieci przedstawione na rys. 3.2:

1. **Funkcja tożsamościowa:** W analizowanej sieci reprezentującej funkcję tożsamościową aktywacja neuronu A skutkuje aktywacją neuronu C. Wynika to z faktu, że neuron C otrzymuje sygnały wyłącznie od neuronu A. Bez aktywacji A, neuron C pozostaje nieaktywny.
2. **Operacja logiczna ′∧′ (i):** Neuron C aktywuje się tylko, gdy oba neurony wejściowe A i B są jednocześnie aktywne, co odpowiada klasycznej realizacji funkcji logicznej 'i'.
3. **Operacja logiczna ′∨′ (lub):** Neuron C jest aktywny, jeśli przynajmniej jeden z neuronów A lub B jest aktywny.
4. **Kombinowana operacja z inhibicją:** Neuron C jest aktywny tylko, gdy otrzymuje sygnał z neuronu A, pod warunkiem, że neuron B jest nieaktywny, co ilustruje złożone interakcje sygnałów pobudzających i hamujących.

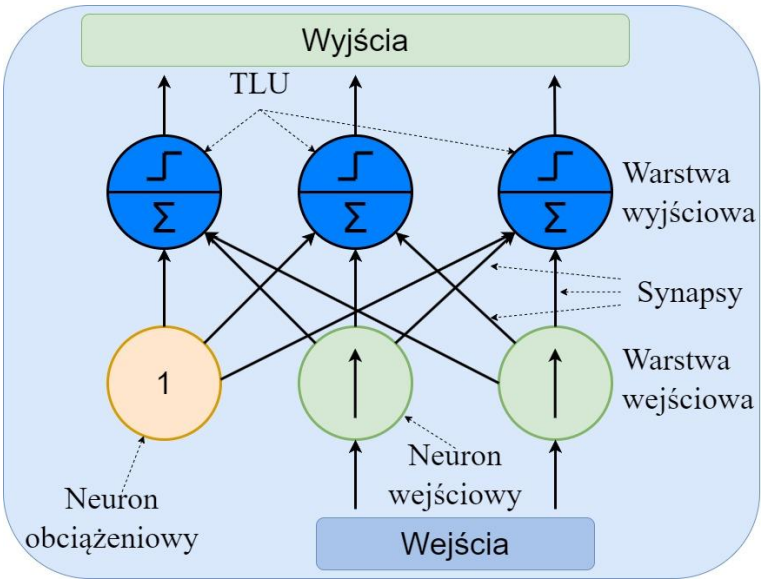
Perceptron opracowany w 1958 roku przez Franka Rosenblatta ([92], [93]) jest jednym z podstawowych modeli w dziedzinie sztucznych sieci neuronowych. Model ten wprowadza innowacje w postaci sztucznego neuronu – progowej jednostki logicznej (rys. 3.3) (ang. *Threshold Logic Unit*, TLU) lub liniowej jednostki progowej (ang. *Linear Threshold Unit*, LTU). W sztucznym neuronie każde połączenie między neuronami z poprzedniej warstwy posiada wagę, która odzwierciedla siłę połączenia synaptycznego. Proces obliczeniowy rozpoczyna się od wyliczenia ważonej sumy sygnałów wejściowych, zgodnie ze wzorem $z = w_1x_1 + w_2x_2 + \cdots + w_nx_n = x^Tw$. Następnie, suma jest wartością wejściową dla funkcji aktywacji, która w klasycznym perceptronie często przybiera postać funkcji skokowej Heaviside'a. Ta funkcja generuje wyjście „1”, jeśli ważona suma przekroczy ustalony próg lub „0” w przeciwnym przypadku, co opisuje się jako $h_w(x) = Heaviside(z = x^Tw)$.



Rys. 3.3 Progowa jednostka logiczna (ang. Threshold Logic Unit TLU): sztuczny neuron obliczający sumę ważoną sygnałów wejściowych, a następnie stosujący funkcję Heaviside'a (opracowano na podstawie m.in. [80],[81],[82])

Jednym z konkurencyjnych rozwiązań dla zastosowania funkcji skokowej Heaviside'a w sztucznych neuronach wielowarstwowych jest funkcja sigmoidalna. Dzięki swej gładkości i ciągłości, funkcja sigmoidalna przekształca sygnały wejściowe w wyjście w sposób, który umożliwia skuteczną propagację wsteczną błędu oraz optymalizację wag za pomocą metod gradientowych.

Perceptron składa się z pojedynczej warstwy wielu jednostek TLU (ang. *Threshold Logic Unit*), tworząc tzw. Warstwę w pełni połączoną lub gęstą, gdzie każdy neuron łączy się z każdym sygnałem wejściowym. Często dodaje się neuron obciążeniowy, wprowadzający stałe obciążenie do sieci, co ułatwia kontrolę nad aktywacją neuronów. Uogólniona budowa perceptronu została przedstawiona na rys. 3.4.



Rys. 3.4 Diagram perceptronu składającego się z dwóch neuronów wejściowych, jednego neuronu obciążeniowego i trzech neuronów wyjściowych (opracowano na podstawie m.in. [80],[81],[82])

Sygnały wejściowe są przekazywane bezpośrednio do neuronów wejściowych, które transmitują dane dalej. Matematycznie działanie perceptronu wyraża równanie (3.1):

$$h_{W,b}(X) = \emptyset(XW + B) \qquad (3.1)$$

gdzie:

- X – reprezentuje macierz cech wejściowych, gdzie każdy wiersz odpowiada poszczególnym przykładom, a każda kolumna odpowiada konkretnej cesze,
- W – macierz wag W zawiera wagi dla wszystkich połączeń, z wyjątkiem tych wychodzących z neuronu obciążeniowego.
- B – wektor obciążeń B zawiera wagi połączeń pomiędzy neuronem obciążeniowym a pozostałymi neuronami w sieci, przy czym dla każdego sztucznego neuronu przypisane jest jedno obciążenie,

- $\emptyset$ – funkcja aktywacji $\emptyset$ w przypadku liniowej jednostki logicznej, przyjmuje formę funkcji skokowej, decydując o aktywacji neuronów na podstawie sumy ważonej sygnałów wejściowych i obciążeń.

Model ten, opisując sposób działania perceptronu, podkreśla jego zdolność do przetwarzania informacji oraz wykonania podstawowych operacji logicznych, stanowiąc fundament dla bardziej złożonych architektur sieci neuronowych. Algorytm uczący, stworzony przez Franka Rosenblatta, czerpie inspirację z reguły Hebba, sformułowanej przez Donalda Hebba w pracy [94] z 1949 roku. Hebb postulował, że gdy jeden neuron biologiczny regularnie pobudza inny neuron, ich połączenie synaptyczne ulega wzmocnieniu. Perceptron jest trenowany przy użyciu modyfikacji tej reguły, która uwzględnia błąd sieci w przewidywaniu wyników, obliczając wyjścia dla każdego z neuronów wyjściowych. W przypadku, gdy wynik jest nieprawidłowy względem oczekiwanej danej uczącej, następuje korekta wag połączeń dla wszystkich sygnałów wejściowych, które przyczyniły się do generacji prognozy.

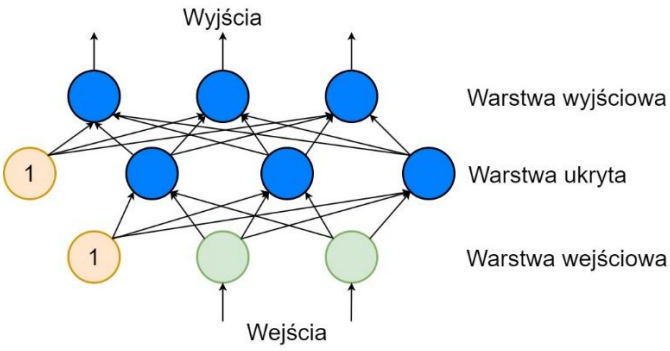
Ten mechanizm dostosowywania wag połączeń w perceptronie jest formalizowany za pomocą równania matematycznego (3.2), które odzwierciedla zasadę dostosowywania wag synaps przyczyniających się do osiągnięcia pożądanych wyników.

$$w_{i,j}^{następny\ krok} = w_{i,j} + \eta_{lr}\left(y_j - \hat{y}_j\right) x_i \qquad (3.2)$$

w tym równaniu:

- $w_{i,j}$ – waga połączenia pomiędzy i-tym neuron wejściowym z j-tym neuron wyjściowym,
- x_i – i-ta wartość wejściowa bieżącego przykładu uczącego,
- $\hat{y}_j$ - wynik j-tego neuronu wyjściowego dla bieżącego przykładu uczącego,
- y_j - docelowy wynik j-tego neuron wyjściowego dla bieżącego przykładu uczącego,
- η_{lr} - współczynnik uczenia.

Marvin Minsky i Seymour Papert w swojej monografii *"Perceptrons"* [95] z 1969 roku zidentyfikowali liczne ograniczenia modelu perceptronu, w tym jego niezdolność do rozwiązania prostych problemów, co jest cechą wspólną dla innych modeli klasyfikacji liniowej, takich jak regresja logistyczna. Wysokie oczekiwania wobec perceptronu przyczyniły się do rozczarowania jego osiągnięciami, co skłoniło wielu badaczy do poszukiwania alternatywnych obszarów badań, takich jak logika, rozwiązywanie problemów czy algorytmy wyszukiwania. Na rys. 3.5 przedstawiono przykładową architekturę perceptronu wielowarstwowego.



Rys. 3.5 Architektura perceptronu wielowarstwowego zawierającego dwa neurony wejściowe, jedną warstwę ukrytą składającą się z trzech neuronów i trzy neurony wyjściowe (opracowano na podstawie m.in. [80],[81],[82])

Po wprowadzeniu wielowarstwowej architektury perceptronu (ang. *Multi Layer Perceptron,* MLP) przezwyciężono niektóre z tych ograniczeń. MLP, zdolny do realizacji operacji „alternatywy rozłącznej" (ang. *eXclusive OR,* XOR), składa się z warstwy wejściowej, co najmniej jednej warstwy ukrytej oraz warstwy wyjściowej, wszystkie z jednostkami TLU. Wyzwaniem było efektywne uczenie MLP, aż do wprowadzenia algorytmu wstecznej propagacji błędu [96] w 1986 roku przez Davida Rumelharta, Geoffreya Hintona i Ronalda Williamsa, który umożliwia efektywną optymalizację wag. Proces uczenia z wykorzystaniem wstecznej propagacji błędu składa się z następujących etapów:

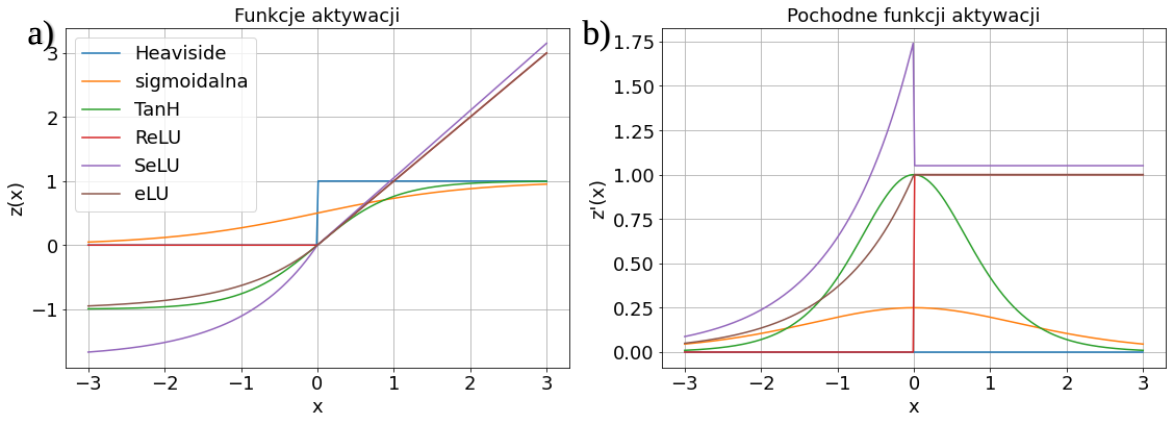
1. **Przetwarzanie w przód**: Każda mini-grupa danych jest wprowadzana do sieci przez warstwę wejściową, a następnie przekazywana sekwencyjnie przez warstwy ukryte. W każdej warstwie obliczany jest wynik dla wszystkich neuronów, bazując na danych z poprzedniej warstwy, co prowadzi do generacji wyjścia końcowego sieci. W trakcie tego procesu zachowywane są wszystkie pośrednie wyniki, niezbędne w dalszym etapie uczenia.
2. **Obliczanie błędu**: Na wyjściu sieci obliczana jest różnica między przewidywanymi a oczekiwanymi wartościami, wykorzystując do tego celu funkcję straty. Wynikiem jest miara błędu, która wskazuje na stopień niezgodności między aktualnymi wynikami sieci a danymi docelowymi.
3. **Propagacja wsteczna**: Algorytm analizuje wkład każdego neuronu w warstwie wyjściowej w wygenerowany błąd, wykorzystując do tego regułę łańcuchową. Proces propagacji wstecznej jest kontynuowany w kierunku warstw wcześniejszych, aż do osiągnięcia warstwy wejściowej. Dzięki temu możliwe jest obliczenie gradientu błędu dla wszystkich wag w sieci.
4. **Aktualizacja wag**: Wykorzystując mechanizm gradientu prostego, algorytm dostosowuje wagi połączeń w sieci na podstawie obliczonych gradientów, w celu zminimalizowania błędu.

Aby skutecznie działać, algorytm perceptronu wielowarstwowego wymagał zmiany architektury przez zastąpienie funkcji skokowej funkcją logistyczną. Zmiana ta była niezbędna, ponieważ funkcja skokowa nie posiada ciągłej pochodnej w każdym punkcie, co uniemożliwia wykorzystanie metody gradientu. W przeciwieństwie do tego funkcja logistyczna mająca zdefiniowaną pochodną niezerową w każdym

punkcie, umożliwia efektywną optymalizację wagi przy użyciu gradientu prostego, co znacząco wpływa na poprawę procesu uczenia sieci neuronowej.

Najbardziej podstawowe funkcje aktywacji (rys. 3.6), poza funkcją Heaviside'a, wykorzystywane w sieciach neuronowych to:

- sigmoidalna,
- tangens hiperboliczny (TanH),
- zniesiona jednostka liniowa, ang. *Rectified Linear Unit* (ReLU),
- skalowana wykładnicza jednostka liniowa, ang. *Scaled exponential Linear Unit* (SeLU),
- wykładnicza jednostka liniowa, ang. *exponential Linear Unit* (eLU).



Rys. 3.6 a) Funkcje aktywacji; b) Pochodne funkcji aktywacji

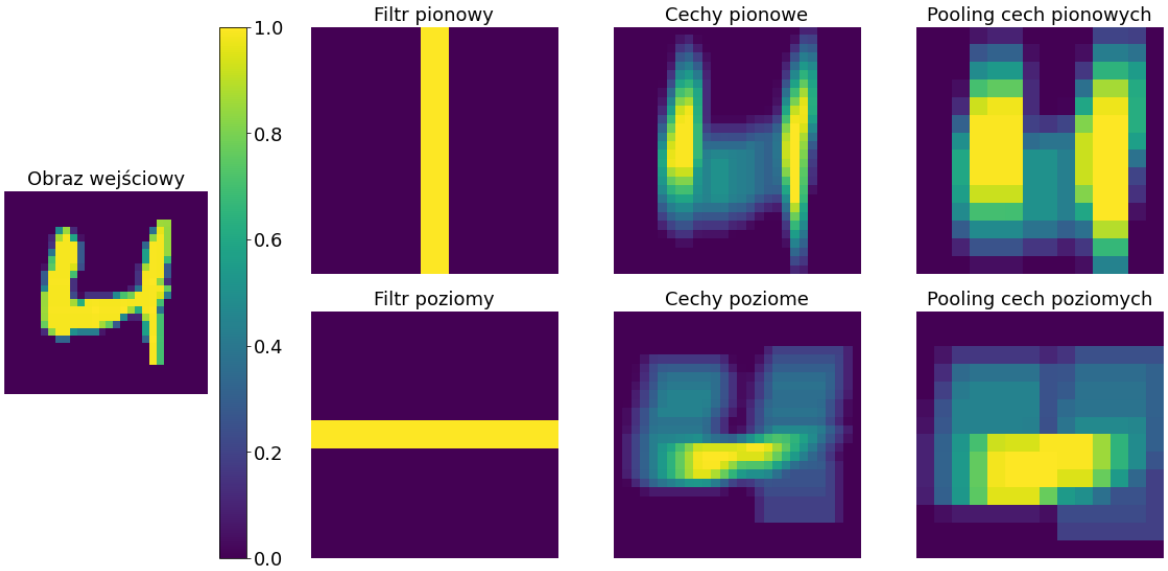
Perceptron wielowarstwowy jest stosowany w zadaniach regresji, gdzie pojedynczy neuron wyjściowy może generować prognozowaną wartość. W przypadku regresji wielowymiarowej, używa się więcej neuronów wyjściowych. Często w warstwie wyjściowej pomija się funkcje aktywacji, co pozwala na generowanie wyników w szerokim zakresie.

3.1.4 Neuronowe sieci splotowe

Sieci konwolucyjne (ang. *Convolutional Neural Networks*, CNN) są kluczowym elementem w dziedzinie rozpoznawania wzorców, szczególnie przy przetwarzaniu obrazów. Ich unikalna architektura, oparta na operacjach splotu (ang. *convolution*) oraz próbkowaniu (ang. *subsampling*), umożliwia efektywną ekstrakcję cech oraz redukcję wymiarowości danych wejściowych ([97], [98]). Próbkowanie w kontekście sieci konwolucyjnych odnosi się do technik redukcji wymiarowości i jest to proces polegający na wybieraniu co n-tej informacji z danych wejściowych. W sieciach konwolucyjnych subsampling jest często realizowany za pomocą operacji agregacji (ang. *pooling*) np. *max pooling* (wybierana jest najsilniejsza informacja z regionu), ang. *average pooling* (wybierana jest uśredniona wartość z regionu).

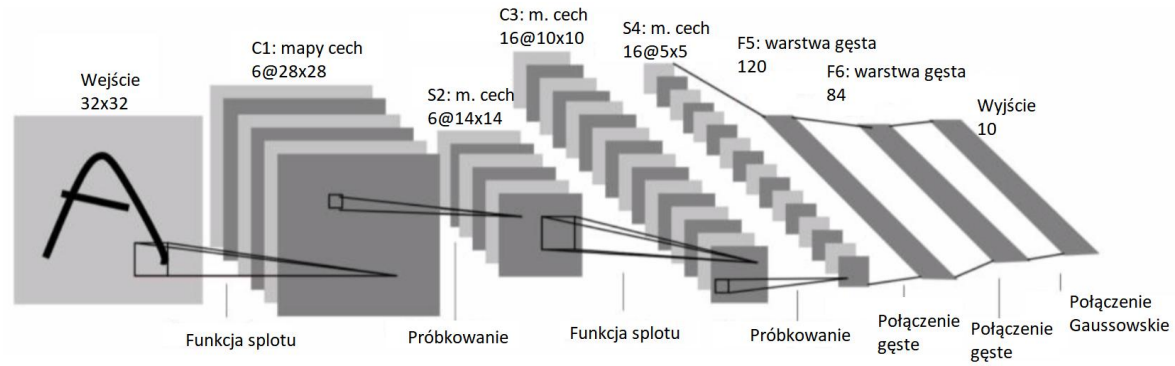
Operacja splotu, będąca podstawą działania CNN, polega na przesuwaniu filtra (jądra konwolucyjnego) przez cały obraz wejściowy, generując mapę cech. Pozwala to na koncentrację na lokalnych cechach obrazu, umożliwiając detekcję krawędzi, kształtów czy tekstur w początkowych warstwach, a w dalszym procesie na stopniowe komponowanie bardziej złożonych wzorców w głębszych warstwach.

Filtry konwolucyjne, pełniące kluczową rolę w procesie ekstrakcji cech, można traktować jako małe, uczące się jądra, skoncentrowane na wykrywaniu specyficznych wzorców w danych wejściowych. Na przykład, filtr wykrywający pionowe krawędzie będzie aktywowany przez pionowe linie na obrazie, ignorując inne cechy. Mechanizm ten został zilustrowany przedstawiony na rys. 3.7, gdzie z obrazu wyjściowego poprzez zastosowanie filtra pionowego oraz poziomego, uzyskano się wybrane cechy. Następnie wykorzystując metodę próbkowania, dokonano redukcji rozmiaru zbioru cech.



Rys. 3.7 Przykład działania filtrów do ekstrakcji cech tj. krawędzie pionowe oraz poziome oraz mechanizm próbkowania ze zmniejszeniem polegający na redukcji mapy cech do mniejszego formatu

Filtry te dostosowują się automatycznie podczas procesu uczenia, co pozwala sieci wyodrębnić najbardziej istotne cechy dla danego zadania. Gdy dane przechodzą przez warstwy konwolucyjne i próbkujące z pomniejszaniem (ang. *subsampling*), wydobywane cechy stają się bardziej złożone i abstrakcyjne co pozwala na interpretację obrazów na różnych poziomach abstrakcji, kluczowych dla skutecznej klasyfikacji i analizy. Operacja subsamplingu, ma na celu dalszą redukcję wymiarowości map cech, jednocześnie zachowując kluczowe informacje. Przez zastosowanie funkcji agregującej, takiej jak maxpooling, sieć zachowuje najbardziej wyróżniające się cechy w każdym obszarze, co przyczynia się do zwiększenia niezmienniczości względem drobnych przesunięć i deformacji obrazu. Maxpooling pozwala również na zmniejszenie liczby parametrów i obliczeń, co poprawia efektywność i generalizację modelu. W splotowych sieciach neuronowych warstwy te są ułożone naprzemiennie. Dodatkowo stosuje się dużą liczbę filtrów oraz warstw, dzięki temu można wyciągać bardziej złożone cechy. Na rys. 3.8 przedstawiono budowę typowej sieci CNN.



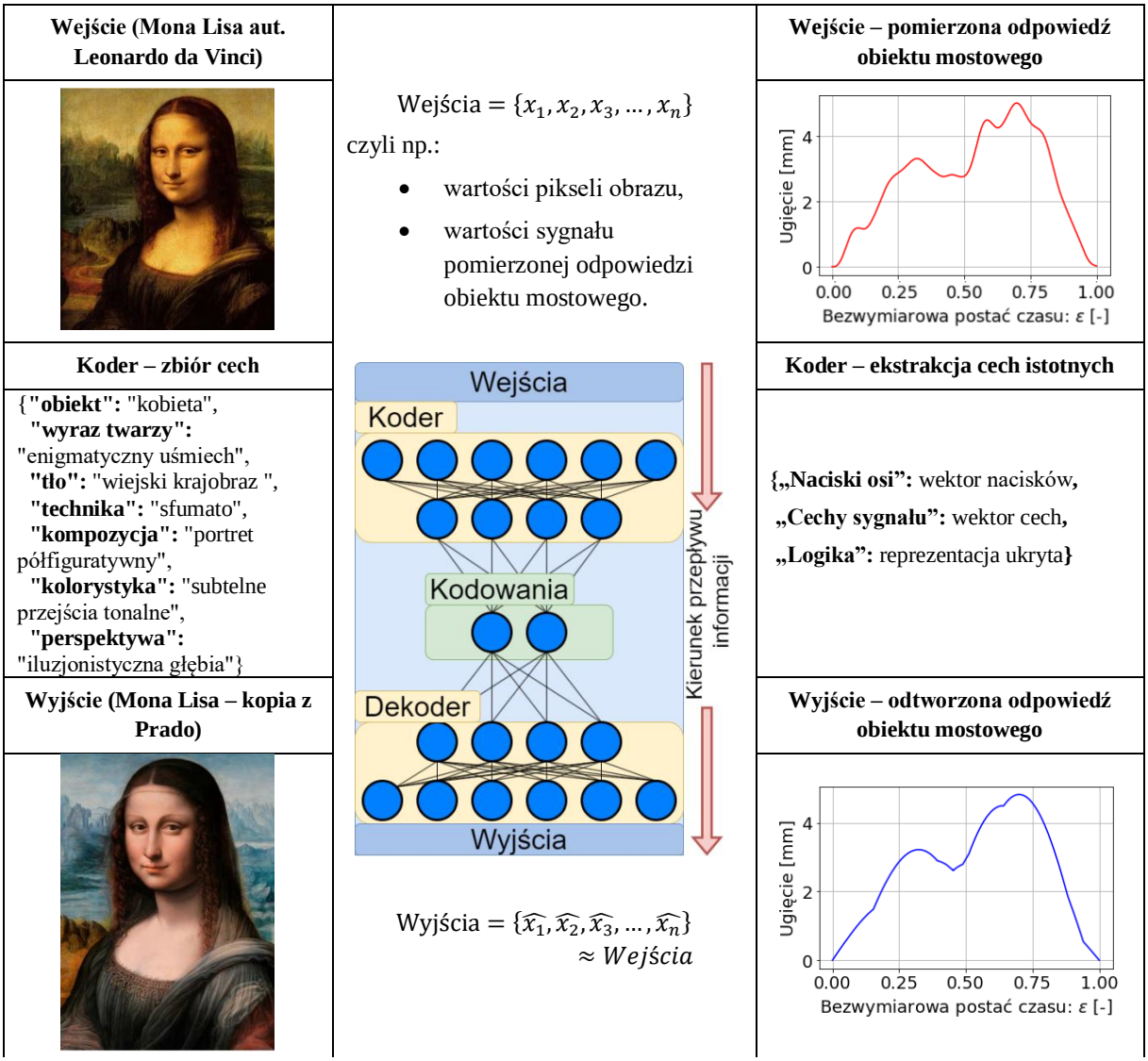
Rys. 3.8 Typowa struktura sieci neuronowej typu CNN [99]

W konstrukcji sieci, warstwy konwolucyjne i poolingowe są ułożone na przemian, co umożliwia stopniowe kompresowanie i abstrahowanie informacji, prowadząc do wydajnego rozpoznawania i klasyfikacji. Proces ekstrakcji cech w sieciach CNN można porównać do stopniowego budowania zrozumienia obrazu od prostych elementów do pełnych obiektów.

3.1.5 Autodekodery

Autodekodery (ang. *autoencoders*) to klasa sztucznych sieci neuronowych, które specjalizują się w generowaniu gęstych reprezentacji danych, znanych jako kodowania lub reprezentacje ukryte, bez zewnętrznego nadzoru. Kodowania charakteryzują się zredukowaną wymiarowością w porównaniu do danych wejściowych, co sprawia, że są idealne do zastosowań w redukcji wymiarowości, zwłaszcza w kontekście wizualizacji danych. Autodekodery są także używane jako detektory cech i narzędzia do wstępnego uczenia nienadzorowanego w kontekście głębokich sieci neuronowych [100].

Mechanizm działania autodekoderów polega na odtwarzaniu sygnałów wejściowych na wyjściach sieci, co może wydawać się prostym zadaniem. Jednak wprowadzenie ograniczeń, takich jak redukcja rozmiaru reprezentacji ukrytych czy dodawanie szumu do danych wyjściowych, znacząco komplikuje proces.



Rys. 3.9 Eksperyment z odtworzeniem obrazu (po lewej) oraz sygnału (po prawej) wraz z reprezentacją ogólnej idei struktury autodekodera (środek)

Strukturalnie autodekoder dzieli się na dwie główne części: koder, który konwertuje dane wejściowe na reprezentację ukrytą, oraz dekoder, który regeneruje dane wyjściowe z tej reprezentacji ukrytej. Przykładem działania autodekodera, zaprezentowanym na rys. 3.9, jest próba rekonstrukcji obrazu. Zawartość obrazu można streścić zbiorem cech, a następnie poprosić malarza, aby na tej podstawie odtworzył obraz. Efektem końcowym powinien być obraz na wzór obrazu pierwotnego. Im więcej cech szczególnych zostanie zawartych w opisie obrazu, tym dokładniejsza rekonstrukcja zostanie uzyskana. Autodekoder może analizować dane nie tylko związane z obrazem, ale także np. sygnał odpowiedzi konstrukcji mostowej na przejeżdżający pojazd.

Autodekodery można klasyfikować według różnych kryteriów ([80],[81],[82]), np. ze względu na stawiane im zadania:

- Ekstrakcja cech (ang. *feature learning*): Autodekodery uczą się reprezentacji danych w sposób umożliwiający wyodrębnienie istotnych cech wejściowych. Takie reprezentacje mogą być następnie wykorzystywane do klasyfikacji, klasteryzacji lub innych zadań analitycznych.
- Redukcja wymiarów (ang. *dimensionality reduction*): Autodekodery mogą działać podobnie do metod takich jak algorytmy ang. *Principal Component Analysis* (PCA), ucząc się nieliniowej reprezentacji danych w przestrzeni o mniejszej liczbie wymiarów, co ułatwia ich wizualizację i analizę.
- Odszumianie (ang. *denoising*): Specjalne wersje autodekoderów uczą się rekonstrukcji sygnału wejściowego na podstawie jego zaszumionej wersji. Pozwala to na efektywne usuwanie szumów z obrazów, sygnałów dźwiękowych czy danych tekstowych.
- Generowanie nowych danych (ang. data generation): Niektóre autodekodery, takie jak wariacyjne autodekodery (ang. *Variational AutoEncoder*, VAE), uczą się probabilistycznych reprezentacji danych wejściowych, umożliwiając generowanie nowych próbek o cechach podobnych do danych treningowych.
- Wykrywanie anomalii (ang. *anomaly detection*): Autodekodery mogą nauczyć się typowych wzorców w danych, a następnie wykrywać nietypowe przypadki, które znacząco odbiegają od oczekiwanej rekonstrukcji. Znajdują zastosowanie w diagnostyce medycznej, cyber bezpieczeństwie i analizie awarii.

Autodekodery można klasyfikować także ze względu na ich budowę ([80],[81],[82]). Do podstawowych typów należą m.in.:

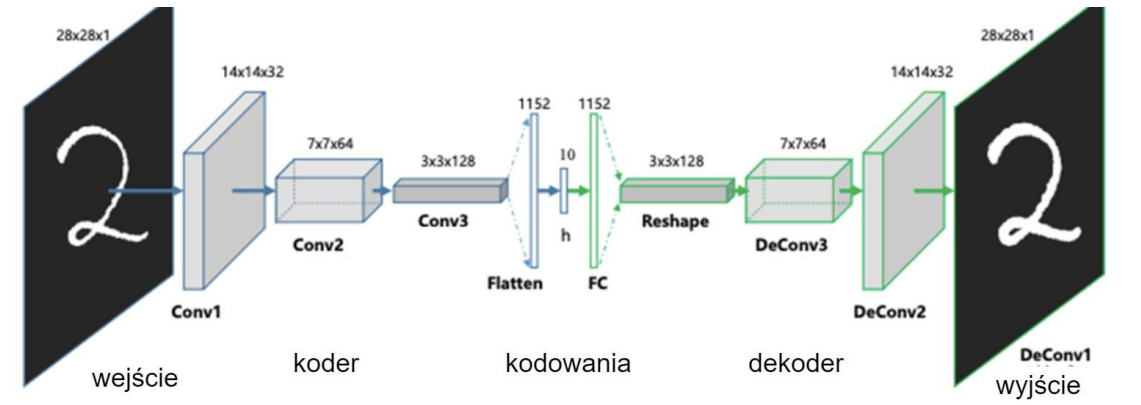
- Podstawowy autodekoder: Klasyczna architektura składająca się z w pełni połączonych warstw gęstych (ang. *dense*), stosowana do redukcji wymiarów i ekstrakcji cech.
- Stosowy autodekoder (ang. *Stacked Autoencoder,* SAE): Składa się z wielu warstw kodera i dekodera, co pozwala na bardziej złożone reprezentacje i lepszą ekstrakcję cech. Może być trenowany warstwa po warstwie w trybie nienadzorowanym.

- Rzadki autodekoder *(*ang. *Sparse AutoEncoder, SpAE)*: Wprowadza regularyzację L1, aby wymusić, by tylko niewielka liczba neuronów w warstwie ukrytej była aktywna. Pomaga to w ekstrakcji istotnych cech.
- Konwolucyjny autodekoder (ang. *convolutional autoencoder*): Wykorzystuje warstwy splotowe, co pozwala na efektywne przetwarzanie danych o strukturze przestrzennej, np. obrazów. Dzięki zastosowaniu warstw Conv i Deconv, zachowuje lokalne zależności w danych.
- Rekurencyjny autodekoder (ang. *Recurrent AutoEncoder*, RAE): Stosuje warstwy rekurencyjne (np. Long-Short Term Memory, Gated Recurrent Unit) zamiast w pełni połączonych, co umożliwia analizę danych sekwencyjnych, takich jak tekst lub szereg czasowy.
- Wariacyjny autodekoder (ang. *Variational AutoEncoder*, VAE): Zamiast zapamiętywać konkretne wartości w warstwie ukrytej, w której reprezentowane są skompresowane cechy danych wejściowych, modeluje ich probabilistyczny rozkład, co umożliwia generowanie nowych danych i interpolację w przestrzeni warstwy ukrytej.
- Kontraktowy autodekoder (ang. *Contractive AutoEncoder*, CAE): Wprowadza dodatkową funkcję kosztu penalizującą duże zmiany w przestrzeni warstwy ukryta, w której reprezentowane są skompresowane cechy danych wejściowych, co poprawia stabilność modelu i jego odporność na szum.
- Autodekoder oparty na sieciach ang. *Generative Adversarial Networks* (GAN): Łączy autodekoder z siecią generatywną, co umożliwia uzyskanie bardziej realistycznych reprezentacji w przestrzeni warstwy ukrytej.

W pracy wykorzystano zdolność autodekoderów do ekstrakcji cech oraz odszumiania danych, opierając się na stosowej, głębokiej architekturze sieci z warstwami splotowymi. Z tego względu poniżej przedstawiono i opisano zastosowane typy autodekoderów oraz w załączniku A zaprezentowano szczegółowe implementacje omawianych modeli autodekoderów.

Podstawowa architektura autodekodera często przypomina perceptron wielowarstwowy, lecz istotną różnicą jest zgodność liczby neuronów w warstwie wyjściowej z liczbą neuronów wejściowych, co wynika z założenia rekonstrukcji danych wejściowych. Autodekodery, będąc rozwinięciem koncepcji sieci neuronowych, mogą być wyposażone w liczne warstwy ukryte, prowadząc do powstania tzw. autodekoderów stosowych lub głębokich. Wprowadzenie dodatkowych warstw ukrytych pozwala na uczenie się bardziej złożonych reprezentacji danych. Architektura tych autodekoderów często charakteryzuje się symetrią względem centralnej warstwy ukrytej zapewniając zbalansowane procesy kodowania i dekodowania. W załączniku A.1 przedstawiono własną implementację autodekodera stosowego, którego zadaniem jest klasyfikowanie odręcznie napisanej cyfry.

Autodekodery splotowe (ang. *convolutional autoencoders*) to specjalna kategoria autodekoderów zaprojektowanych do przetwarzania danych wizualnych, takich jak obrazy. Trudno jest przypisać konkretną osobę jako pierwszego twórcę autodekodera splotowego. Można zauważyć, że wczesne prace nad autodekoderami splotowymi zaczęły pojawiać się krótko po tym, jak splotowe sieci neuronowe stały się popularne dzięki ich sukcesom w rozpoznawaniu obrazów. Idea splotowych autodekoderów została opisana np. w publikacji [101]. Stanowią one odmianę tradycyjnych sieci splotowych, które składają się z warstw splotowych i warstw próbkujących. W strukturze autodekodera splotowego kodująca część sieci ma za zadanie kompresję danych wejściowych do gęstej reprezentacji ukrytej. Proces ten obejmuje zastosowanie wielu warstw splotowych i próbkujących, które sukcesywnie zmniejszają wymiary przestrzenne danych, przy jednoczesnym zwiększaniu głębokości – liczby map cech. Taka struktura umożliwia efektywne wyodrębnienie hierarchii cech obrazów, co jest kluczowe dla skutecznego kodowania informacji wizualnych. Część dekodująca autodekodera splotowego odpowiedzialna za rekonstrukcję danych wejściowych z gęstej reprezentacji ukrytej musi dokonać odwrotnej transformacji. Obejmuje to zwiększenie wymiarów przestrzennych reprezentacji ukrytej oraz redukcję głębokości do poziomu pierwotnych danych wejściowych. W tym celu stosuje się tzw. transponowane warstwy splotowe (ang. *convolutional transpose layers*), które umożliwiają zwiększenie rozdzielczości danych przy jednoczesnej redukcji liczby kanałów. Proces ten pozwala na odtworzenie danych wyjściowych, które w jak największym stopniu odpowiadają oryginalnym obrazom wejściowym. Rys. 3.10 ilustruje schematycznie działanie autodekodera.



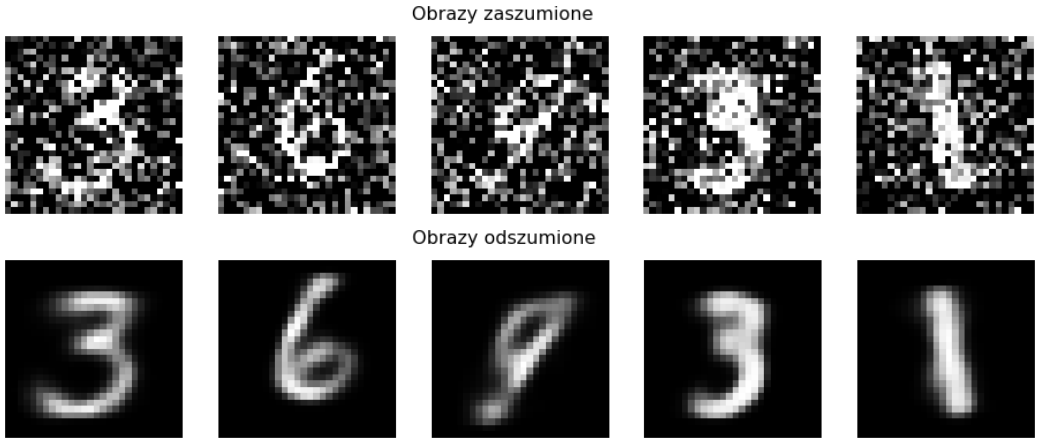
Rys. 3.10 Schemat działania autodekodera splotowego [101]

W prezentowanej na rys. 3.10 strukturze autodekodera zawarte są 3 warstwy splotowe (*Conv1, Conv2, Conv3*) stanowiące koder, 3 warstwy gęste (*Flatten, h, FC*) stanowiące kodowania oraz 3 warstwy splotowe (*DeConv3, DeConv2, DeConv1*) stanowiące dekoder. W poszczególnych warstwach kodera znajduje się odpowiednio 32, 64 i 128 filtrów. Po każdej operacji splotu następuje redukcja obrazów. Dlatego z pojedynczego sygnału na końcu kodera znajduje się 128 map cech o wymiarach 3x3, czyli 1152 informacje. Te parametry są następnie sprowadzane do kodera, który uczy się analizować cechy

szczególne. W najwęższym miejscu, czyli w warstwie h, system „pamięta” tylko 10 informacji. Następnie w procesie odwrotnym, następuje odtworzenie pierwotnego obrazu.

Autodekodery odszumiające to innowacyjne narzędzia w uczeniu maszynowym, które korzystają z umiejętności autodekoderów do ekstrakcji kluczowych cech. Pierwsze zastosowania tej techniki datują się na lata osiemdziesiąte XX wieku, ale ich potencjał w ekstrakcji cech został dokładnie przeanalizowany i udokumentowany w późniejszych badaniach (na przykład w [102], [103]).
W praktycznej implementacji autodekoder odszumiający często korzysta z konwencjonalnej architektury autodekodera stosowego, rozszerzonej o dodatkową warstwę ang. *dropout*. Ta warstwa jest aktywna tylko podczas fazy uczenia, co umożliwia efektywne trenowanie modelu do odszumiania danych. Jej mechanizm polega na wyłączaniu niektórych połączeń w sztucznej sieci neuronowej, jeśli siła połączeń pomiędzy neuronami jest poniżej pewnej wartości.

Przykładowe zaszumione obrazy oraz ich rekonstrukcje przez autodekoder pokazują zdolność modelu do rekonstruowania oryginalnych obrazów, nawet przy znacznym zaszumieniu, co potwierdza jego skuteczność w "odgadywaniu" brakujących szczegółów. W załączniku A.2 przedstawiono autorską implementację autodekodera odszumiającego do oczyszczenia obrazów z szumu gaussowskiego. Prezentowane na rys. 3.11 obrazy liczb z szumem oraz ich oczyszczone wersje zostały uzyskane na podstawie wspomnianej implementacji.

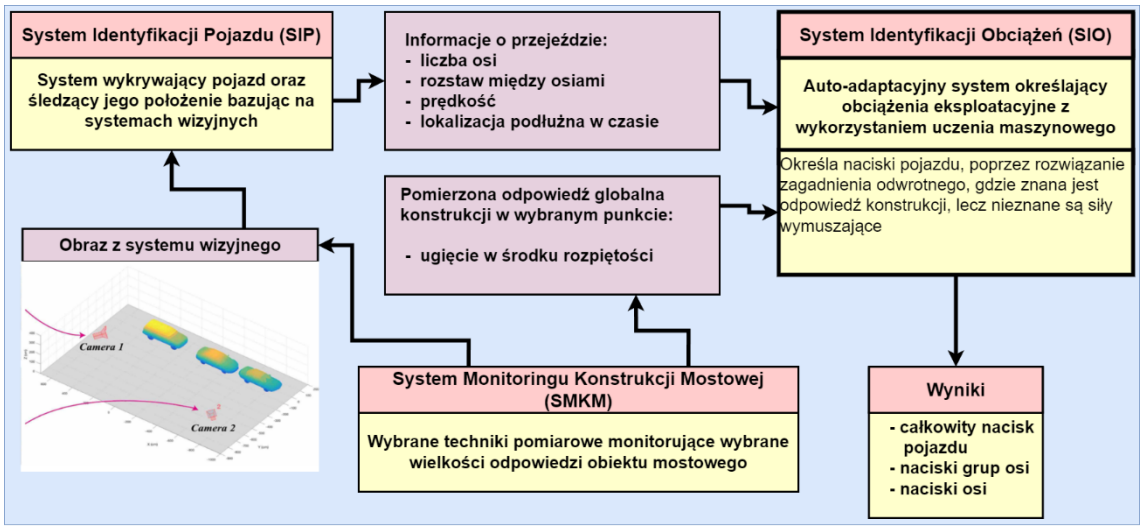


Rys. 3.11 Przykład działania autodekodera odszumiającego z warstwą dropout. Pierwszy rząd prezentuje obrazy liczb z nałożonym szumem gaussowskim. Drugi rząd reprezentuje obraz odtworzony przez autodekoder

Autodekodery odszumiające, oprócz wizualizacji procesu usuwania szumu i zastosowań jako narzędzie do nienadzorowanego uczenia wstępnego, stanowią skuteczną metodę oczyszczania danych. Ich wykorzystanie może znacznie poprawić jakość danych wejściowych dla kolejnych etapów przetwarzania, szczególnie w złożonych systemach wizyjnych.

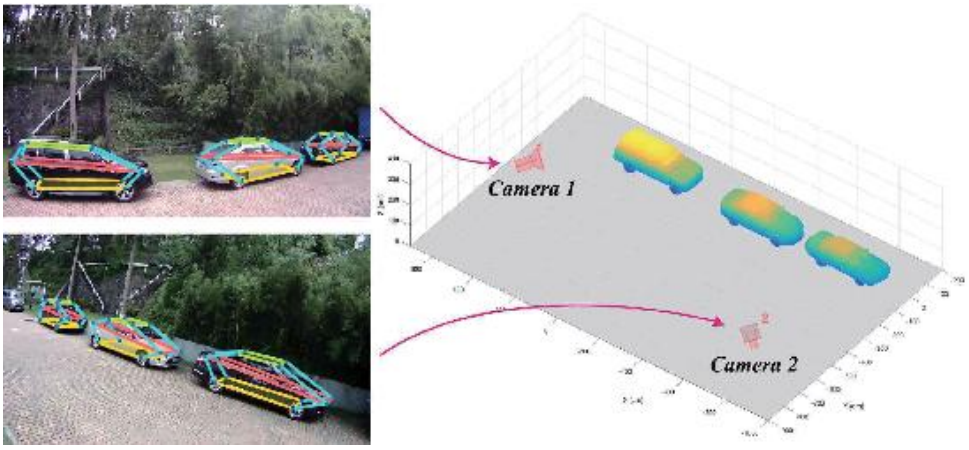
3.2 Koncepcja budowy systemu identyfikacji obciążeń obiektów mostowych

Podstawowym zadaniem rozprawy, jest stworzenie auto-adaptacyjnego systemu identyfikacji obciążeń obiektów mostowych pojazdami z wykorzystaniem uczenia maszynowego. Zgodnie z ogólnym schematem przedstawionym na rys. 2.2, mostowy system identyfikacji obciążeń typu Bridge Weigh-in-Motion składa się z dwóch kluczowych komponentów: Systemu Identyfikacji Pojazdów (SIP) oraz Systemu Identyfikacji Obciążeń (SIO). Uzupełnienie koncepcji stanowi System Monitoringu Konstrukcji Mostowej (SMKM), który za pomocą wybranych technik pomiarowych dostarczać będzie odpowiednich informacji dla kluczowych systemów. Architektura proponowanego nowego systemu, jest przedstawiona na rys. 3.12.



Rys. 3.12 Schemat blokowy proponowanej struktury systemu AutoSIO (Auto-adaptacyjny System Identyfikacji Obciążeń)

W zaproponowanym systemie AutoSIO System Identyfikacji Pojazdów (SIP) ma za zadanie określenie fundamentalnych parametrów pojazdu, takich jak liczba osi, rozstaw między osiami, prędkość, lokalizacja podłużna w czasie. Literatura przedmiotowa omówiona w podrozdziale 2.3 wskazuje na istnienie licznych algorytmów zdolnych do dostarczania tych danych. Z analizy wynika, że najbardziej efektywne wydają się być systemy wizyjne, które wykorzystując obraz z kamer, rejestrują przejazdy w czasie rzeczywistym, zapewniając ciągłość danych i wyższą precyzję w porównaniu do punktowych pomiarów, jak na przykład te realizowane przez mechaniczne detektory osi wbudowane w nawierzchnię. Rozwój tych technologii, wspomagany przez badania nad autonomicznymi systemami pojazdów, w których sieci neuronowe sterują ruchem pojazdów na podstawie rejestracji otoczenia przez kamery, umożliwia pewną klasyfikację pojazdów oraz określenie trajektorii ich ruchu (m.in. [31], [104], [105]). Najbardziej obiecującą technologią jest ta z dziedziny ang. *vehicle pose estimation*, której przykład działania został zaprezentowany na rys. 3.13.



Rys. 3.13 Oszacowanie konfiguracji pojazdu na podstawie obrazu z kamer [31]

Kluczowym elementem jest system AutoSIO, określający naciski pojazdów wykorzystując techniki uczenia maszynowego, który jest auto-adaptacyjny. Oznacza to, że algorytm, w miarę gromadzenia danych, sam poprawia jakość swojego działania. Znając konfigurację pojazdu, na podstawie danych z systemu identyfikacji pojazdów oraz sygnał odpowiedzi obiektu mostowego system AutoSIO określa nacisk całkowity pojazdu.

Podstawowe założenia proponowanego systemu AutoSIO to:

- System musi być auto-adaptacyjny poprzez wykorzystanie autodekodera, który jest oparty na uczeniu nienadzorowanym. System dzięki zastosowaniu autodekoderów będzie dążyć do oszacowania nacisku całkowitego pojazdu, a następnie będzie odtwarzać sygnał wejściowy. Dane wejściowe stanowią zarazem dane wyjściowe, co sprawia, że sieć jest w stanie określić dokładność odtworzenia. Podstawowym wyzwaniem jest więc stworzenie odpowiedniej architektury sieci, aby uzyskać oczekiwane naciski poszczególnych osi pojazdu. Jest to zadanie trudne, co też wykazano na podstawie algorytmu opisanego w podrozdziale 5.4, który poprawnie odtwarzał sygnał wejściowy, lecz niepoprawnie określał naciski pojazdu.
- Sygnał odpowiedzi konstrukcji mostu jest sygnałem dyskretnym w postaci wektora informacji i przedstawia on pionowe przemieszczenie konstrukcji w środku rozpiętości wywołane przejazdem pojazdu. Sygnały z czujników mogą być przetwarzane w sposób analogowy bądź cyfrowy. Z uwagi na fakt, że sieć neuronowa pracuje na wartościach numerycznych w postaci list, tablic i wielowymiarowych tablic, wszelkie informacje muszą zostać sprowadzone do wspomnianych formatów.
- Na obiekcie mostowym znajduje się 1 pojazd o 5 osiach maksymalnie. Zdecydowana większość pojazdów poruszających się po obiektach mostowych to pojazdy do 5 osi. Z uwagi na fakt, że auto-adaptacyjny system był implementowany od podstaw, przyjęto na początku pracy nad rozprawą doktorską pewne założenia wstępne co do liczby osi i liczby pojazdów.
- Znana jest funkcja wpływu mierzonej wielkości odpowiedzi konstrukcji mostowej. Sieć neuronowa to narzędzie zdolne do wyszukiwania nieoczywistych wzorców w pozornie

losowych danych. Konieczne jest dostarczenie informacji o fizycznych, realnych parametrach mostu. Najwygodniejszą i najłatwiejszą do uzyskania jest funkcja wpływu, która może być teoretyczna, ale co też najważniejsze, może być też uzyskana na podstawie obciążenia próbnego na istniejącym obiekcie. Uzyskana doświadczalnie funkcja wpływu opisuje dokładnie zachowanie się konstrukcji przy przejeździe quasi-statycznym.

- Głównym celem jest określenie całkowitego nacisku pojazdu. Celem dodatkowym jest określenie nacisków statycznych na poszczególne osie oraz grupy osi.

Stworzenie architektury oraz implementacja systemu stanowią złożone zadanie. Zdolność do uczenia się systemu w trakcie działania metod wykorzystujących uczenie maszynowe jest jednocześnie atutem jak i problemem. Problemem, który polega na braku kontroli nad tworzoną logiką w połączeniach między poszczególnymi warstwami sieci. Oznacza to, że niekoniecznie wartości cech zredukowanych w jądrze sieci będą odpowiadać wartościom nacisku na poszczególne osie. W pracy zaprezentowano 3 warianty systemu wykorzystujące uczenie maszynowe dla różnych architektur i opisano je w rozdziale 5. Proces implementacji systemu jest kompleksowym zadaniem wymagającym dokładnego planowania, testowania i wdrażania. Działania te mają na celu zapewnienie, że nowy system spełnia wszystkie oczekiwania funkcjonalne, jest stabilny i może być skutecznie zintegrowany z istniejącą infrastrukturą technologiczną. W podrozdziale 3.3 przedstawiono zaproponowaną i zastosowaną oryginalną metodykę implementacji systemu identyfikacji obciążeń obiektów mostowych.

3.3 Metodyka implementacji systemu identyfikacji obciążeń

Na potrzeby tej rozprawy opracowano i zastosowano autorską metodę implementacji systemu. Zastosowano technikę testowania systemu za pomocą obiektów atrap, znanych również w literaturze jako ang. *mock objects* [106]. Pełnią one kluczową rolę w procesie tworzenia oprogramowania, zwłaszcza w testowaniu jednostkowym. Dzięki nim możliwe jest symulowanie zachowań i interakcji obiektów w izolowanym środowisku, co umożliwia dokładne i kontrolowane testowanie komponentów oprogramowania.

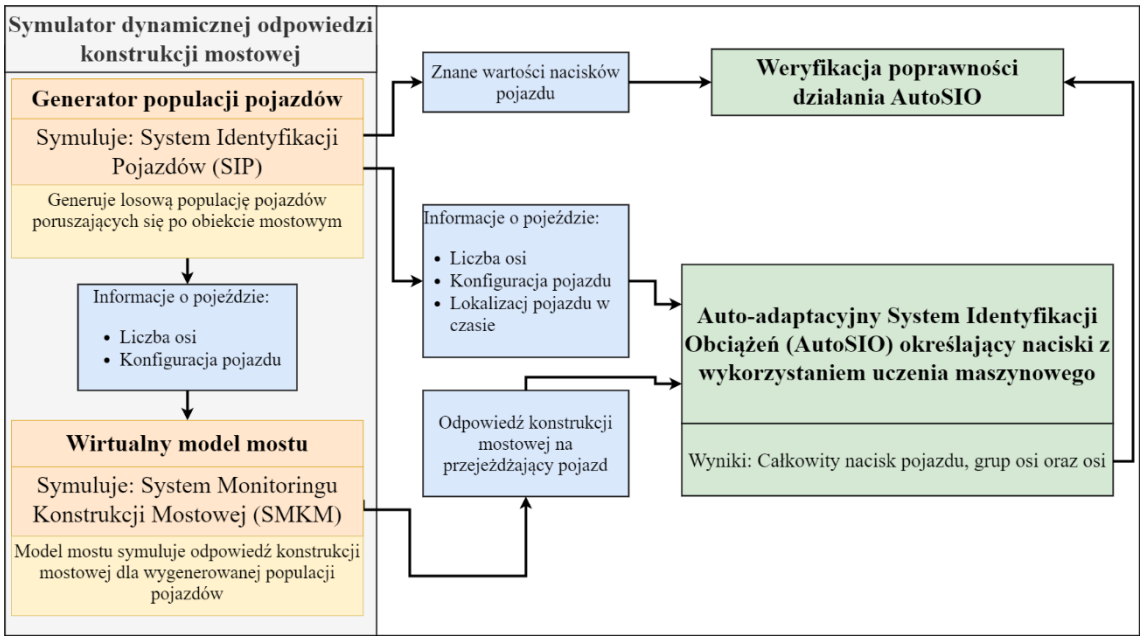
Zalety stosowania obiektów atrap to:

1. **Izolacja testów:** Obiekty atrapy pozwalają na izolację testowanego komponentu od reszty systemu. Dzięki temu, testy mogą skoncentrować się wyłącznie na funkcjonalności danego komponentu bez ryzyka zakłóceń wynikających z błędów w innych częściach systemu.
2. **Konfigurowalność:** Atrapy mogą być konfigurowane tak, aby odpowiadały różnym scenariuszom testowym. Można je zdefiniować tak, aby zwracały określone wartości, zwracały wyjątki (zazwyczaj błędy) lub reagowały na wywołania w specyficzny sposób, co jest szczególnie użyteczne przy testowaniu przypadków brzegowych.
3. **Zmniejszenie zależności:** Wiele systemów wymaga zdefiniowania skomplikowanych zależności i konfiguracji, które mogą być trudne lub czasochłonne do ustalenia w środowisku testowym. Obiekty atrapy eliminują potrzebę konfiguracji tych zależności, symulując ich zachowanie, co upraszcza proces testowania.
4. **Szybkość wykonania:** Testy wykorzystujące prawdziwe obiekty są z reguły czasochłonne, jeśli wymagają interakcji z bazami danych, sieciami lub zbieraniem informacji w czasie rzeczywistym. Zastosowanie atrap eliminuje te zależności, znacząco przyspieszając proces testowania.
5. **Dokładność testów:** Umożliwiają precyzyjne testowanie reakcji systemu na różne sytuacje, które mogą być trudne do wywołania przy użyciu prawdziwych obiektów, takie jak błędy systemowe, błędy aparatury pomiarowej czy inne wyjątkowe warunki.

Schemat implementacji systemu zaproponowano na rys. 3.14. Zastosowano symulator dynamicznej odpowiedzi obiektu mostowego, składający się z dwóch głównych komponentów (atrap):

1. **Generator populacji pojazdów:** Symuluje efekt działania systemu identyfikacji pojazdów.
2. **Wirtualny model mostu:** Zastępuje system monitorowania konstrukcji mostowej, który określa odpowiedź konstrukcji na przejazd pojazdu.

Takie podejście pozwala na dynamiczne modelowanie i testowanie systemu w kontrolowanych warunkach.



Rys. 3.14 Schemat implementacji oraz testowania z wykorzystaniem atrap

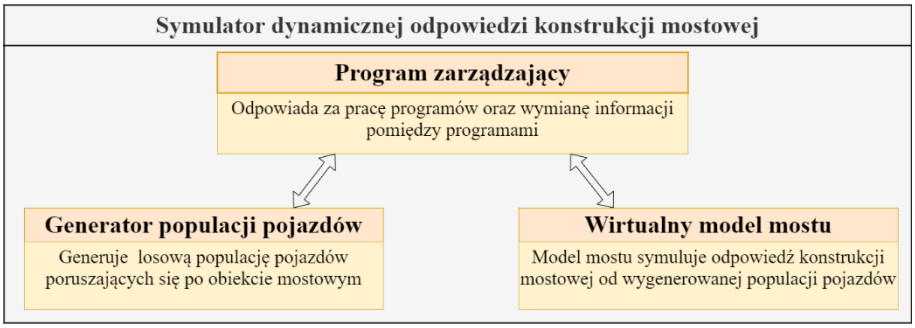
Celem testowania jest uzyskanie danych odpowiadających informacjom, które byłyby zebrane przy wykorzystaniu systemu monitoringu rzeczywistego obiektu mostowego. Dzięki kontrolowanemu środowisku możliwa jest dokładna weryfikacja stabilności oraz poprawności działania algorytmu. Pozwala to na szczegółową analizę potencjalnych problemów oraz ocenę efektywności systemu przed planowaną implementacją na obiekcie rzeczywistym. Wyniki takich symulacji są kluczowe dla dalszych etapów projektowania i rozwoju, umożliwiając optymalizację systemu przed jego rzeczywistym wdrożeniem.

Symulator dynamicznej odpowiedzi konstrukcji mostowej został szczegółowo opisany w rozdziale 4.
Implementacja algorytmu określającego naciski, wykorzystującego techniki uczenia maszynowego, jest przedstawiona w rozdziale 5.
Weryfikacja poprawności działania tego algorytmu oraz testy jego stabilności zostały opisane w rozdziale 6.

4 Symulator dynamicznej odpowiedzi konstrukcji mostowej na obciążenia pojazdami

4.1 Wprowadzenie

Zgodnie z założeniami przedstawionymi w podrozdziale 3.3, do poprawnego działania systemu wymagana jest symulacja dynamicznej odpowiedzi obiektu mostowego na przejazdy pojazdów użytkowych. Proponowany symulator składa się z kilku programów, współpracujących ze sobą, zgodnie ze schematem przedstawionym na rys. 4.1.



Rys. 4.1 Symulator dynamicznej odpowiedzi konstrukcji mostowej

Symulator składa się z 3 podprogramów:

1. **Generatora populacji pojazdów:** Program generujący populację pojazdów poruszających się po obiekcie mostowym.
2. **Wirtualny model mostu:** Program symulujący odpowiedź konstrukcji przy zadanych warunkach brzegowych.
3. **Program zarządzający:** Program stworzony w celu zarządzania podprogramami, wątkami symulacji oraz zapisem i transferem danych.

W następujących załącznikach opisano szczegółowo programy:

- Załącznik B.2 przedstawia implementację generatora populacji pojazdów.
- Załącznik B.3 przedstawia implementację programu zarządzającego.
- Załącznik B.4 przedstawia opis implementacji wirtualnego modelu mostu.

W podrozdziale 4.2 opisano przyjęte modele obliczeniowe.

W podrozdziale 4.3 dokonano walidacji poprawności uzyskiwanych rezultatów uzyskiwanych przy użyciu symulatora dynamicznej odpowiedzi konstrukcji mostowej z wynikami analiz przeprowadzonych przy wykorzystaniu MES:

- w podpunkcie 4.3.2 przedstawiono dobrane parametry obiektu mostowego,
- w podpunkcie 4.3.3 przedstawiono modelowanie nierówności nawierzchni na obiekcie,
- w podpunkcie 4.3.4 przedstawiono przyjętą metodą modelowania zawieszenia,
- w podpunkcie 4.3.5 zaprezentowano przyjęte referencyjne modele pojazdów,
- w podpunkcie 4.3.6 oraz 4.3.7 przedstawiono wyniki walidacji rozwiązań.

W podrozdziale 4.4 przedstawiono analizę wpływu wybranych parametrów symulatora na wyniki symulacji:

- w podpunkcie 4.4.1 przedstawiono zakres analiz,
- w podpunkcie 4.4.2 przedstawiono wpływ prędkości przejazdu,
- w podpunkcie 4.4.3 przedstawiono wpływ tłumienia konstrukcji,
- w podpunkcie 4.4.4 przedstawiono wpływ masy własnej konstrukcji,
- w podpunkcie 4.4.5 przedstawiono wpływ rozpiętości belki,
- w podpunkcie 4.4.6 przedstawiono wpływ sztywności zawieszenia pojazdu,
- w podpunkcie 4.4.7 przedstawiono wpływ tłumienia zawieszenia pojazdu,
- w podpunkcie 4.4.8 przedstawiono wpływ nierówności nawierzchni,
- w podpunkcie 4.4.9 przedstawiono wpływ różnych profili nierówności nawierzchni.

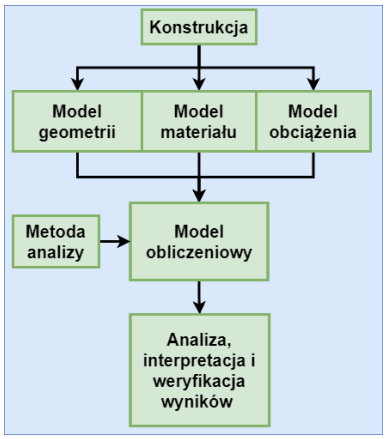
W podrozdziale 4.5 omówiono szczegółowo generator populacji pojazdów, odpowiednio:

- w podpunkcie 4.5.1 omówiono wymagania oraz przepisy dot. podstawowych parametrów pojazdu,
- w podpunkcie 4.5.2 przedstawiono opracowany algorytm generacyjny rozstawów oraz nacisków na osie w pojazdach,
- w podpunkcie 4.5.3 przedstawiono weryfikację wyników generatora.

4.2 Opis modeli obliczeniowych

4.2.1 Wprowadzenie

Modelowanie polega na uproszczonym odwzorowaniu rzeczywistej konstrukcji lub fragmentu otoczenia. Przed przystąpieniem do wyznaczania sił wewnętrznych zachodzi zawsze konieczność opracowania modelu konstrukcji. Sposób modelowania zależy w dużej mierze od przewidywanej metody rozwiązania zadania. Na rys. 4.2 przedstawiono ogólny proces tworzenia modelu obliczeniowego.



Rys. 4.2 Ogólny schemat procesu modelowania i analizy konstrukcji (opracowano na podstawie [107], [108])

Model obliczeniowy konstrukcji jest wyidealizowaną reprezentacją rzeczywistego układu fizycznego, zawierającą jedynie istotne cechy wpływające na badane zagadnienie. Upraszczając rzeczywistą konstrukcję, model obliczeniowy umożliwia efektywną analizę przy zachowaniu akceptowalnej dokładności wyników. Wybór odpowiedniego modelu obliczeniowego jest zadaniem trudnym i wymaga kompromisu pomiędzy wiernym odwzorowaniem rzeczywistej konstrukcji a racjonalnym nakładem pracy. Zgodnie z podaną klasyfikacją w publikacjach (m.in. w [107], [108]), model obliczeniowy konstrukcji składa się z trzech kluczowych elementów:

1. Modelu geometrii.
2. Modelu materiału.
3. Modelu obciążeń.

Uwzględniając klasyfikację podziału elementów modelu obliczeniowego przyjęto w pracy następujące założenia:

- **Model geometrii**: Z uwagi na teoretyczny charakter pracy analizowano konstrukcję jako belkę jednoprzęsłową, swobodnie podpartą.
- **Model materiału**: Z uwagi na konieczność uwzględnienia interakcji dynamicznej obiektu mostowego z pojazdem w trakcie przejazdu, przyjęto jednorodny, izotropowy, liniowo-sprężysty materiał.
- **Model obciążenia:** W analizach przeprowadzonych na potrzeby niniejszej pracy rozpatrywano następujące modele obliczeniowe pojazdu, różniące się stopniem złożoności:
  - **Model podstawowy pojazdu** symulujący przejazd zbioru sił po belce.
  - **Model rozszerzony pojazdu** symulujący przejazd punktów masowych po belce.
  - **Model szczegółowy pojazdu** symulujący przejazd mas resorowanych oraz nieresorowanych, które odzwierciedlają elementy nieresorowane zawieszenia, takie jak np. koła oraz elementy resorowane np. nadwozie wraz z ładunkiem.

  Dodatkowo dla modeli obciążenia wrażliwych na nierówności nawierzchni uwzględniono nierówności nawierzchni.

W tablicy tab. 4.1 zestawiono uwzględnione parametry w poszczególnych modelach pojazdu.

Tab. 4.1 Zestawienie uwzględnianych parametrów w poszczególnych modelach pojazdu

Model pojazdu	Uwzględniane parametry					
	Nacisk statyczny	Masy resorowane	Parametry zawieszenia pojazdu	Masy nieresorowane	Charakterystyki opon i nieresorowanych elementów zawieszenia	Wrażliwość modelu na nierówności nawierzchni
Model podstawowy	+	-	-	-	-	-
Model rozszerzony	+	+	+	-	-	+
Model szczegółowy	+	+	+	+	+	+
Oznaczenia: (+) uwzględnione; (-) nieuwzględnione						

Dla przyjętego modelu obliczeniowego konstrukcji wybrano dwie metody analizy, służące do rozwiązania odpowiedzi modelu konstrukcji na przejazd modelu obciążenia.

- **Metoda analityczno-numeryczna** wykorzystuje analitycznie sformułowane równania interakcji pojazd–most. Otrzymany układ równań różniczkowych rozwiązywany jest numerycznie.
- **Metoda elementów skończonych** rozwiązuje zagadnienie w oparciu o dyskretny model konstrukcji. W tym przypadku równania ruchu pojazd–most uzyskiwane są bezpośrednio z dyskretnego modelu numerycznego, a ich rozwiązanie następuje również metodami numerycznymi charakterystycznymi dla MES.

W tablicy tab. 4.2 przedstawiono przyjęte w pracy modele obliczeniowe układu pojazd–most wraz z odniesieniami do odpowiednich podpunktów pracy, w których opisano zastosowane rozwiązania dla każdego z modeli.

Tab. 4.2 Zestawienie modeli obliczeniowych

Nr modelu	Metoda analizy	Model geometrii	Model materiału	Model obciążenia pojazdu	Podpunkt
1	Metoda analityczno-numeryczna	Model ciągły swobodnie podpartej belki	Jednorodny, izotropowy, liniowo-sprężysty	Podstawowy	4.2.2 Model podstawowy
2	Metoda analityczno-numeryczna			Rozszerzony	4.2.3 Model rozszerzony
3	Metoda analityczno-numeryczna			Szczegółowy	4.2.4 Model szczegółowy
4	Metoda Elementów Skończonych	Model dyskretny swobodnie podpartej belki		Szczegółowy	4.2.5 Model MES

W pracy przyjęto łącznie cztery modele obliczeniowe układu pojazd–most, obejmujące trzy modele pojazdów o rosnącym stopniu złożoności oraz dwie metody analizy odpowiedzi konstrukcji. Zastosowanie dwóch niezależnych podejść umożliwia porównanie wyników oraz ocenę wpływu stopnia szczegółowości modelu pojazdu i metody analizy na uzyskiwane wyniki.

4.2.2 Model podstawowy

Przyjęto model ciągły konstrukcji obiektu mostowego oparty na klasycznej teorii belki jednoprzęsłowej według Euler'a-Bernoulliego. Model ten jest jednym z najczęściej stosowanych w analizie konstrukcji mostowych, ze względu na jego prostotę i jednocześnie wystarczającą dokładność dla wielu zastosowań inżynierskich. Przyjęto następujące założenia:

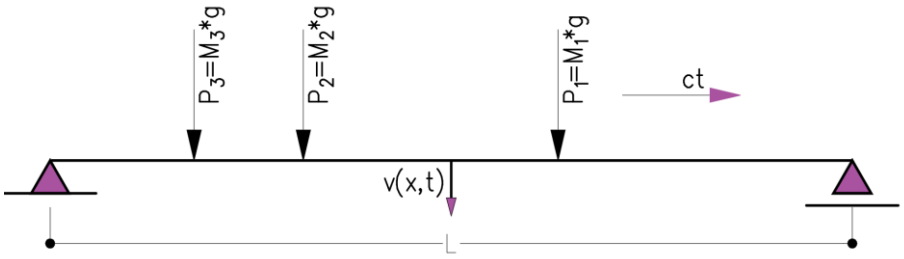
- W analizie założono, że belka jest wykonana z liniowo-sprężystego materiału. Właściwości materiałowe takie jak moduł Younga (E) oraz moment bezwładności przekroju (I) są stałe dla całego elementu.
- W modelu nie uwzględniono wpływu tarcia, oddziaływań termicznych ani innych nieliniowości.

Najprostszą metodą jest uzyskanie rozwiązania przemieszczenia belki od poruszającej się ze stałą prędkością siły. Całkowity nacisk statyczny osi na podłoże wynosi:

$$P = M \cdot g \tag{4.1}$$

gdzie:

- M – masa osi,
- g – przyspieszenie grawitacyjne.



Rys. 4.3 Ruchome siły skupione poruszające się po belce Euler'a

Dla pojedynczej siły, można zapisać więc następujące równanie:

$$\mu \frac{\partial^2 v(x,t)}{\partial t^2} + 2\mu\omega_b \frac{\partial v(x,t)}{\partial t} + EI \frac{\partial^4 v(x,t)}{\partial x^4} = \delta(x - ct)P \tag{4.2}$$

gdzie:

- x – współrzędna analizowanego punktu,
- t – czas,
- $v\,(x, t)$ – pionowe przemieszczenie punktu x w chwili czasowej t,
- EI – sztywność belki wyrażona jako iloczyn modułu Younga oraz momentu bezwładności,
- μ – liniowa masa pręta,
- ω_b – częstość kołowa tłumienia belki,
- $\delta\,(x - ct)$ – funkcja Diraca,

- P – siła wymuszająca,
- L – rozpiętość teoretyczna belki,
- c – prędkość przemieszczania się siły P.

Założono warunki brzegowe oraz warunki początkowe funkcji (4.2) jako następujące:

$$v(0,t)\ oraz\ v(l,t) = 0 \ \rightarrow \ \left.\frac{\partial^2 v(x,t)}{\partial x^2}\right|_{x=0} = 0 \ \ i \ \left.\frac{\partial^2 v(x,t)}{\partial x^2}\right|_{x=L} = 0$$

$$v(x,0) = 0 \ \rightarrow \ \left.\frac{\partial v(x,t)}{\partial t}\right|_{t=0} = 0$$

Jeśli przyjmiemy, że i-ta modalna forma belki jest wyrażona jako $\sin(\frac{i\pi x}{L})$, wtedy rozwiązanie wyrażenia (4.2) przyjmie formę:

$$v(x,t) = \sum_{i=1}^{\infty} \sin\frac{i\pi x}{L} V_i(t) \qquad (4.3)$$

gdzie:

- $V_i(t)$ – modalne przemieszczenia.

Po podstawieniu równania (4.3) do (4.2) , równanie (4.2) jest mnożone przez modalną funkcję kształtu dla *j*-tej formy $\sin(\frac{j\pi x}{L})$, gdzie j = 1,2…. Rozwiązanie następnie jest całkowane względem *x* w przedziale od 0 do *L*. Wykorzystując własność funkcji Diracka uzyskane zostanie rozwiązanie w postaci (4.4):

$$\ddot{V}_j(t) + 2\omega_b \dot{V}_j(t) + \omega_j^2 V_j(t) = \frac{2P}{\mu l}\sin\frac{j\pi ct}{l} \ (\mathrm{j} = 1,2,\dots,\infty) \qquad (4.4)$$

gdzie:

- $V_j(t), \dot{V}_j(t), \ddot{V}_j(t)$ – funkcja przemieszczenia pręta dla *j*-tej postaci modalnej oraz jej pochodne,
- $\omega_j^2 = EI\frac{j^4\pi^4}{L^4}$.

Zakładając, że belka obciążona jest *k* ruchomymi siłami, można stworzyć analogiczny układ wielu równań.

$$\begin{Bmatrix} \ddot{V}_1 \\ \ddot{V}_2 \\ \vdots \\ \ddot{V}_j \end{Bmatrix} + \begin{Bmatrix} 2\omega_b\dot{V}_1 \\ 2\omega_b\dot{V}_2 \\ \vdots \\ 2\omega_b\dot{V}_j \end{Bmatrix} + \begin{Bmatrix} \omega_1^2 V_1 \\ \omega_2^2 V_2 \\ \vdots \\ \omega_j^2 V_j \end{Bmatrix} =$$

$$= \frac{2}{\mu L}\begin{bmatrix} \sin\frac{\pi(ct-\hat{x}_1)}{L} & \sin\frac{\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{\pi(ct-\hat{x}_k)}{L} \\ \sin\frac{2\pi(ct-\hat{x}_1)}{L} & \sin\frac{2\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{2\pi(ct-\hat{x}_k)}{L} \\ \vdots & \vdots & & \vdots \\ \sin\frac{j\pi(ct-\hat{x}_1)}{L} & \sin\frac{j\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{j\pi(ct-\hat{x}_k)}{L} \end{bmatrix} \begin{Bmatrix} P_1 \\ P_2 \\ \vdots \\ P_k \end{Bmatrix} \qquad (4.5)$$

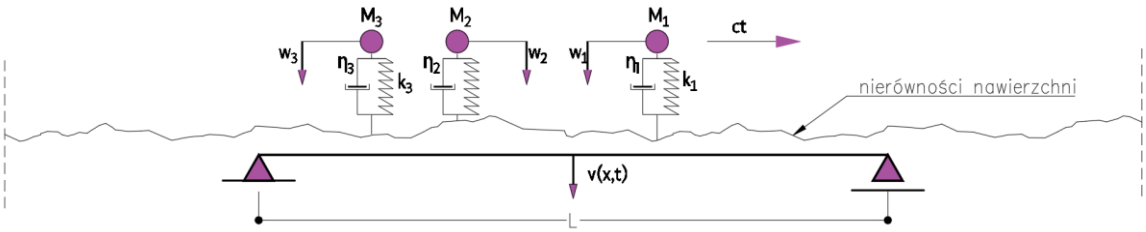
gdzie:

- $\hat{x}_k$ – dystans pomiędzy k-tą siłą, a siłą pierwszą. Przy takim założeniu $\hat{x}_1 = 0$.

Przy modelowaniu przejazdu całego pojazdu należy założyć, że analiza zaczyna się w momencie wjazdu pierwszej osi na obiekt mostowy, natomiast interakcja kończy się, kiedy ostatnia oś zjeżdża z obiektu. Oznacza to, że przy założeniu, że pojazd jest krótszy od belki, należy rozwiązać $2k - 1$ układów równań, gdzie k oznacza liczbę osi. Można przyjąć warunki brzegowe dla momentu, kiedy pierwsza oś wjeżdża na belkę, lecz w kolejnych układach, warunkiem początkowym jest rozwiązanie z poprzedniego przedziału czasowego.

4.2.3 Model rozszerzony

Zagadnieniem bardziej złożonym jest rozwiązanie przejazdu resorowanych oraz tłumionych punktów masowych po belce z uwzględnieniem nierówności nawierzchni (rys. 4.4).



Rys. 4.4 Model belki z nierównościami obciążonej poruszającym się zestawem punktów masowych reprezentujących poszczególne osie pojazdu

Masy, które reprezentują masy elementów pojazdu oraz ładunku, są połączone z podłożem sprężyną i tłumikiem drgań. Przedstawiony problem dla *i*-tej osi może zostać opisany jako zestaw dwóch równań różniczkowych (4.6) i (4.7). Wprowadzając współrzędną x_i, która opisuje lokalizację koła w czasie, można określić siły wymuszające działające na punkt masowy jako te, pochodzące od deformacji zawieszenia, np. jako skutek nierówności nawierzchni (4.8).

$$\mu \frac{\partial^2 v(x,t)}{\partial t^2} + 2\mu\omega_b \frac{\partial v(x,t)}{\partial t} + EI \frac{\partial^4 v(x,t)}{\partial x^4} = \delta(x_i(t))P_i(t) \qquad (4.6)$$

$$M_i \frac{d^2 w(t)}{dt^2} + \eta_i \frac{dw(t)}{dt} + k_i w_i(t) = R_i(x_i(t)), \qquad (4.7)$$

$$R_i(t) = \begin{cases} k_i[w_i(t) - v(x = x_i(t), t) - p_d(x = x_i(t))] \geq 0 & \text{jeśli } x_i(t) \geq 0 \text{ oraz } x_i(t) \leq L \\ k_i[w_i(t) - p_d(x = x_i(t))] & \text{jeśli } x_i(t) < 0 \text{ lub } x_i(t) > L \end{cases} \qquad (4.8)$$

gdzie:

- $P_i(t) = \left[M_i g - M_i \frac{d^2 w_i(t)}{dt^2}\right]$ – siła nacisku na most jest zmienna w czasie i zależy od nacisku statycznego oraz dodatkowych sił bezwładności układu masowego,
- $x_i(t) = x - c(t - t_i)$ – lokalizacja *i*-tego koła w czasie,
- t_i – chwila czasowa wjazdu *i*-tej osi na belkę. Dla pierwszej osi $t_1 = 0$,

- M_i – masa i-tego punktu masowego,
- $w_i(t)$ oraz pochodne – pionowe przemieszczenia, prędkości oraz przyspieszenia i-tego punktu masowego,
- k_i – sztywność zawieszenia,
- η_i - tłumienie zawieszenia,
- $p_d(x_i(t))$ – wartość nierówności nawierzchni drogowej dla i-tego koła w chwili czasowej t.

Zakładając k ruchomych mas oraz przeprowadzając analogiczne operacje jak dla (4.1) uzyskamy układ równań opisujący oddziaływania przejazdu punktów masowych po belce.

$$\begin{Bmatrix} \ddot{V}_1 \\ \ddot{V}_2 \\ \vdots \\ \ddot{V}_j \end{Bmatrix} + \begin{Bmatrix} 2\omega_b \dot{V}_1 \\ 2\omega_b \dot{V}_2 \\ \vdots \\ 2\omega_b \dot{V}_j \end{Bmatrix} + \begin{Bmatrix} \omega_1^2 V_1 \\ \omega_2^2 V_2 \\ \vdots \\ \omega_j^2 V_j \end{Bmatrix} =$$

$$= \frac{2}{\mu L} \begin{bmatrix} \sin\frac{\pi(ct-\hat{x}_1)}{L} & \sin\frac{\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{\pi(ct-\hat{x}_k)}{L} \\ \sin\frac{2\pi(ct-\hat{x}_1)}{L} & \sin\frac{2\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{2\pi(ct-\hat{x}_k)}{L} \\ \vdots & \vdots & \vdots & \vdots \\ \sin\frac{j\pi(ct-\hat{x}_1)}{L} & \sin\frac{j\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{j\pi(ct-\hat{x}_k)}{L} \end{bmatrix} \begin{Bmatrix} P_1 - M_1\frac{d^2w_1(t)}{dt^2} \\ P_2 - M_2\frac{d^2w_2(t)}{dt^2} \\ \vdots \\ P_k - M_k\frac{d^2w_k(t)}{dt^2} \end{Bmatrix} \quad (4.9)$$

Dodatkowo należy również podać równania opisujące zachowanie punktów masowych:

$$\begin{Bmatrix} M_1\frac{d^2w_1(t)}{dt^2} \\ M_2\frac{d^2w_2(t)}{dt^2} \\ \vdots \\ M_k\frac{d^2w_k(t)}{dt^2} \end{Bmatrix} + \begin{Bmatrix} \eta_1\frac{dw_1(t)}{dt} \\ \eta_2\frac{dw_2(t)}{dt} \\ \vdots \\ \eta_k\frac{dw_k(t)}{dt} \end{Bmatrix} + \begin{Bmatrix} k_1\, w_1(t) \\ k_2\, w_2(t) \\ \vdots \\ k_k\, w_k(t) \end{Bmatrix} = \begin{Bmatrix} R_1(t) \\ R_2(t) \\ \vdots \\ R_k(t) \end{Bmatrix} \quad (4.10)$$

$$R_i(t) =$$

$$= \begin{cases} k_i[w_i(t) - v(x = c(t-t_0), t) - p_d(x = c(t-t_0))] \geq 0 & \text{jeśli } x \geq 0 \text{ oraz } x \leq L \\ k_i[w_i(t) - p_d(x = c(t-t_0))] \geq 0 & \text{jeśli } x < 0 \text{ lub } x > L \end{cases} \quad (4.11)$$

Przy modelowaniu przejazdu całego pojazdu należy założyć, że analiza zaczyna się, kiedy pierwsza oś znajduje się w znacznej odległości od obiektu mostowego. Punkty masowe z uwagi na nierówności nawierzchni oscylują, symulując zachowanie prawdziwego pojazdu na nierównej drodze. Przyjęto następujące warunki brzegowe dla obiektu mostowego, jak i punktów masowych.

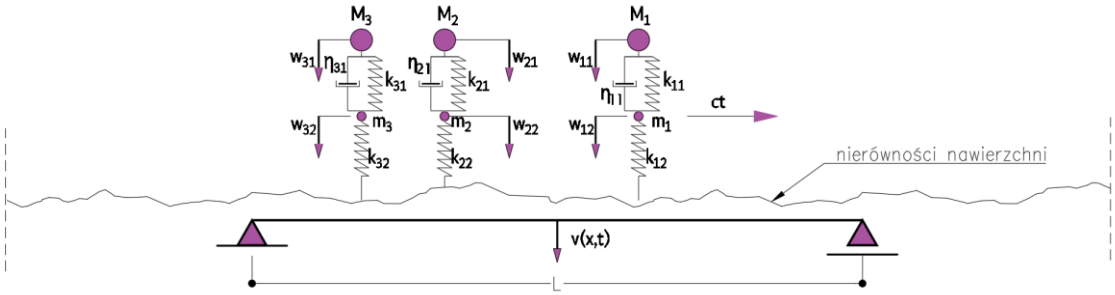
$$v(0,t)\ oraz\ v(l,t) = 0 \quad \rightarrow \left.\frac{\partial^2 v(x,t)}{\partial x^2}\right|_{x=0} = 0 \quad i \quad \left.\frac{\partial^2 v(x,t)}{\partial x^2}\right|_{x=L} = 0$$

$$v(x,0) = 0 \quad \rightarrow \quad \left.\frac{\partial v(x,t)}{\partial t}\right|_{t=0} = 0 \quad , \quad \left.\frac{\partial w_i(t)}{\partial t}\right|_{t=0} = 0 \ \ i\ w_i(t)|_{t=0} = 0$$

Z uwagi na konieczność uwzględnienia interakcji pojazd-podłoże przed pojawieniem się pojazdu na moście, oraz przy założeniu, że pojazd jest krótszy od belki, należy rozwiązać $2k$ układów równań, gdzie k oznacza liczbę osi.

4.2.4 Model szczegółowy

Rozwinięciem przejazdu mas resorowanych po obiekcie będzie przejazd bardziej złożonego układu składającego się z mas nieresorowanych i resorowanych połączonych ze sobą sprężynami oraz tłumikami. Elementy nieresorowane pojazdów to np. koła oraz ogumienie, oski napędowe, wahacze oraz wybrane elementy konstrukcyjne zawieszenia. Nierówności nawierzchni są przekazywane bezpośrednio na koła pojazdu, które charakteryzują się w stosunku do zawieszenia dużą sztywnością oraz bardzo niskim tłumieniem drgań. Elementy resorowane, to natomiast wszelkie elementy pojazdu nie znajdujące się w bezpośrednim kontakcie z nawierzchnią. Wszelkie nierówności nawierzchni są wstępnie niwelowane i tłumione przez zawieszenie.



Rys. 4.5 Model belki z nierównościami obciążonej poruszającymi się zestawami punktów masowych reprezentujących poszczególne osie pojazdu

Analogicznie jak dla modelu rozszerzonego można stworzyć dla wybranej osi pojazdu następujące równania:

$$\mu \frac{\partial^2 v(x,t)}{\partial t^2} + 2\mu\omega_b \frac{\partial v(x,t)}{\partial t} + EI \frac{\partial^4 v(x,t)}{\partial x^4} = \delta(x_i(t))P_i(t) \qquad (4.12)$$

$$M_i \frac{d^2 w_{i1}(t)}{dt^2} + \eta_{\mathrm{i}} \left[\frac{dw_{i1}(t)}{dt} - \frac{dw_{i2}(t)}{dt}\right] + k_{i1}[w_{i1}(t) - w_{i2}(t)] = 0 \qquad (4.13)$$

$$m_i \frac{d^2 w_{i1}(t)}{dt^2} - \eta_{\mathrm{i}} \left[\frac{dw_{i1}(t)}{dt} - \frac{dw_{i2}(t)}{dt}\right] - k_{i1}[w_{i1}(t) - w_{i2}(t)] = R_i(t) \qquad (4.14)$$

$$R_i(t) = \begin{cases} k_i[w_i(t) - v(x = x_i(t), t) - p_d(x = x_i(t))] \geq 0 & \text{jeśli } x_i(t) \geq 0 \text{ oraz } x_i(t) \leq L \\ k_i[w_i(t) - p_d(x = x_i(t))] & \text{jeśli } x_i(t) < 0 \text{ oraz } x_i(t) > L \end{cases} \qquad (4.15)$$

gdzie:

- $P_i(t) = \left[(M_i + m_i)g - M_i \frac{d^2 w_{i1}(t)}{dt^2} - m_i \frac{d^2 w_{i2}(t)}{dt^2}\right]$ – siła nacisku na most jest zmienna w czasie i zależy od nacisku statycznego oraz dodatkowych sił bezwładności punktów masowych,
- $\mathrm{M_i}$ – masa resorowanego punktu masowego,

- m_i – masa nieresorowanego punktu masowego,
- $w_{i1}(t)$ oraz pochodne – pionowe przemieszczenia, prędkości oraz przyspieszenia resorowanego i-tego punktu masowego,
- $w_{i2}(t)$ oraz pochodne – pionowe przemieszczenia, prędkości oraz przyspieszenia nieresorowanego i-tego punktu masowego.

Zakładając k – ruchomych mas oraz przeprowadzając analogiczne operacje jak dla (4.1) uzyskamy układ równań opisujący oddziaływania przejazdu punktów masowych po belce.

$$\begin{Bmatrix}\ddot{V}_1\\ \ddot{V}_2\\ \vdots\\ \ddot{V}_j\end{Bmatrix}+\begin{Bmatrix}2\omega_b\dot{V}_1\\ 2\omega_b\dot{V}_2\\ \vdots\\ 2\omega_b\dot{V}_j\end{Bmatrix}+\begin{Bmatrix}\omega_1^2V_1\\ \omega_2^2V_2\\ \vdots\\ \omega_j^2V_j\end{Bmatrix}=\frac{2}{\mu L}\begin{bmatrix}\sin\frac{\pi(ct-\hat{x}_1)}{L} & \sin\frac{\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{\pi(ct-\hat{x}_k)}{L}\\ \sin\frac{2\pi(ct-\hat{x}_1)}{L} & \sin\frac{2\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{2\pi(ct-\hat{x}_k)}{L}\\ \vdots & \vdots & \vdots & \vdots\\ \sin\frac{j\pi(ct-\hat{x}_1)}{L} & \sin\frac{j\pi(ct-\hat{x}_2)}{L} & \dots & \sin\frac{j\pi(ct-\hat{x}_k)}{L}\end{bmatrix}\times$$
$$\times\begin{Bmatrix}P_1-M_1\frac{d^2w_{11}(t)}{dt^2}-m_1\frac{d^2w_{12}(t)}{dt^2}\\ P_2-M_2\frac{d^2w_{21}(t)}{dt^2}-m_2\frac{d^2w_{22}(t)}{dt^2}\\ \vdots\\ P_k-M_k\frac{d^2w_{k1}(t)}{dt^2}-m_k\frac{d^2w_{k2}(t)}{dt^2}\end{Bmatrix} \quad (4.16)$$

Dodatkowo należy również podać równania opisujące punkty masowe:

$$\begin{Bmatrix}M_1\frac{d^2w_1(t)}{dt^2}\\ M_2\frac{d^2w_2(t)}{dt^2}\\ \vdots\\ M_k\frac{d^2w_k(t)}{dt^2}\end{Bmatrix}+\begin{Bmatrix}\eta_1\frac{dw_{11}(t)}{dt}\\ \eta_2\frac{dw_{21}(t)}{dt}\\ \vdots\\ \eta_k\frac{dw_{k1}(t)}{dt}\end{Bmatrix}-\begin{Bmatrix}\eta_1\frac{dw_{12}(t)}{dt}\\ \eta_2\frac{dw_{22}(t)}{dt}\\ \vdots\\ \eta_k\frac{dw_{k2}(t)}{dt}\end{Bmatrix}+\begin{Bmatrix}k_{11}\,w_{11}(t)\\ k_{21}\,w_{21}(t)\\ \vdots\\ k_{k1}\,w_{k1}(t)\end{Bmatrix}-\begin{Bmatrix}k_{11}\,w_{12}(t)\\ k_{21}\,w_{22}(t)\\ \vdots\\ k_{k1}\,w_{k2}(t)\end{Bmatrix}=\begin{Bmatrix}0\\0\\ \vdots\\0\end{Bmatrix} \quad (4.17)$$

$$\begin{Bmatrix}m_1\frac{d^2w_1(t)}{dt^2}\\ m_2\frac{d^2w_2(t)}{dt^2}\\ \vdots\\ m_k\frac{d^2w_k(t)}{dt^2}\end{Bmatrix}-\begin{Bmatrix}\eta_1\frac{dw_{11}(t)}{dt}\\ \eta_2\frac{dw_{21}(t)}{dt}\\ \vdots\\ \eta_k\frac{dw_{k1}(t)}{dt}\end{Bmatrix}+\begin{Bmatrix}\eta_1\frac{dw_{12}(t)}{dt}\\ \eta_2\frac{dw_{22}(t)}{dt}\\ \vdots\\ \eta_k\frac{dw_{k2}(t)}{dt}\end{Bmatrix}-\begin{Bmatrix}k_{11}\,w_{11}(t)\\ k_{21}\,w_{21}(t)\\ \vdots\\ k_{k1}\,w_{k1}(t)\end{Bmatrix}+\begin{Bmatrix}k_{11}\,w_{12}(t)\\ k_{21}\,w_{22}(t)\\ \vdots\\ k_{k1}\,w_{k2}(t)\end{Bmatrix}=\begin{Bmatrix}R_1(t)\\ R_2(t)\\ \vdots\\ R_k(t)\end{Bmatrix} \quad (4.18)$$

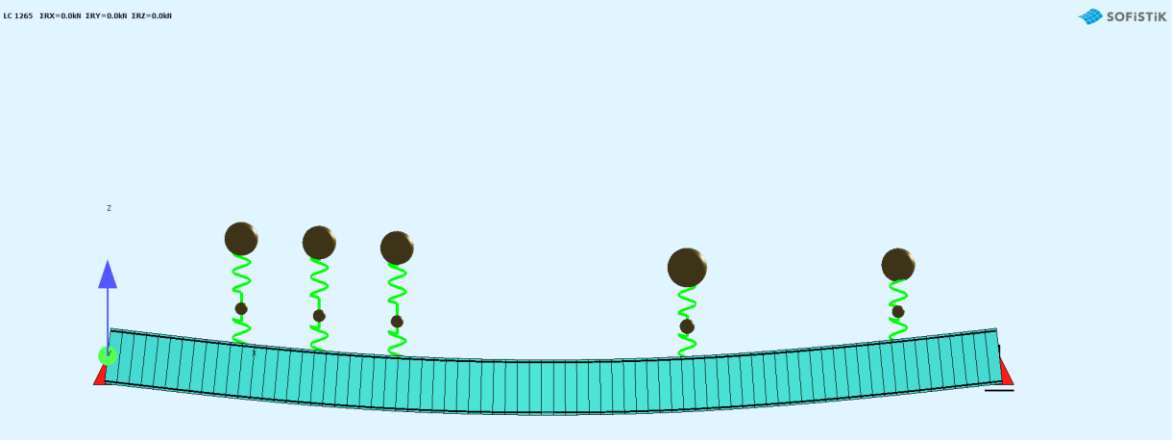
$$R_i(t)=\begin{cases}k_{i2}[w_{i2}(t)-v(x=x_i(t),t)-p_d(x=x_i(t))]\geq 0 & \text{jeśli } x_i(t)\geq 0 \text{ oraz } x_i(t)\leq L\\ k_{i2}[w_{i2}(t)-p_d(x=x_i(t))]\geq 0 & \text{jeśli } x_i(t)<0 \text{ lub } x_i(t)>L\end{cases} \quad (4.19)$$

Przyjmuje się analogiczne jak dla modelu rozszerzonego warunki brzegowe.

$$v(0,t)\ oraz\ v(l,t)=0\rightarrow\ \left.\frac{\partial^2v(x,t)}{\partial x^2}\right|_{x=0}=0;\ \left.\frac{\partial^2v(x,t)}{\partial x^2}\right|_{x=L}=0;$$
$$v(x,ts)=0\rightarrow\ \left.\frac{\partial v(x,t)}{\partial t}\right|_{t=0}=0\ ;\quad \left.\frac{\partial w_{i1}(t)}{\partial t}\right|_{t=0}=0;$$
$$w_{i1}(t)|_{t=0}=0;\left.\frac{\partial w_{i2}(t)}{\partial t}\right|_{t=0}=0;\ w_{i2}(t)|_{t=0}=0$$

4.2.5 Model MES

Rozwiązanie równań ruchu, nawet układów o jednym stopniu swobody, jest wymagającym zadaniem. Analitycznie można otrzymać rozwiązania tylko dla bardzo podstawowych zagadnień. Niezbędnym krokiem przy analizie dynamicznej złożonych układów jest przejście na model numeryczny. Interakcję pojazd-most zamodelowano przy wykorzystaniu programu SOFiSTiK z analizą dynamiczną wykonaną w module DYNA [109]. Na rys. 4.6 przedstawiono model pojazdu jako zestaw połączonych mas nieresorowanych oraz resorowanych poruszających się po belce.



Rys. 4.6 Masy resorowane oraz nieresorowane poruszające się po belce

W najbardziej ogólnym przypadku, równania ruchu można rozwiązać poprzez bezpośrednie całkowanie sprzężonych równań różniczkowych drugiego rzędu. Aby tego dokonać należy podzielić czas na dyskretne chwile czasowe $t \rightarrow \{t_1, \dots t_n\}$, a następnie dokonać dyskretyzacji równania i przekształcić je do równania w postaci macierzowej (4.20):

$$\boldsymbol{M}\ddot{\boldsymbol{u}}(t_{i+1}) + \boldsymbol{C}\dot{\boldsymbol{u}}(t_{i+1}) + \boldsymbol{K}\boldsymbol{u}(t_{i+1}) = \boldsymbol{p}(t_{i+1}) \text{ dla } i = 1, \dots, n-1 \qquad (4.20)$$

gdzie:

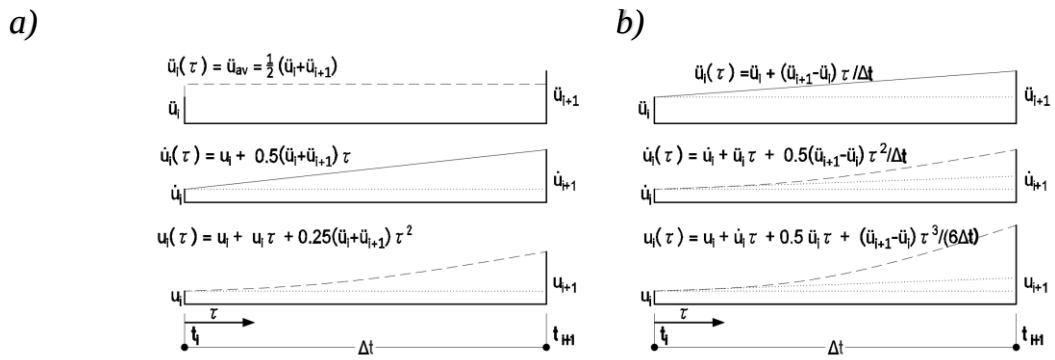
- $t_{i+1} = t_i + \Delta t$ - krok czasowy,
- n – liczba kroków czasowych.

Uogólniając, poprzez opuszczenie zapisu w postaci macierzowej, równania dla kroku czasowego t_{i+1} dla przemieszczeń (4.22) oraz prędkości przemieszczeń (4.21) bazują na znanych już rozwiązaniach układu uzyskanych w poprzednich krokach czasowych uwzględniających warunki początkowe.

$$\dot{u}(t_{i+1}) = \dot{u}(t_i) + \int_0^{\Delta t} \ddot{u}(\tau) d\tau \qquad (4.21)$$

$$u(t_{i+1}) = u(t_i) + \int_0^{\Delta t} \dot{u}(\tau) d\tau \qquad (4.22)$$

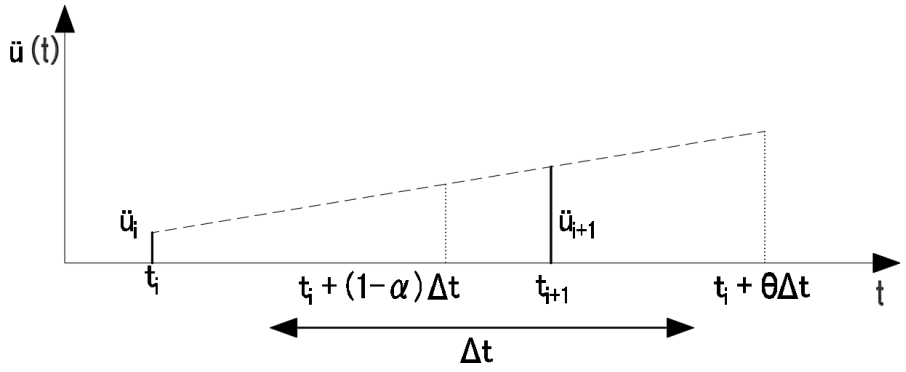
Równania odzwierciedlają prędkość oraz przemieszczenie na końcu kroku czasowego $\Delta t = t_{i+1} - t_i$ jako funkcję znanych na początku kroku czasowego wartości $\dot{u}(t_i)$ i $u(t_i)$. Aby znaleźć rozwiązanie, koniecznym jest przyjęcie zmiany przyspieszenia $\ddot{u}(t_i)$ w trakcie kroku czasowego np. jak stałą wartość przyspieszenia węzła w czasie kroku czasowego, albo jako funkcję liniową (rys. 4.7).



Rys. 4.7 a) Uśrednione wartości przyspieszeń, b) Liniowe wartości przyspieszeń [109]

Następnie należy dokonać wyboru pomiędzy pięcioma różnymi możliwościami wyboru momentu $t_i + \theta\Delta t$ (rys. 4.8) dla którego równanie równowagi jest spełnione:

- całkowanie bezpośrednie ($\theta = 0.0$),
- metoda Newmarka ($\theta = 1.0, \delta = 0.5, \beta \geq 0.25(0.5 + \delta)^2$),
- metoda Wilson-θ ($\theta \geq 1.37$),
- metoda α Hilber-Hughes-Taylor ($\theta < 1.0, \delta = \frac{1-2\alpha}{2}, \beta = \frac{(1-\alpha)^2}{4}$,
- analiza modalna.



Rys. 4.8 Całkowanie po czasie – metody α oraz Wilson-θ [109]

Dla ogólnej metody Newmarka rozwiązania przyjmą następującą formę (4.23), (4.24):

$$\dot{u}(t_{i+1}) = \dot{u}(t_i) + \Delta t[(1-\delta)\ddot{u}(t_i) + \delta\ddot{u}(t_{i+1})] \quad (4.23)$$

$$u(t_{i+1}) = u(t_i) + \Delta t\dot{u}(t_i) + \Delta t^2[(1/2-\beta)\ddot{u}(t_i) + \beta\ddot{u}(t_{i+1})] \quad (4.24)$$

4.3 Walidacja modeli obliczeniowych

4.3.1 Założenia

W celu walidacji rozpatrywanych modeli porównano rozwiązania uzyskane z 3 modeli (podstawowy, rozszerzony, szczegółowy) z wynikami modelu MES z wykorzystaniem przyjętych parametrów: konstrukcji obiektu mostowego, nierówności nawierzchni oraz konfiguracji pojazdów.

Rozważono następujące sytuacje obliczeniowe:

- Przejazd pojazdów po obiekcie mostowym bez uwzględnienia nierówności nawierzchni.
- Przejazd pojazdów po obiekcie mostowym z uwzględnieniem nierówności nawierzchni.

Przyjęto następujące założenia:

- Prędkość przejazdu wynosi 25 m/s, czyli 90 km/h.
- Pojazd zaczyna ruch za obiektem mostowym.
    - W przypadku nieuwzględniania nierówności nawierzchni, przyjmuje się, że początek analizy zaczyna się w momencie, gdy pierwsza oś pojazdu wjeżdża na obiekt.
    - W przypadku analiz z uwzględnieniem nierówności nawierzchni, przyjęto za początek analizy moment, gdy pierwsza oś znajduje się 125 m przed obiektem; na pojazd działają tylko siły od nierówności nawierzchni i dopiero po wjeździe na przęsło, dochodzi do interakcji pojazd-most.
- Analizowano również drgania swobodne obiektu mostowego od momentu, kiedy ostatnia oś pojazdu zjeżdża z obiektu przez okres równy okresowi bezpośredniej interakcji pojazd-most.

Szczegółowym celem analiz jest:

- Ocena różnic pomiędzy rozwiązaniami uzyskanymi za pomocą różnych modeli.
- Ocena złożoności obliczeniowej wyrażonej funkcją czasu niezbędnego do uzyskania wyniku dla różnych modeli.

4.3.2 Dobór parametrów obiektu mostowego

W celach walidacji modeli przyjęto referencyjny model przęsła obiektu mostowego. Założono, że konstrukcja obiektu mostowego będzie belką jednoprzęsłową, swobodnie podpartą.
Przy doborze parametrów elementu konstrukcyjnego, założono, że obiekt musi spełniać warunki ugięcia dla drogowych mostów żelbetowych. Z uwagi na fakt, że Eurokody [110],[111],[112],[113] nie precyzują warunków ugięcia, jako podstawę do określenia przemieszczeń przyjęto wymagania SGU wg PN-85/S-10030 [114], PN-82/S-10052 [115] oraz PN-91/S-10042 [116]. Normy te podają różne dopuszczalne warunki ugięcia w zależności od materiału konstrukcji, przeznaczenia, schematu statycznego oraz typu dźwigarów głównych. W niniejszej pracy zastosowano parametryczny model

konstrukcji, który jedynie w sposób ogólny odnosi się do rzeczywistych obiektów. Przyjęto więc arbitralnie, że ekstremalne przemieszczenia pionowe wywołane charakterystycznymi obciążeniami ruchomymi nie powinny przekraczać wartości $\frac{L_t}{600}$, gdzie L_t to rozpiętość teoretyczna. Przyjęto także obciążenie klasy B, tj. pojazd K o sile oddziaływania $600\ kN$ oraz obciążenie równomiernie rozłożone $3\frac{kN}{m^2}$. Założono, że na obiekcie występuje tylko 1 pas ruchu o szerokości 4 m. Pojazd K porusza się środkiem wyznaczonego pasa ruchu. Dla przyjętej rozpiętość teoretycznej równej 15 m strzałka dopuszczalna ugięcia wynosi: $\frac{L_t}{600} = \frac{15m}{600} = 0.025\ m$. Przyjęto ciężar związany z wyposażeniem wynoszący, zgodnie z tab. 4.3:

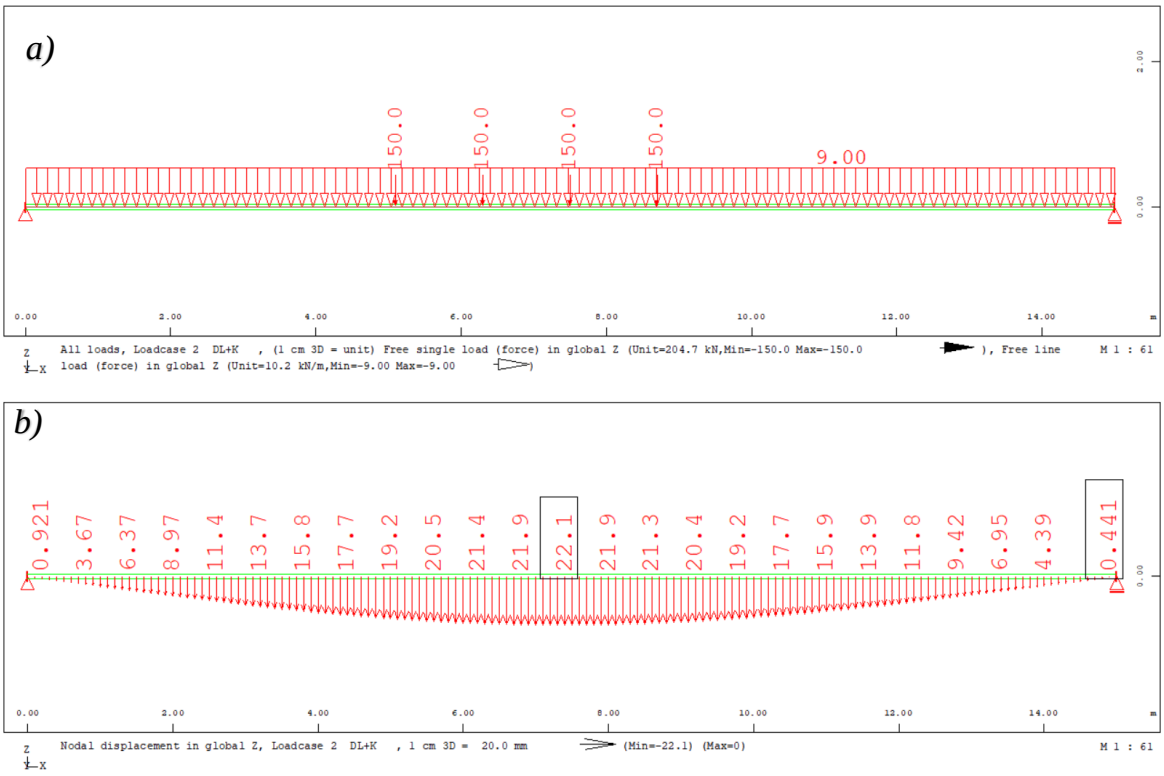
Tab. 4.3 Zestawienie przyjętych obciążeń

Element	Wartość obciążenia
Kapy chodnikowe	$2 \cdot 0.4\ m^2 \cdot 25\frac{kN}{m^3} = 20\frac{kN}{m}$
Krawężniki	$2 \cdot 1\frac{kN}{m} = 2\frac{kN}{m}$
Barieroporęcz	$2 \cdot 1\frac{kN}{m} = 2\frac{kN}{m}$
Nawierzchnia drogi	$23\frac{kN}{m^3} \cdot 0.1m \cdot 4\ m = 9.2\frac{kN}{m}$
Izolacja	$17\frac{kN}{m^3} \cdot 0.01m \cdot 6\ m = 1.02\frac{kN}{m}$
Łącznie	$34.22\frac{kN}{m} = 3422\frac{kg}{m}$

Założono, że materiałem konstrukcyjnym będzie stal o module Young'a wynoszącym $E = 210\ GPa$ oraz parametry przekroju konstrukcji nośnej jak dla 2 profili HE900B. Uwzględniając powyższe, przyjęto następujące parametry belki (tab. 4.4), a następnie w programie MES sprawdzono ugięcie od charakterystycznej kombinacji obciążeń ciężarem własnym, wyposażenia oraz użytkowym (rys. 4.9).

Tab. 4.4 Założone podstawowe parametry belki

Parametry ogólne	
$E = 210\ GPa$	$I_y = 2 \times 4.95E + 5\ cm^4$
$m_{własna} = 2 \times 292\frac{kg}{mb}$	$m_{wyposażenia} = 3422\frac{kg}{mb}$
$L_t = 15\ m$	



Rys. 4.9 a) Przyjęte obciążenie pojazdem K klasy B; b) Przemieszczenia pionowe od obciążenia pojazdem K klasy B

Wartość ugięcia wynosi 22.1 mm, a więc spełniony jest przyjęty warunek. Przyjęto także wartość tłumienia modalnego dla analizowanej konstrukcji. Typowe wartości dla tłumienia modalnego zaprezentowano w tab. 4.5.

Tab. 4.5 Wartości tłumienia modalnego [117]

Typ konstrukcji	Wartość tłumienia	
	Analiza sprężysta [%]	Analiza plastyczna [%]
Żelbet	1-2	7
Konstrukcje sprężone	0.8	5
Nitowane stalowe konstrukcje	1	7
Spawane stalowe konstrukcje	0.4	7

Przyjęto tłumienie na poziomie 0.5% tłumienia krytycznego. Dla modelu MES przyjęto tłumienie Rayleigha, które jest modelem tłumienia stosowanym w analizach dynamicznych. Ten model łączy dwa typy tłumienia: tłumienie proporcjonalne do masy i tłumienie proporcjonalne do sztywności i wyraża się to wzorem (4.25):

$$C = \alpha[M] + \beta[K] \qquad (4.25)$$

gdzie:

- C – macierz tłumienia,

- M – macierz masy,
- K – macierz sztywności,
- α i β – są współczynnikami tłumienia Rayleigha.

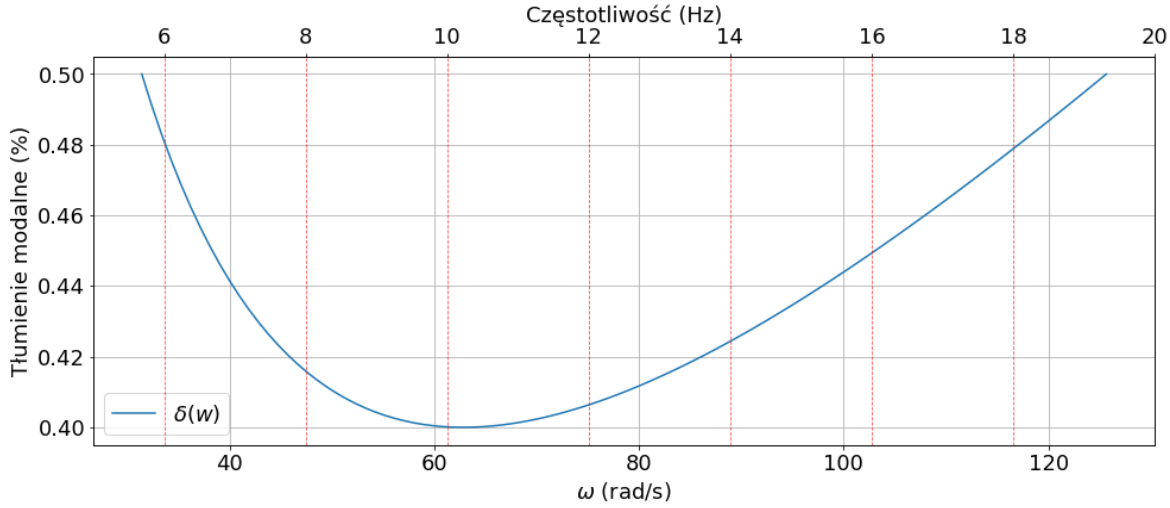
Model tłumienia Rayleigha pozwala na bardziej elastyczne i dokładne modelowanie tłumienia w systemach dynamicznych, ponieważ uwzględnia zarówno tłumienie zależne od prędkości (proporcjonalne do masy), jak i tłumienie zależne od odkształceń (proporcjonalne do sztywności).
Przyjęto wartość tłumienia modalnego na poziomie 0.5% oraz dla modelu MES określono 2 pierwsze częstotliwości własne na poziomie 5 Hz i 20 Hz.
Określono współczynnik tłumienia:

$$\alpha = 2\omega_1\omega_2 \frac{\varepsilon_1\omega_2 - \varepsilon_2\omega_1}{\omega_2^2 - \omega_1^2} \text{ oraz } \beta = 2 \frac{\varepsilon_2\omega_2 - \varepsilon_1\omega_1}{\omega_2^2 - \omega_1^2}$$

Jeśli $\varepsilon_1 = \varepsilon_2 = 0.5\ \%$ oraz $\omega_i = 2\pi f_i$, uzyskamy następujące wartości:

$$A = 4\pi\varepsilon \frac{f_1 f_2}{f_1 + f_2} = 0.2513 \text{ oraz } B = \varepsilon \frac{1}{\pi(f_1 + f_2)} = 0.0000636$$

Na rys. 4.10 przedstawiono przyjęte wartości tłumienia modalnego dla częstotliwości.



Rys. 4.10 Wartości tłumienia modalnego modelu MES wg. teorii Rayleigha

4.3.3 Modelowanie nierówności nawierzchni

Nierówności nawierzchni drogowej wpływają na dynamikę pojazdu w czasie jazdy. Te nierówności podłoża mogą prowadzić do różnych skutków, począwszy od prostego drżenia kierownicy, aż po znaczące obciążenia dla układu zawieszenia pojazdu. Aby scharakteryzować i sklasyfikować nierówności nawierzchni drogowej, przyjęto międzynarodową normę ISO 8608 [118]. Kluczowym parametrem wykorzystywanym w tej normie do opisu nierówności jest widmowa gęstość mocy (ang. *Power Spectral Density*, PSD) odchylenia pionowej współrzędnej profilu drogi, będąca narzędziem matematycznym, które pozwala analizować i opisywać nierówności drogowe w zakresie różnych

częstotliwości. Norma ISO 8608 definiuje osiem głównych klas nierówności drogowych, symbolizowane literami od A do H. Klasa A odnosi się do nawierzchni o wyjątkowo gładkiej powierzchni, idealnej dla komfortu podróżujących. Z kolei klasa H reprezentuje powierzchnie wyjątkowo nierówne. Dla każdej klasy określono zakresy wartości widmowej gęstości mocy.
Istotna jest również sama metoda tworzenia profilu nierówności drogi. Zgodnie z literaturą [119], istnieje kilka metod stosowanych przy generowaniu wspomnianego profilu:

- metody oparte na rzeczywistych pomiarach,
- metody oparte na modelach matematycznych,
- metody losowe.

W rozprawie przyjęto do generowania metody losowe, polegające na tworzeniu profilu drogi poprzez generowanie losowych wartości, które są następnie filtrowane i sprawdzane pod kątem wybranej metryki nierówności nawierzchni. Najlepiej, aby profil drogi był elementem stacjonarnego procesu losowego, który można zdefiniować jako sygnał, którego właściwości statystyczne nie zmieniają się w czasie, a w przypadku drogi, nie zmieniają się na długości profilu drogi. Koncepcja, zastosowania oraz implikacje jednorodności i izotropii są analizowane i wyjaśniane artykule [120]. W dalszej części pracy przyjęto za wzór pierwszą metodę określoną w ISO 8608, która jest zawarta w załączniku B do obowiązującego Eurokodu [111], i z tego względu jest dla projektantów konstrukcji mostowych wiążąca.
Profil drogi wyraża się jako suma funkcji cosinus, gdzie każdy wyraz reprezentuje nierówność o odpowiedniej amplitudzie oraz częstotliwości. W celu wyznaczenia wartości amplitud nierówności skorzystano ze wzorów:

$$r(x) = \sum_{i=1}^{N} \alpha_i \cos\left(2\pi n_i x + \varphi_i\right) \tag{4.26}$$

$$\alpha_i^2 = 4G_d(n_i)\Delta n, \;\; \Delta n = \frac{n_u - n_l}{N} \tag{4.27}$$

$$G_d(n_i) = G_d(n_0)\left(\frac{n_i}{n_0}\right)^{-2}, \;\; n_i = n_l + i\Delta n \tag{4.28}$$

gdzie:

- α_i [m] – amplituda i-tej przestrzennej nierówności nawierzchni,
- n_i [cykl/m] – i-ta częstotliwość przestrzenna nierówności,
- n_u [cykl/m] – założona górna granica częstotliwości przestrzenna nierówności,
- n_l [cykl/m] – założona dolna granica częstotliwości przestrzenna nierówności,
- x [m] – współrzędna drogi,
- φ_i [rad] – losowa wartość przesunięcia fazowego dla i-tej składowej nierówności,
- N [-] – liczba członów wyrażenia,
- $G_d(n_0)$ [m^3] – współczynnik nierówności zależny od klasy nierówności nawierzchni,

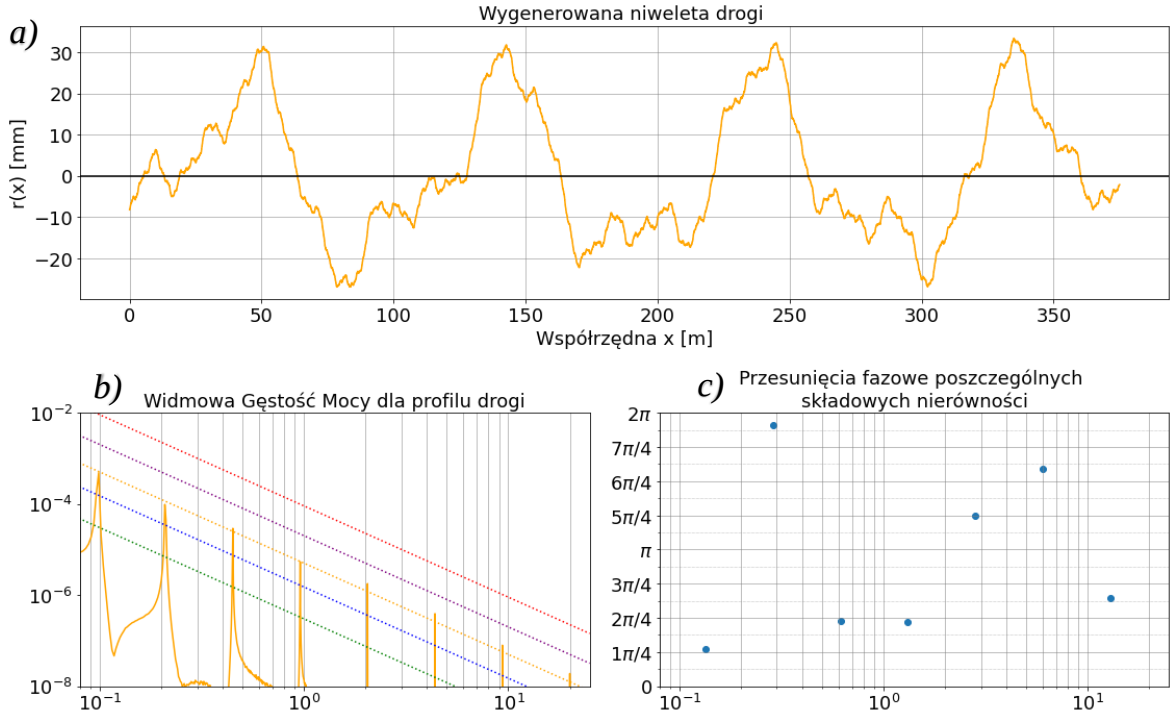
- n_0 [cykl/m] – podstawowa częstotliwość przestrzenna nierówności.

Tab. 4.6 Wartości współczynnika nierówności w zależności od klasy nawierzchni [111]

Klasa drogi	Stopień nierówności			
	Jakość nawierzchni	$G_d(n_0)^{*)}[10^{-6}\mathrm{m}^3]$		
		Granica dolna	Średnia geometryczna	Granica górna
A	bardzo dobra	-	16	32
B	dobra	32	64	128
C	średnia	128	256	512
D	niska	512	1024	2048
E	bardzo zła	2048	4096	8192
*) $n_0 = 0.1\ cykl/m$				

W załączniku B.4.2 przedstawiono autorski program do generowania profilu nierówności drogi.

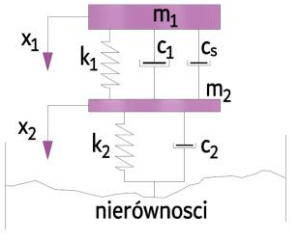
Przyjęto również założenie, że nawierzchnia drogi na obiekcie jest nawierzchnią o nierównościach klasy C. Na podstawie stworzonego algorytmu (Załącznik B.4.2.2) wygenerowano funkcję nierówności nawierzchni na podstawie wzoru $r(x) = \sum_{i=1}^{N} \alpha_i \cos(2\pi n_i x + \varphi_i)$ (4.26). Profil nierówności drogi zaprezentowano na rys. 4.11a. Analizowany most znajduje się pomiędzy 125, a 140 metrem rzędnej długości drogi. Dodatkowo przeprowadzono analizę widmowej gęstości mocy przemieszczeń pionowych (rys. 4.11b) oraz zaprezentowano dobrane przesunięcia fazowe poszczególnych nierówności (rys. 4.11c).



Rys. 4.11 a) Wygenerowana funkcja nierówności drogi;
b) Widmowa Gęstość Mocy dla profilu drogi;
c) Przesunięcia fazowe poszczególnych składowych.

4.3.4 Modelowanie zawieszenia pojazdów

W podpunkcie 4.3.4 przedstawiono mechanizm analizy oraz diagnostyki zawieszenia pojazdów. Na podstawie przedstawionych w tym podpunkcie metod zostaną dobrane parametry zawieszenia pojazdów referencyjnych w podpunkcie 4.3.5. W metodach diagnozowania zawieszenia wykorzystuje się tzw. ćwiartkowy model pojazdu. Jest to teoretyczna konstrukcja, obejmująca masę resorowaną oraz nieresorowaną pojazdu. Model ten charakteryzuje się sprężystością, sztywnością i tłumieniem zawieszenia oraz sztywnością i tłumieniem kół. Wszelkie elementy układu oraz działające między nimi siły tworzą układ pionowy, który został przedstawiony na rys. 4.12.



Rys. 4.12 Model ćwiartkowy pojazdu

gdzie:

- m_1 – masa nadwozia,
- m_2 – masa zawieszenia koła,
- k_1 – sztywność sprężyny,
- k_2 – sztywność opony,
- c_1 – tłumienie amortyzatora,
- c_s – tłumienie zawieszenia koła (tłumienie na ruchomych elementach zawieszenia np. z uwagi na tarcie w przegubach i połączeniach),
- c_2 – tłumienie koła.

Wymuszenie pracy zawieszenia może się odbywać w warunkach użytkowych, czyli w trakcie ruchu pojazdu lub w warunkach laboratoryjnych, co zostało przedstawione na rys. 4.13.

a) 

b) 

Rys. 4.13 a) Obciążenie elementów zawieszenia w warunkach naturalnych [W 4.1];
b) Obciążenie elementów zawieszenia w warunkach laboratoryjnych. [W 4.2]

Równania wykorzystywane do analizy modelu ćwiartkowego pojazdu są analogiczne do równań wykorzystanych w podpunkcie 0 i można je przedstawić w postaci (4.29) oraz (4.30):

$$m_1\ddot{x_1} + c_1(\dot{x_1} - \dot{x_2}) + k_1(x_1 - x_2) = 0 \quad (4.29)$$

$$m_2\ddot{x_2} + c_1(\dot{x_2} - \dot{x_1}) + c_2(\dot{x_2} - \dot{R}) + k_1(x_2 - x_1) + k_2(x_2 - R) = 0 \quad (4.30)$$

gdzie:

- R – to funkcja nierówności nawierzchni.

W pierwszej kolejności należy rozwiązać zagadnienie własne i określić częstotliwości własne układu. Równania w takim razie przyjmą następującą formę w postaci (4.31) oraz (4.32):

$$m_1\ddot{x_1} + c_1(\dot{x_1} - \dot{x_2}) + k_1(x_1 - x_2) = 0 \quad (4.31)$$

$$m_2\ddot{x_2} + c_1(\dot{x_2} - \dot{x_1}) + c_2\dot{x_2} + k_1(x_2 - x_1) + k_2x_2 = 0 \quad (4.32)$$

Powyższe równania można przedstawić w postaci macierzowej:

$$\begin{bmatrix} m_1 & 0 \\ 0 & m_2 \end{bmatrix}\begin{bmatrix} \ddot{x_1} \\ \ddot{x_2} \end{bmatrix} + \begin{bmatrix} c_1 & -c_1 \\ -c_1 & c_1 + c_2 \end{bmatrix}\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} + \begin{bmatrix} k_1 & -k_1 \\ -k_1 & k_1 + k_2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0$$

Rozwiązania przyjmą postać:

$$x_1(t) = X_1e^{i\omega t} \text{ oraz } x_2(t) = X_2e^{i\omega t}$$

$$\dot{x}_1(t) = i\omega X_1e^{i\omega t} \text{ oraz } \dot{x}_2(t) = i\omega X_2e^{i\omega t}$$

$$\ddot{x}_1(t) = -\omega^2 X_1e^{i\omega t} \text{ oraz } \ddot{x}_2(t) = -\omega^2 X_2e^{i\omega t}$$

gdzie:

- ω – częstość kołowa drgań własnych,
- X_1 i X_2 - amplitudy.

Podstawiając przyjęte rozwiązania do równań ruchu i dzieląc przez $e^{i\omega t}$ otrzymamy:

$$-\omega^2 m_1X_1 + i\omega c_1(X_1 - X_2) + k_1(X_1 - X_2) = 0 \quad (4.33)$$

$$-\omega^2 m_2X_2 + i\omega(c_1(X_2 - X_1) + c_2X_2) + k_1(X_2 - X_1) + k_2X_2 = 0 \quad (4.34)$$

Następnie układ równań można zapisać w formie macierzowej:

$$\begin{bmatrix} -\omega^2 m_1 + i\omega c_1 + k_1 & -i\omega c_1 - k_1 \\ -i\omega c_1 - k_1 & -\omega^2 m_2 + i\omega(c_1 + c_2) + k_1 + k_2 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = 0$$

W celu rozwiązania układu należy wyznaczyć wyznacznik, który przyjmie postać:

$$(-\omega^2 m_1 + i\omega c_1 + k_1)(-\omega^2 m_2 + i\omega(c_1 + c_2) + k_1 + k_2) - (i\omega c_1 - k_1)^2 = 0$$

Powyższe równanie można podzielić na części urojone i rzeczywiste:

$$i\omega(4c_1k_1 + c_2k_1 + c_1k_2 - c_1m_1\omega^2 - c_2m_1\omega^2 - c_1m_2\omega^2) = 0$$

$$m_1m_2\omega^4 - (c_1c_2 + k_1m_1 + k_1m_2 + k_2m_1)\omega^2 + k_1k_2 = 0$$

Ostatecznie równanie kwadratowe będzie miało postać:

$$A\omega^4 + B\omega^2 + C = 0$$

gdzie, A, B i C są współczynnikami zależnymi od m_1, m_2, c_1, c_2, k_1, k_2. Rozwiązanie równania kwadratowego da dwie wartości, które będą częstościami drgań własnych układu. Rozwiązanie równania dla części rzeczywistej:

$$\omega^2 = \frac{B \pm \sqrt{B^2 - 4AC}}{2A}$$

Rozwiązanie tego równania kwadratowego można uzyskać stosując standardowe metody algebraiczne lub numeryczne, np. za pomocą funkcji w programach matematycznych takich jak MATLAB, Mathematica lub za pomocą bibliotek numerycznych w Pythonie (np. numpy).

Na podstawie [121] przyjęto parametry oceny jakości amortyzatorów samochodów. Ważnym zagadnieniem jest analiza zawieszenia pod kątem przesunięcia fazowego oraz przylegania koła do podłoża. Kąt przesunięcia fazowego jest miarą różnicy fazy między dwoma drganiami harmonicznymi, które mają tę samą częstotliwość, ale mogą być przesunięte względem siebie w czasie. W kontekście układów mechanicznych, takich jak zawieszenie, kąt przesunięcia fazowego może wskazywać na opóźnienie odpowiedzi układu (np. masy) na wymuszenie (np. drgania nawierzchni).
Przyleganie koła do nawierzchni jest kluczowe dla bezpieczeństwa i komfortu jazdy. Funkcja przylegania określa, jak dobrze koło styka się z nawierzchnią w różnych warunkach dynamicznych. Przyleganie może być opisane jako minimalny procent nacisku statycznego koła w trakcie ruchu po nierównej nawierzchni. Przyleganie określa jak mocno koło jest dociskane do podłoża. Ma to niebagatelny wpływ na zdolność prowadzenia i sterowania pojazdem.

W artykule [121] podano również wymogi Europejskiego Stowarzyszenia Producentów Amortyzatorów, które opracowało metodę oceny efektywności tłumienia. Zasada polega na procentowym określaniu siły przylegania koła do podłoża. Ocena skuteczności tłumienia amortyzatora określa wskaźnik EUSAMA:

$$WE = \frac{W_{min}}{W_{st}} * 100\%$$

gdzie:

- W_{min} – zmierzona minimalna siła dynamiczna przylegania opony do podłoża.
- W_{st} – statyczna siła przylegania opony do podłoża (spoczynkowa).

W trakcie badania koło samochodu spoczywa na nieruchomej płycie urządzenia wymuszającego. Dokonywany jest wówczas pomiar statyczny. Następnie uruchamiany jest układ wymuszający. Płyta osiąga drgania o amplitudzie 4-8 mm w zakresie około 25 Hz. Przyjęte kryterium oceny są następujące:

- $WE = 0 - 20$ % - zły stan techniczny,
- $WE = 21 - 40$ % - dopuszczalna wartość tłumienia,
- $WE = 41 - 60$ % - dobra wartość tłumienia,

- $WE > 60$ % - bardzo dobra wartość tłumienia.

Dodatkowo w artykule [121] przedstawiono metodę, pod nazwą EUSAMA PLUS, która polega na pomiarze płytowym o zmiennej częstotliwości drgań wymuszających. Proces składa się z 2 faz:

1. Rozgrzewanie płynu w amortyzatorze celem uzyskania właściwej lepkości.
2. Proces pomiarowy ze zmieniającą się częstotliwością co 1 Hz w sposób malejący od 30 Hz do 8 Hz.

Szczegółowo analizowany jest przedział 13 do 18 Hz gdzie oczekiwany jest rezonans masy nieresorowanej. Dodatkowo firma Hunter Engineering Company zaproponowała uwzględnienie dodatkowej mierzalnej wartości – kąt przesunięcia fazowego pomiędzy sygnałami przemieszczenia płyty i mas nieresorowanych. Wartość kąta przemieszczenia fazowego jest wielkością charakteryzującą wielkość tłumienia. Kiedy zawieszenie samochodu ma odpowiednią wartość tłumienia to minimalny kąt przesunięcia fazowego pomiędzy częstotliwościami rezonansowymi masy resorowanej i nieresorowanej osiągnie wartość powyżej 90^o.

4.3.5 Przyjęcie referencyjnych modeli pojazdów

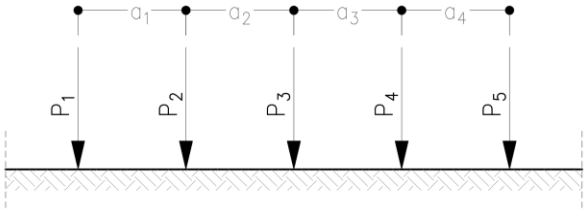
Do analiz obliczeniowych przyjęto cztery modele pojazdów drogowych. Na rys. 4.14 przedstawiono pojazd ciężarowy o 2 osiach, budowlane wywrotki o 3 lub 4 osiach, oraz 2-osiowy ciągnik siodłowy wraz z 3-osiową naczepą.

a) b)

c) d)

Rys. 4.14 Przyjęte do weryfikacji pojazdy: a) Mercedes Atego 4x2 [W 4.3];
b) Mercedes Arocs 6x4 [W 4.4];
c) Mercedes Arocs 8x4/4 [W 4.5];
d) Mercedes Actros 4x2 + Wywrotka budowlana [W 4.6]

Na podstawie internetowego konfiguratora pojazdów producenta [W 4.7] przyjęto oraz przedstawiono w tab. 4.7 rozstawy oraz dopuszczalne naciski na osie, natomiast na rys. 4.15 przedstawiono przyjęte oznaczenia.



Rys. 4.15 Schemat podstawowego modelu pojazdu

Tab. 4.7 Przyjęte wartości dla podstawowych modeli pojazdu

Wariant pojazdu	Rozstawy pomiędzy osiami				Przyjęte naciski na poszczególne osie.				
	a_1	a_2	a_3	a_4	P_1	P_2	P_3	P_4	P_5
	[m]	[m]	[m]	[m]	[kN]	[kN]	[kN]	[kN]	[kN]
Pojazd 2-osiowy	4.76	-	-	-	51	105	-	-	-
Pojazd 3-osiowy	4.8	1.3	-	-	75	95	95	-	-
Pojazd 4-osiowy	1.6	4.75	1.31	-	75	75	130	130	-
Pojazd 5-osiowy	3.55	4.88	1.31	1.31	80	130	80	80	80

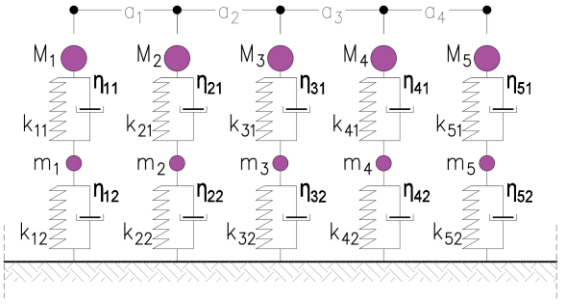
Wartości w tab. 4.7 dotyczą podstawowego modelu pojazdu składającego się z sił skupionych. Przy doborze modelu pojazdu z punktami masowymi należy dokonać dokładniejszej analizy. W pierwszej kolejności dobrano parametry modelu szczegółowego. Przyjęto, że masa resorowana stanowi 95 % dopuszczalnej, maksymalnej masy pojazdu. W skład masy nieresorowanej wchodzą elementy takie jak opona wraz z felgą, układ hamulcowy oraz wahacze. Przy modelowaniu parametrów zawieszenia dla pojazdów modelu szczegółowego przyjęto kryteria podane w podpunkcie 4.3.4:

- Kąt przesunięcia fazowego, który powinien wynosić 90^o dla częstotliwości rezonansowych masy resorowanej i nieresorowanej.
- Bardzo dobra jakość amortyzatora, a więc wartość wskaźnika EUSAMA powinna być większa niż 60 %.

Dodatkowo uwzględniono następujące założenia dotyczące zawieszenia na podstawie [12]:

- Częstotliwość rezonansowa masy resorowanej powinna mieścić się w przedziale 1-3 Hz.
- Częstotliwość rezonansowa masy nieresorowanej powinna mieścić się w przedziale 8-18 Hz.

Każdą oś rozpatrywano niezależnie. Przyjęto, że pomimo różnych nacisków, charakterystyka zawieszenia wszystkich osi pojazdów będzie identyczna. Z uwagi na występowanie wielu osi o różnych naciskach, pełną procedurę doboru parametrów przedstawiono dla drugiej osi pojazdu 5-osiowego o nacisku 130 kN. Na rysunku rys. 4.16 przedstawiono schemat modelu szczegółowego, natomiast w tab. 4.8, tab. 4.9 oraz w tab. 4.10 przedstawiono przyjęte wartości.



Rys. 4.16 Schemat szczegółowego modelu pojazdu i modelu MES pojazdu

Tab. 4.8 Przyjęte masy dla poszczególnych osi szczegółowych modeli pojazdów i modeli MES

Podstawowe parametry pojazdu					
Wariant pojazdu	Przyjęte naciski techniczne na poszczególne osie.				
	M_1 / m_1	M_2 / m_2	M_3 / m_3	M_4 / m_4	M_5 / m_5
	[t] / [t]	[t] / [t]	[t] / [t]	[t] / [t]	[t] / [t]
Pojazd 2-osiowy	4.845 / 0.255	9.975 / 0.525	-	-	-
Pojazd 3-osiowy	7.125 / 0.375	9.025 / 0.475	9.025 / 0.475	-	-
Pojazd 4-osiowy	7.125 / 0.375	7.125 / 0.375	12.35 / 0.65	12.35 / 0.65	-
Pojazd 5-osiowy	7.6 / 0.4	12.35 / 0.65	7.6 / 0.4	7.6 / 0.4	7.6 / 0.4

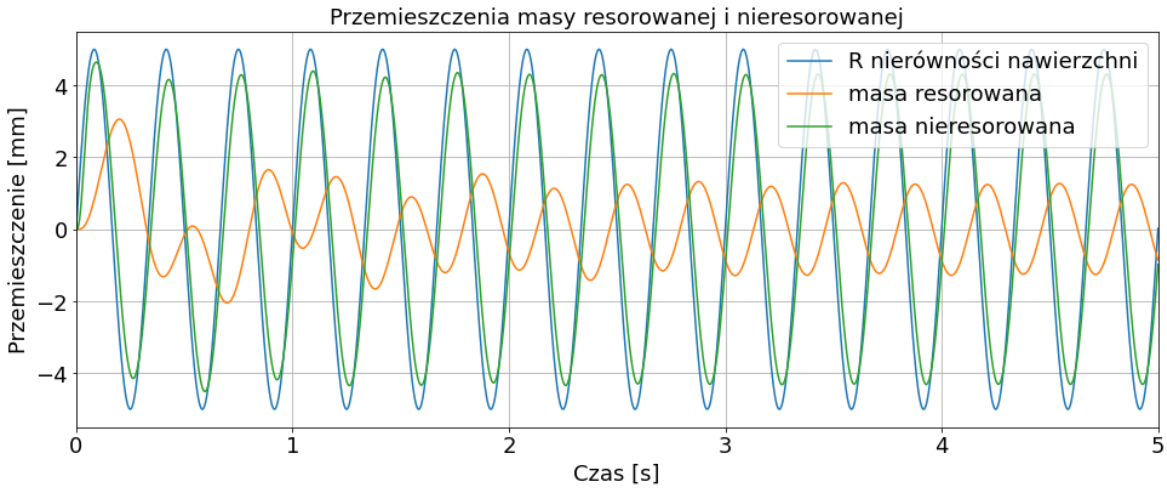
Tab. 4.9 Przyjęte sztywności oraz tłumienia zawieszenia dla poszczególnych osi szczegółowych modeli pojazdów i modeli MES

Wariant pojazdu	Sztywność zawieszenia					Wartość tłumienia zawieszenia				
	k_{11}	k_{21}	k_{31}	k_{41}	k_{51}	η_{11}	η_{21}	η_{31}	η_{41}	η_{51}
	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$
Pojazd 2-osiowy	1530	3150	-	-	-	0.04	0.08	-	-	-
Pojazd 3-osiowy	2250	2850	2850	-	-	0.05	0.07	0.07	-	-
Pojazd 4-osiowy	2250	2250	3900	3900	-	0.05	0.05	0.10	0.10	-
Pojazd 5-osiowy	2400	3900	2400	2400	2400	0.06	0.10	0.06	0.06	0.06

Tab. 4.10 Przyjęte sztywności oraz tłumienia kół (masy nieresorowane) dla poszczególnych osi szczegółowych modeli pojazdów i modeli MES

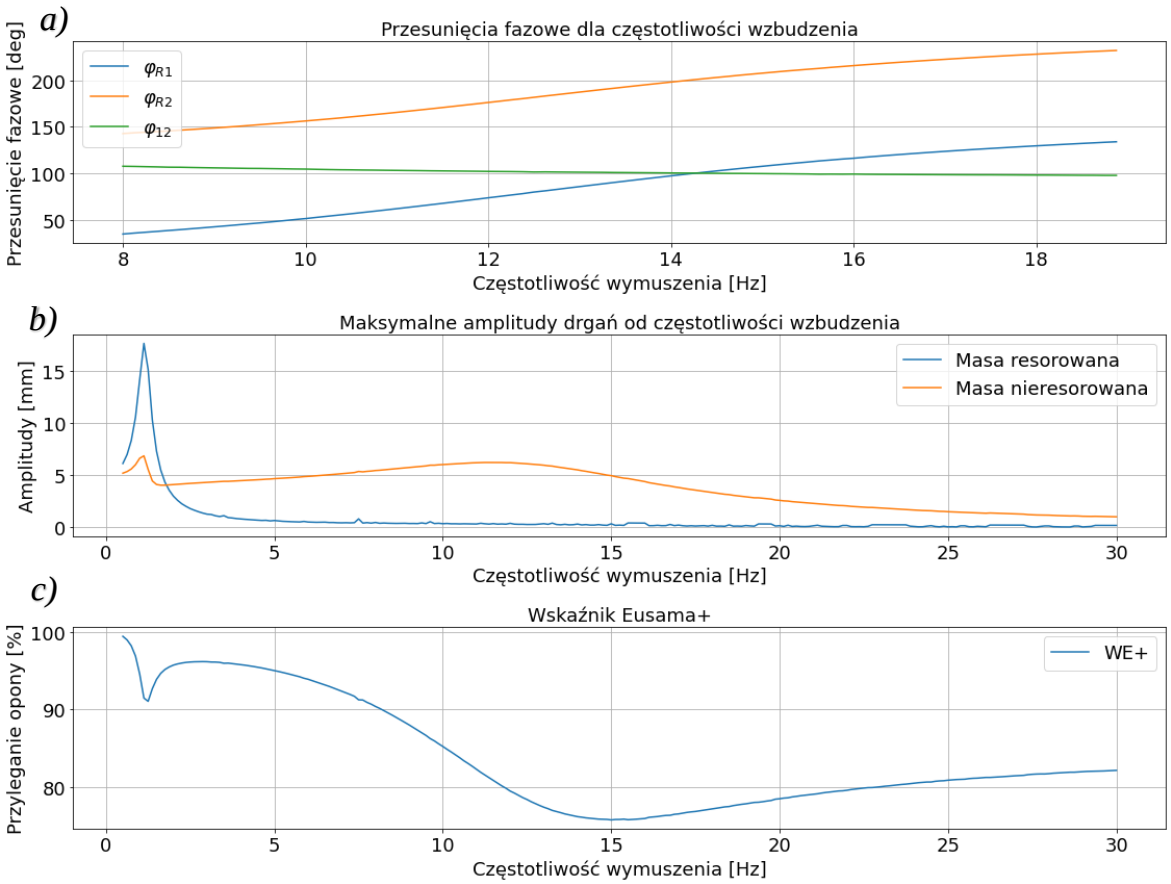
Wariant pojazdu	Sztywność zawieszenia					Wartość tłumienia zawieszenia				
	k_{12}	k_{22}	k_{32}	k_{42}	k_{52}	η_{12}	η_{22}	η_{32}	η_{42}	η_{52}
	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$	$[\frac{kN}{m \cdot s}]$
Pojazd 2-osiowy	306	630	-	-	-	15.4	31.6	-	-	-
Pojazd 3-osiowy	450	570	570	-	-	22.6	28.6	28.6	-	-
Pojazd 4-osiowy	450	450	780	780	-	22.6	22.6	39.2	39.2	-
Pojazd 5-osiowy	480	780	480	480	480	24	39.2	24	24	24

Na rys. 4.17 przedstawiono obliczone przemieszczenia drugiej osi pojazdu 5-osiowego od harmonicznego wymuszenia drganiami podłoża o amplitudzie 5 mm oraz częstotliwości 3 Hz.



Rys. 4.17 Przemieszczenia w czasie masy resorowanej i nieresorowanej drugiej osi pojazdu 5-osiowego

Następnie dokonano sprawdzenia częstotliwości rezonansowej oraz wskaźnika EUSAMA PLUS oraz przesunięcia fazowego w pełnym zakresie częstotliwości. Wyniki zaprezentowano na rys. 4.18:

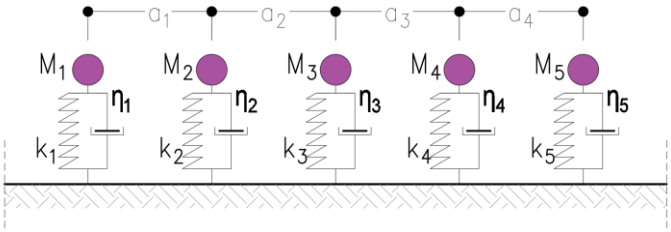


Rys. 4.18 Weryfikacja charakterystyk zawieszenia pojazdu
a) Przesunięcia fazowe dla częstotliwości wzbudzenia; b) Maksymalne amplitudy drgań od częstotliwości wzbudzenia; c) Wskaźnik Eusama+

Na podstawie wykresów można stwierdzić następujące fakty:

- Przesunięcie fazowe pomiędzy masą resorowaną oraz nieresorowaną w pełnym zakresie częstotliwości wynosi ponad 90 stopni.
- Wskaźnik przylegania opony do podłoża mieści się powyżej 75 %, co znaczy, że amortyzator charakteryzuje się bardzo dobrymi parametrami.
- Częstotliwość rezonansowa masy nieresorowanej wynosi ok. 12 Hz, czyli mieści się w oczekiwanym przedziale.
- Częstotliwość rezonansowa masy resorowanej jest poniżej granicy 1-3 Hz. Efekty od rezonansu, powodujące zmniejszenie przylegania opony do podłoża, nie przekraczają wartości dopuszczalnych.

Na podstawie dobranych parametrów dla modelu szczegółowego, dobrano charakterystyki zawieszenia do modelu rozszerzonego, tak aby zachowanie się punktów masowych z modelu rozszerzonego odzwierciedlało zachowanie się punktów masowych z modelu szczegółowego. Na rysunku rys. 4.19 przedstawiono schemat modelu rozszerzonego:



Rys. 4.19 Schemat rozszerzonego modelu pojazdu

W tab. 4.11 oraz tab. 4.12 przedstawiono przyjęte parametry pojazdów:

Tab. 4.11 Przyjęte masy dla poszczególnych osi rozszerzonych modeli pojazdów

Wariant pojazdu	Przyjęte masy przypadające na poszczególne osie.				
	M_1	M_2	M_3	M_4	M_5
	[t]	[t]	[t]	[t]	[t]
Pojazd 2-osiowy	5.1	10.5	-	-	-
Pojazd 3-osiowy	7.5	9.5	9.5	-	-
Pojazd 4-osiowy	7.5	7.5	13.0	13.0	-
Pojazd 5-osiowy	8.0	13.0	8.0	8.0	8.0

Tab. 4.12 Przyjęte sztywności oraz tłumienia zawieszenia dla poszczególnych osi rozszerzonych modeli pojazdów

Wariant pojazdu	Sztywność zawieszenia					Wartość tłumienia zawieszenia				
	k_1	k_2	k_3	k_4	k_5	η_1	η_2	η_3	η_4	η_5
	[kN/m]	[kN/m]	[kN/m]	[kN/m]	[kN/m]	[kN/m*s]	[kN/m*s]	[kN/m*s]	[kN/m*s]	[kN/m*s]
Pojazd 2-osiowy	255	525	-	-	-	14.42	29.7	-	-	-
Pojazd 3-osiowy	375	475	475	-	-	21.21	26.87	26.87	-	-
Pojazd 4-osiowy	375	375	650	650	-	21.21	21.21	36.77	36.77	-
Pojazd 5-osiowy	400	650	400	400	400	22.63	36.77	22.63	22.63	22.63

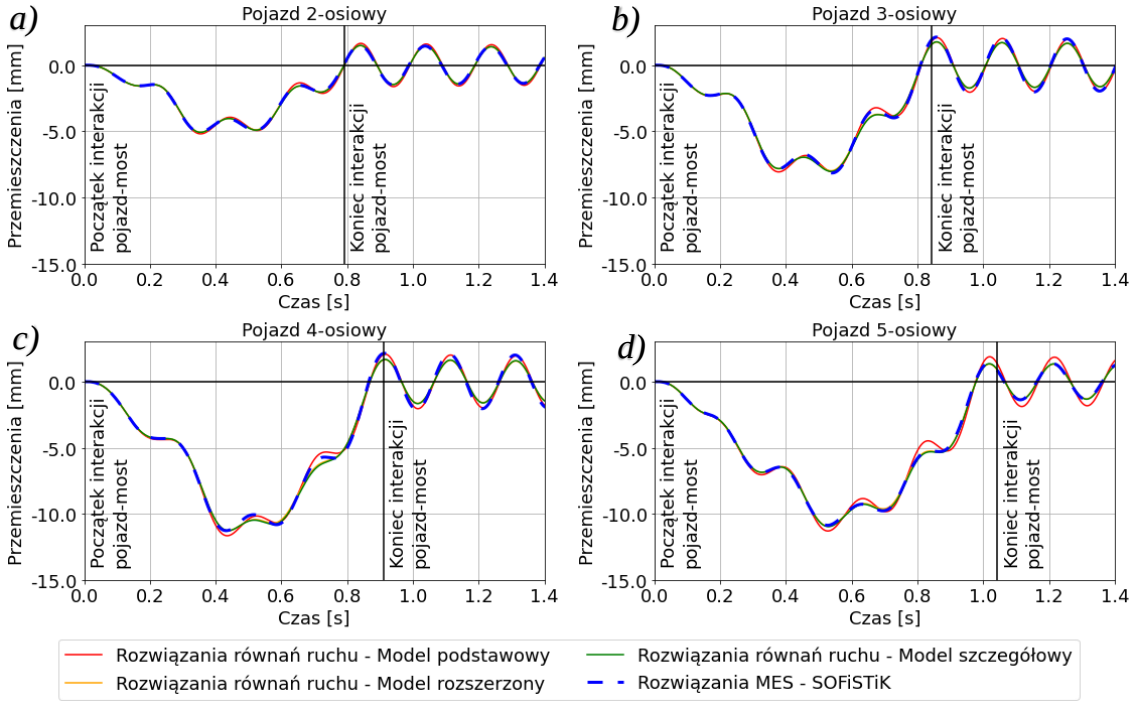
4.3.6 Walidacja za pomocą MES rozwiązań bez uwzględniania nierówności nawierzchni

Sprawdzono poprawność rozwiązania poprzez porównanie różnych modeli pojazdu.
Przyjęto, że:

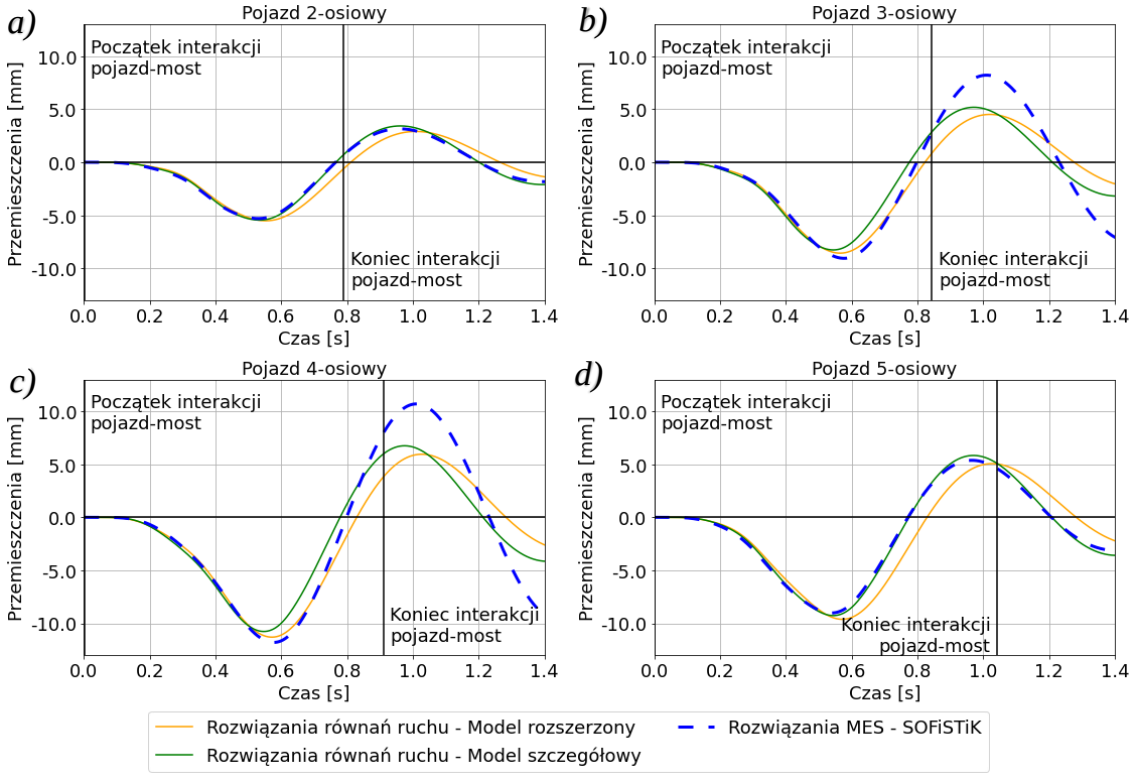
- sprawdzenie zostanie wykonane dla każdego z pojazdów referencyjnych,
- prędkość przejazdu wyniesie 25 m/s,
- model obiektu mostowego jest zgodny z podpunktem 4.3.2,
- parametry modeli pojazdów są zgodne z podpunktem 4.3.5.

Wyniki porównujące rezultaty uzyskane za pomocą programu obliczeniowego oraz równań różniczkowych przedstawiono na:

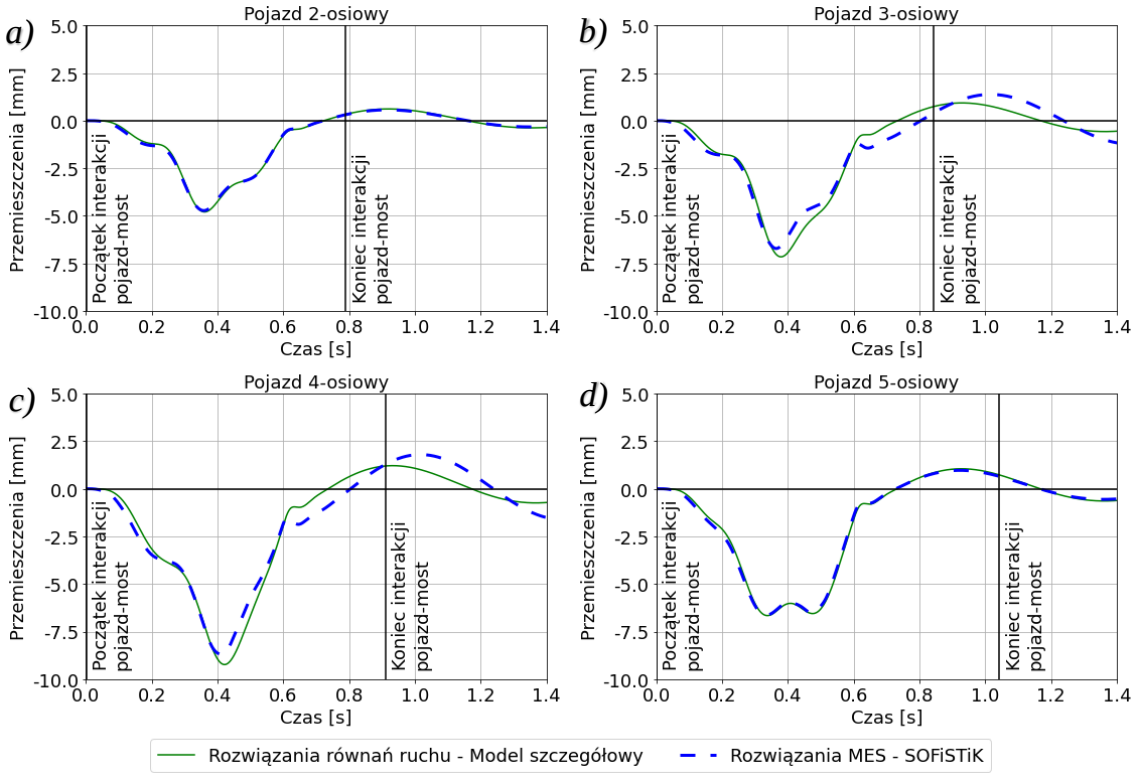
- rys. 4.20, gdzie pokazano przemieszczenia pionowe w środku rozpiętości modelowej belki, wywołane przejazdem pojazdów referencyjnych; analiza ta uwzględnia wszystkie modele obliczeniowe,
- rys. 4.21, gdzie zaprezentowano przemieszczenia masy resorowanej pierwszej osi pojazdów referencyjnych dla modeli obliczeniowych; model podstawowy nie został uwzględniony na tym wykresie, ponieważ nie zawiera masy resorowanej,
- rys. 4.22, gdzie przedstawiono przemieszczenia masy nieresorowanej pierwszej osi pojazdów referencyjnych dla różnych modeli obliczeniowych; w tym przypadku, zarówno rozwiązanie wg modelu rozszerzonego, jak i modelu podstawowego, nie zostało zaprezentowane, gdyż oba te modele nie zawierają mas nieresorowanych,
- rys. 4.23, gdzie przedstawiono różnice przemieszczenia pionowego w środku rozpiętości modelowej belki dla modeli względem modelu MES.



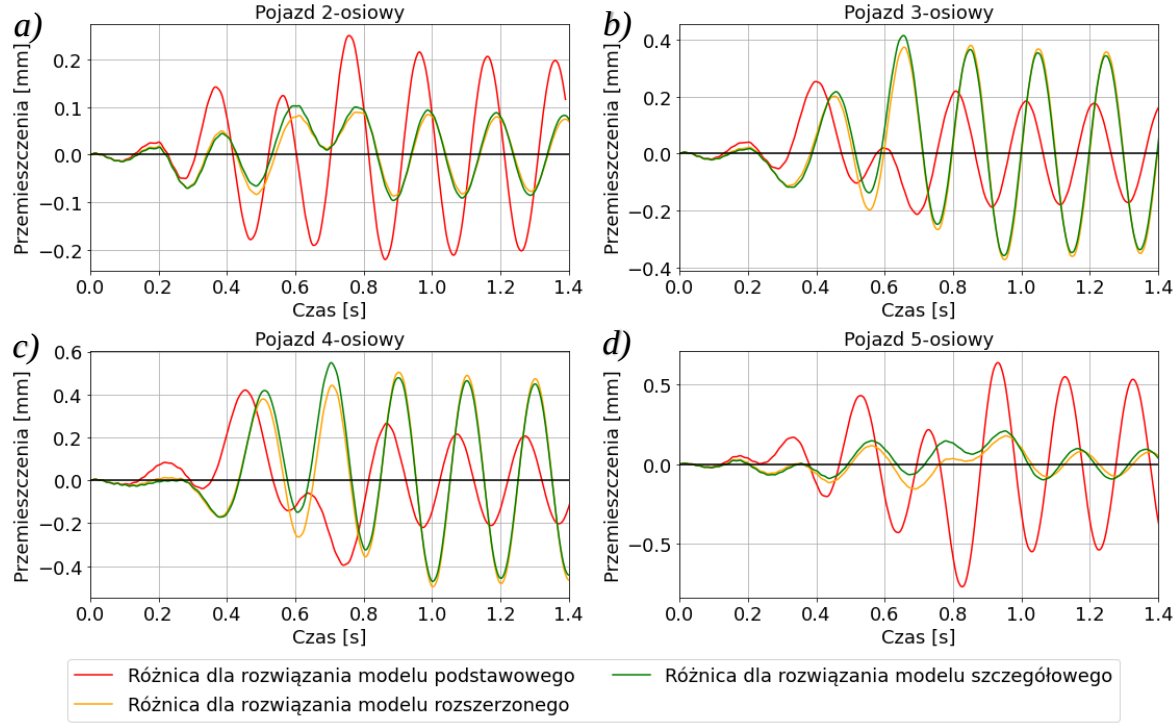
Rys. 4.20 Wykresy przemieszczeń pionowych modelowego przęsła w środku rozpiętości od przejazdów pojazdów referencyjnych dla różnych modeli obliczeniowych, odpowiednio: a) pojazd 2-osiowy, b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy



Rys. 4.21 Wykresy przemieszczeń masy resorowanej pierwszej osi pojazdów referencyjnych dla różnych modeli obliczeniowych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy



Rys. 4.22 Wykresy przemieszczeń masy nieresorowanej pierwszej osi pojazdów referencyjnych dla różnych modeli obliczeniowych, odpowiednio:
a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy



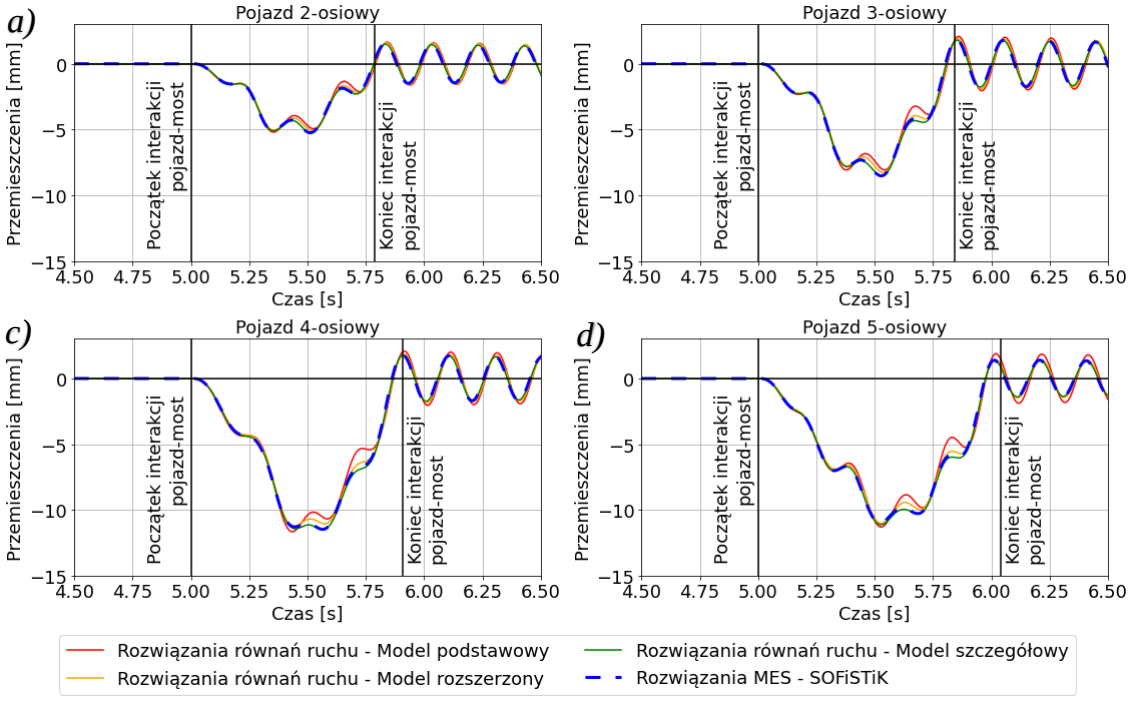
Rys. 4.23 Różnice uzyskanych rozwiązań przemieszczeń pionowych modelowego przęsła w środku rozpiętości od przejazdów pojazdów referencyjnych dla różnych modeli obliczeniowych względem modelu MES, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

Na podstawie rys. 4.23, można stwierdzić, że rozwiązania są zbieżne. Uzyskane różnice przemieszczeń, pionowych konstrukcji, w ekstremalnych przypadkach, nie przekraczają 0.6 mm. Dla celów symulacyjnych, można stwierdzić, że modele analityczne jak i MES podają akceptowalne rozwiązania.

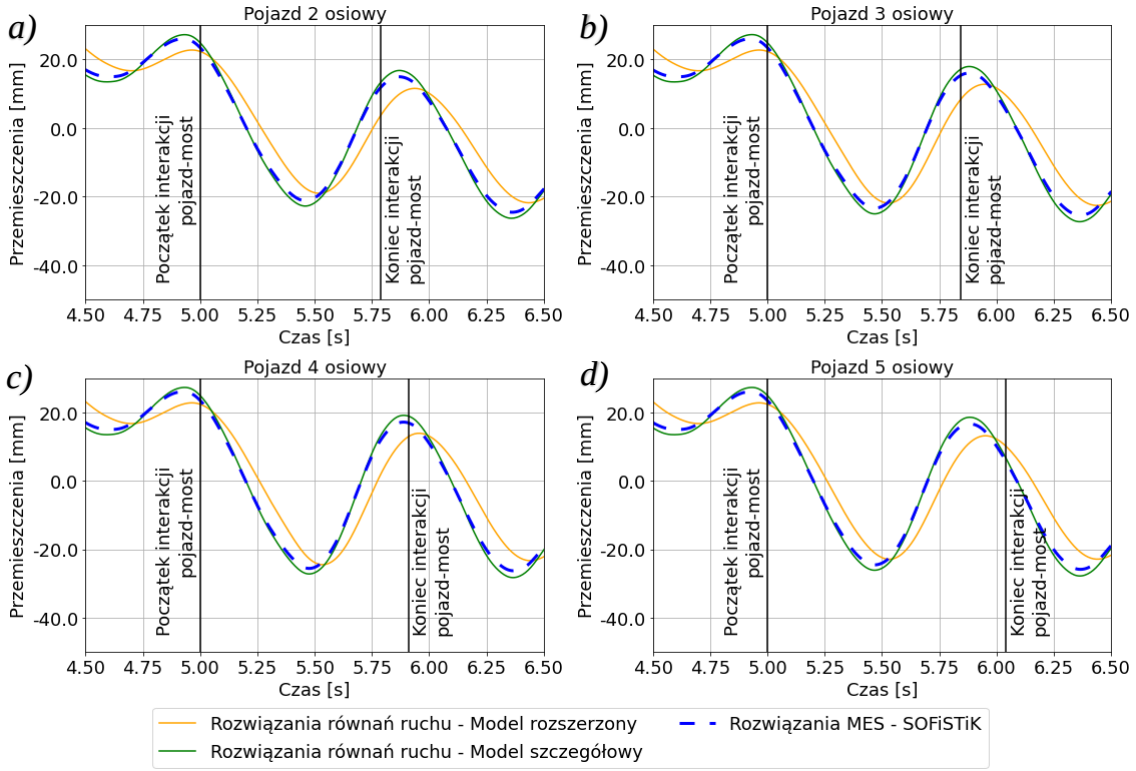
4.3.7 Walidacja za pomocą MES rozwiązań z uwzględnieniem nierówności nawierzchni

Wykonano również walidację rozwiązań z uwzględnieniem nierówności nawierzchni, co pokazano na:

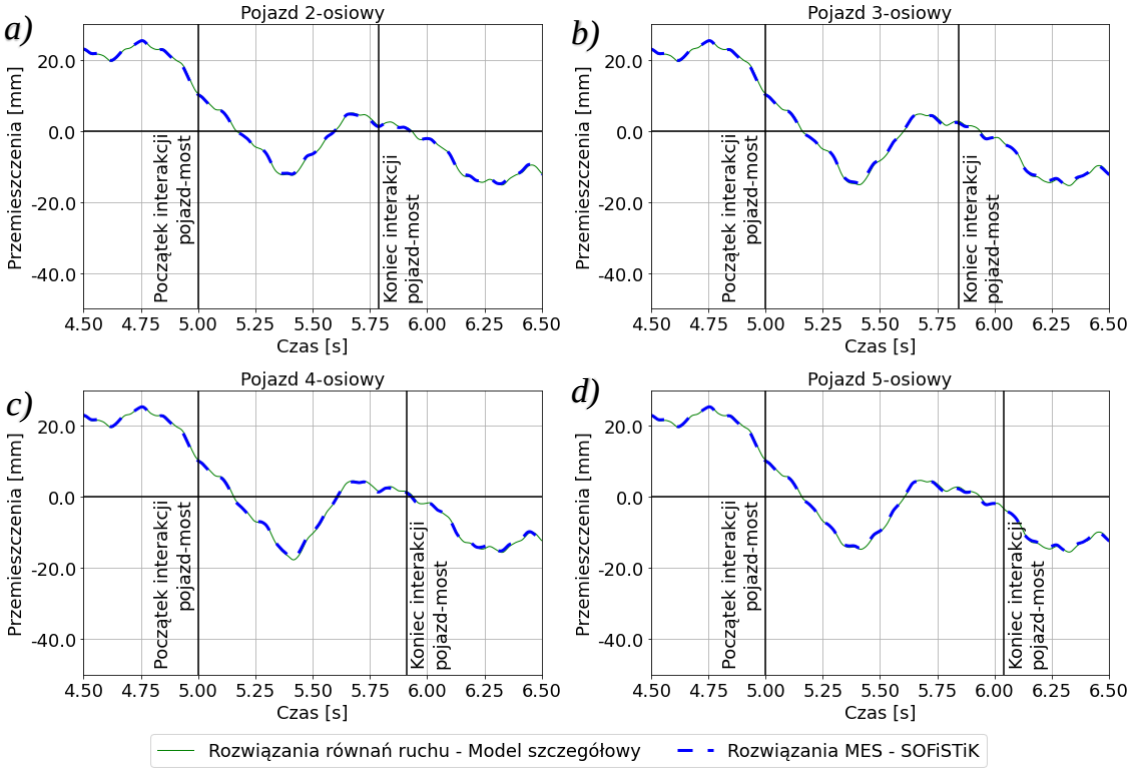
- rys. 4.24, gdzie przedstawiono przemieszczenia pionowe w środku rozpiętości belki, wywołane przejazdem pojazdów referencyjnych,
- rys. 4.25, gdzie zaprezentowano przemieszczenia masy resorowanej pierwszej osi pojazdów referencyjnych dla rozpatrywanych modeli obliczeniowych; ruch przedstawiono dla całego okresu drgania i obejmuje czas dojazdu do obiektu, jak również drgania swobodne po zjechaniu z obiektu,
- rys. 4.26, gdzie przedstawiono przemieszczenia masy nieresorowanej pierwszej osi pojazdów referencyjnych dla różnych modeli obliczeniowych,
- rys. 4.27, gdzie przedstawiono różnice przemieszczenia pionowego w środku rozpiętości modelowej belki dla modeli względem modelu MES.



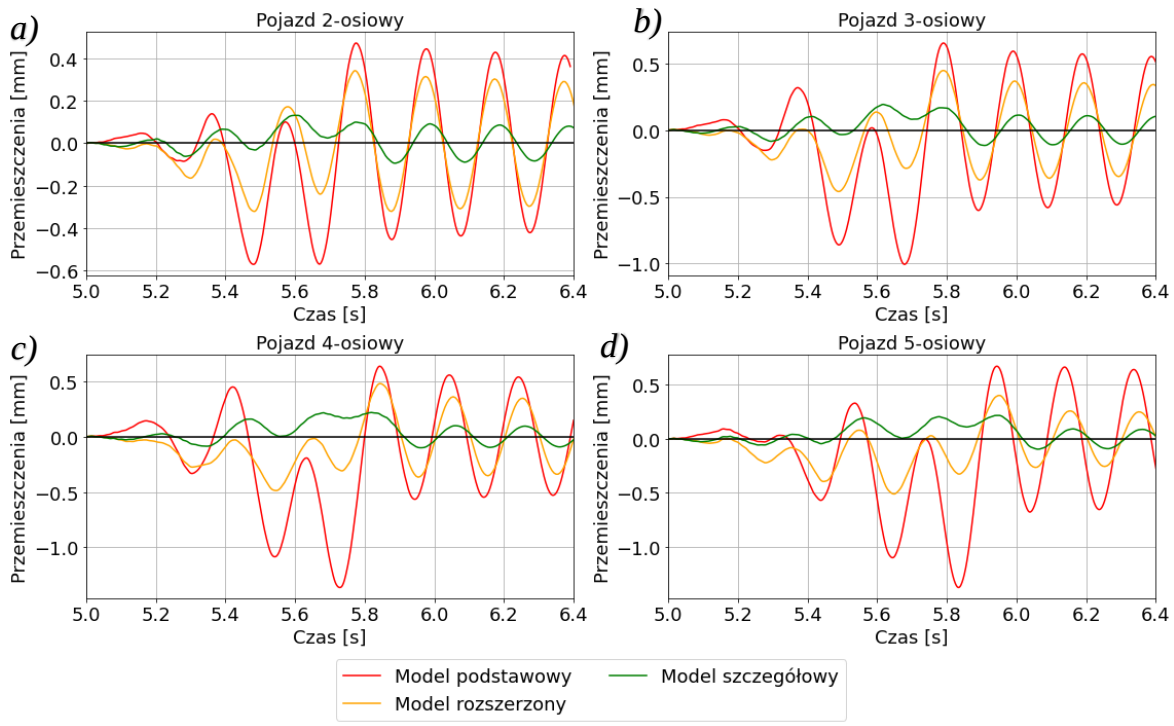
Rys. 4.24 Wykresy przemieszczeń pionowych modelowej belki w środku rozpiętości od przejazdów pojazdów referencyjnych dla różnych modeli obliczeniowych z uwzględnieniem nierówności nawierzchni, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy



Rys. 4.25 Wykresy przemieszczeń masy resorowanej pierwszej osi pojazdów referencyjnych dla różnych modeli obliczeniowych z uwzględnieniem nierówności nawierzchni, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy



Rys. 4.26 Wykresy przemieszczeń masy nieresorowanej pierwszej osi pojazdów referencyjnych dla różnych modeli obliczeniowych z uwzględnieniem nierówności nawierzchni, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy



Rys. 4.27 Różnice uzyskanych rozwiązań przemieszczeń pionowych modelowego przęsła w środku rozpiętości od przejazdów pojazdów referencyjnych dla różnych modeli obliczeniowych względem modelu MES, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

Na podstawie rys. 4.27, można stwierdzić, że rozwiązania są zbieżne. Zauważana jest jednak jakościowa różnica pomiędzy rozwiązaniami uwzględniającymi masy resorowane i nieresorowane, a tymi które traktują pojazd jako zbiór sił. Różnice uzyskane dla modeli rozszerzonych i szczegółowych nie przekraczają odpowiednio 0.25 mm i 0.5 mm, podczas gdy różnica dla modelu podstawowego, zwłaszcza w końcowej fazie interakcji pojazd-most, przekracza 1 mm. Model szczegółowy reprezentuje bardzo dużą dokładność i zgodność z rozwiązaniem uzyskanym za pomocą modelu MES.

4.4 Analiza wpływu wybranych parametrów na wyniki symulacji

4.4.1 Analizowane parametry

W prezentowanym podpunkcie przedstawiono analizę wpływu poszczególnych parametrów symulacji na odpowiedź dynamiczną konstrukcji.

Celem analizy jest:

- określenie konieczności uwzględnienia poszczególnych parametrów w proponowanym algorytmie identyfikacji obciążeń,
- dobór odpowiedniego modelu pojazdu do szkolenia algorytmu identyfikacji obciążeń,
- dobór odpowiednich cech dynamicznych konstrukcji mostowej do algorytmu identyfikacji obciążeń.

W analizie uwzględniono parametry przedstawione w tab. 4.13.

Tab. 4.13 Zakres analizy parametrów poszczególnych modeli obliczeniowych

Analizowane parametry modelu	Analizowany model pojazdu		
	Podstawowy	Rozszerzony	Szczegółowy
Prędkość pojazdu	●	●	●
Tłumienie belki	●	●	●
Masa belki	●	●	●
Rozpiętość belki przy „stałej podatności”	●	●	●
Nierówności nawierzchni	○	●	●
Tłumienie zawieszenia	○	●	●
Sztywność zawieszenia	○	●	●
● – Pełna możliwość analizy parametru dla wybranego modelu, ○ – Wybrany model nie uwzględnia analizowanego parametru; uzyskane wyniki są wyłącznie poziomem porównawczym.			

4.4.2 Wpływ prędkość przejazdu

Przeprowadzona analiza skupia się na badaniu wpływu prędkości pojazdów na ugięcie przęsła w środku rozpiętości. Rozpatrzono prędkości pojazdów mieszczące się w przedziale od 1 m/s (3,6 km/h) do 55 m/s (198 km/h), z krokiem iteracyjnym wynoszącym 0.25 m/s.

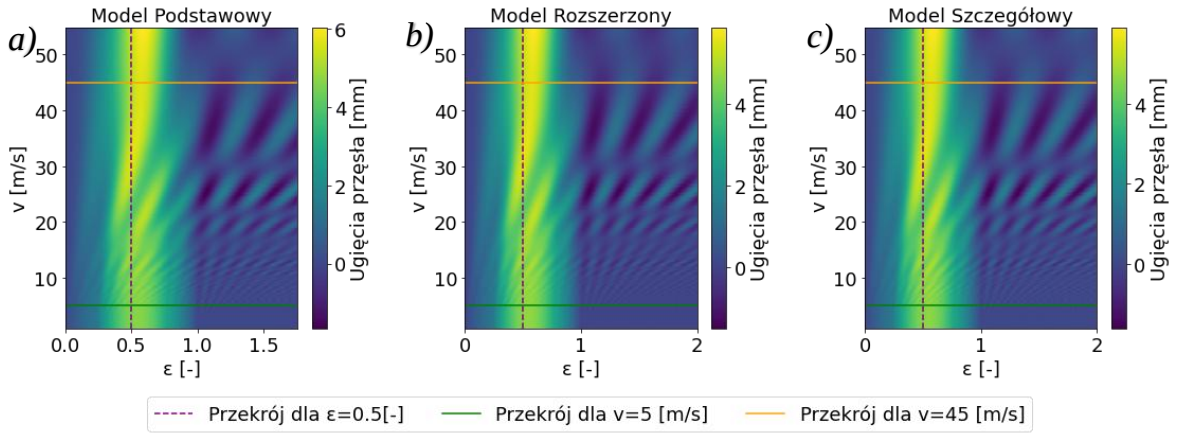
Do analizy wybrano cztery pojazdy, zgodne z podpunktem 4.3.5, reprezentujące grupy pojazdów odpowiednio dwu-, trzy-, cztero- i pięcioosiowych. Każdy pojazd został zamodelowany wg modelu podstawowego, rozszerzonego i szczegółowego.

W celu zobrazowania wyników na przykładzie pojazdu dwuosiowego przedstawiono je w postaci mapy rozkładu rys. 4.28, gdzie kolorami oznaczono wartości przemieszczeń pionowych przęsła w środku rozpiętości.

Z uwagi na różne prędkości wyniki zostały przeskalowane do bezwymiarowej postaci czasu ε, gdzie:

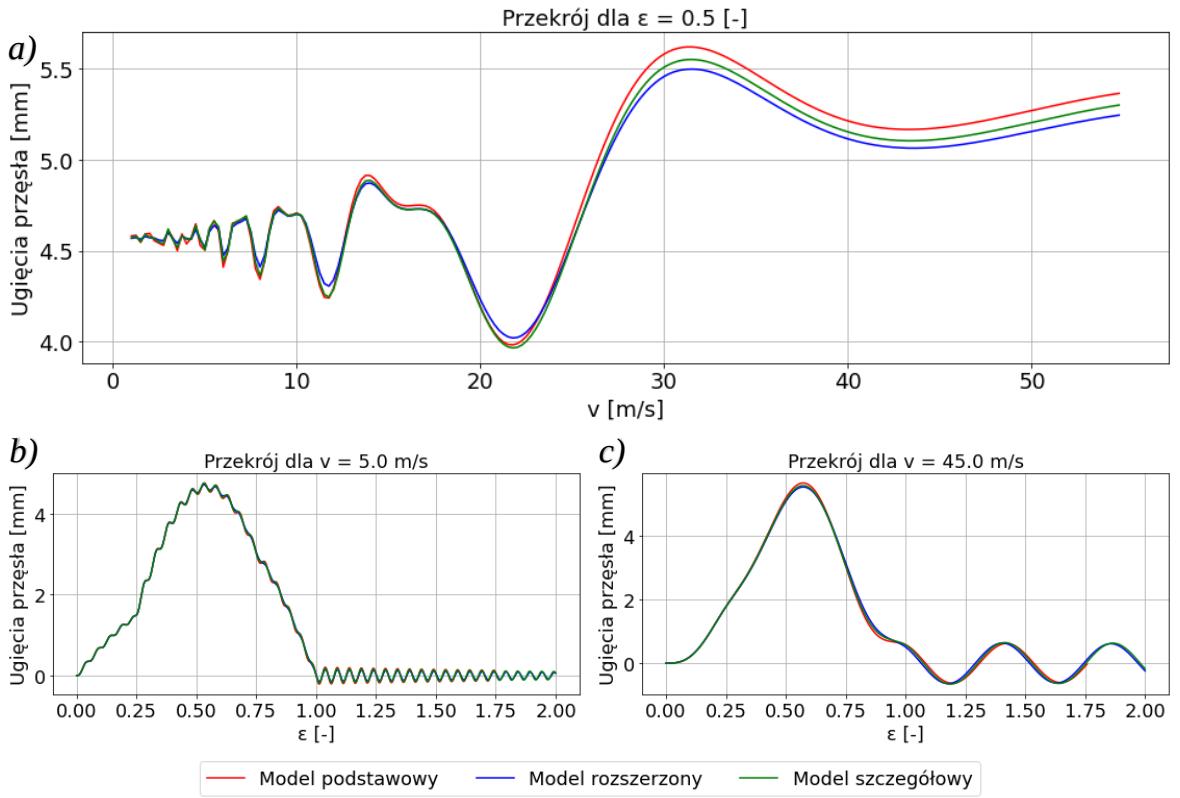
- $\varepsilon = 0$ odpowiada momentowi, w którym pierwsza oś pojazdu wjeżdża na belkę,
- $\varepsilon = 1$, gdy ostatnia oś zjeżdża z belki.

Wykresy obejmują też wartości $\varepsilon > 1$, celem zilustrowania również drgań swobodnych belki po zakończeniu ruchu pojazdu po belce.



Rys. 4.28 Mapy rozkładu ugięć przęsła w środku rozpiętości w funkcji czasu i prędkości dla modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

W celu porównania różnic pomiędzy modelami na mapie zaznaczono płaszczyzny dla wybranych prędkości oraz wybranej chwili czasowej. Na rys. 4.29 przedstawiono ugięcia dla zaznaczonych przekrojów na mapie cieplnej ugięć dla pojazdu 2-osiowego przy przyjęciu różnych modeli obliczeniowych.



Rys. 4.29 Przekroje map ugięć dla różnych modeli pojazdu 2-osiowego: a) Przekrój dla chwili czasowej ε= 0.5 b) Przekrój dla $v = 5.0\ m/s$; c) Przekrój dla $v = 45.0\ m/s$

Na podstawie przedstawionych wykresów można stwierdzić, że maksymalne ugięcia są zależne od prędkości przejazdu sił. Dodatkowo zauważono, że o ile dla przejazdu quasi-statycznego ugięcie maksymalne osiągnięto dla ustawienia pojazdu w środku rozpiętości, to dla wyższych prędkości wartość maksymalna cechuje się pewnym opóźnieniem.

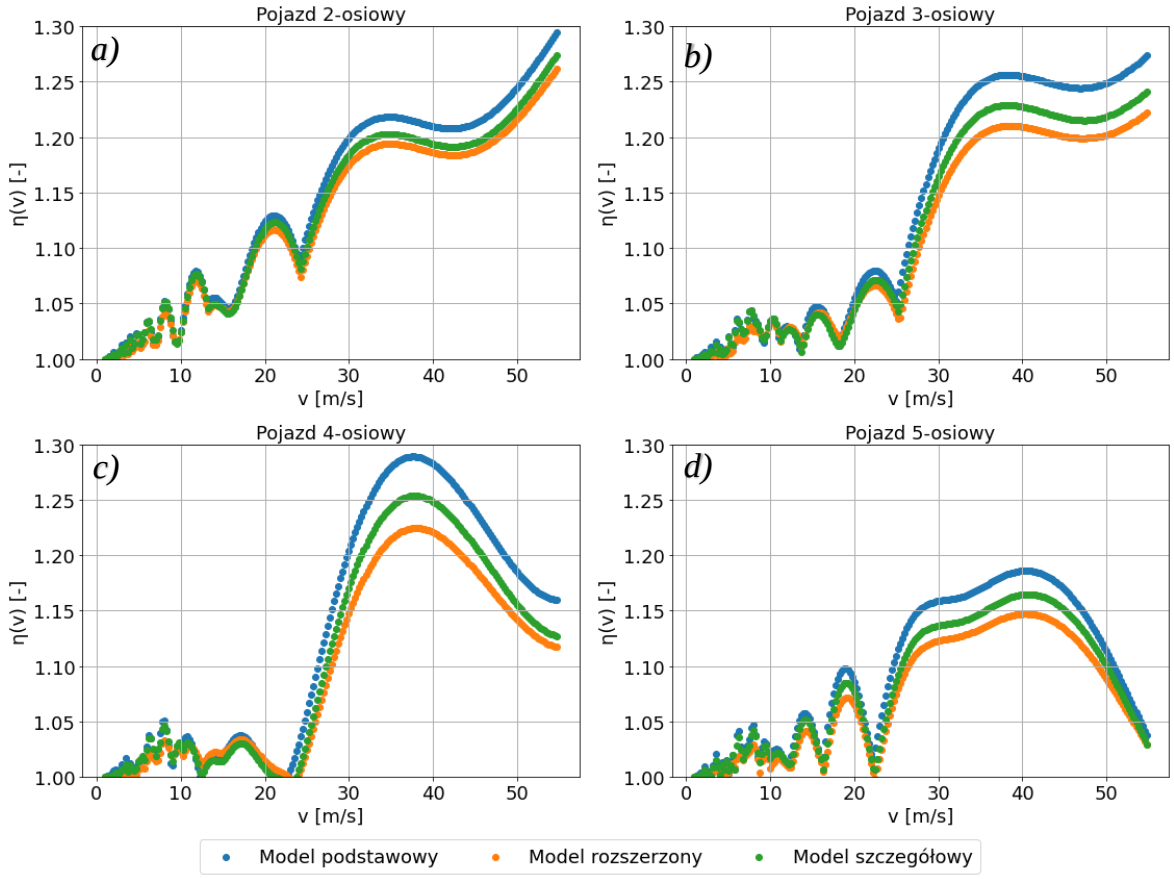
W celu oceny efektów wpływu prędkości przyjęto współczynnik η(v), który można wyrazić jako:

$$\eta(v) = \frac{\mathrm{u}_{\max}(v)}{\mathrm{u}_{\max}(v_* = 1\,m/s)}$$

gdzie:

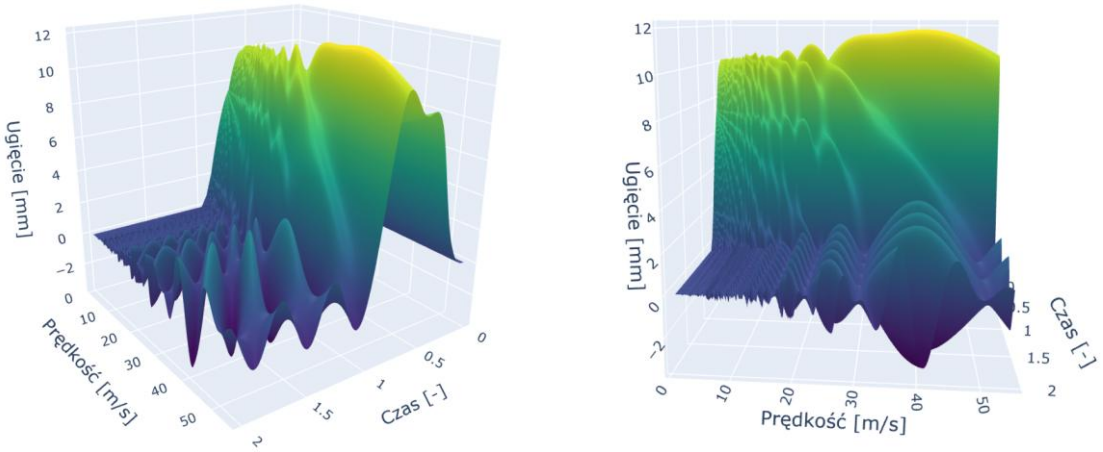
- $\mathrm{u}_{\max}(v)$ – maksymalne ugięcie przęsła w środku rozpiętości dla wybranej prędkości przejazdu,
- $\mathrm{u}_{\max}(v_* = 1\,m/s)$ – maksymalne ugięcie przęsła w środku rozpiętości dla wybranej quasi-statycznej prędkości przejazdu $v_* = 1\,m/s$.

Na rys. 4.30 przedstawiono wyniki $\eta(v)$ dla wszystkich rozpatrywanych typów/klas pojazdów.



Rys. 4.30 Zestawienie współczynnika $\eta(v)$ dla porównywanych modeli pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

W celu weryfikacji oraz precyzyjnej analizy kształtu wykresów współczynnika $\eta(v)$ dla modelu szczegółowego pojazdu 5-osiowego stworzono wykres ugięcia w postaci funkcji czasu i prędkości, który przedstawiono na rys. 4.31.



Rys. 4.31 Wykres ugięcia w funkcji czasu i prędkości (przedstawienie z różnych perspektyw)

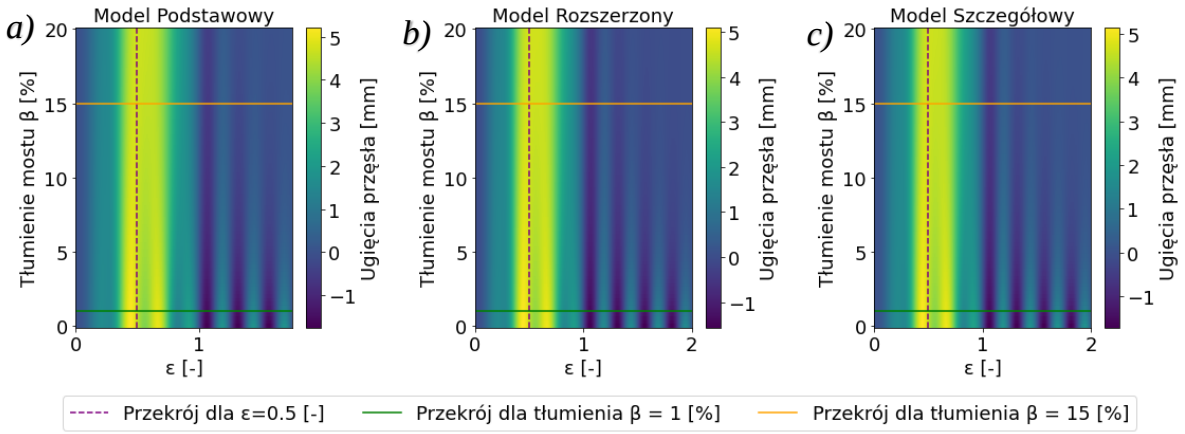
Wykres ten bardzo dobrze obrazuje istotę pojawiania się nieregularnych załamań na stworzonych wykresach współczynnika $\eta(v)$.

Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski:

- Wartość współczynnika $\eta(v)$ zależy od prędkości przejazdu, liczby osi oraz przyjętego rodzaju pojazdu.
- Model podstawowy charakteryzuje się największymi wartościami współczynnika $\eta(v)$.
- Maksymalne wartości współczynnika $\eta(v)$ są mocno zróżnicowane dla każdego typu pojazdu i zależą od konfiguracji nacisków oraz odległości między osiami.
- Zależność współczynnika $\eta(v)$ od prędkości nie jest zależnością liniową.
- Prędkość jest znaczącym parametrem wpływającym na oddziaływania dynamiczne i powinna zostać uwzględniona w algorytmie.

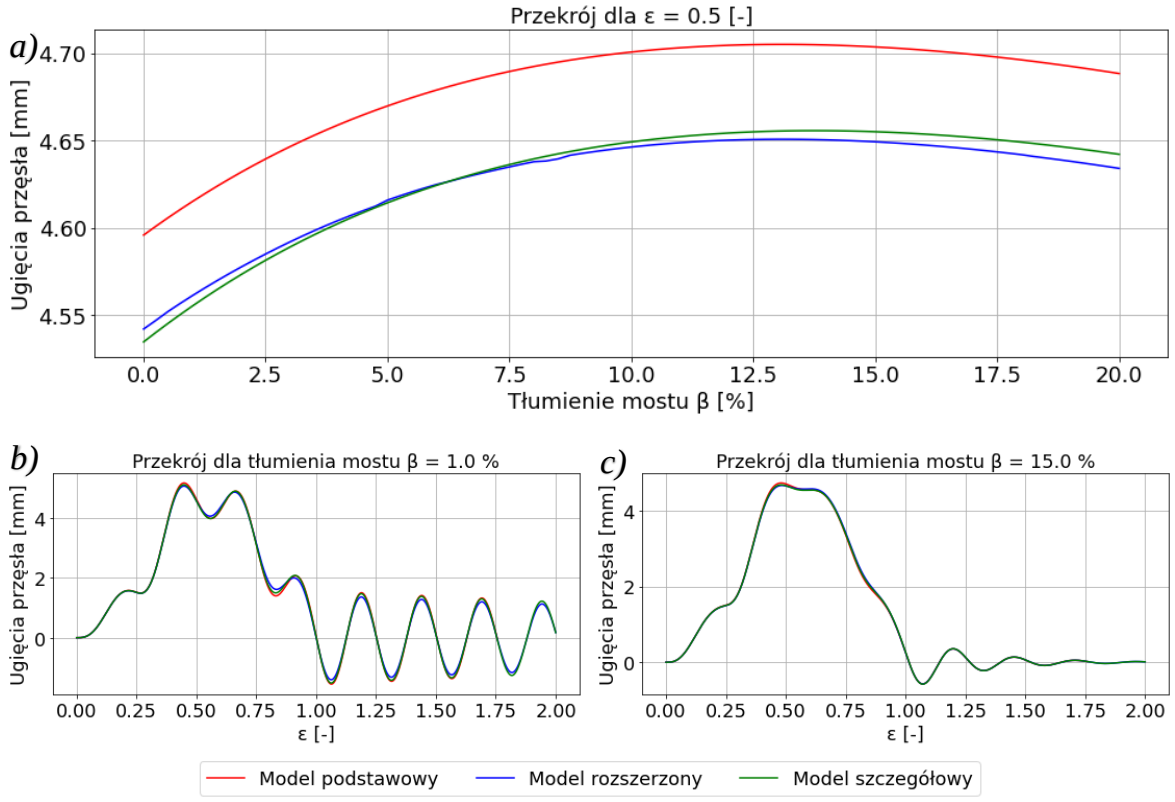
4.4.3 Wpływ tłumienia konstrukcji

Rozpatrzono wpływ tłumienia belki na przemieszczenie przęsła w środku rozpiętości. Iteracyjnie zmieniano dekrement tłumienia obiektu mostowego w zakresie od 0 do 20% tłumienia krytycznego, stosując krok co 0.25%. Przyjęto stałą prędkość przejazdu pojazdów na poziomie 25 m/s (90 km/h). Analizę przeprowadzono dla wszystkich modeli obliczeniowych pojazdów referencyjnych. Na ich podstawie opracowano mapy cieplne (rys. 4.32) dla pojazdu dwuosiowego, przedstawiające zależność ugięcia przęsła w środku rozpiętości od czasu ε oraz stosunku tłumienia belki do tłumienia krytycznego β. Analogicznie do poprzedniego podpunktu 4.4.2, zastosowano bezwymiarową skalę czasu ε.



Rys. 4.32 Mapy rozkładu ugięć przęsła w środku rozpiętości w funkcji czasu i tłumienia dla modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

W celu porównania różnic pomiędzy modelami na rys. 4.33 przedstawiono (w miejscach zaznaczonych na rys. 4.32) dla wybranych parametrów tłumienia β oraz wybranej chwili czasowej ε.



Rys. 4.33 Wybrane przekroje map ugięcia obliczonego dla różnych modeli pojazdu 2-osiowego: a) Przekrój dla chwili czasowej ε= 0.5; b) Przekrój dla β = 1.0 %; c) Przekrój dla β = 15.0 %

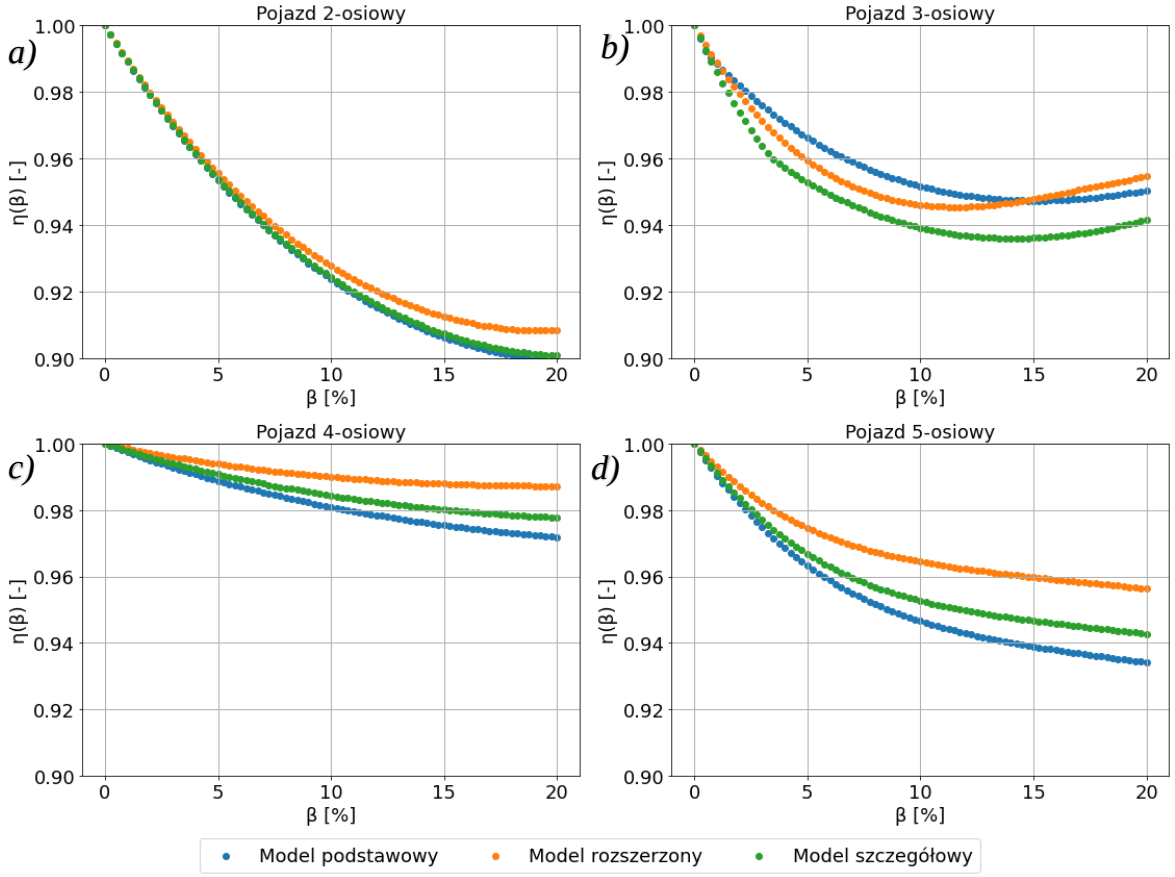
W celu oceny efektów wpływu tłumienia przyjęto współczynnik $\eta(\beta)$, który można wyrazić jako:

$$\eta(\beta) = \frac{u_{max}(\beta)}{u_{max}(\beta_* = 0\ \%)}$$

gdzie:

- $u_{max}(\beta)$ – maksymalne przemieszczenie pionowe dla wybranego procentowego stosunku tłumienia do wartości tłumienia krytycznego belki,
- $u_{max}(\beta_* = 0\ \%)$ – maksymalne przemieszczenie pionowe dla nietłumionej konstrukcji belki.

Na rys. 4.34 przedstawiono zbiorcze wyniki η(β) dla wybranych pojazdów z podziałem na liczbę osi oraz model pojazdu.



Rys. 4.34 Zestawienie współczynnika $\eta(\beta)$ dla porównywanych modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

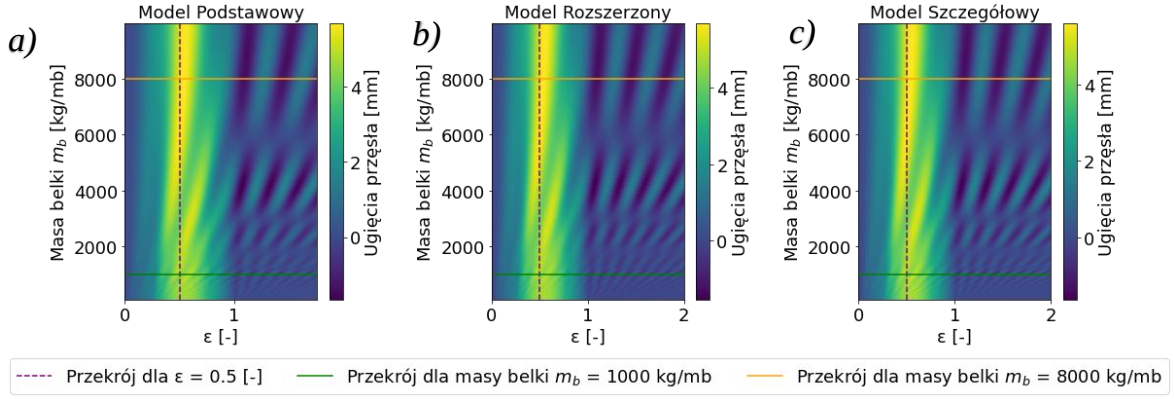
Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski:

- Wartość współczynnika η(β) zależy od liczby osi oraz przyjętego modelu pojazdu.
- Model podstawowy charakteryzuje się największymi zmianami współczynnika η(β).
- Zależność współczynnika η(β) od prędkości nie jest zależnością liniową.
- W zakresie tłumienia dla typowych konstrukcji budowlanych (tj. dla $\beta \leq 3\ \%$), zmiany te są praktycznie pomijalne (poniżej 1 %).

- Tłumienie konstrukcji jest parametrem wpływającym na oddziaływania dynamiczne i powinno zostać uwzględnione w algorytmie.

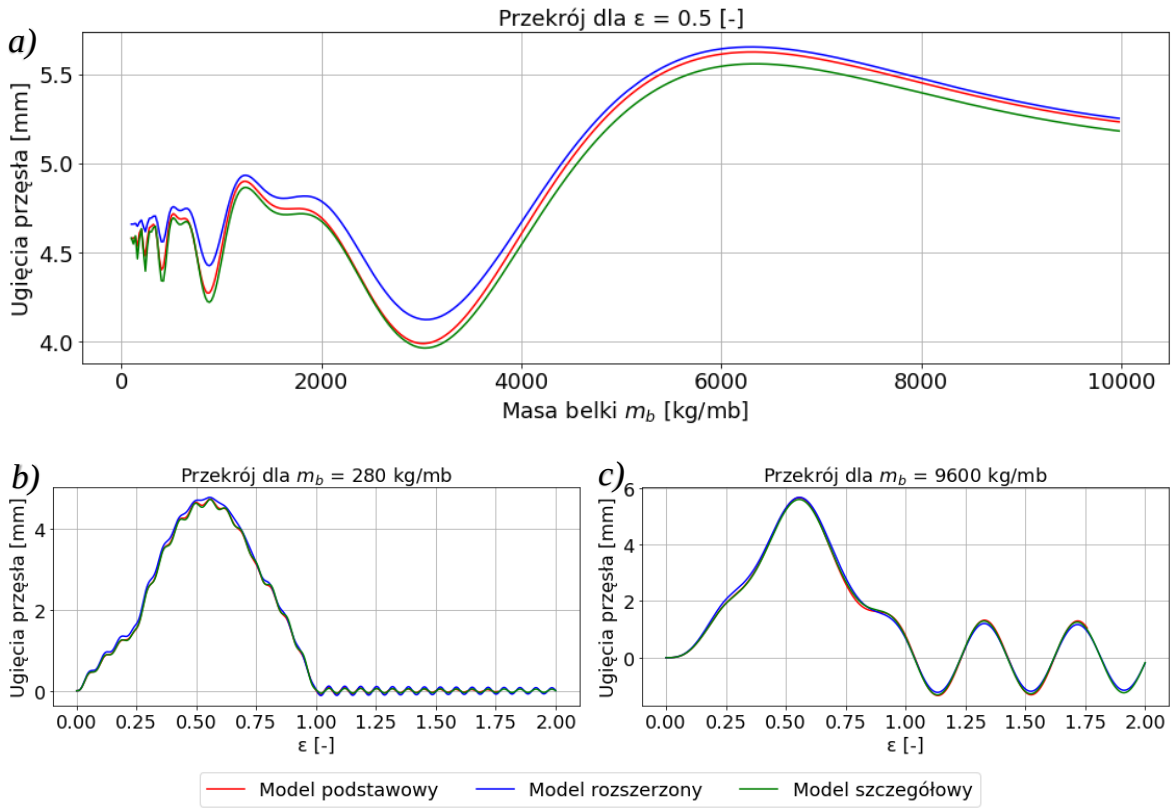
4.4.4 Wpływ masy własnej konstrukcji

Rozpatrzono wpływ masy belki na przemieszczenie przęsła w środku rozpiętości. Iteracyjnie zmieniano równomiernie rozłożoną masę belki w zakresie od $100\ \frac{kg}{mb}$ do $10000\ \frac{kg}{mb}$, stosując krok $20\ \frac{kg}{mb}$. Przyjęto stałą prędkość przejazdu pojazdów na poziomie 25 m/s (90 km/h). Analizę przeprowadzono dla wszystkich modeli obliczeniowych pojazdów referencyjnych. Opracowano mapy cieplne (rys. 4.32) dla pojazdu dwuosiowego, przedstawiające zależność ugięcia przęsła w środku rozpiętości od czasu oraz równomiernie rozłożonej masy belki wyrażonych odpowiednio jako ε oraz m_b.



Rys. 4.35 Mapy rozkładu ugięć przęsła w środku rozpiętości w funkcji czasu i masy belki dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

W celu porównania różnic pomiędzy modelami na mapie cieplnej zaznaczono płaszczyzny dla wybranych wartości równomiernie rozłożonej masy belki m_b oraz wybranej chwili czasowej ε. Różnice pomiędzy modelami dla pojazdu 2-osiowego dla wybranych wartości przedstawiono na rys. 4.36.



Rys. 4.36 Wybrane przekroje map dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Przekrój dla chwili czasowej ε= 0.5; b) Przekrój dla $m_b = 280\ kg/mb$; c) Przekrój dla $m_b = 9600\ kg/mb$

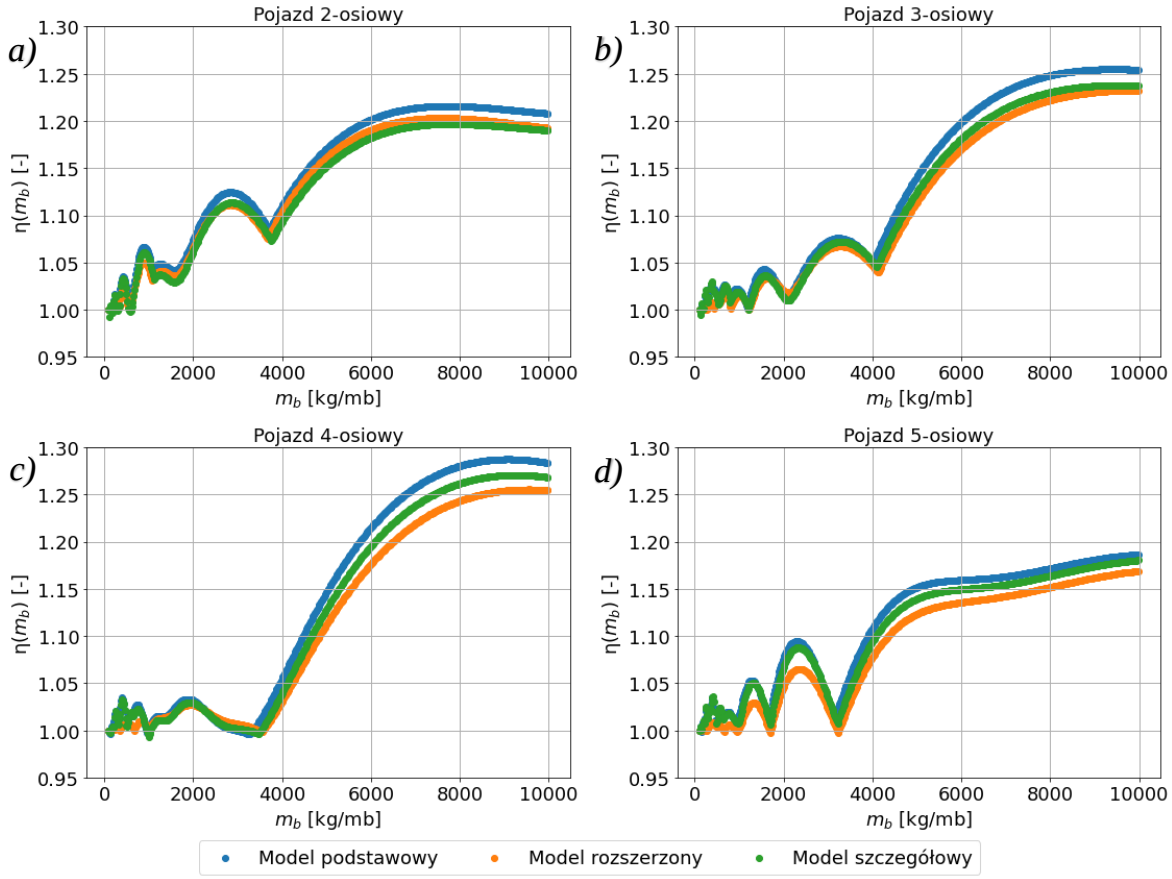
W celu oceny efektów wpływu masy przyjęto współczynnik przeciążenia $\eta(m_b)$, który można wyrazić jako:

$$\eta(m_b) = \frac{u_{max}(m_b)}{u_{max}(m_{b_1}=100kg/mb)},$$

gdzie:

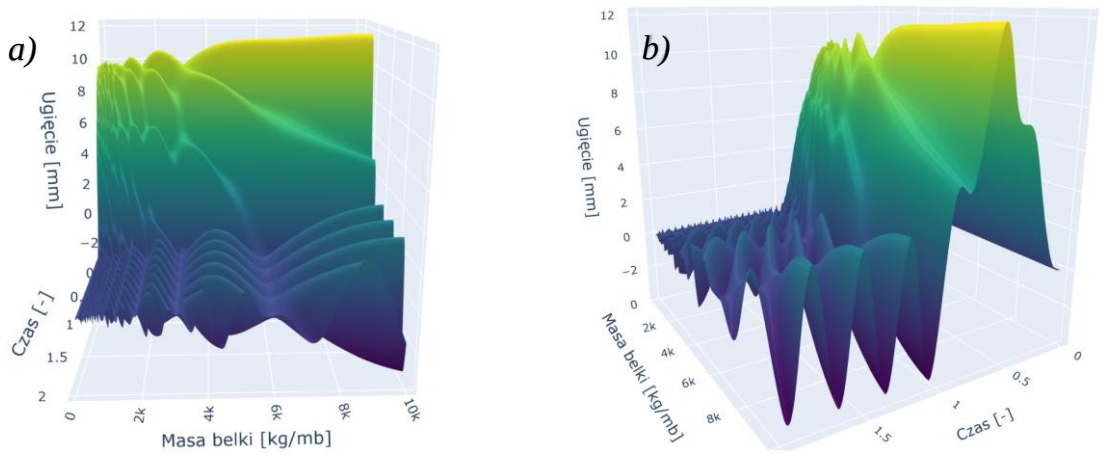
- $u_{max}(m_b)$ – maksymalne przemieszczenie pionowe dla wybranej, równomiernie rozłożonej masy obiektu mostowego,
- $u_{max}(m_{b_1} = 100kg/mb)$ – maksymalne przemieszczenie pionowe dla wybranej masy równomiernie rozłożonej obiektu mostowego równej 100 kg/mb.

Na rys. 4.37 przedstawiono zbiorcze wyniki $\eta(m_b)$ dla wybranych pojazdów z podziałem na liczbę osi oraz model pojazdu.



Rys. 4.37 Zestawienie współczynnika $\eta(m_b)$ dla porównywanych modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

Interesujący jest przebieg wartości szczegółowego modelu dla pojazdu 5-osiowego, przedstawiony na rys. 4.38, na którym pokazano wykres ugięcia w postaci funkcji czasu i masy belki.



Rys. 4.38 Wykres ugięcia w funkcji czasu i masy belki (przedstawienie z różnych perspektyw)

Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski:

- Wartość współczynnika $\eta(m_b)$ zależy od liczby osi oraz przyjętego modelu pojazdu.
- Model podstawowy charakteryzuje się największymi zmianami współczynnika $\eta(m_b)$.

- Maksymalne wartości współczynnika $\eta(m_b)$ są indywidualne dla każdego typu pojazdu. Zależą od konfiguracji nacisków oraz odległości między osiami.
- Zależność współczynnika $\eta(m_b)$ od masy obiektu mostowego nie jest zależnością liniową.
- Równomiernie rozłożona masa obiektu mostowego jest parametrem wpływającym na oddziaływania dynamiczne i powinna zostać uwzględniona w algorytmie.

4.4.5 Wpływ rozpiętości przęsła

Rozpatrzono wpływ rozpiętości przęsła przy założeniu stałej podatności, rozumianej jako miara, która określa, jak łatwo konstrukcja ulega odkształceniu pod wpływem przyłożonej siły. Matematycznie, podatność jest odwrotnością sztywności i jest zdefiniowana jako stosunek odkształcenia do siły.

$$f = \frac{\delta}{P}$$

gdzie:

- δ – ugięcie belki w środku rozpiętości,
- P – zewnętrzna siła działająca na konstrukcję.

Maksymalne ugięcie belki w środku rozpiętości wywołane obciążeniem statycznym jest funkcją, która zależy od np. wartości obciążenia zewnętrznego, parametrów geometrycznych oraz materiałowych przekroju, rozpiętości teoretycznej oraz schematu statycznego. Zakładając więc, stałą siłę przy zmieniającej się rozpiętości należałoby zwiększać sztywność w takim stopniu, aby uzyskiwane ugięcia w każdym kroku były takie same.

Dobierając parametry belki przyjęto następujące założenia:

- analizowana rozpiętość mieści się w zakresie o 10 do 50 m,
- obciążenia zewnętrznego w postaci siły skupionej o wartości 600 kN, w środku rozpiętości
- dopuszczalna strzałka ugięcia wynosi $u_{dop} = \frac{L_t}{600}$.

Dobór przekroju w zależności od rozpiętości określono z wykorzystaniem algorytmu bazującego na podstawowym wzorze (4.35).

$$u_{extr} = \frac{PL_t^3}{48EI} \qquad (4.35)$$

Stosując się do poczynionych założeń, można określić wymagany moment bezwładności:

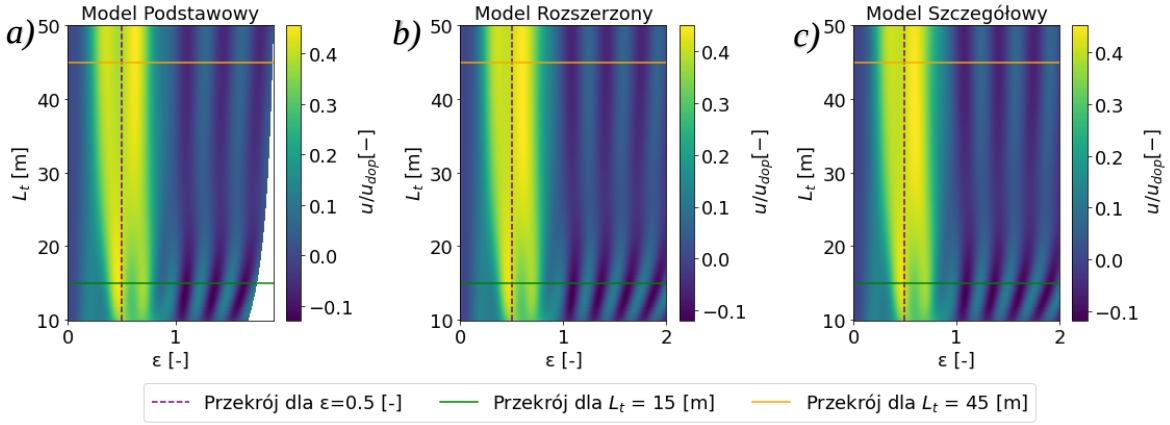
$$\frac{L_t}{600} = \frac{PL_t^3}{48EI_{req}}$$

$$I_{req} = \frac{600PL_t^2}{48E}$$

Rozpatrzono wpływ rozpiętości przęsła obiektu mostowego, w zakresie granicznych wartości od 10 do 50 m przy założonej dopuszczalnej strzałce ugięcia od obciążenia referencyjnego, na odpowiedź

dynamiczną konstrukcji. Procedura analizy miała charakter iteracyjny, w którym dla wybranych wartości rozpiętości obiektu mostowego, dobierano parametry przekrojowe obiektu mostowego, a następnie sprawdzano odpowiedź konstrukcji na przejazd pojazdów. Założono prędkość przejazdu na poziomie 25 m/s (90 km/h). Analizy przeprowadzono dla wszystkich modeli obliczeniowych pojazdów referencyjnych.

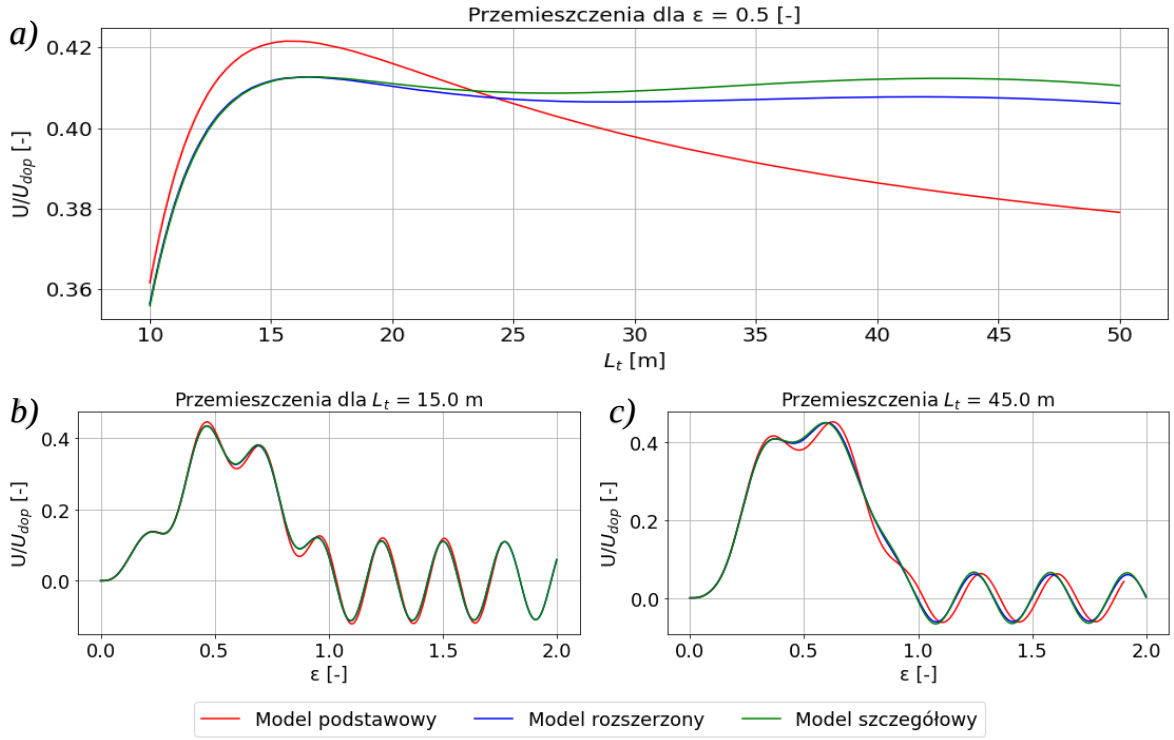
Z uwagi na fakt, że wraz ze wzrostem rozpiętości, rośnie również dopuszczalna strzałka ugięcia wprowadzono więc wykres ugięcia przęsła w stosunku do maksymalnego ugięcia dopuszczalnego dla danej rozpiętości, czyli $\frac{u}{u_{dop}}$. Na ich podstawie opracowano mapy cieplne (rys. 4.39) dla pojazdu dwuosiowego, przedstawiające zależność $\frac{u}{u_{dop}}$ w środku rozpiętości od czasu oraz rozpiętości teoretycznej belki wyrażonych odpowiednio jako ε oraz L_t.



Rys. 4.39 Mapy rozkładu ugięć przęsła w środku rozpiętości w funkcji czasu i rozpiętości teoretycznej dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

Na rys. 4.39 zauważalny jest fakt, że dla wybranego pojazdu 2-osiowego osiągnięty maksymalny stosunek jest podobny dla szerokiego zakresu rozpiętości niezależnie od rozpiętości – zwłaszcza dla bardziej złożonych modeli.

W celu porównania różnic pomiędzy modelami na mapie cieplnej zaznaczono płaszczyzny dla wybranych rozpiętości teoretycznych L_t oraz wybranej chwili czasowej ε. Różnice pomiędzy modelami dla pojazdu 2-osiowego dla wybranych wartości przedstawiono na rys. 4.40.



Rys. 4.40 Wybrane przekroje map dla różnych modeli pojazdu 2-osiowego: a) Przekrój dla chwili czasowej ε= 0.5; b) Przekrój dla L_t=15.0 m; c) Przekrój dla L_t=45.0 m

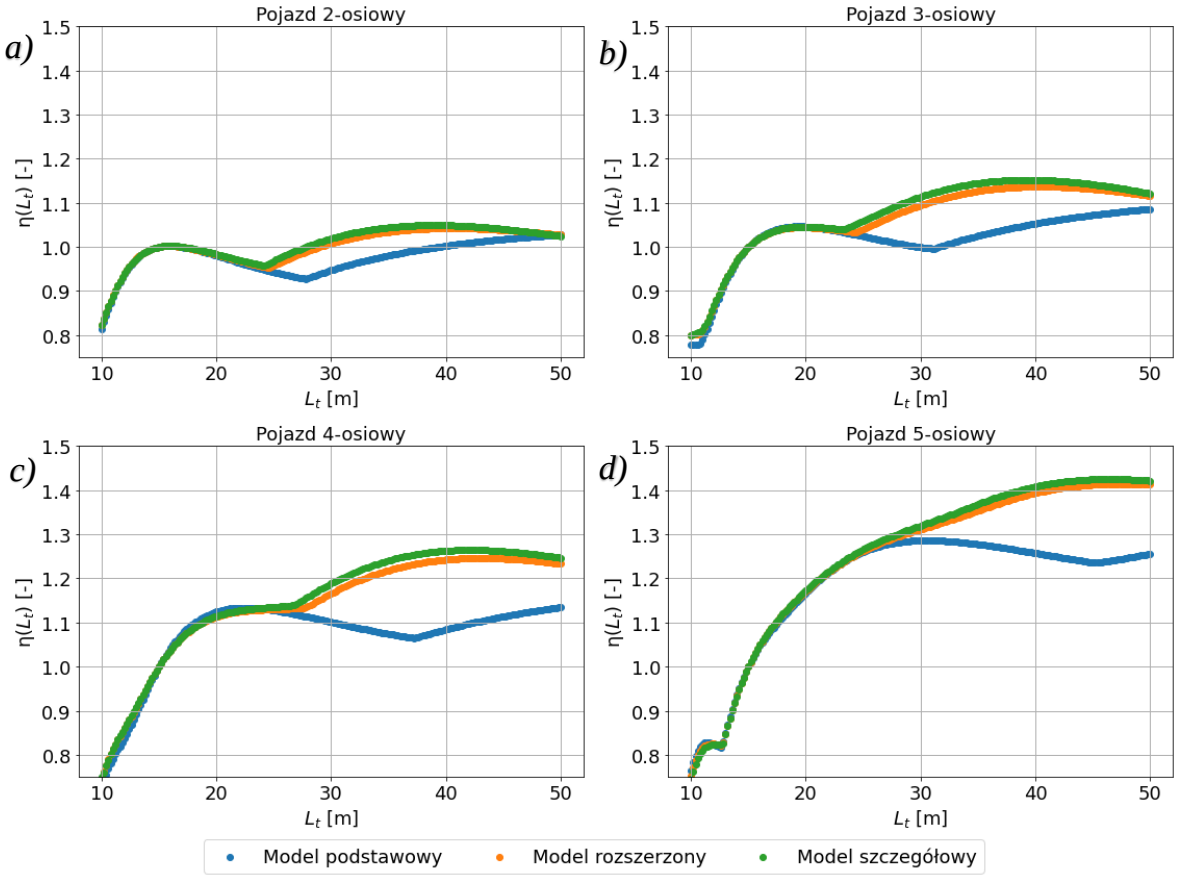
W celu oceny efektów wpływu rozpiętości teoretycznej przyjęto współczynnik $\eta(L_t)$, który można wyrazić jako:

$$\eta(L_t) = \frac{u_{max}(L_t)}{u_{max}(L_{t*} = 10\ m\ )}$$

gdzie:

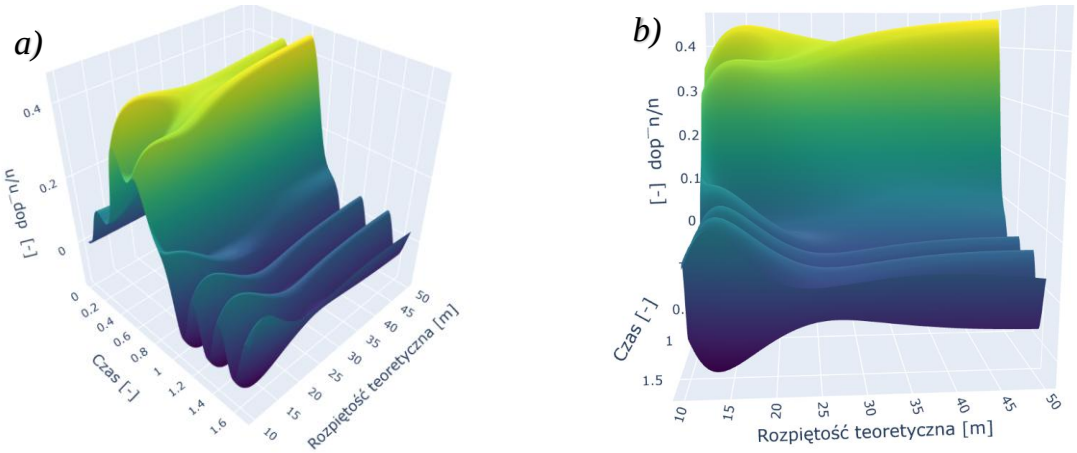
- $u_{max}(L_t)$ – maksymalne wartość ugięcia belki do ugięcia dopuszczalnego dla belki o wybranej rozpiętości teoretycznej,
- $u_{max}(L_{t*} = 10m)$ – wartość ugięcia belki do ugięcia dopuszczalnego dla belki o rozpiętości teoretycznej równej 10 m.

Na rys. 4.41 przedstawiono zbiorcze wyniki $\eta(L_t)$ dla wybranych pojazdów z podziałem na liczbę osi oraz model pojazdu.



Rys. 4.41 Zestawienie współczynnika $\eta(L_t)$ dla porównywanych modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

W celu weryfikacji poprawności wykresów, na rys. 4.42 przedstawiono wykres $\frac{u}{u_{dop}}$ w postaci funkcji czasu i rozpiętości teoretycznej dla modelu szczegółowego pojazdu 5-osiowego.



Rys. 4.42 Wykres $\frac{u}{u_{dop}}$ w funkcji czasu i rozpiętości teoretycznej (przedstawienie z różnych perspektyw)

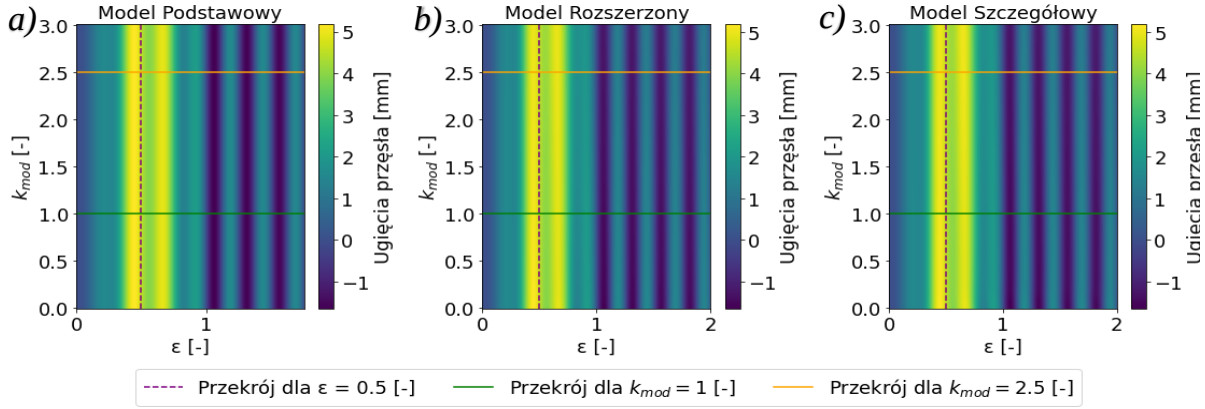
Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski:

- Zależność współczynnika $\eta(L_t)$ od L_t nie jest zależnością liniową.

- Maksymalne wartości współczynnika $\eta(L_t)$ są indywidualne dla każdego typu pojazdu – zależą od konfiguracji osi pojazdu.
- Największą zmianę współczynnika zanotowano dla szczegółowego modelu pojazdu 5-osiowego.
- Rozpiętość konstrukcji jest parametrem wpływającym na oddziaływania dynamiczne.

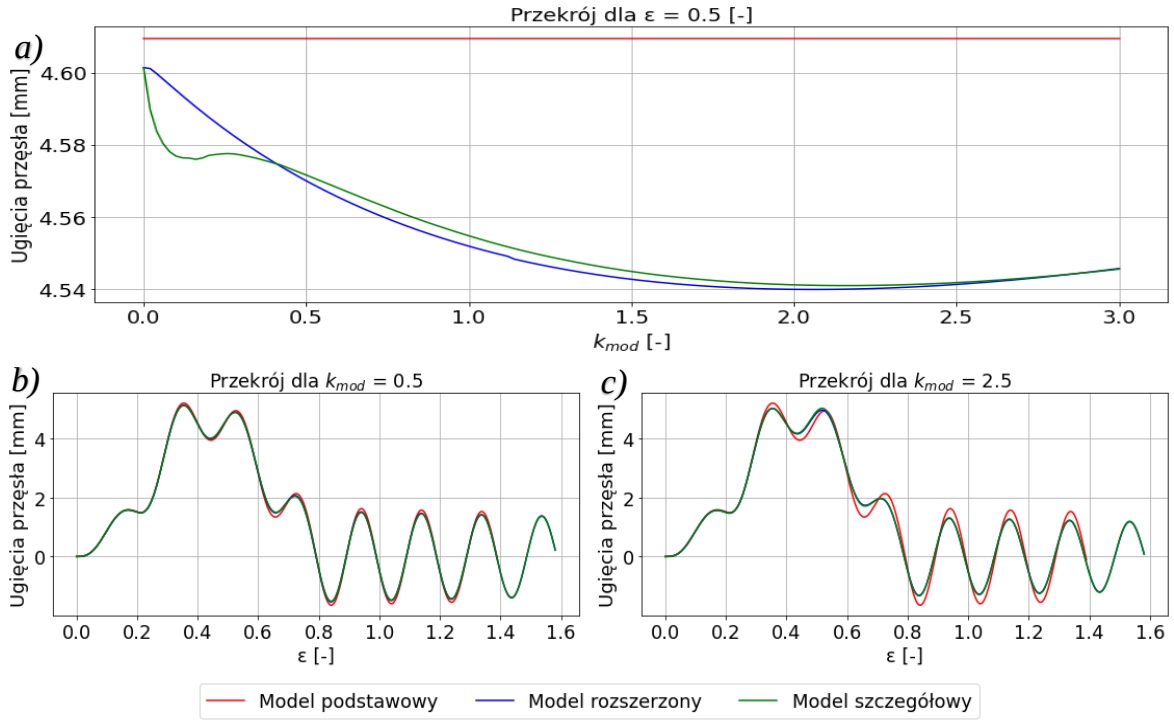
4.4.6 Wpływ sztywności zawieszenia pojazdu

Rozpatrzono wpływ sztywności zawieszenia pojazdu na odpowiedź dynamiczną konstrukcji poprzez krokową zmianę sztywności zawieszenia. Przyjęto, że parametry zawieszenia opisane w 4.3.5, są wartościami referencyjnymi oraz przyjęto te wartości jako poziom odniesienia. Zastosowano parametr k_{mod}, który opisuje stosunek sztywności zawieszenia analizowanego w kroku obliczeniowej do sztywności zawieszenia przyjętych jako wartości referencyjne. Dla przyjętych pojazdów wartość $k_{mod} = 1.0$. Następnie sprawdzono parametr k_{mod} w zakresie $< 0; 3.0 >$ z krokiem iteracyjnym 0.02. Pozostałe parametry analizy pozostały niezmienne. Analizy przeprowadzono dla wszystkich modeli obliczeniowych pojazdów referencyjnych. W celu zobrazowania wyników dla pojazdu dwuosiowego przedstawiono je w postaci mapy cieplnej rys. 4.43.



Rys. 4.43 Mapy cieplne ugięć przęsła w środku rozpiętości w funkcji czasu i sztywności zawieszenia dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

W celu porównania różnic pomiędzy modelami na mapie cieplnej zaznaczono płaszczyzny dla wybranych współczynników sztywności zawieszenia oraz wybranej chwili czasowej. Różnice pomiędzy modelami dla pojazdu 2-osiowego dla wybranych wartości przedstawiono na rys. 4.44.



Rys. 4.44 Wybrane przekroje map dla różnych modeli pojazdu 2-osiowego:
a) Przekrój dla chwili czasowej ε= 0.5; b) Przekrój dla $k_{mod} = 0.5$*; c) Przekrój dla* $k_{mod} = 2.5$

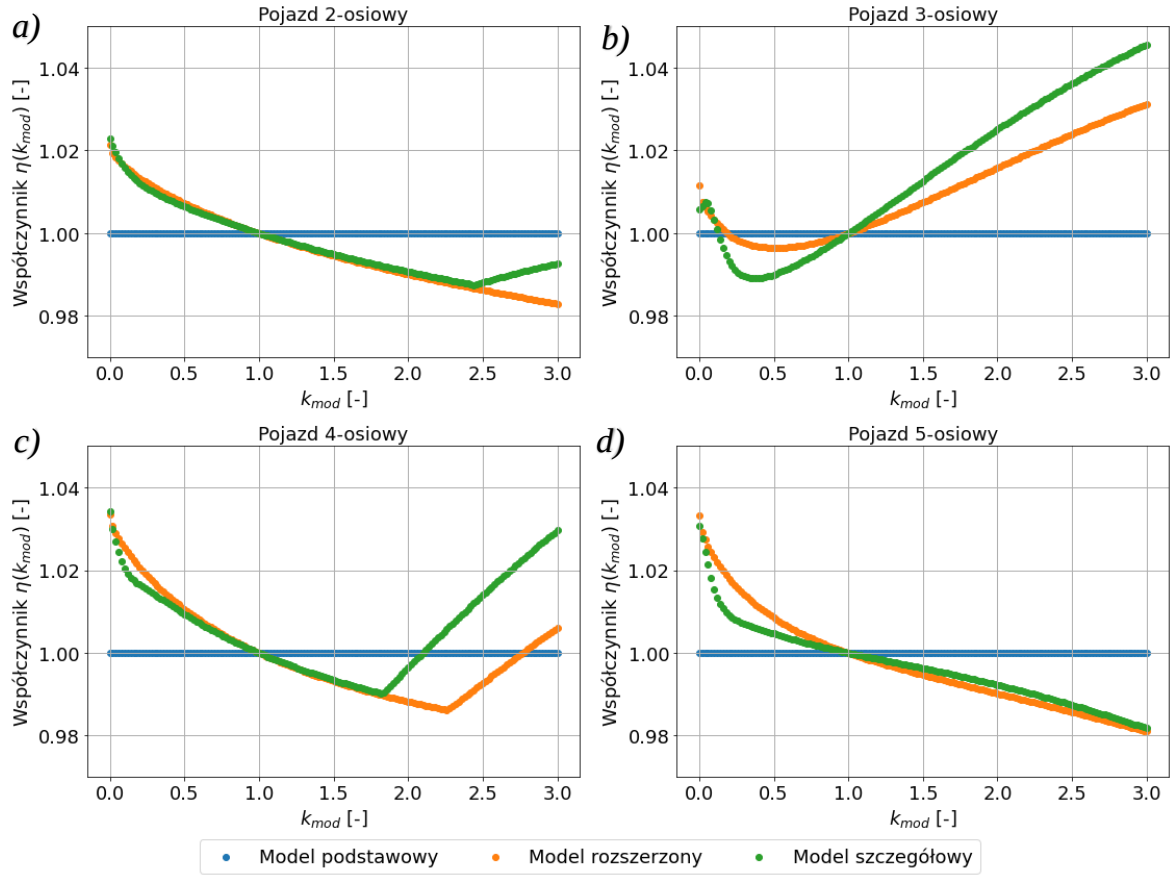
W celu oceny efektów wpływu sztywności zawieszenia przyjęto współczynnik $\eta(k_{mod})$, który można wyrazić jako:

$$\eta(k_{mod}) = \frac{\mathrm{u}_{\max}(k_{mod})}{\mathrm{u}_{\max}(k_{mod*} = 1)}$$

gdzie:

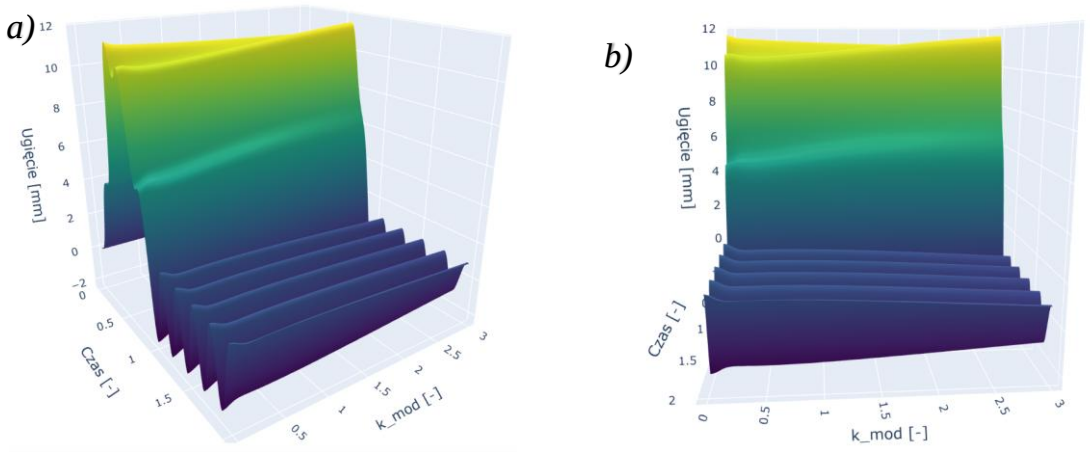
- $\mathrm{u}_{\max}(k_{mod})$ – maksymalne przemieszczenie pionowe dla wybranej wartości sztywności zawieszenia,
- $\mathrm{u}_{\max}(k_{mod*} = 1)$ – maksymalne przemieszczenie pionowe dla referencyjnej sztywności zawieszenia.

Na rys. 4.45 przedstawiono zbiorcze wyniki $\eta(k_{mod})$ dla wybranych pojazdów z podziałem na liczbę osi oraz model pojazdu.



Rys. 4.45 Zestawienie współczynnika $\eta(k_{mod})$ dla porównywanych modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

W celu weryfikacji poprawności wykresów, na rys. 4.46 przedstawiono ugięcia w postaci funkcji czasu i k_{mod} dla modelu szczegółowego pojazdu 4-osiowego.



Rys. 4.46 Wykres ugięcia belki w funkcji czasu i k_{mod} (przedstawienie z różnych perspektyw)

Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski:

- Wartość współczynnika $\eta(k_{mod})$ nieznacznie wpływają na odpowiedź dynamiczną konstrukcji mostowej.
- Maksymalne wartości współczynnika $\eta(k_{mod})$ są indywidualne dla każdego typu pojazdu i zależą od konfiguracji osi pojazdu.
- Zależność współczynnika $\eta(k_{mod})$ nie jest zależnością liniową.

4.4.7 Wpływ tłumienia zawieszenia pojazdu

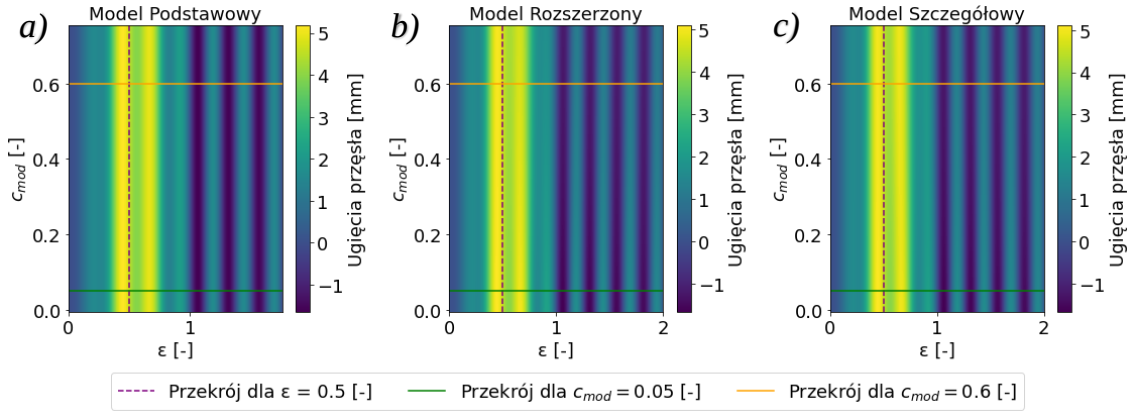
Przeprowadzona analiza skupiła się na badaniu wpływu tłumienia zawieszenia pojazdów na przemieszczenie przęsła w środku rozpiętości. Przyjęto parametr c_{mod}, który wyraża się jako:

$$c_{mod} = \frac{c}{c_{kr}}$$

gdzie:

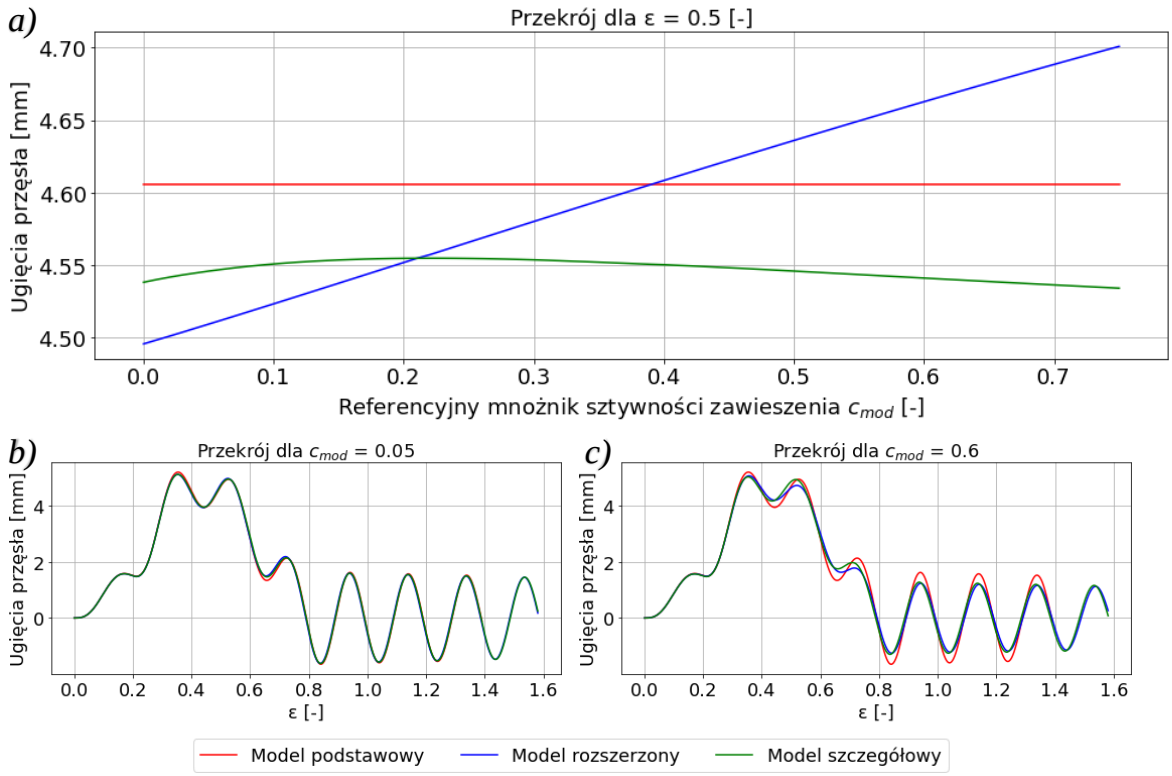
- c – tłumienie belki,
- c_{kr} – tłumienie krytyczne belki.

Następnie przeprowadzono symulacje dla parametrów c_{mod} w zakresie $< 0; 0.7 >$ z krokiem iteracyjnym 0.025. Analizy przeprowadzono dla wszystkich modeli obliczeniowych pojazdów referencyjnych. Na ich podstawie opracowano mapy cieplne (rys. 4.47) dla pojazdu dwuosiowego, przedstawiające zależność ugięcia przęsła w środku rozpiętości od czasu ε oraz wartości c_{mod}.



Rys. 4.47 Mapy rozkładu ugięć przęsła w środku rozpiętości w funkcji czasu i tłumienia zawieszenia dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

W celu porównania różnic pomiędzy modelami na mapie cieplnej zaznaczono płaszczyzny dla wybranych wartości c_{mod} oraz wybranej chwili czasowej. Różnice pomiędzy modelami dla pojazdu 2 - osiowego dla wybranych wartości przedstawiono na rys. 4.48.



Rys. 4.48 Wybrane przekroje map dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Przekrój dla chwili czasowej ε= 0.5; b) Przekrój dla $c_{mod} = 0.05$*; c) Przekrój dla* $c_{mod} = 0.6$

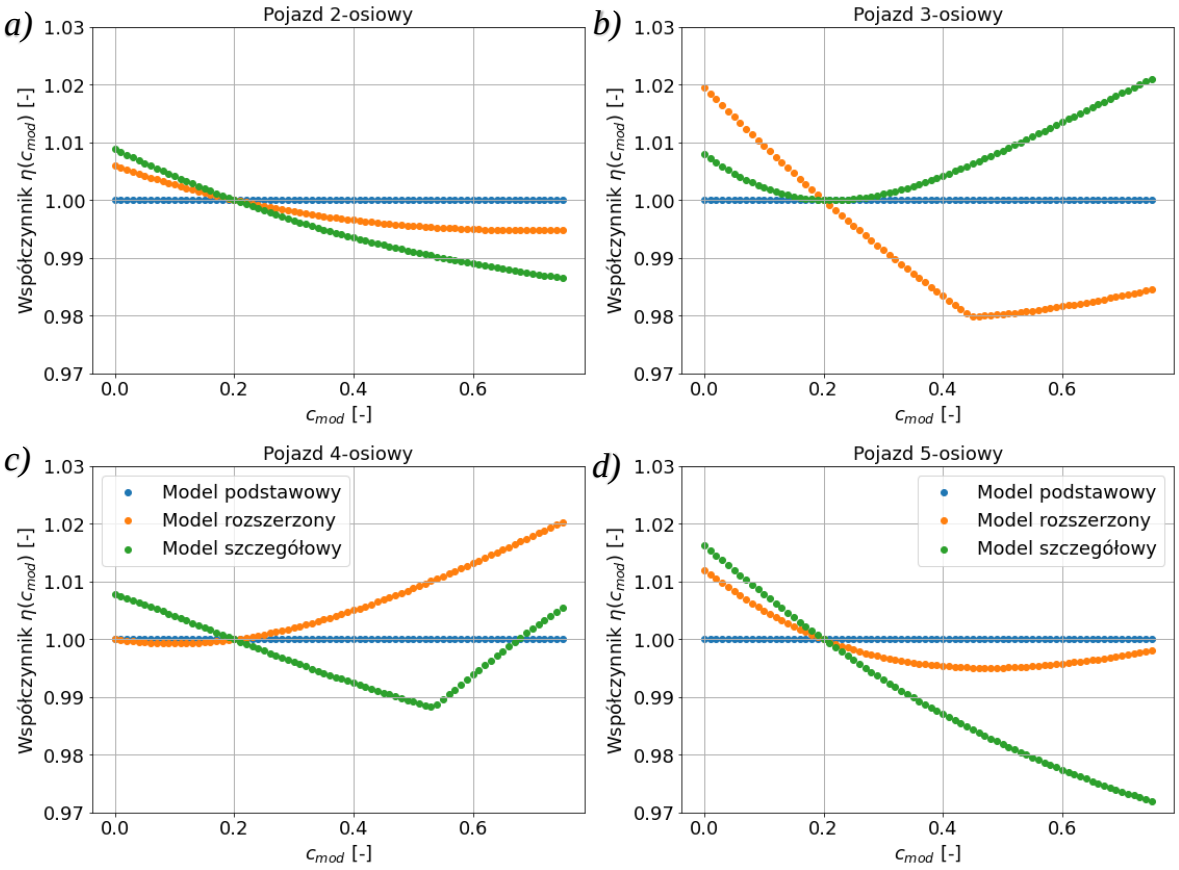
W celu oceny efektów wpływu tłumienia zawieszenia przyjęto współczynnik $\eta(c_{mod})$, który można wyrazić jako:

$$\eta(c_{mod}) = \frac{u_{max}(c_{mod})}{u_{max}(c_{mod*} = 0.2\ )}$$

gdzie:

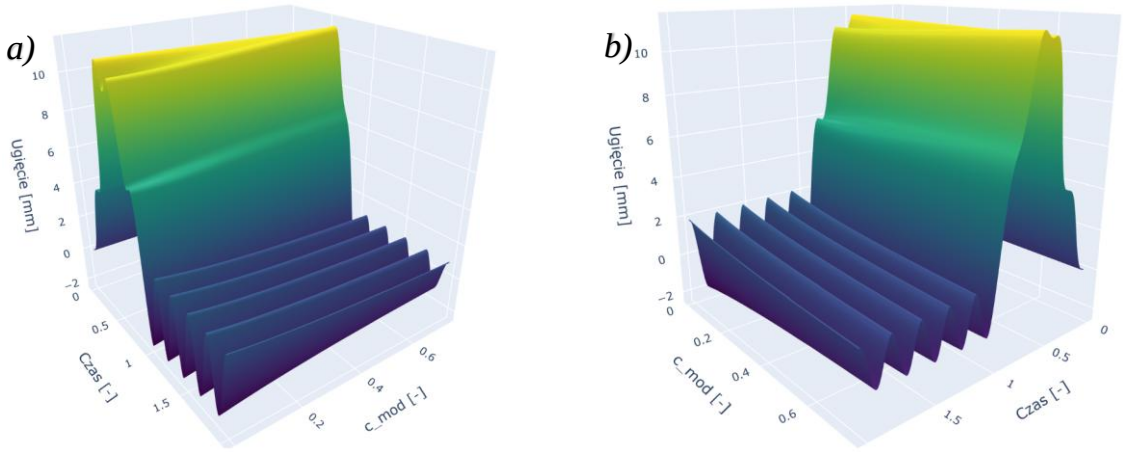
- $u_{max}(c_{mod})$ – maksymalne przemieszczenie pionowe dla wybranej wartości dekrementu tłumienia zawieszenia,
- $u_{max}(c_{mod*} = 0.2)$ – maksymalne przemieszczenie pionowe dla referencyjnej wartości tłumienia zawieszenia.

Na przedstawiono zbiorcze wyniki $\eta(c_{mod})$ dla wybranych pojazdów z podziałem na liczbę osi oraz model pojazdu.



Rys. 4.49 Zestawienie współczynnika $\eta(c_{mod})$ dla porównywanych modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

W celu weryfikacji poprawności wykresów, na rys. 4.50 przedstawiono ugięcia w postaci funkcji czasu i c_{mod} dla modelu szczegółowego pojazdu 4-osiowego.



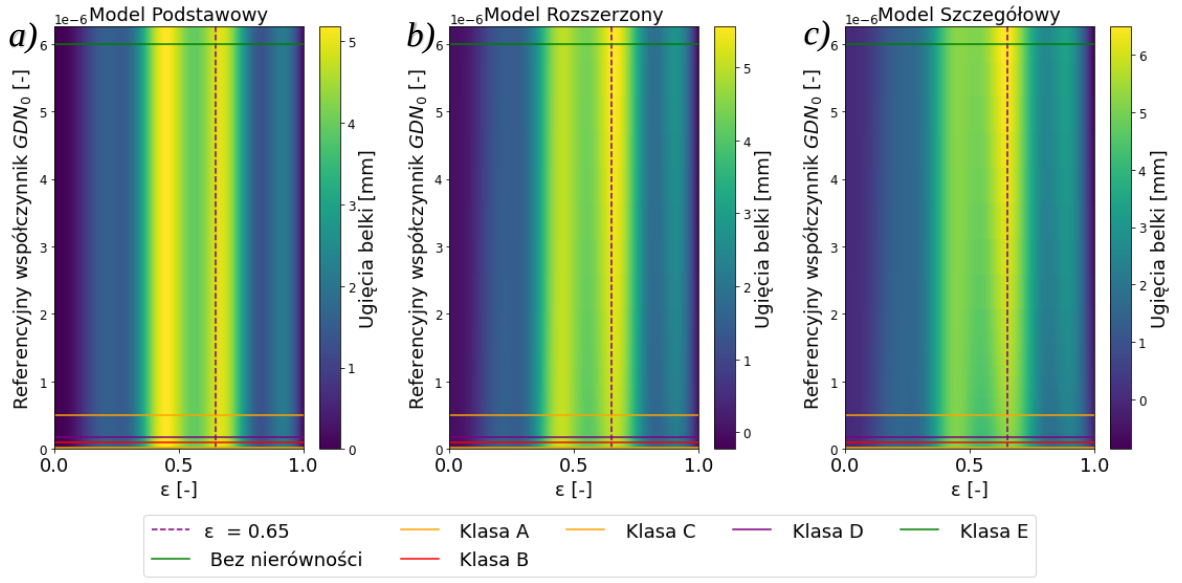
Rys. 4.50 Wykres ugięcia belki w funkcji czasu i c_{mod} (przedstawienie z różnych perspektyw)

Na podstawie przedstawionego wykresu można zweryfikować pojawienie się załamania, które wynikają z charakteru przebiegu funkcji ugięcia. Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski:

- Wartość współczynnika $\eta(c_{mod})$ zależy od prędkości przejazdu, liczby osi oraz przyjętego modelu pojazdu.
- Maksymalne wartości współczynnika $\eta(c_{mod})$ są indywidualne dla każdego typu pojazdu. Zależą od konfiguracji nacisków oraz odległości między osiami.
- Wartość c_{mod} ma niewielki wpływ na wartość $\eta(c_{mod})$.
- Zależność współczynnika $\eta(c_{mod})$ nie jest zależnością liniową.

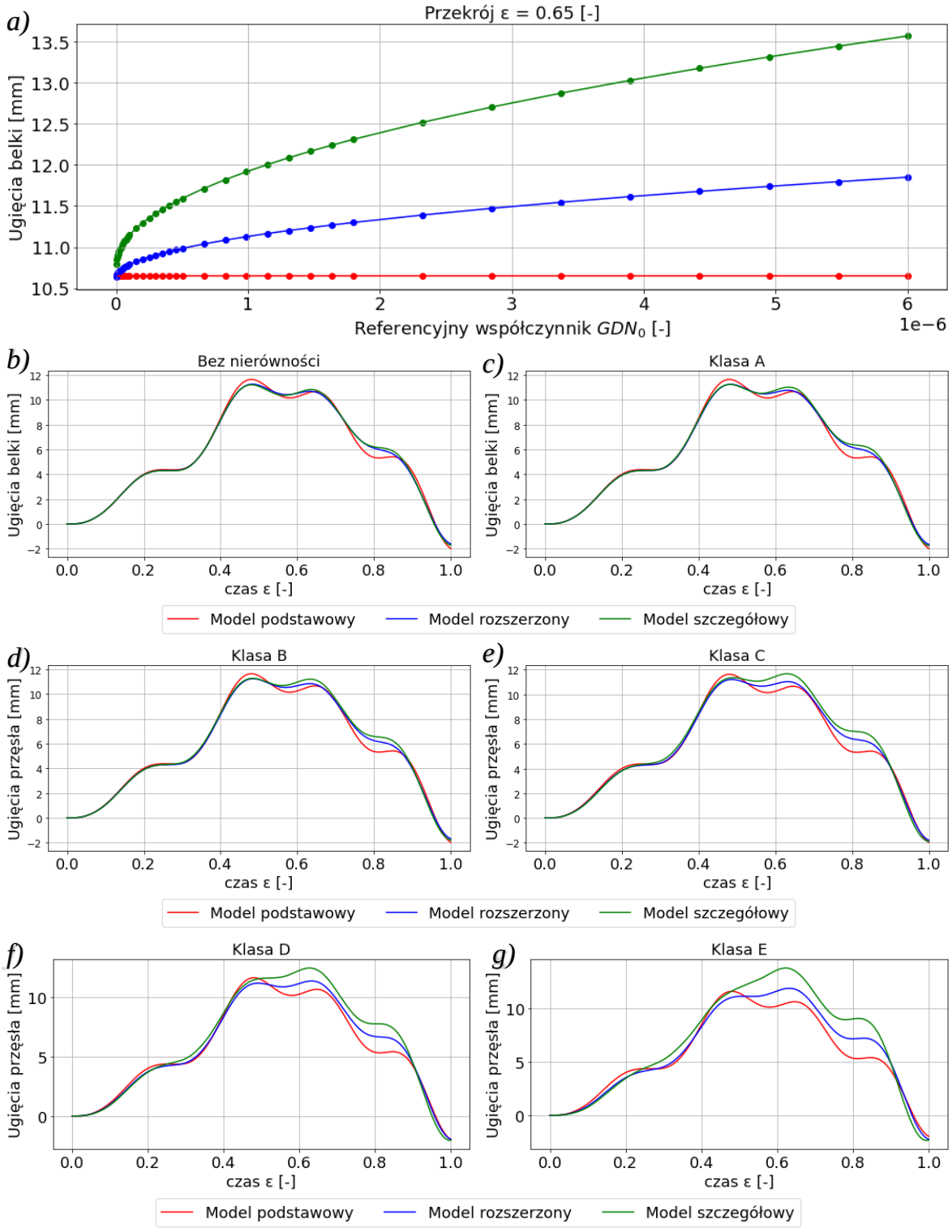
4.4.8 Wpływ nierówności nawierzchni

Przeprowadzona analiza skupiła się na badaniu wpływu nierówności nawierzchni na odpowiedź dynamiczną konstrukcji w środku rozpiętości przęsła. Wygenerowano funkcje nierówności nawierzchni, które różniły się wartościami amplitud poszczególnych składowych funkcji nierówności. Sprawdzano różne wartości klasy jakości nawierzchni w zakresie od idealnie gładkiej nawierzchni, przez nawierzchnię o bardzo dobrej jakości klasy A do drogi klasy E. Następnie dla każdego modelu oraz każdego pojazdu referencyjnego wykonano analizę wpływu ugięcia przęsła w środku rozpiętości od czasu oraz stopnia nierówności nawierzchni. W celu zobrazowania wyników dla pojazdu dwuosiowego przedstawiono je w postaci mapy cieplnej rys. 4.51.



Rys. 4.51 Mapy rozkładu ugięć przęsła w środku rozpiętości w funkcji czasu i nierówności drogi dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Model Podstawowy; b) Model Rozszerzony; c) Model Szczegółowy

W celu porównania różnic pomiędzy modelami na mapie cieplnej zaznaczono płaszczyzny dla różnych profili klas drogi oraz przekrój dla wybranej chwili czasowej. Różnice pomiędzy modelami dla pojazdu 2-osiowego dla wybranych przekroi przedstawiono na rys. 4.52.



Rys. 4.52 Wybrane przekroje map dla różnych modeli pojazdu 2-osiowego, odpowiednio: a) Przekrój dla chwili czasowej ε= 0.65 b) Przekrój dla drogi bez nierówności nawierzchni; c) Przekrój dla drogi klasy A; d) Przekrój dla drogi klasy B; e) Przekrój dla drogi klasy C; f) Przekrój dla drogi klasy D; g) Przekrój dla drogi klasy E

Z uwagi na fakt, że model podstawowy jest niewrażliwy na zmiany jakości nawierzchni, przebieg funkcji ugięcia przęsła w czasie dla różnych wartości nierówności nawierzchni jest taki sam. Zauważalna jest wrażliwość modelu rozszerzonego i szczegółowego na wpływ nierówności nawierzchni.

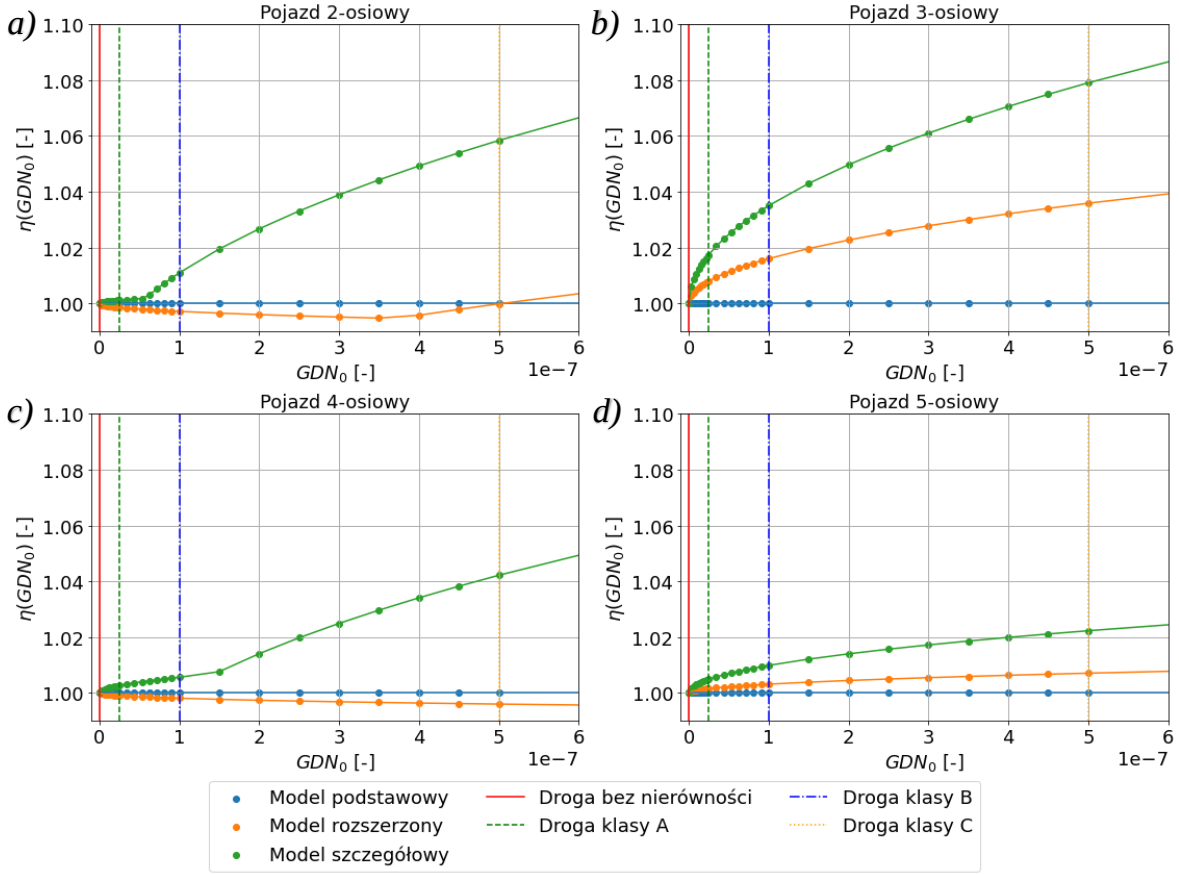
W celu oceny efektów wpływu nierówności nawierzchni przyjęto współczynnik $\eta(GDN_0)$, który można wyrazić jako:

$$\eta(GDN_0) = \frac{u_{max}(GDN_0)}{u_{max}(GDN_{0*} = 0)}$$

gdzie:

- $u_{max}(GDN_0)$ – maksymalne przemieszczenie pionowe dla klasy nierówności nawierzchni drogi,
- $u_{max}(GDN_{0*} = 0)$ – maksymalne przemieszczenie pionowe dla konstrukcji bez nierówności nawierzchni.

Na rys. 4.53 przedstawiono zbiorcze wyniki $\eta(GDN_0)$ dla wybranych pojazdów z podziałem na liczbę osi oraz model pojazdu.



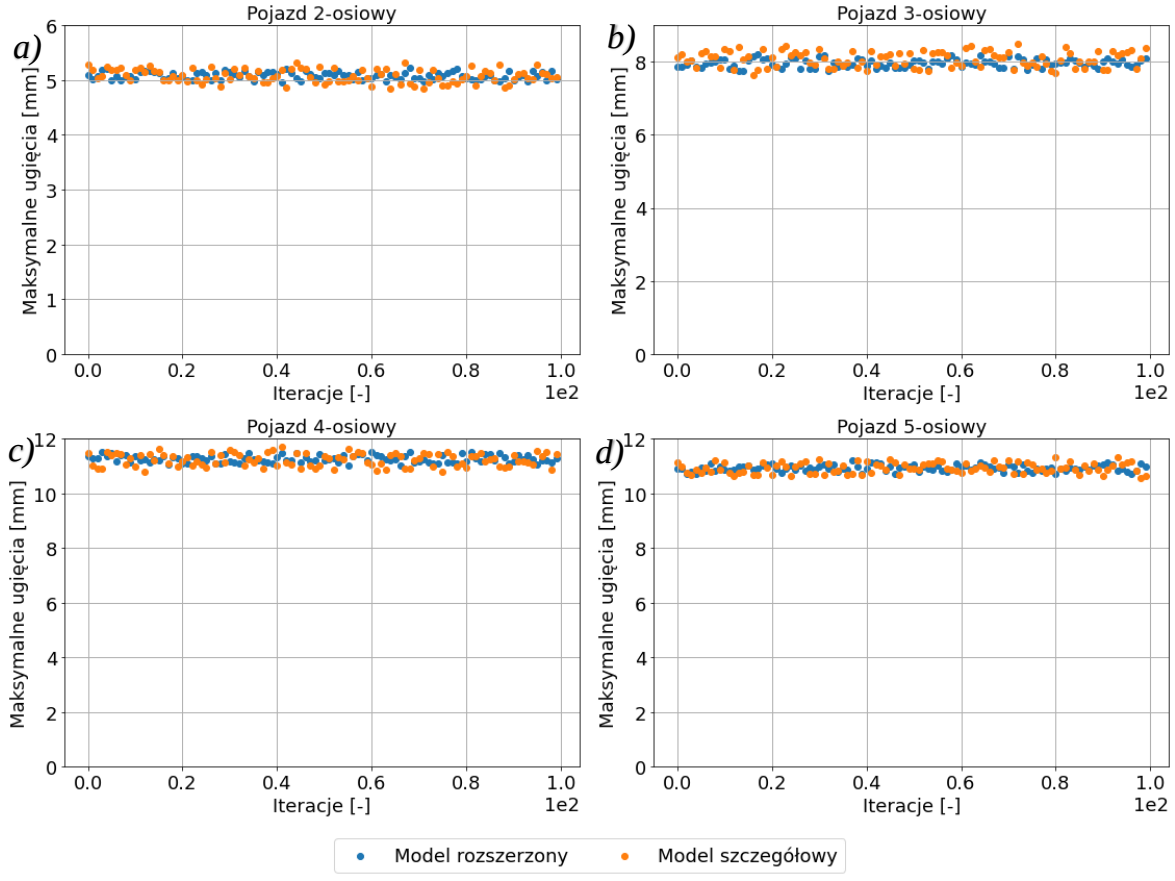
Rys. 4.53 Zestawienie współczynnika $\eta(GDN_0)$ dla porównywanych modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

Na podstawie przedstawionych wykresów można wyciągnąć następujące wnioski.:

- Nierówności nawierzchni wpływają na wartość współczynnika $\eta(GDN_0)$.
- Maksymalne wartości współczynnika $\eta(GDN_0)$są indywidualne dla każdego typu pojazdu i zależą od konfiguracji osi pojazdu.
- Zależność współczynnika $\eta(GDN_0)$ nie jest zależnością liniową.
- W zakresie dla drogi o klasy A oraz B, wartości przeciążenia dla pojazdów referencyjnych wahały się w granicach do 3.5 %. Te klasy oznaczają bardzo dobrą oraz dobrą jakość nawierzchni.

4.4.9 Wpływ różnych profili nierówności nawierzchni

W podpunkcie 4.4.8 założono we wszystkich analizach analogiczną funkcję nierówności nawierzchni i zmieniano jedynie amplitudę poszczególnych składowych. W przypadku analizy w podpunkcie 4.4.9 sprawdzono wpływ losowości przy generowaniu funkcji o tej samej klasie nierówności nawierzchni. Wygenerowano 100 różnych profili nierówności drogi klasy C z różnymi przesunięciami fazowymi poszczególnych składowych. Wszelkie analizy wykonano dla modeli rozszerzonych oraz szczegółowych modeli pojazdów referencyjnych. Wszystkie pozostałe parametry analizy pozostały niezmienne. Wyniki analiz przedstawiono na rys. 4.54.



Rys. 4.54 Wykresy maksymalnych ugięć przęsła dla modeli wybranych pojazdów referencyjnych, odpowiednio: a) pojazd 2-osiowy; b) pojazd 3-osiowy; c) pojazd 4-osiowy; d) pojazd 5-osiowy

Ukształtowanie nierówności przed i na obiekcie mostowym ma wpływ na odpowiedź dynamiczną konstrukcji. Pomimo zastosowania różnych funkcji nierówności nawierzchni, uwzględniających odmienne fazy przesunięcia ich składowych, uzyskiwane wyniki są do siebie zbliżone, co pozwala przyjąć, że losowo wybrany pojedynczy profil w wystarczająco dokładny sposób odwzorowuje rzeczywiste nierówności drogi.

4.5 Opis generatora populacji pojazdów

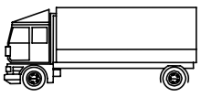
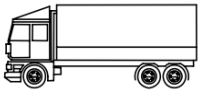
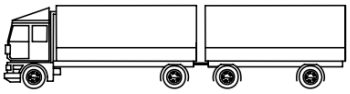
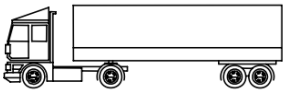
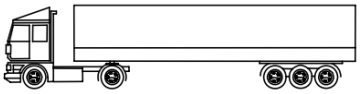
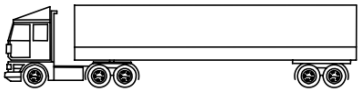
4.5.1 Wymagania oraz przepisy dotyczące podstawowych parametrów pojazdu.

Zgodnie z podrozdziałem 3.3 stworzono generator pojazdów, którego zadaniem będzie tworzenie danych wejściowych dla symulatora.

W celu zasymulowania realnego ruchu pojazdów po obiekcie mostowym, konieczne było wygenerowanie wielu różnych pojazdów poruszających się z różnymi prędkościami po obiekcie. Pierwszym założeniem, jest przyjęcie maksymalnej liczy osi równej 5.

Na podstawie badań i analiz przeprowadzonych oraz przedstawionych w [122] przyjęto do analizy następujące i przedstawiono w tab. 4.14 sylwetki pojazdów.

Tab. 4.14 Przyjęte typy oraz sylwetki pojazdów [122]

Typy pojazdów		Sylwetka pojazdu	Symbol pojazdu
Pojazdy 2-osiowe	Pojazdy osobowe	2P	
	Dostawcze	2P	
	Autobusy	2P	
Pojazdy 3-osiowe	Dostawcze	3P	
	Autobusy	3P	
Pojazdy 4-osiowe	Ciężarówki 4-osiowe	4P	
	Ciężarówki 2-osiowe + przyczepia 2-osiowa	2P+2P	
	Ciągnik siodłowy 2-osiowy + naczepa 2-osiowa	2C+2N	
Pojazdy 5-osiowe	Ciągnik siodłowy 2-osiowy + naczepa 3-osiowa	2C+3N	
	Ciągnik siodłowy 3-osiowy + naczepa 2-osiowa	3C+2N	

Przyjęto ogólne założenia dotyczące rozstawów oraz maksymalnych nacisków dla różnych typów pojazdów na podstawie Dyrektywy Unijnej 96/53/WE z dnia 25 lipca 1996 [12] oraz przedstawiono w tab. 4.15 proponowane zakresy nacisków na poszczególne osie.

Tab. 4.15 Maksymalne naciski na poszczególne osie [12]

Typy pojazdów		Maksymalne naciski na poszczególne osie [tony]					
		Oś 1	Oś 2	Oś 3	Oś 4	Oś 5	Masa całkowita
Pojazdy 2-osiowe	Pojazdy osobowe	1.5	3	-	-	-	3.5
	Dostawcze	10	11.5	-	-	-	18
	Autobusy	10	11.5	-	-	-	18
Pojazdy 3-osiowe	Dostawcze	10	9	9	-	-	25
	Autobusy	10	9	9	-	-	25
Pojazdy 4-osiowe	Ciężarówki 4-osiowe	9	9	9	9	-	32
	Ciężarówki 2-osiowy + przyczepa 2-osiowa	10	11.5	10	10	-	36
	Ciągnik siodłowy 2-osiowy + naczepa 2-osiowa	10	11.5	10	10	-	40
Pojazdy 5-osiowe	Ciągnik siodłowy 2-osiowy + naczepa 3 osiowa	10	11.5	8	8	8	40
	Ciągnik siodłowy 3-osiowy + naczepa 2-osiowa	10	9	9	10	10	40

Na podstawie danych z Głównego Pomiaru Ruchu [5] oraz informacji przedstawionych w podrozdziale 1.1 przyjęto i przedstawiono w tab. 4.16 udziały w ruchu poszczególnych typów pojazdów.

Tab. 4.16 Przyjęty udział procentowy poszczególnych typów pojazdów w ruchu drogowym [5]

Typy pojazdów		Udział poszczególnych typów pojazdów w ruchu drogowym	Przyjęty SDRR
		[%]	[poj./dobę]
Pojazdy 2-osiowe	Pojazdy osobowe	66	22440
	Dostawcze	6	2040
	Autobusy	0.12	40.8
Pojazdy 3-osiowe	Dostawcze	4.9	1666
	Autobusy	0.08	27.2
Pojazdy 4-osiowe	Ciężarówki 4-osiowe	2	680
	Ciężarówki 2-osiowe + przyczepa 2-osiowa	1.5	510
	Ciągnik siodłowy 2-osiowy + naczepa 2-osiowa	3.5	1190
Pojazdy 5-osiowe	Ciągnik siodłowy 2-osiowy + naczepa 3-osiowa	15	5100
	Ciągnik siodłowy 3-osiowy + naczepa 2-osiowa	0.9	306
Suma		100	34 000

4.5.2 Algorytm generacyjny rozstawów oraz nacisków na osie w pojazdach

Na podstawie pomiarów nacisków osi i mas pojazdów wykonywanych w pełnym zakresie występowania na drodze nr 11 w miejscowości Byczyna i zaprezentowanych w pracy [122], przyjęto i przedstawiono w tab. 4.17 wartości średnie μ nacisków na poszczególne osie oraz odchylenia standardowe σ dla różnych sylwetek pojazdów.

Tab. 4.17 Wartości nacisków na poszczególne osie na podstawie pomiarów ruchu nad DK 11 w miejscowości Byczyna [122]

Parametry	Sylwetka pojazdu												
	2P		3P			4P				2P + 2P			
Numer osi [-]	1	2	1	2	3	1	2	3	4	1	2	3	4
μ [kN]	33.4	41.2	50	61	41	44	45	47	39	44.6	64.1	37.4	36.9
σ [kN]	10.8	19.6	14.2	21.7	17.3	30.5	21.1	25.8	17.2	11.2	20.9	15.3	14.7
Parametry	Sylwetka pojazdu												
	2C + 2N				3C + 2N					2C + 3N			
Numer osi [-]	1	2	3	4	1	2	3	4	5	1	2	3	4	5
μ [kN]	47.1	47.5	32.6	32.9	54	69	46	48	49	54	71.9	51.2	51.9	51.8
σ [kN]	9.95	16.9	12.3	12.4	8.42	19.2	14.1	16.2	15.8	10.7	23.8	18.6	18.9	19.1

W pracy [122] sformułowano tezę o występowaniu trzech typów krzywych opisujących rozkłady nacisków osi pojazdów: krzywej typu normalnego, wykładniczego oraz gamma:
Funkcja gęstości rozkładu normalnego jest opisana wzorem:

$$f(P) = \frac{1}{\sigma \cdot \sqrt{2\pi}} e^{-\frac{(P-\mu)^2}{2\sigma^2}} \quad (4.36)$$

gdzie:

- P – obciążenie osi pojazdu,
- μ – wartość średnia,
- σ^2 – wariancja.

Funkcja gęstości rozkładu wykładniczego określona jest wzorem:

$$f(P) = \lambda e^{-\lambda P} \; dla \; P > 0 \quad (4.37)$$

gdzie:

- P – obciążenie osi pojazdu,
- $\mu = \frac{1}{\lambda}$ – wartość średnia,
- $\sigma^2 = \frac{1}{\lambda^2}$ – wariancja.

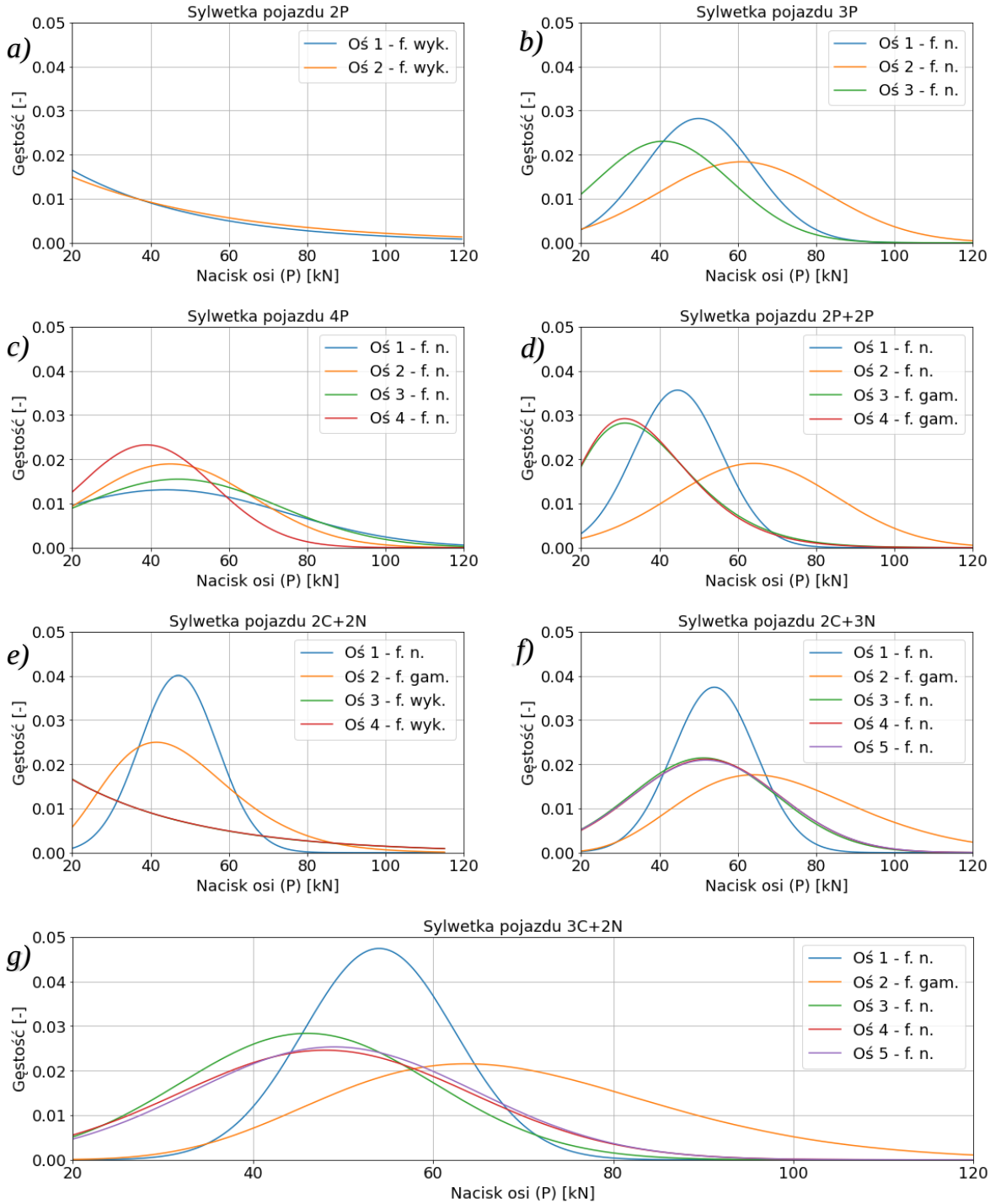
Funkcja gęstości rozkładu gamma jest opisana wzorem:

$$f(P) = \frac{1}{\Gamma(\alpha)\beta^{\alpha}} P^{\alpha-1} e^{-\frac{P}{\beta}} \; dla \; P > 0, \qquad (4.38)$$

gdzie:

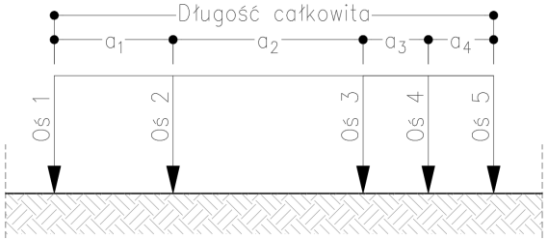
- P – obciążenie osi pojazdu,
- α – parametr kształtu,
- β – parametr skali,
- $\mu = \alpha \cdot \beta$ – wartość średnia,
- $\sigma^2 = \alpha \cdot \beta^2$ – wariancja.

Na podstawie badań [122] przyjęto i przedstawiono na rys. 4.55 następujące funkcje rozkładu dla poszczególnych sylwetek pojazdów.



Rys. 4.55 Przyjęte funkcje gęstości prawdopodobieństwa wartości nacisków osi dla sylwetek pojazdów, odpowiednio: a) sylwetka 2P; b) sylwetka 3P; c) sylwetka 4P; d) sylwetka 2P+2P; e) sylwetka 2C+2N; f) sylwetka 2C+3N; g) sylwetka 3C+2N

Zaznaczyć trzeba, że rozkłady te opierają się na pomiarach dokonanych na drodze krajowej w miejscowości Byczyna. Obejmują one stosunkowo krótki okres monitoringu i dlatego nie stanowią pełnego obrazu struktury ruchu. Aby jednoznacznie określić konfigurację osi pojazdu, konieczne jest przyjęcie ogólnego modelu pojazdu, który został zaprezentowany na rys. 4.56.



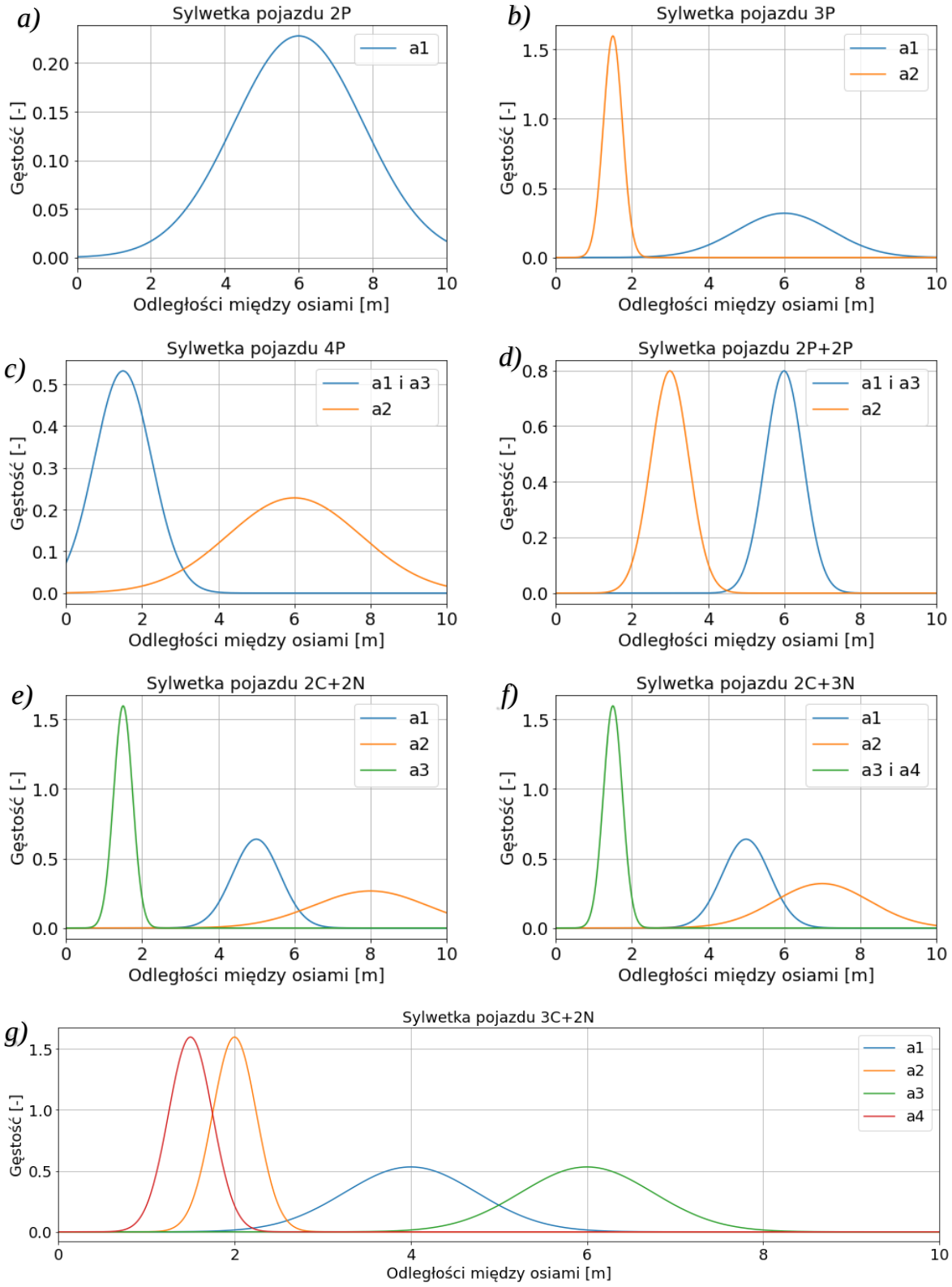
Rys. 4.56 Model ogólny pojazdu do 5 osi

W nawiązaniu do przyjętych oznaczeń oraz dyrektywy unijnej [12] przyjęto oraz przedstawiono w tab. 4.18 maksymalne i oczekiwane wartości rozstawów między osiami.

Tab. 4.18 Przyjęte zakresy rozstawów pomiędzy osiami

Liczba osi	Sylwetka pojazdu	Rozstawy pomiędzy osiami								Maksymalny rozstaw skrajnych osi
		a_1 [m]		a_2 [m]		a_3 [m]		a_4 [m]		
		max	σ	max	σ	max	σ	max	σ	
2	2P	10	6	-	-	-	-	-	-	12
3	3P	9	6	2	1.5	-	-	-	-	12
4	4P	4	1.5	8	6	4	1.5	-	-	12
	2P+2P	7	6	4	3	7	6	-	-	18.75
	2C+2N	6	5	10	8	2	1.5	-	-	16.5
5	2C+3N	6	5	9	7	2	1.5	2	1.5	16.5
	3C+2N	6	4	2.5	2	8	6	2	1.5	16.5

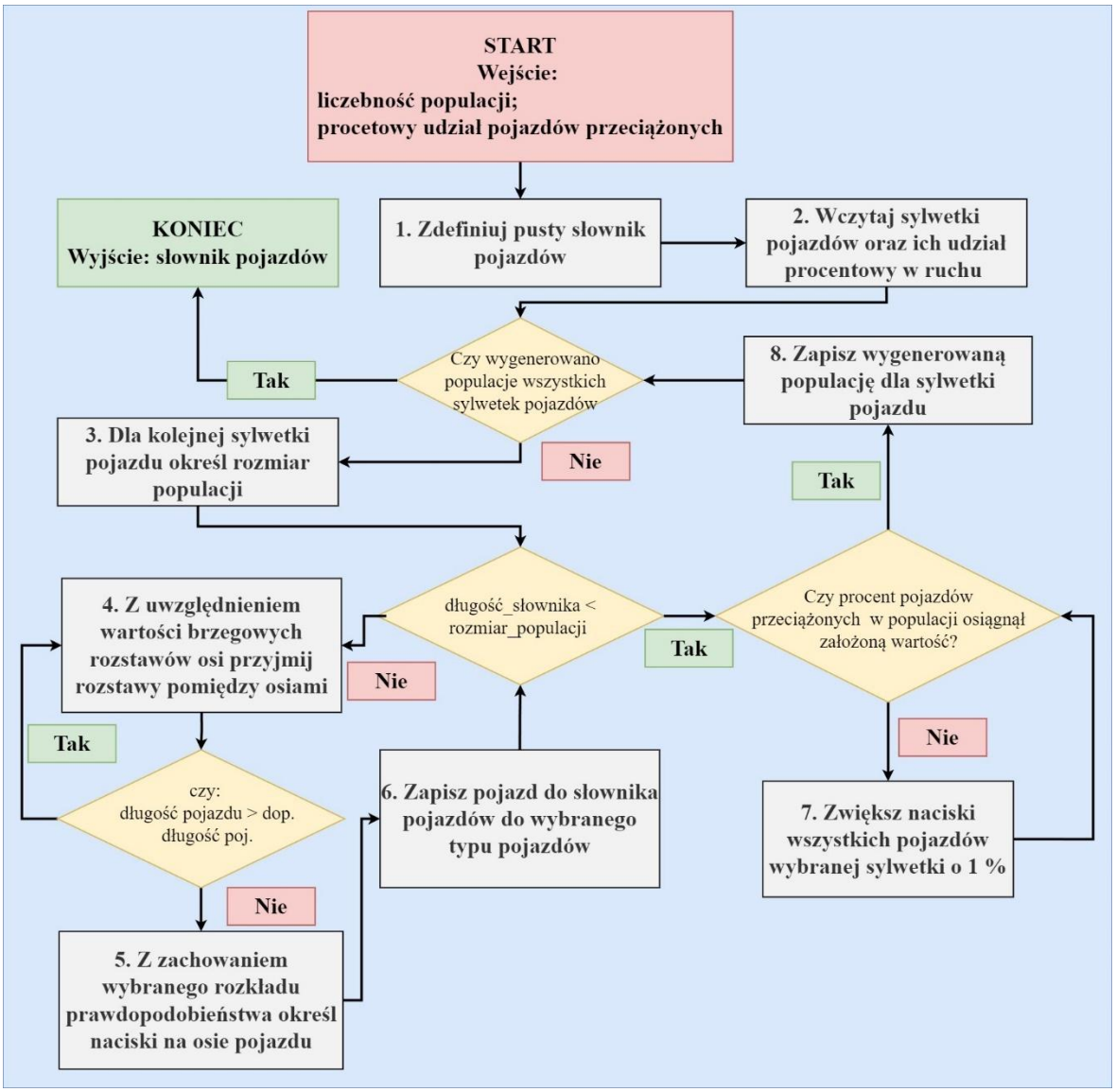
Przyjęto, że rozstaw między osiami jest opisany funkcją rozkładu normalnego. Przyjęte rozkłady przedstawiono na rys. 4.57.



Rys. 4.57 Przyjęte funkcje gęstości prawdopodobieństwa rozstawów osi dla sylwetek pojazdów, odpowiednio: a) sylwetka 2P; b) sylwetka 3P; c) sylwetka 4P; d) sylwetka 2P+2P; e) sylwetka 2C+2N; f) sylwetka 2C+3N; g) sylwetka 3C+2N

Uwzględniając wszystkie powyższe założenia napisano algorytm do automatycznego generowania modeli pojazdów o różnych parametrach rozkładu z uwzględnieniem ograniczeń. Uwzględniono możliwość „wymuszenia przeciążania pojazdów", które należy rozumieć jako ręcznie definiowany procent populacji pojazdów przekraczający wartości dopuszczalnej.

Schemat działania programu przedstawiono na rys. 4.58. Program przyjmuje jako parametry wejściowe procentowy udział pojazdów przeciążonych oraz liczebność populacji. Następnie uwzględniając ograniczenia oraz założenia generuje słownik pojazdów.



Rys. 4.58 Schemat blokowy Generatora Populacji Pojazdów

4.5.3 Weryfikacja wyników generatora

W poniższym podpunkcie przedstawiono wygenerowane populacje, które będą służyć do treningu oraz testowania sieci. W kontekście trenowania sztucznych sieci neuronowych ważne jest, aby testować je

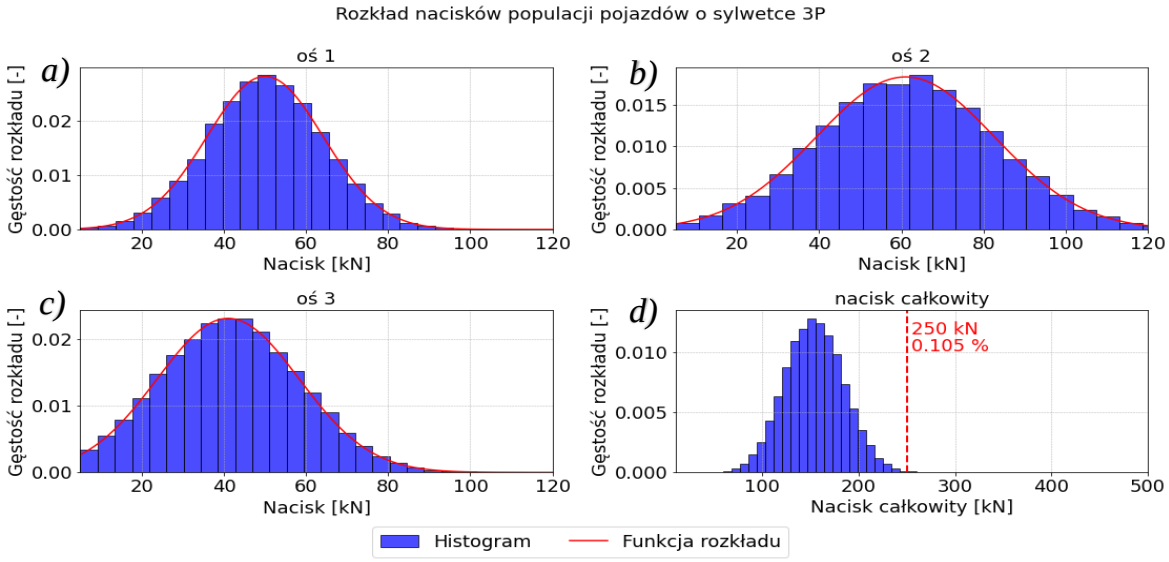
na danych, odrębnych i niezależnych od danych uczących, aby uniknąć problemu nadmiernego dopasowania się do danych uczących.

W tab. 4.19 przedstawiono przyjęty ilościowy udział poszczególnych typów i sylwetek pojazdów w ruchu drogowym dla populacji treningowej oraz testowej.

Tab. 4.19 Przyjęty udział poszczególnych modeli pojazdów w wygenerowanych populacjach

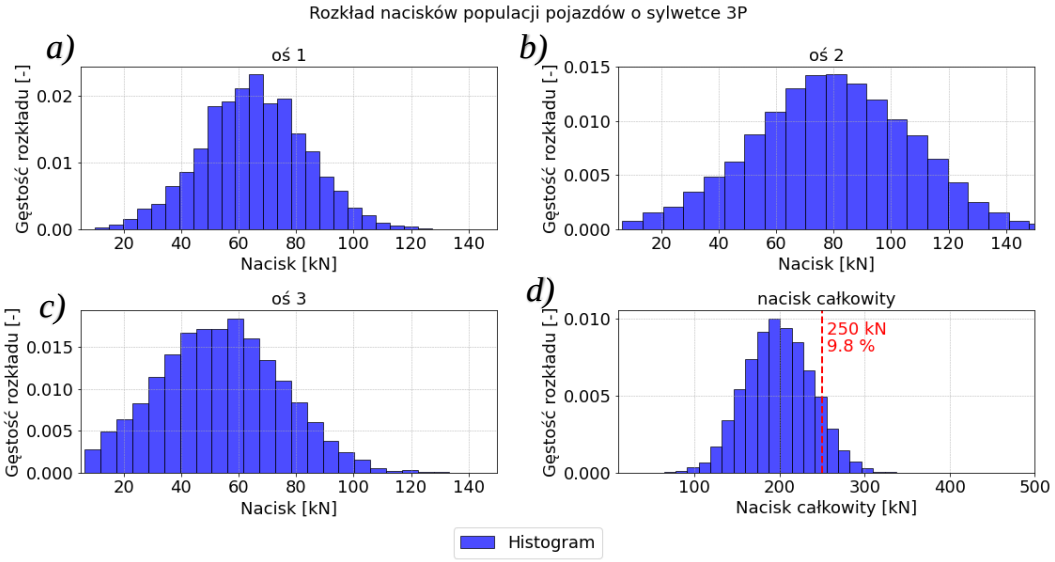
Liczba osi	Typ pojazdu	Sylwetka pojazdu	Udział poszczególnych modeli pojazdów w populacji	Wygenerowana populacja pojazdów treningowych	Wygenerowana populacja pojazdów testowych
			[%]	[szt.]	[szt.]
2	Dostawcze	2P	17.6	8800	1760
	Autobusy	2P	0.4	200	40
3	Dostawcze	3P	14.4	7200	1440
	Autobusy	3P	0.2	100	20
4	Ciężarówki 4 osiowe	4P	5.9	3000	600
	Ciężarówki 2 osiowe + przyczepia 2 osiowa	2P+2P	4.4	2200	440
	Ciągnik siodłowy 2 osie + naczepa 2 osiowa	2C+2N	10.3	5150	1030
5	Ciągnik siodłowy 2 osie + naczepa 3 osiowa	2C+3N	44.1	22050	4410
	Ciągnik siodłowy 3 osiowy + naczepa 2 osiowa	3C+2N	2.6	1300	260
Suma			100	50000	10000

Rys. 4.59 przedstawia rozkład nacisków pojazdów populacji o sylwetce 3P wygenerowany przy założeniu funkcji rozkładu przedstawionych na rys. 4.55.



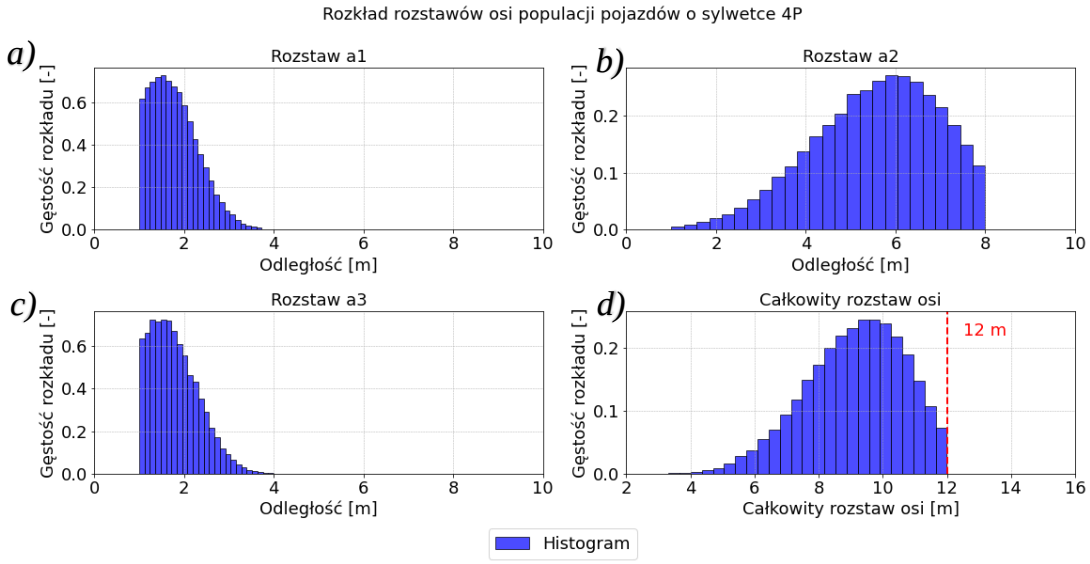
Rys. 4.59 Rozkład nacisków na osie populacji pojazdów o sylwetce 3P, odpowiednio: a) 1. oś, b) 2. oś, c) 3. oś, d) całkowity nacisk pojazdu

Zauważalny jest bardzo niski poziom pojazdów przeciążonych, których masa całkowita przekracza dopuszczalne wartości. Dokonano więc modyfikacji nacisków populacji zgodnie z krokiem 7 przedstawionego algorytmu (rys. 4.58). Przy założeniu występowania 10 % pojazdów przeciążonych program wygenerował następującą populację pojazdów (rys. 4.60).



Rys. 4.60 Rozkład nacisków populacji pojazdów o sylwetce 3P z założonym 10% poziomem przeciążenia, odpowiednio: a) 1. oś; b) 2. oś; c) 3. oś; d) całkowity nacisk pojazdu

Zgodnie z rys. 4.57 oraz z tab. 4.18 przyjęto odpowiednie wartości graniczne odległości między osiami i na tej podstawie wygenerowano wartości rozstawów. Na rys. 4.61 przedstawiono rozstawy dla populacji pojazdów o sylwetce 4P.



Rys. 4.61 Rozkład rozstawów pomiędzy osiami populacji pojazdów o sylwetce 4P, odpowiednio: a) rozstaw a_1; b) rozstaw a_2; c) rozstaw a_3; d) całkowity rozstaw osi

Ostatecznie stworzono więc algorytm, który generuje populację pojazdów o określonej liczebności, sylwetkach i rozkładach nacisków. Program ten będzie elementem składowym systemu zgodnie z przedstawionym w podrozdziale 3.3 opisem implementacji systemu.

4.6 Podsumowanie

W rozdziale 4 stworzono symulator dynamicznej odpowiedzi konstrukcji mostowej, zgodnie z schematem budowy systemu przedstawionym na rys. 4.1, który składa się z wirtualnego modelu mostu, generatora populacji pojazdów oraz programu zarządzającego (Program zarządzający, będący kluczowym elementem informatycznym systemu, odpowiada za integrację pozostałych komponentów i zapewnienie ich poprawnej współpracy. Ze względu na jego techniczny charakter oraz szczegółowe aspekty implementacyjne, został on opisany w załączniku B.3, a nie w głównej części pracy).

W podrozdziale 4.2 i 4.3 opisano i zweryfikowano działanie wirtualnego modelu mostu oraz modeli pojazdów. Na podstawie przeprowadzonej analizy w podpunkcie 4.3.6 oraz 4.3.7 stwierdzono zbieżność uzyskanych wyników dla różnych modeli. Oceniono złożoność obliczeniową jako funkcję czasu niezbędnego do uzyskania wyniku dla różnych modeli. W trakcie przeprowadzanych obliczeń wyznaczono uśredniony czas potrzebny do wykonania jednej analizy dynamicznej. Wyniki przedstawiono w tab. 4.20.

Tab. 4.20 Uśredniony czas potrzebny do wykonania 1 analizy

Model	Uśredniony czas wykonania 1 analizy	Możliwość analizy wielowątkowej	Uśredniony czas wykonania 1 analizy z wykorzystaniem wielowątkowego programowania
	[s]		[s]
Podstawowy	0.04	Tak	0.0025
Rozszerzony	0.1	Tak	0.00625
Szczegółowy	0.4	Tak	0.025
MES	6.20	Nie	6.20

Autorskie implementacje modeli napisane w języku programowania Python działają znacznie szybciej niż model MES stworzony przy użyciu komercyjnego programu SOFiSTiK, w którym jest nałożone ograniczenie licencyjne umożliwiające wykonanie tylko jednej analizy dynamicznej w tym samym czasie. W stworzonym oprogramowaniu możliwe jest wykonywanie operacji obliczeniowej na wielu rdzeniach jednocześnie.

Z uwagi na złożoność obliczeniową, do dalszej analizy nie uwzględnia się modelu MES z uwagi na znaczny czas obliczeniowy wymagany do analizy 1 przejazdu. Z uwagi na uproszczenia i brak możliwości uwzględniania parametrów zawieszenia, które wpływają na wyniki odpowiedzi konstrukcji, nie wykorzystuje się w dalszej części pracy podstawowego modelu pojazdu.

Obliczenia wykonywano na domowym komputerze autora, wyposażonym w procesor Ryzen 3800X. Szacunkowo symulator dynamicznej odpowiedzi konstrukcji mostowej potrafił generować 140 tys.

przejazdów na godzinę modeli szczegółowych oraz 500 tys. przejazdów modeli rozszerzonych. Uznano, że na cele rozprawy ta wydajność jest wystarczająca.

W podrozdziale 4.4 dokonano analizy wpływu wybranych parametrów na wyniki symulacji i szczegółowe wnioski zostały przedstawione w każdym z podpunktów podrozdziału. W tab. 4.21 zestawiono analizowane parametry symulacji z podziałem na parametry dotyczące mostu oraz pojazdu.

Tab. 4.21 Analizowane parametry symulatora odpowiedzi konstrukcji mostowej.

Parametr	Pojazd	Obiekt mostowy	Czy parametr jest stały dla populacji pojazdów
Prędkość przejazdu	●		Nie, pojazdy poruszają się z losowymi prędkościami. Cechują się zmiennymi naciskami oraz stanem zawieszenia.
Sztywność zawieszenia	●		
Tłumienie zawieszenia	●		
Tłumienie konstrukcji		●	Tak, parametry konstrukcji obiektu mostowego są stałe w czasie. Ewentualne zmiany z uwagi na procesy degradacji zachodzą tak wolno, że postanowiono je pominąć.
Masa własna konstrukcji		●	
Rozpiętość konstrukcji		●	
Nierówność nawierzchni		●	

Dodatkowo zaznaczono parametry zmienne i stałe, w kontekście tworzonego systemu. Parametry badanego obiektu, takie jak rozpiętość, masa własna, tłumienie konstrukcji nie ulegają zmianom w czasie. System, więc będzie musiał automatycznie stworzyć za pomocą sieci neuronowych ukrytą reprezentację obiektu. Pojazdy poruszające się obiekcie mostowym, cechują się dużą różnorodnością parametrów przejazdu. System musi być zdolny albo określać te parametry, np. prędkość przejazdu.
W tab. 4.22 przedstawiono zbiorcze zakresy współczynnika przeciążenia konstrukcji dla analizowanych wpływów parametrów pojazdu oraz obiektu mostowego na odpowiedź konstrukcji mostowej. Przedstawiono maksymalną oraz minimalną wartość współczynnika przeciążenia $\boldsymbol{\eta}$.

Tab. 4.22 Zakresy zmian wartości współczynnika przeciążenia dla analizowanych parametrów

Parametr [jednostka]	Zakres wartości <min, max>	Model *)	2-osiowy		3-osiowy		4-osiowy		5-osiowy	
			η(min)	η(max)	η(min)	η(max)	η(min)	η(max)	η(min)	η(max)
Prędkość przejazdu [m/s]	$v =< 0{,}30 >$	P	1.00	1.20	1.00	1.19	1.00	1.20	1.00	1.16
		R	1.00	1.17	1.00	1.15	1.00	1.15	1.00	1.12
		S	1.00	1.18	1.00	1.17	1.00	1.17	1.00	1.14
Sztywność zawieszenia [-]	$k_{mod} = < 0.5{,}2.5 >$	P	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		R	0.99	1.01	1.00	1.02	0.99	1.01	0.99	1.01
		S	0.99	1.01	0.99	1.04	0.99	1.01	0.99	1.00
Tłumienie zawieszenia [-]	$c_{mod} = < 0.0{,}0.75 >$	P	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
		R	0.99	1.01	0.98	1.02	1.0	1.02	1.00	1.01
		S	0.99	1.01	1.0	1.02	0.99	1.01	0.97	1.02
Tłumienie konstrukcji [-]	$\beta =< 0, 0.03 >$	P	0.97	1.00	0.98	1.00	0.99	1.00	0.98	1.00
		R	0.97	1.00	0.97	1.00	1.00	1.00	0.98	1.00
		S	0.97	1.00	0.97	1.00	0.99	1.00	0.98	1.00
Masa własna konstrukcji [kg/mb]	$m_b = < 100{,}9980 >$	P	1.0	1.22	1.0	1.25	1.0	1.29	1.0	1.19
		R	1.0	1.2	1.0	1.23	1.0	1.26	1.0	1.17
		S	0.99	1.2	0.99	1.24	0.99	1.27	1.0	1.18
Rozpiętość konstrukcji [m]	$L_t =< 10, 50 >$	P	0.81	1.03	0.78	1.09	0.73	1.13	0.76	1.29
		R	0.82	1.04	0.8	1.14	0.75	1.25	0.75	1.41
		S	0.82	1.05	0.8	1.15	0.75	1.26	0.74	1.43
Nierówność nawierzchni [-]	$GDN_0 =$ < 0, klasa C nawierzchni >	P	1.00	1.0	1.00	1.0	1.00	1.0	1.00	1.0
		R	0.99	1.0	1.00	1.03	0.99	1.0	1.00	1.01
		S	1.00	1.05	1.00	1.07	1.00	1.04	1.00	1.02
*) Skróty oznaczeń modeli: P – Model podstawowy; R – Model rozszerzony; S – Model szczegółowy.										

Kluczowymi parametrami, mającymi wpływ na odpowiedź konstrukcji mostowej jest prędkość pojazdu, masa własna, rozpiętość oraz tłumienie konstrukcji, a także nierówności nawierzchni.

Zmienne parametry pojazdu zostały uwzględnione w Generatorze Populacji Pojazdów, który tworzy odpowiednio liczną i zgodną z przyjętymi założeniami populację pojazdów. Generator Populacji Pojazdów uwzględnia losowość ruchu oraz stanu zawieszenia zgodnie z podpunktem 4.3.4. Działanie generatora zostało opisane i zweryfikowane w podrozdziale 4.5. Generator umożliwia generowanie dowolnie licznej populacji pojazdów.

5 Implementacja systemu i analiza błędów określenia nacisków

5.1 Podstawowe założenia

W rozdziale 5 przedstawiono implementację oraz weryfikację analizowanych wariantów Auto-adaptacyjnych Systemów Identyfikacji Obciążenia (AutoSIO). Rozpatrzono 3 warianty:

1. **AutoSIO_S**: Quasi-statyczny wariant systemu, który został przedstawiony w podrozdziale 5.2.
2. **AutoSIO_SD**: Wariant systemu uwzględniający podejście statyczne i dynamiczne, który został omówiony w podrozdziale 5.3.
3. **AutoSIO_D**: Dynamiczny wariant systemu, który został omówiony w podrozdziale 5.4.

Podstawowe założenia dla wszystkich wariantów systemu:

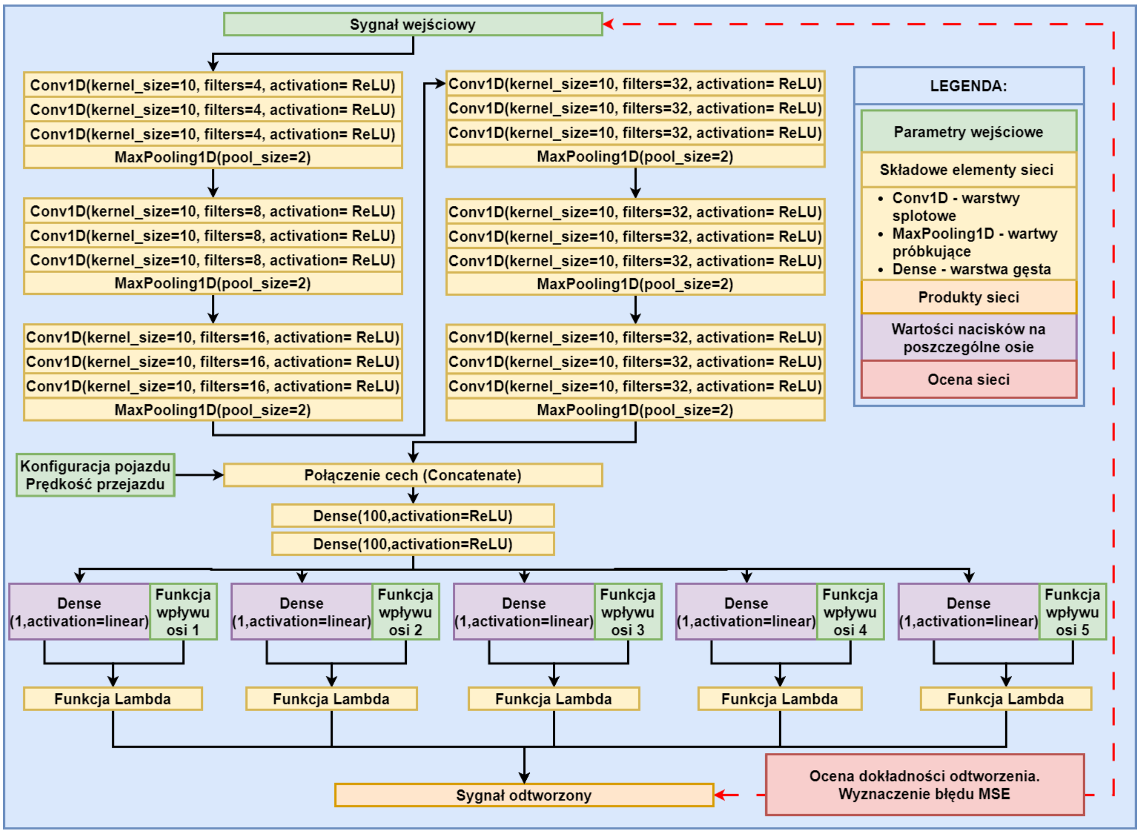
- Stworzony system musi być auto-adaptacyjny poprzez wykorzystanie uczenia nienadzorowanego. Musi poprawiać swoją dokładność działania, bez konieczności nadzoru ludzkiego.
- Sygnały w formie dyskretnej analizowane przez sieć są przeskalowane do pojedynczego wektora o stałej długości k. Pierwsza wartość sygnału jest dla chwili czasowej, w której pierwsza oś pojawia się na obiekcie mostowym. Ostatnia wartość sygnału odnosi się do chwili czasowej, w której ostatnia oś zjeżdża z obiektu. Krok czasowy między poszczególnymi wartościami sygnału jest stały.
- Analizowana jest odpowiedź mostu jako pionowe przemieszczenie w środku rozpiętości.
- Generator populacji pojazdów, który symuluje System Identyfikacji Pojazdu, dostarcza informacje o prędkości przejazdu, liczbie osi, rozstawach osi oraz współrzędnej lokalizacji osi pojazdu w trakcie przejazdu.
- Na obiekcie mostowym znajduje się jeden pojazd, który ma maksymalnie 5 osi.
- Głównym celem systemu jest określenie całkowitego nacisku pojazdu, ponieważ to do niej odnoszą się wszystkie rozporządzenia oraz przepisy dot. ruchu pojazdów.
- Celem dodatkowym jest określenie nacisków statycznych na poszczególne osie oraz grupy osi.

5.2 System AutoSIO_S odtwarzający quasi-statyczną odpowiedź obiektu mostowego

5.2.1 Struktura systemu

W opracowanym systemie AutoSIO_S zastosowano podejście Moses'a. Zadaniem sieci jest na podstawie otrzymanych parametrów wejściowych przewidzieć quasi-statyczną odpowiedź konstrukcji obiektu mostowego na przejazd pojazdu. W przeciwieństwie do tradycyjnych podejść, wykorzystano nienadzorowaną sieć neuronową, której zadaniem jest wypracowanie własnej logiki i „zrozumienia" zagadnienia interakcji pojazd-most.

Schemat budowy i działania został zaprezentowany na rys. 5.1. Dokładny opis implementacji został przedstawiony w podrozdziale C.3, załącznika C.



Rys. 5.1 Schemat blokowy systemu AutoSIO_S

Rozpatrywany system charakteryzuje się liniową budową, w której informacja wejściowa jest przesyłana przez warstwy sieci neuronowej w jednym kierunku, bez rozgałęzień, aż do warstwy wyjściowej oraz do określenia błędu odtworzenia funkcji.

Wariant AutoSIO_S wymaga następujących parametrów wejściowych:

1. **Sygnał wejściowy:** Jest dyskretnym wektorem wartości mierzonej odpowiedzi konstrukcji mostowej. Pierwsza wartość opisuje odpowiedź konstrukcji mostowej w chwili pojawienia się pierwszej osi na obiekcie mostowym, a ostatnia wartość, moment, kiedy ostatnia oś zjeżdża z obiektu. Założono stały krok czasowy między poszczególnymi wartościami.

2. **Konfiguracja pojazdu:** Wektor odległości między osiami o stałej długości. Domyślnie maksymalna liczba osi, którą uwzględniono to 5, dlatego wektor ma 4 parametry. Jeśli rozpatrywany pojazd posiada mniejszą niż maksymalna liczbę osi, to odpowiednie odległości przyjmują wartość 0.
3. **Prędkość pojazdu**: Wartość prędkości przejazdu pojazdu w m/s. Założono stałą wartość prędkości przejazdu.
4. **Funkcja wpływu**: Funkcja wpływu mierzonej wielkości odpowiedzi obiektu mostowego na zmieniające się położenie pojazdu określonego w dziedzinie czasu. Tradycyjną linią wpływu wybranej wielkości nazywamy wykres zależności wybranej wielkości w dziedzinie współrzędnej określającej położenie uogólnionej siły jednostkowej. Funkcję wpływu można uzyskać analitycznie z wykorzystaniem znajomości mechaniki lub doświadczalnie, poprzez próbne obciążenie pojazdem o znanej masie oraz konfiguracji osi. Tak opisana funkcja wpływu jest funkcją $FW(x)$, gdzie x oznacza współrzędną przyłożenia siły. W przypadku pojazdu, który składa się z wielu osi dla każdej osi możemy stworzyć funkcję wpływu danej osi. Zakładając, że pojazd porusza się ze stałą prędkością, to funkcja wpływu i-tej osi będzie wyrażać się jako $FW_i\big(x_i = c(t - t_i)\big)$, gdzie:
    a. $\boldsymbol{c}$: Prędkość przejazdu.
    b. $\boldsymbol{t}$: Zmienna czasu.
    c. $\boldsymbol{t_i}$: Różnica czasu pomiędzy pojawieniem się i-tej, a pierwszej osi na obiekcie mostowym.
    d. $\boldsymbol{x_i}$: Lokalizacja osi w czasie.

    Funkcje Wpływu dla rzędnych znajdujących się poza obiektem przyjmą wartość 0.

Wykorzystano następujące elementy składowe sieci:

1. **Conv1D(kernel_size, filters, activation)**: Warstwa konwolucyjna 1D, która analizuje lokalne wzorce w danych sekwencyjnych, takich jak sygnały czasowe. Poniżej przedstawiono opis parametrów:
    a. **kernel_size**: Rozmiar okna splotowego (liczba sąsiednich punktów, które są analizowane jednocześnie).
    b. **filters**: Liczba filtrów używanych do przetwarzania danych, co określa liczbę wyodrębnionych cech.
    c. **activation**: Funkcja aktywacji, np. ReLU (Rectified Linear Unit), stosowana w celu wprowadzenia nieliniowości do sieci neuronowej.
2. **MaxPooling1D(pool_size):** Warstwa redukująca wymiary danych, wybierająca maksymalną wartość w zadanym oknie danych, co pozwala na redukcję liczby przetwarzanych danych i zmniejszenie ryzyka przeuczenia. Poniżej przedstawiono opis parametrów:

   a. **pool_size:** Rozmiar okna, w którym następuje próbkowanie danych.
3. **Dense(k, activation)**: Warstwa w pełni połączona, gdzie każdy neuron jest połączony z każdym neuronem w poprzedniej warstwie. Przekształca wyodrębnione cechy w bardziej złożone reprezentacje lub końcowe wyniki. Poniżej przedstawiono opis parametrów:
   a. **k**: Liczba neuronów w warstwie.
   b. **activation**: Funkcja aktywacji, np. ReLU lub linearna, w zależności od potrzeb sieci.
4. **Concatenate**: Operacja łączenia cech wyodrębnionych z różnych ścieżek sieci, co umożliwia wspólne przetwarzanie różnych źródeł informacji. Łączy wyniki z różnych warstw w jedną sekwencję cech, którą można dalej przetwarzać.
5. **Funkcja Lambda**: Niestandardowa funkcja używana do przekształceń na wyjściu warstwy. W tym przypadku funkcja przemnaża szacowaną wartość nacisku na wybraną oś, przez funkcję wpływu wybranej osi.

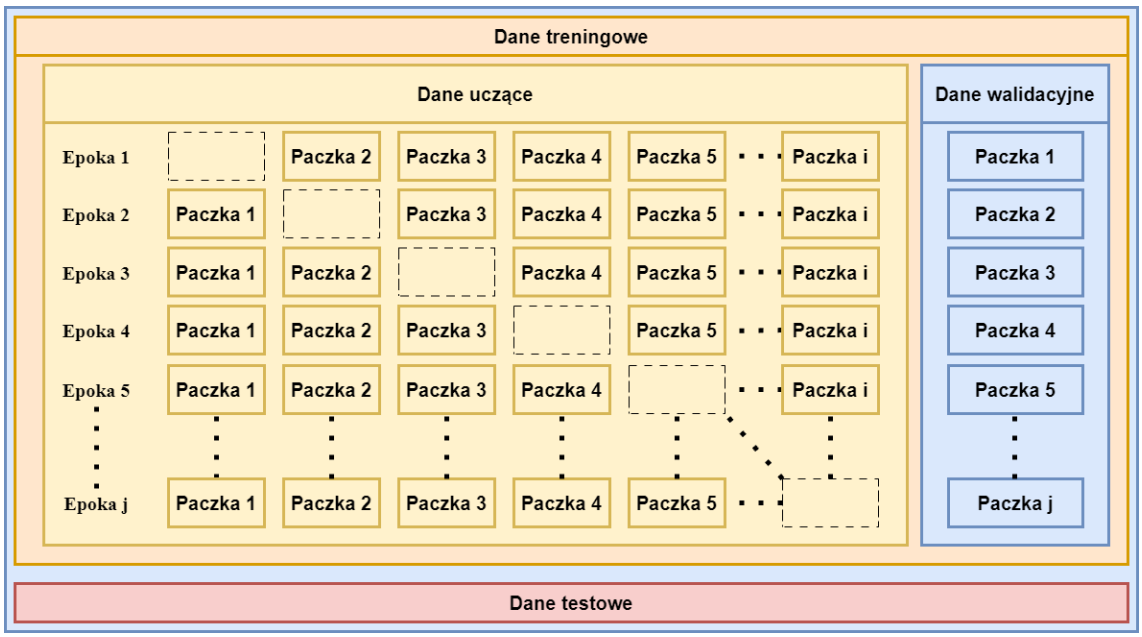
Na schemacie (rys. 5.1) zostały wyszczególnione następujące bloki sieci neuronowej:

- Wejściem do bloku Conv1D (warstwy CNN) jest sygnał odpowiedzi konstrukcji mostowej. Zadaniem warstw CNN jest ekstrakcja cech poprzez wykorzystanie filtrów oraz funkcji splotu. Analizowany sygnał jest wektorem, dlatego wykorzystano 1-wymiarowe warstwy splotowe. Jako funkcję aktywacji zastosowano funkcję ReLU, a następnie redukcję wymiarowości sygnału. Elementem wyjściowym jest wektor cech sygnału.
- Do uzyskanego wektora cech dodawane są informacje o konfiguracji pojazdu oraz prędkości przejazdu. Tak przygotowany wektor cech jest ze sobą łączony, a następnie analizowany przez warstwy gęste sieci neuronowej. W najwęższym miejscu, sieć neuronowa posiada tylko 5 neuronów, które odpowiadają za wartości nacisków poszczególnych osi.
- Wartości wyznaczone przez neurony są mnożone przez Funkcje Wpływu dla odpowiednich osi w funkcji lambda *(dop. Funkcja lambda jest funkcją anonimową w języku Python, zdefiniowaną za pomocą słowa kluczowego lambda co oznacza, że nie musi mieć nazwy i może być zdefiniowana tam, gdzie jest potrzebna. Funkcje te mogą przyjmować dowolną liczbę argumentów, ale mogą mieć tylko jedno wyrażenie, co oznacza, że nie można w nich mieć złożonych operacji, jak pętli czy wielu linii kodu.)*. Produkty są następnie dodawane do siebie i wynikiem jest sygnał wyjściowy.
- System porównuje różnice pomiędzy sygnałem wejściowym, a sygnałem wyjściowym poprzez wyznaczenie błędu średniokwadratowego. Ponieważ sygnały są w formie dyskretnej, dla każdego punktu sygnału można określić różnicę między wartością oczekiwaną (wejściową), a otrzymaną (wyjściową). Następnie algorytm optymalizacyjny dąży do minimalizacji błędu poprzez np. wsteczną propagację.

5.2.2 Proces treningu sieci

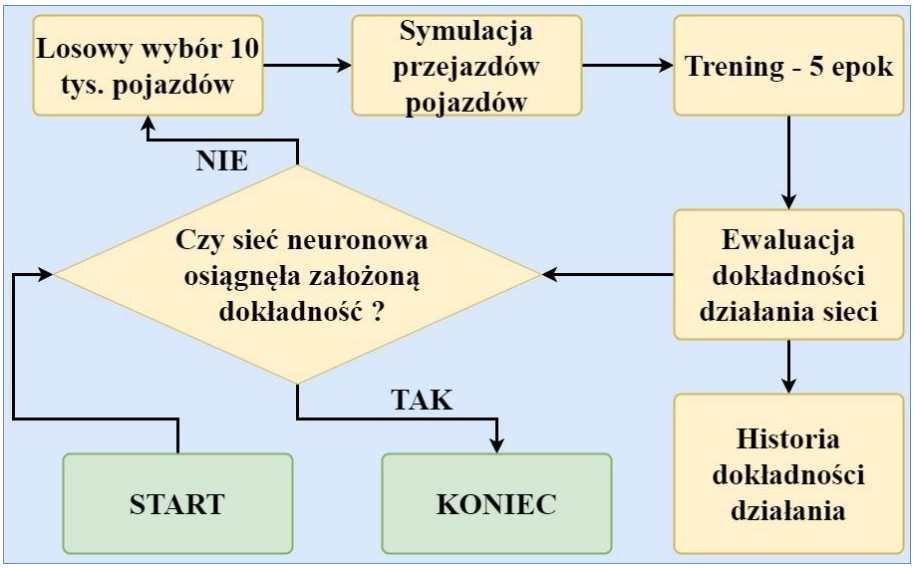
Trenowanie oraz testowanie systemu opartego o sieci neuronowe, odbywało się na 2 niezależnych populacjach pojazdów wygenerowanych na podstawie algorytmu opisanego w podrozdziale 4.5.2. Przyjęto populacje treningową oraz testową zgodnie z tab. 4.19.

- Pierwszą grupę stanowiły dane treningowe, na których sieć neuronowa była uczona odtwarzać sygnały i szacować naciski. Dodatkowo w trakcie uczenia dane treningowe można podzielić na dane treningowe oraz dane walidacyjne.
- Druga grupa, o wielkości 20% grupy treningowej, stanowiła zbiór testowy, na którym sprawdzano dokładność oszacowań systemu. Dane te nie były wcześniej wykorzystane do treningu sieci.



Rys. 5.2 Podział danych na dane treningowe (uczące oraz walidacyjne) oraz testowe

Dane uczące i walidacyjne są zbiorem danych treningowych i są wykorzystywane do bezpośredniego uczenia sieci neuronowej, tj. do dostosowania jej parametrów w taki sposób, aby jak najlepiej odwzorować zależności występujące w tych danych. Z kolei dane walidacyjne służą do oceny dokładności działania sieci neuronowej w trakcie procesu trenowania i pozwalają one na monitorowanie i detekcję problemu przeuczenia. W każdej epoce jest nowy podział na dane walidacyjne oraz uczące.
Dane testowe to zbiór informacji, które nigdy nie były wykorzystane w procesie uczenia. Służą do monitorowania ostatecznej efektywności działania sieci. Różnica między tymi zestawami polega na ich roli w procesie tworzenia sieci neuronowej systemu: dane treningowe kształtują połączenia między poszczególnymi neuronami w sieci, natomiast dane testowe pozwalają ocenić, jak dobrze sieć będzie działała na nowych, niewidzianych wcześniej danych.
Na rys. 5.3 przedstawiono schemat blokowy cyklu treningowego sieci.



Rys. 5.3 Schemat blokowy cyklu treningowego sieci

Celem analiz było nie tylko potwierdzenie dokładności oszacowań, ale także zbadanie, w jaki sposób różne parametry pojazdów, takie jak masa czy prędkość, wpływają na dokładność działania sieci neuronowej. Z uwagi na znaczną liczbę przejazdów oraz rozmiar pamięci operacyjnej, który należy zarezerwować do analizy danych, jednorazowo do sieci neuronowej importowano 10 000 pojazdów, symulowano ich przejazd, a następnie dzielono dane na mniejsze paczki danych zgodnie z rys. 5.2. Cykl treningowy obejmował 5 epok, po których następowała ewaluacja sieci neuronowej poprzez testowanie jej na danych, z którymi wcześniej nie miała styczności, aby ocenić błąd i dokładność oszacowania. W trakcie ewaluacji dla danych testowych oceniano błędy oszacowań w stosunku do wartości rzeczywistych. Ponieważ zbiór testowy był stały dla każdego cyklu, możliwe było śledzenie zmian dokładności działania sieci oraz obserwacja, jak błędy dla poszczególnych pojazdów zmieniały się w czasie.

Krzywa uczenia to graficzna reprezentacja zmiany wartości miary dokładności systemu, na przykład błędu odwzorowania, w zależności od cykli bądź epok trenowania. Prezentuje, jak sieć neuronowa uczy się rozwiązywania zadania w trakcie treningu, co pozwala na ocenę konwergencji sieci oraz identyfikację potencjalnych problemów, takich jak przeuczenie lub niedouczenie.
Jako funkcję błędu opisującą różnicę pomiędzy oczekiwanym efektem pracy sieci, a rzeczywistą wartością, przyjęto błąd średniokwadratowy (*ang. Mean Squared Error,* MSE), który jest popularną funkcją stosowaną w regresji. MSE mierzy średni kwadrat różnic między przewidywanymi, a rzeczywistymi wartościami. Jest wyrażony wzorem:

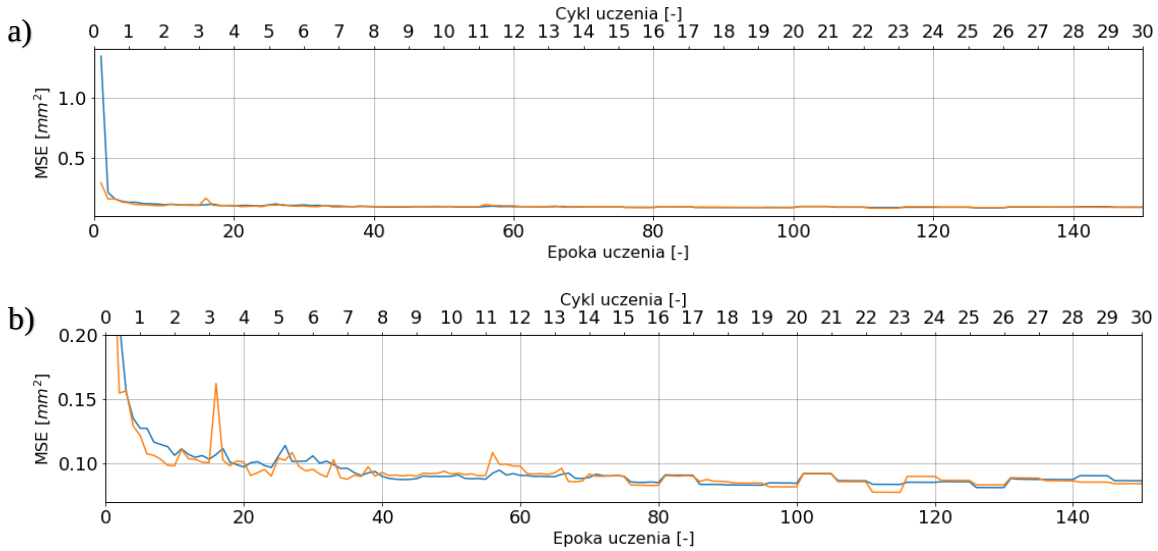
$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{Y_i}\right)^2$$

gdzie:

- Y_i - rzeczywiste wartości,

- $\widehat{Y}_i$ - to wartości przewidziane przez sieć neuronową systemu,
- n - liczba próbek.

Wykresy krzywych uczenia przedstawionych na rys. 5.4 prezentują zmianę błędu średniokwadratowego w trakcie procesu trenowania sieci neuronowej.

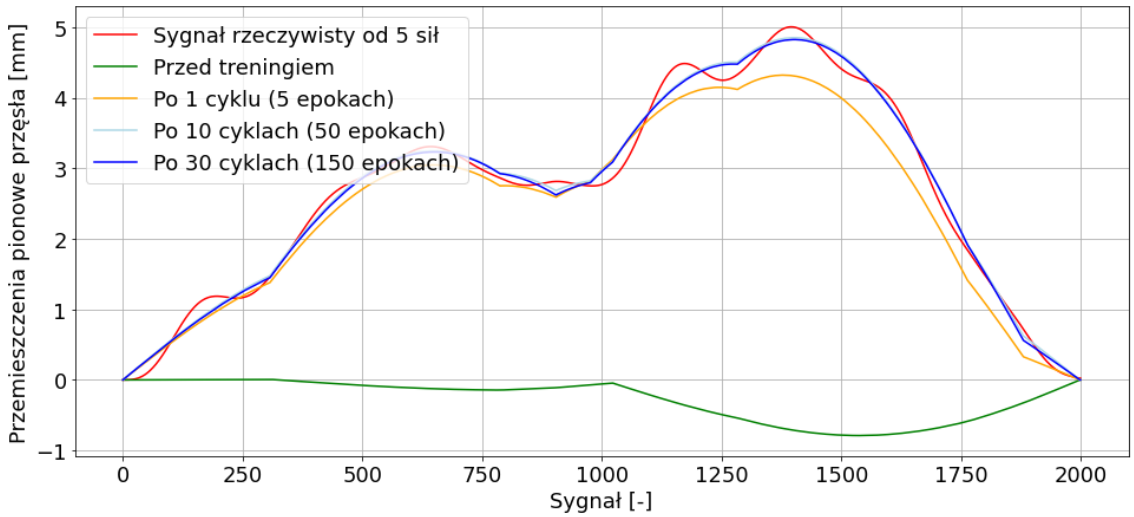


Rys. 5.4 Wartości krzywych uczenia w trakcie uczenia systemu AutoSIO_S
a) Wykres przedstawia pełny zakres wartości MSE;
b) Wykres przedstawia zakres 0.08-0.2 MSE

Pierwszy wykres obejmuje pełen zakres 30 cykli uczenia (150 epok), pokazując początkowy wysoki poziom błędu, który znacząco spada w pierwszych fazach treningu. Następnie następuje dalsza minimalizacja błędu, by ostatecznie zatrzymał się on na względnie stałym poziomie.

Istotną obserwacją jest zbieżność krzywych błędu dla danych treningowych i walidacyjnych, co świadczy o niewystąpieniu przeuczenia (ang. *overfitting*) lub niedouczenia (ang. *underfitting*) systemu. Dodatkowo, charakterystyczne dla procesu są nieregularności pojawiające się co 5 epok, manifestujące się przez skoki wartości błędu. Zjawisko to wynika z wprowadzenia do sieci nowych danych co cykl, czyli, co 5 epok.

Na rys. 5.5 przedstawiono efekty działania sieci neuronowej w trakcie uczenia się. Wybrano pojazd o sylwetce 2C+3N (ciągnik siodłowy + naczepa) z populacji testowej, a następnie w różnych fazach treningu sprawdzano dokładność odtworzenia sygnału wejściowego.



Rys. 5.5 Porównanie sygnału odtworzonego przez wariant systemu AutoSIO_S z sygnałem wejściowym dla różnych faz treningu

Dodatkowo dla analizowanego pojazdu porównano rzeczywiste naciski pojazdu podane w symulatorze z efektami nacisków określonymi przez wariant systemu AutoSIO_S w różnych fazach treningu, włączając w to fazę przed rozpoczęciem treningu. Wyniki przedstawiono w tab. 5.1.

Tab. 5.1 Określone wartości nacisków statycznych na różnych etapach treningu systemu

Etap treningu	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
Zamodelowana wartość	54.82	54.26	48.57	47.96	53.87	109.07	150.40	259.47
Przed treningiem	0.30	-4.49	2.28	1.29	-26.42	-4.19	-22.85	-27.04
1 cykl	51.78	55.15	42.42	68.60	26.69	106.92	137.72	244.64
10 cykl	51.78	58.24	38.17	60.20	49.16	110.02	147.52	257.54
30 cykl	52.36	58.72	36.00	64.77	48.78	111.08	149.55	260.63

Prezentowane wartości stanowią wyniki działania poszczególnych neuronów zlokalizowanych na końcu jądra dekodującego sieci neuronowej i korespondują z naciskami statycznymi generowanymi przez poszczególne osie pojazdów. Wartości przed treningiem przyjmują wartości losowe, ponieważ wszystkie połączenia w sieci zostały zainicjowane z losowymi wartościami. System, oparty o sieć neuronową, nauczył się odtwarzać sygnał konstrukcji i poprawnie szacować naciski na poszczególne osie. Przy 10 cyklu treningowym osiągnięto optymalną dokładność działania. Dalszy proces trenowania nie przynosi wzrostu dokładności działania. Zauważalne jest dokładne określenie całkowitego nacisku pojazdu oraz nacisków grup osi. Większymi różnicami cechują się natomiast określone naciski na poszczególne osie.

5.2.3 Analiza błędu oszacowania całkowitego nacisku pojazdu

W procesie ewaluacji sieci neuronowej, dokonywanej po każdej zmianie zestawu danych treningowych, weryfikowano na danych testowych dokładność działania systemu w określaniu całkowitego nacisku pojazdu. Sumowano naciski wyznaczone przez system dla pięciu osi i porównywano z całkowitym naciskiem pojazdu z symulatora.

Dodatkowo, dla wyników uzyskanych dla populacji testowej analizowano parametry statystyczne takie jak:

- **Skośność** (ang. *skewness*): Skośność jest miarą asymetrii rozkładu danych względem ich średniej. Rozkład symetryczny ma skośność równą 0, wartości dodatnie wskazują na ogon rozkładu ciągnący się po prawej stronie średniej, a wartości ujemne – po lewej.

$$Skośność = \frac{E[(X-\mu)^3]}{\sigma^3}$$

- **Kurtoza** (ang. *kurtosis*): Kurtoza opisuje "ostrość" szczytu rozkładu i grubość jego ogonów. Rozkład normalny ma kurtozę równą 0. Wartości dodatnie wskazują na bardziej spiczasty szczyt niż w rozkładzie normalnym, wartości ujemne na płaski.

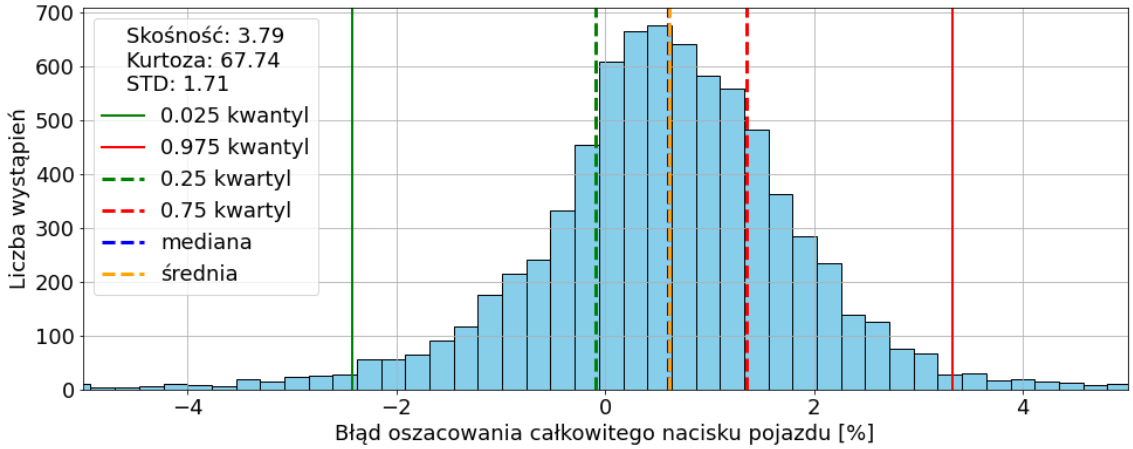
$$Kurtoza = \frac{E[(X-\mu)^4]}{\sigma^4} - 3$$

- **Odchylenie standardowe** (ang. *standard deviation*): Odchylenie standardowe mierzy rozproszenie danych wokół średniej. Im większa wartość, tym większe rozproszenie danych.

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2}$$

- **Kwantyle i kwartyle** to pojęcia statystyczne używane do opisywania rozkładu i dystrybucji danych. Oba terminy odnoszą się do punktów podziału w zestawie danych, które dzielą ten zestaw na równoliczne grupy. Kwantyl określony przez q w zbiorze danych N jest wartością, poniżej której znajduje się q procent obserwacji w zbiorze danych. Kwartyle są specyficznym przypadkiem kwantyli, które dzielą uporządkowany zestaw danych na cztery równoliczne części.

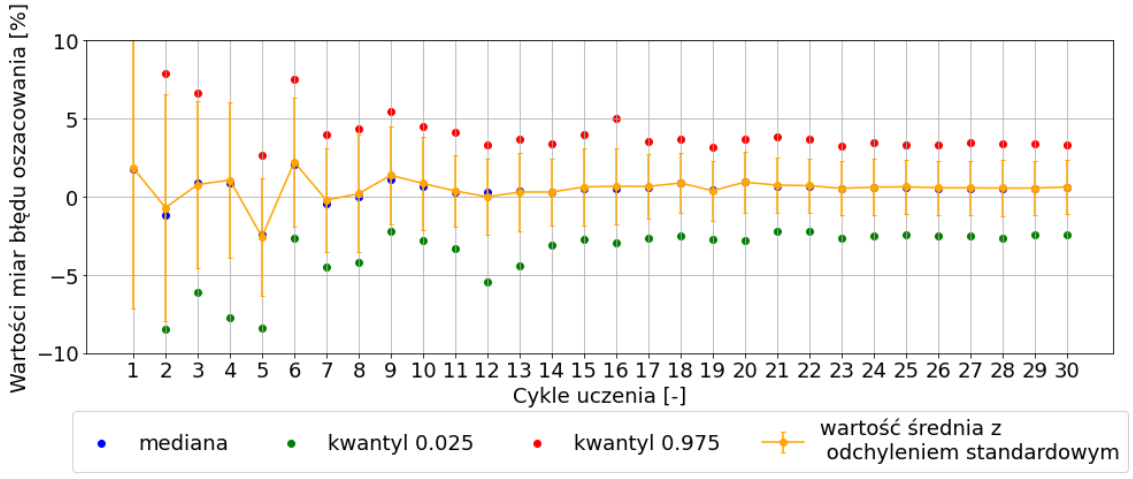
Na rys. 5.6 przedstawiono rozkład wartości błędu oszacowania całkowitego nacisku pojazdów dla całej populacji testowej ostatniego cyklu treningu.



Rys. 5.6 Rozkład wartości błędu oszacowania całkowitego nacisku pojazdów dla ostatniego cyklu treningu systemu AutoSIO_S

Z histogramu rozkładu błędów oszacowania całkowitego nacisku pojazdów wynika, że 95% błędów mieści się w przedziale od -2.41% do 3.32%. Mediana i średnia osiągnęły podobne wartości odpowiednio 0.60% oraz 0.62% co wskazuje na symetryczny rozkład błędów wokół średniej wartości. Odchylenie standardowe na poziomie 1.71% od mediany również podkreśla koncentrację 50% wartości wokół średniej.

Dodatkowo, analiza ewaluacyjna przeprowadzona dla każdego cyklu umożliwia obserwację procesu uczenia się systemu i zmian w rozkładzie błędów. Na rys. 5.7 przedstawiono zmiany wartości miar błędu w trakcie uczenia.



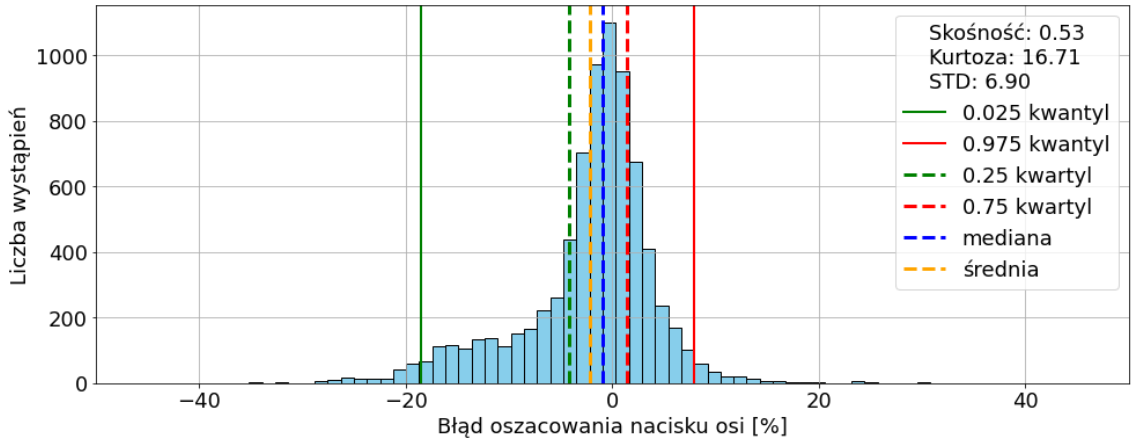
Rys. 5.7 Zmiany wartości miar błędu w trakcie uczenia systemu AutoSIO_S

Początkowe cykle uczenia charakteryzowały się znacznie większymi błędami, z większym odchyleniem standardowym. W miarę postępu treningu, wartości odchylenia standardowego stawały się coraz

mniejsze i bardziej skoncentrowane wokół wartości średniej, co świadczy o stopniowej poprawie systemu i jej coraz lepszej adaptacji do zadania określenia nacisków

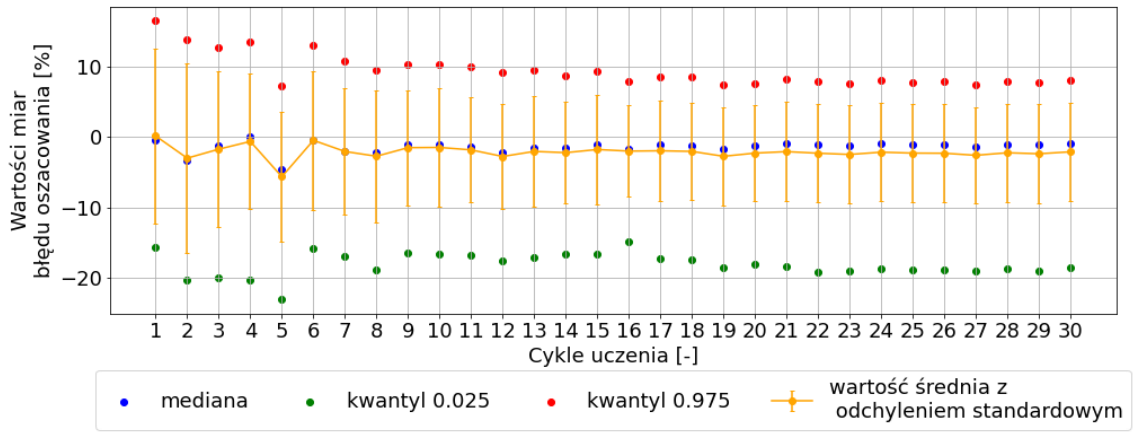
5.2.4 Analiza błędu oszacowania nacisków na poszczególne osie

Przeanalizowano statystyczny rozkład oszacowania nacisków na poszczególne osie pojazdów. Na rys. 5.8 przedstawiono rozkład oszacowań nacisku na pierwszą oś dla ostatniego cyklu uczenia, ukazując kluczowe statystyki takie jak kwantyle (0,025 i 0,975), pierwszy i trzeci kwartyl, medianę, średnią oraz dodatkowe miary takie jak skośność, kurtoza i odchylenie standardowe (STD).



Rys. 5.8 Rozkład błędów oszacowania nacisku 1. osi dla systemu AutoSIO_S

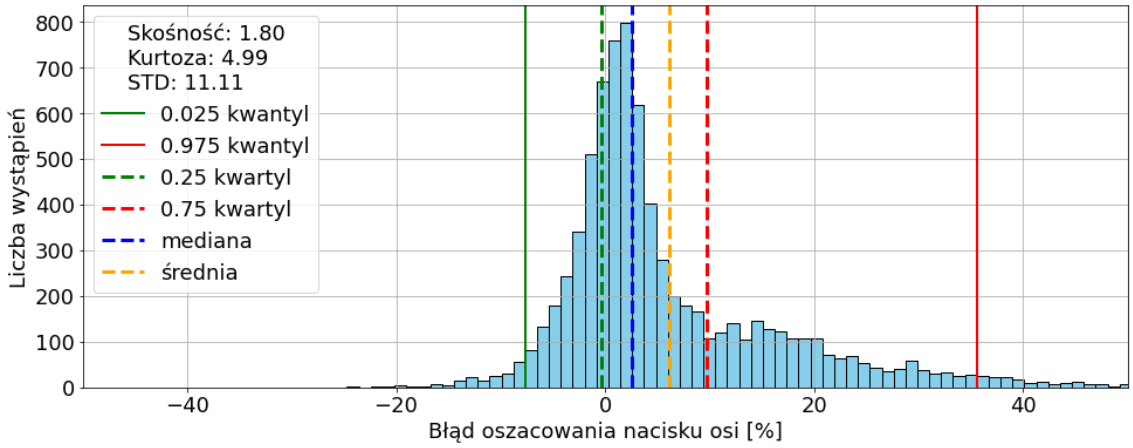
Analiza ta wskazuje na zwiększoną zmienność błędów, z wartością średnią równą -2.13%, co sugeruje, że system w uśrednieniu dokonuje precyzyjnych oszacowań, jednak rozpiętość błędów dla 95% próbek, mieszcząca się w przedziale od -18.55% do 7.96%, jest znacznie większa niż w przypadku całkowitego nacisku pojazdu. Rys. 5.9 ukazuje ewolucję kluczowych statystyk błędu w trakcie procesu nauczania.



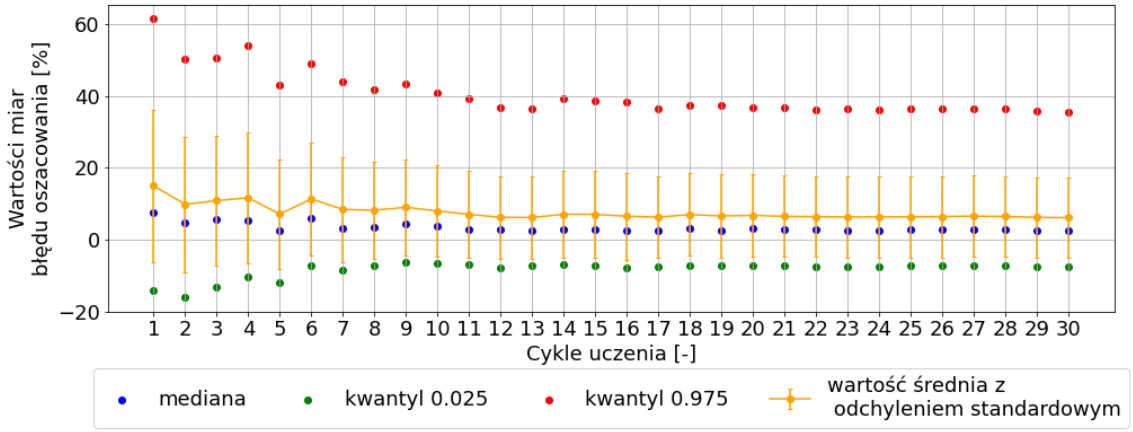
Rys. 5.9 Zmiany wartości miar błędu oszacowania nacisku 1. osi w trakcie uczenia systemu AutoSIO_S

Od 15 cyklu uczenia, wartości te ustabilizowały się na pewnym stałym poziomie. System osiągnął pewien stały poziom dokładności.

Na rys. 5.10 i rys. 5.11 przedstawiono rozkład błędów oszacowania dla ostatniego cyklu oraz zmiany wartości miar błędu w trakcie uczenia dla 2. osi.



Rys. 5.10 Rozkład błędów oszacowania nacisku 2. osi w trakcie uczenia systemu AutoSIO_S

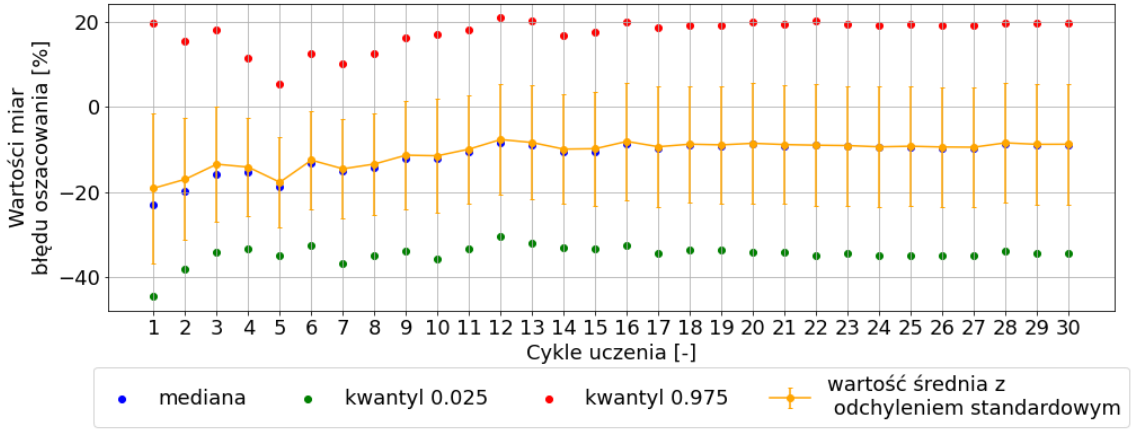


Rys. 5.11 Zmiany wartości miar błędu oszacowania 2. osi w trakcie uczenia systemu AutoSIO_S

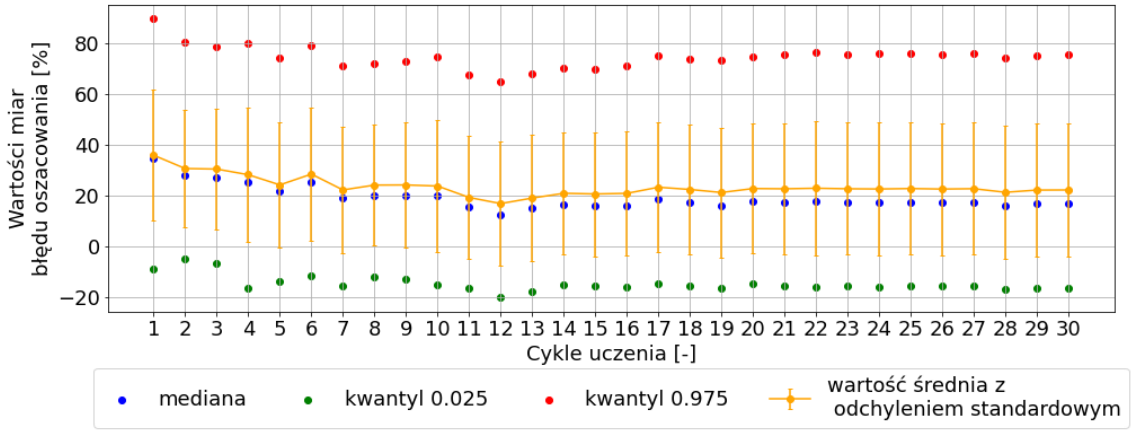
Analiza rozkładu błędów oszacowania nacisku 2. osi ukazuje znacznie szerszy zakres wahań, z grupą 95% próbek mieszczących się w przedziale od -7.61% do 35.61%. Taka szeroka rozpiętość błędów wskazuje na dużą zmienność w oszacowaniach systemu dla tej konkretnej osi. Średnia wartość błędu wynosi około 6.02%, co świadczy o tendencji systemu do przeszacowania nacisku na drugą oś. Mediana, osiągająca poziom 2.53%, może sugerować mniejszą skłonność do takiego przeszacowania, lecz nadal podkreśla istotną różnicę pomiędzy oszacowaniami, a wartościami rzeczywistymi.

Proces nauczania sieci neuronowej systemu, obserwowany na wykresie zmian wartości i miar błędów w trakcie uczenia, charakteryzuje się szybkim wzrostem dokładności w początkowych cyklach (od 0 do 6), co świadczy o efektywnej adaptacji systemu do danych treningowych w tej początkowej fazie. Od 6 do 30 cyklu, błąd stabilizuje się, wskazując na osiągnięcie przez system pewnego optymalnego poziomu, gdzie dalsze uczenie nie przynosi znaczącej poprawy dokładności oszacowań.

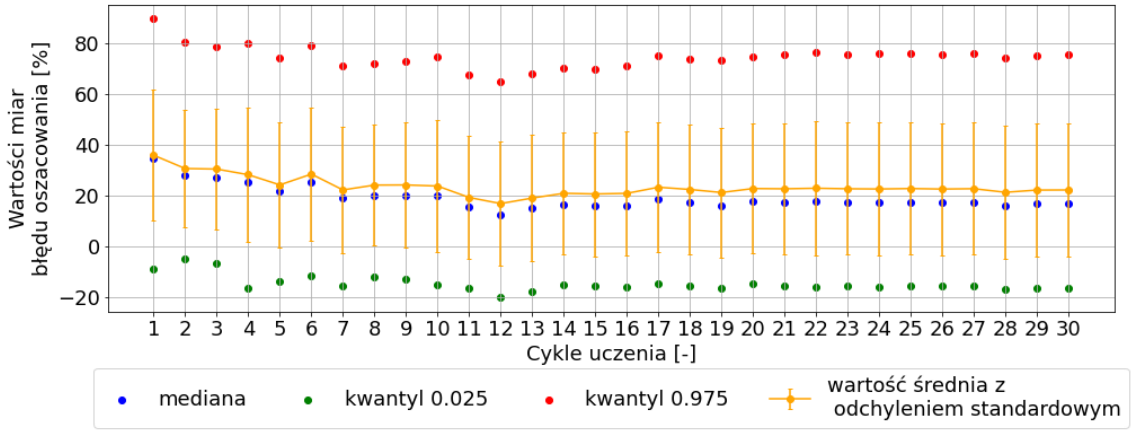
Analiza zmian miar błędów dla 3., 4. i 5. osi w trakcie całego procesu uczenia została przedstawiona na rys. 5.12, rys. 5.13, oraz rys. 5.14. Zrezygnowano z prezentacji rozkładu błędów dla ostatniego cyklu dla wymienionych osi ze względu na ich duże podobieństwo do rozkładów zaobserwowanych dla pierwszej (rys. 5.9) i drugiej osi (rys. 5.10).



Rys. 5.12 Zmiany wartości miar błędu oszacowania dla 3. osi w trakcie uczenia systemu AutoSIO_S



Rys. 5.13 Zmiany wartości miar błędu oszacowania dla 4. osi w trakcie uczenia systemu AutoSIO_S

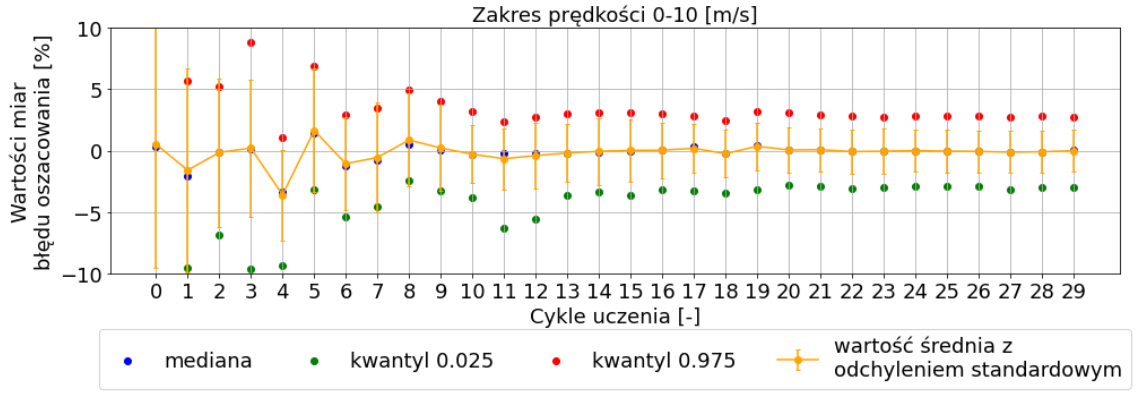


Rys. 5.14 Zmiany wartości miar błędu oszacowania dla 5. osi w trakcie uczenia systemu AutoSIO_S

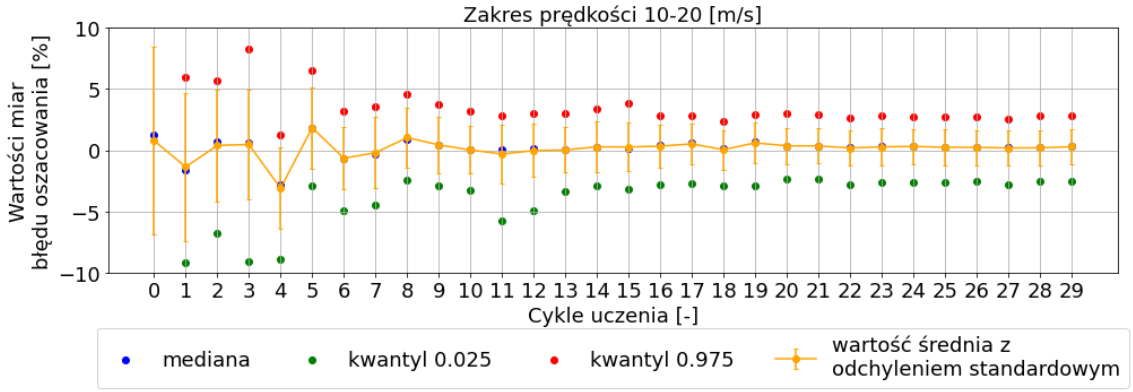
Analiza zmian miar błędów oszacowania nacisków na trzecią, czwartą i piątą oś w cyklu uczenia wykazuje podobieństwo do wyników uzyskanych dla osi pierwszej i drugiej. Dla tych osi, początkowo zauważalny jest znaczny błąd, który w ciągu pierwszych dziesięciu cykli szybko maleje, a następnie stabilizuje się na pewnym, relatywnie stałym poziomie. Charakterystyczną cechą jest przewaga ujemnych wartości średnich błędów dla trzeciej, czwartej i piątej osi, co wskazuje na tendencję do niedoszacowania nacisków w przeciwieństwie do oszacowań dla osi drugiej, gdzie obserwowano przeszacowanie. Ponadto, rozpiętość przedziału, w którym zawiera się 95% próbek jest znaczna, co świadczy o szerokiej zmienności w oszacowaniach nacisków na te osie. Tak duże rozbieżności wskazują na wyraźne trudności systemu w precyzyjnym oszacowaniu nacisków na poszczególne osie, co może być uznane za aspekt wymagający dalszych usprawnień systemu.

5.2.5 Analiza wpływu prędkości na błąd odtworzenia

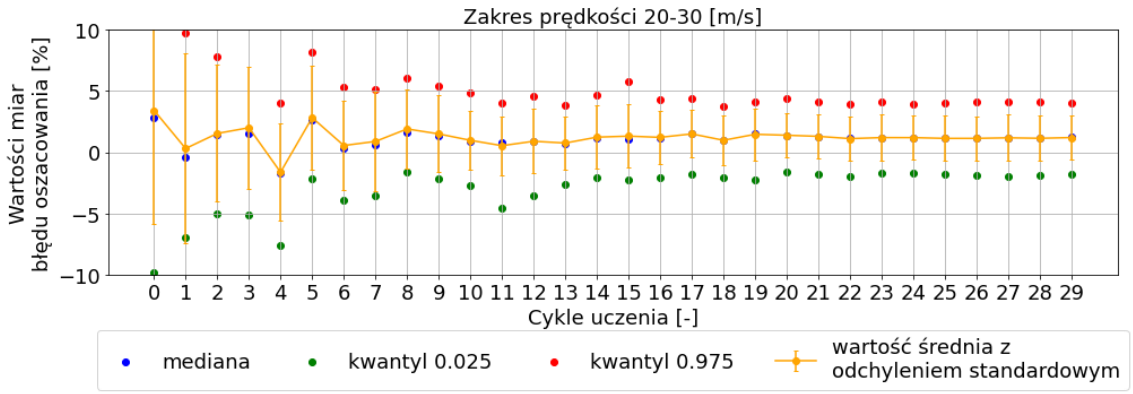
Analiza zmian rozkładu błędu oszacowania całkowitego nacisku pojazdu w zależności od prędkości jego ruchu została przedstawiona na wykresach, z podziałem na trzy zakresy prędkości: 0–10 m/s, 10–20 m/s i 20–30 m/s, odpowiednio przedstawionych na rys. 5.15, rys. 5.16 i rys. 5.17. Dane treningowe zostały przefiltrowane w celu izolacji pojazdów poruszających się w tych konkretnych przedziałach prędkości, co umożliwiło dokładniejszą analizę wpływu prędkości na dokładność oszacowań systemu.



Rys. 5.15 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o prędkości od 0 do 10 m/s



Rys. 5.16 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o prędkości od 10 do 20 m/s



Rys. 5.17 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o prędkości od 20 do 30 m/s

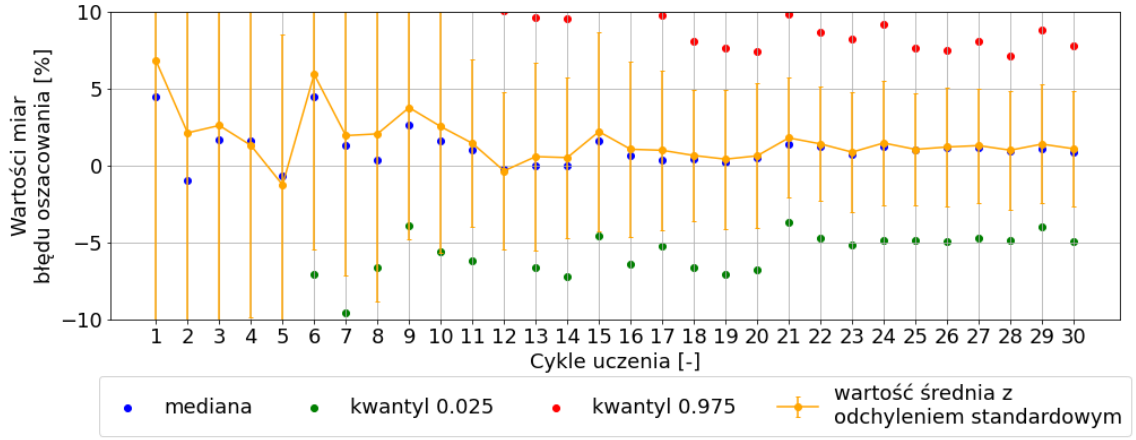
Wyniki wykazały, że średni błąd oszacowania całkowitego nacisku pojazdu dla ostatniego cyklu uczenia jest zbliżony w każdym z badanych zakresów prędkości, co wskazuje na brak istotnych różnic w dokładności systemu w zależności od prędkości pojazdu. Takie wyniki sugerują, że zastosowany system, wykorzystujący sieci neuronowe, wykazuje odporność na zmiany prędkości, co potwierdza jego skuteczność w określaniu nacisków na poszczególne osie niezależnie od prędkości poruszania się pojazdu.

5.2.6 Analiza wpływu masy pojazdu na błąd odtworzenia

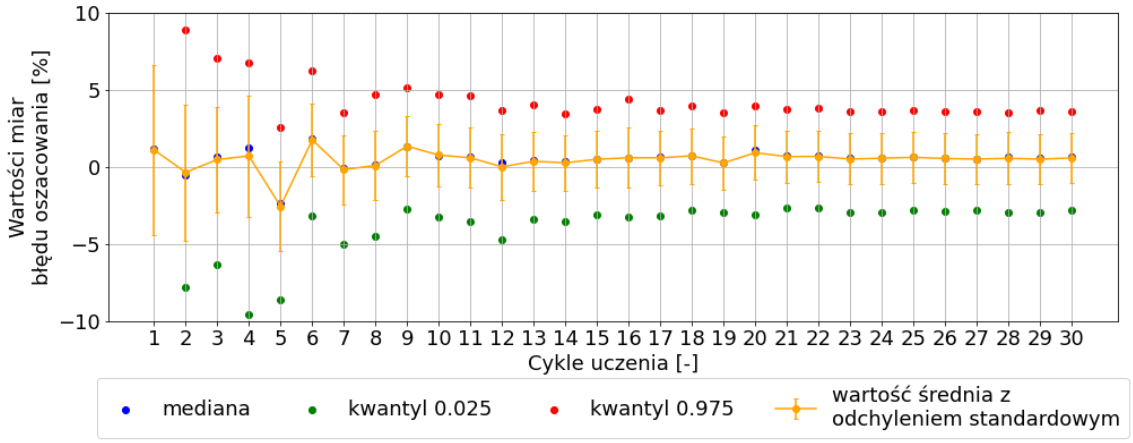
Analiza wpływu masy pojazdu na błąd oszacowania całkowitego nacisku pojazdu została przeprowadzona poprzez podział rozwiązań na zbiory w zależności od symulowanej masy pojazdu. Zgrupowane rozwiązania dla 4 zakresów masy pojazdu:

- 0 – 10 000 kg,
- 10 000 kg – 20 000 kg,
- 20 000 kg – 30 000 kg,
- powyżej 30 000 kg.

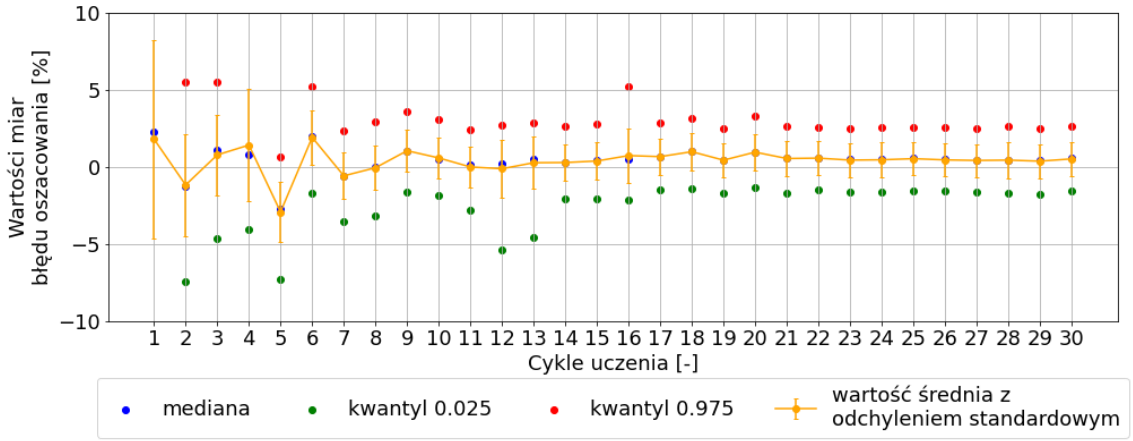
Wykresy przedstawione na rys. 5.18, rys. 5.19, rys. 5.20, rys. 5.21, ilustrują zmianę miar błędu, w tym mediany, wartości średniej, odchylenia standardowego oraz przedziału niepewności obejmującego 95% wartości populacji, w zależności od masy pojazdu i cyklu uczenia systemu.



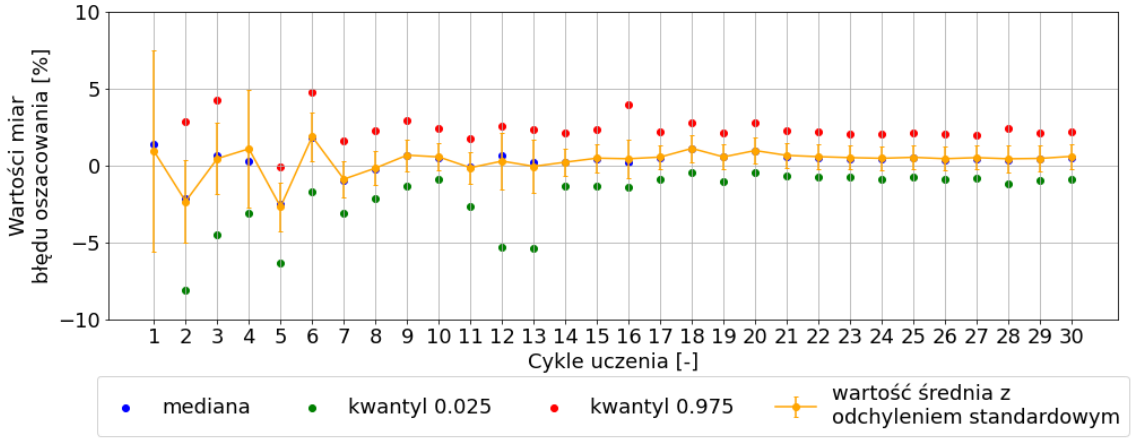
Rys. 5.18 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o masie całkowitej od 0 do 10 000 kg



Rys. 5.19 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o masie całkowitej od 10 000 do 20 000 kg



Rys. 5.20 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o masie całkowitej od 20 000 do 30 000 kg



Rys. 5.21 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_S przy uwzględnieniu pojazdów o masie całkowitej powyżej 30 000 kg

Obserwacje wykazały, że dla lekkich pojazdów (0-10 000 kg) średni błąd szacowania całkowitego nacisku osiąga poziom około 1.09%, a przedział błędu dla 95% populacji zawiera się od -4.94% do 7.77%, co wskazuje na znaczną niepewność oszacowań dla tej kategorii. W przypadku cięższych pojazdów, szczególnie tych o masie powyżej 30 ton, charakterystycznych dla ciężarówek poruszających się po drogach, błąd oszacowania staje się znacznie mniejszy, a 95% wartości błędów mieści się w węższym przedziale od -0.92% do 2.22%, co świadczy o większej dokładności systemu w oszacowaniu całkowitego nacisku cięższych pojazdów.

Takie wyniki sugerują, że system lepiej radzi sobie z oszacowaniem nacisków pojazdów ciężkich, co jest istotne z punktu widzenia ochrony infrastruktury drogowej przed uszkodzeniami spowodowanymi przez nadmierne obciążenia.

5.2.7 Analiza korelacji pomiędzy parametrami wejściowymi

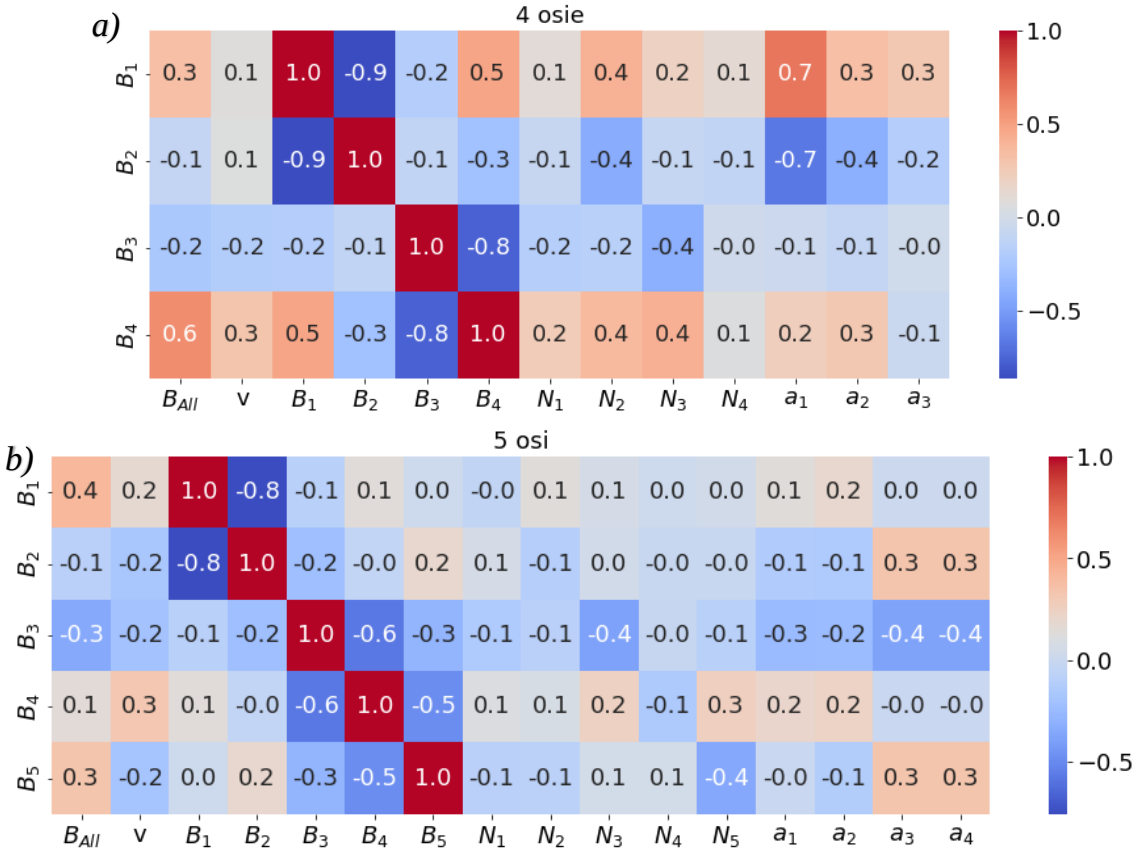
W podpunkcie podjęto próbę analizy korelacji pomiędzy różnymi parametrami pojazdów, a błędami w oszacowaniu nacisków na poszczególne osie oraz całkowitym naciskiem pojazdu. Rozpatrywano związki między błędami, a takimi zmiennymi, jak prędkość pojazdu, całkowity nacisk oraz rozstawy osi. Zastosowano tablice korelacji, gdzie dla każdej osi pojazdu analizowano zależności błędu oszacowania od prędkości, nacisków poszczególnych osi oraz rozstawów między nimi.
Analiza ta miała na celu zidentyfikowanie potencjalnych wzorców lub trendów, które mogą wpływać na dokładność oszacowań systemu, opartego o sieci neuronowe. Poprzez badanie korelacji między błędami, a poszczególnymi parametrami pojazdu, możliwa była lepsza optymalizacja sieci neuronowej pod kątem precyzyjniejszego oszacowania nacisków.

Korelacje Pearsona [123], Spearmana [124] i Kendalla [125] to trzy popularne metody statystyczne używane do oceny siły i kierunku związku między dwiema zmiennymi. Zastosowano korelację Spearmana, która jest nieparametryczną miarą zależności statystycznej między dwoma zmiennymi. Opiera się na rangach danych zamiast na ich rzeczywistych wartościach, co czyni ją odpowiednią do

analizy związków między zmiennymi, które nie spełniają założeń korelacji Pearsona, takich jak brak normalności rozkładów czy zależności nieliniowe. Wartość współczynnika korelacji Spearmana, oznaczana jako ρ, mieści się w zakresie od -1 do 1, gdzie 1 oznacza doskonałą korelację dodatnią, -1 doskonałą korelację ujemną, a 0 brak korelacji.

Rozpatrywano zależność błędów i-tej osi (B_i) od:

- B_{All} – błąd w określeniu całkowitego nacisku osi,
- v – prędkość,
- B_i – błąd oszacowania nacisku i-tej osi,
- N_i – nacisk i-tej osi,
- a_i – rozstaw i między osią i-tą, a osią $i+1$.



Rys. 5.22 Wybrane korelacje Spearmana, odpowiednio dla pojazdów: a) 4-osiowych; b) 5-osiowych

Analiza nie wykazała istotnej korelacji liniowej między błędami oszacowania, a prędkością pojazdu, co potwierdza wcześniejsze wnioski, że prędkość nie wpływa znacząco na dokładność oszacowań systemu. Interesujące spostrzeżenie dotyczy korelacji między błędami oszacowania na poszczególnych osiach i globalnym naciskiem pojazdu. Mimo znaczących błędów dla indywidualnych osi, ogólna dokładność oszacowania całkowitego nacisku pojazdu utrzymuje się na wysokim poziomie, co sugeruje występowanie kompensacji błędów między poszczególnymi osiami.

Analiza korelacji pomiędzy błędami, a rozstawami osi wykazała, że zmiany w rozstawach osi mogą wpływać na dokładność oszacowania nacisków na poszczególne osie, pokazując złożoność wzajemnych zależności pomiędzy parametrami pojazdu, a reakcjami konstrukcji mostowej. Zauważono, że zwiększenie odległości między osiami może poprawić oszacowanie nacisku na niektóre osie, podczas gdy na inne wpływa negatywnie.

Podsumowując, analiza korelacji w systemie o podejściu quasi-statycznym ujawnia złożone wzorce zależności między parametrami pojazdu, a błędami oszacowania nacisków. Chociaż sieci neuronowe są w stanie zidentyfikować i wykorzystać te wzorce do poprawy dokładności oszacowań, zrozumienie tych zależności w pełni wymaga dalszych badań.

5.2.8 Podsumowanie

Podsumowując wyniki analiz opracowanego systemu AutoSIO_S o podejściu quasi-statycznym, stwierdza się, że system oparty na sieciach neuronowych skutecznie radzi sobie z oceną całkowitego nacisku pojazdu. Świadczy o tym fakt, że 95% błędów oszacowania mieści się w przedziale od -2.41% do 3.32%.

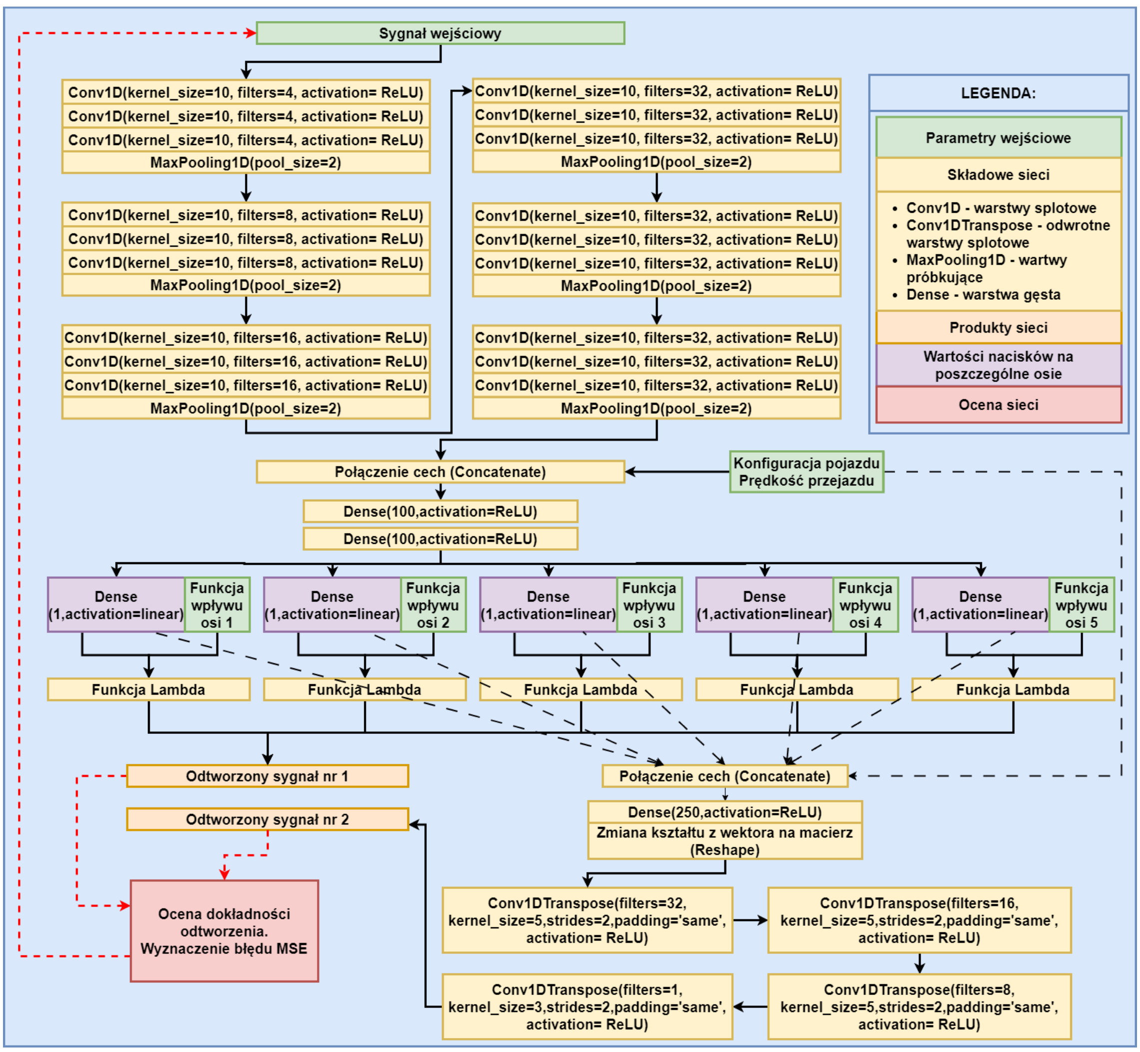
Wykazano odporność systemu na zmiany prędkości pojazdów, co jest istotną cechą dla aplikacji monitorujących ruch drogowy. Wariant systemu wykazuje tendencję do znacznego przeszacowywania lub niedoszacowania nacisków na indywidualne osie, co skutkuje dużą niepewnością tych oszacowań. Dotyczy to szczególnie osi, znajdujących się w niewielkich odległościach od siebie. System poprawnie określa jednak naciski grup osi np. zestawu potrójnych kół w naczepie siodłowej. Szczególnie obiecujące wydaje się zastosowanie systemu w ocenie nacisków pojazdów ciężkich, gdzie błędy oszacowań okazują się być minimalne. To wskazuje na potencjał systemu w identyfikacji i monitorowaniu pojazdów stanowiących największe obciążenie dla dróg i mostów.

Wnioski z analiz potwierdzają zasadność wykorzystania autodekoderów w ocenie nacisków pojazdów na obiekty infrastruktury drogowej. System o podejściu quasi-statycznym, mimo pewnych ograniczeń, prezentuje się jako obiecujące narzędzie, które po dalszej optymalizacji może znaleźć zastosowanie.

5.3 System AutoSIO_SD odtwarzający dynamiczną i quasi-statyczną odpowiedź obiektu mostowego

5.3.1 Struktura systemu

Podstawowym zadaniem omawianego w podrozdziale 5.2 systemu AutoSIO_S jest określenie całkowitego nacisku pojazdu i odtworzenie odpowiedzi konstrukcji jak dla quasi-statycznego przejazdu. Dokładny opis implementacji został przedstawiony w załączniku C.3. W kolejnym wariancie systemu zdecydowano się na odtworzenie sygnału zarówno jak dla przejazdu quasi-statycznego jak również odtworzenie sygnału zawierającego efekty pochodzenia dynamicznego. Oczekuje się, że system AutoSIO_SD z wykorzystaniem funkcji wpływu odtworzy quasi-statyczny sygnał odpowiedzi konstrukcji obiektu mostowego. Dodatkowo system, oparty o sieci neuronowe, znając tylko określone przez siebie naciski pojazdów oraz wybrane parametry przejazdu, niezależnie od funkcji wpływu, odtworzy sygnał odpowiedzi z uwzględnieniem dynamicznej natury zjawiska. Strukturę sieci przedstawiono na rys. 5.23. Sieć wymaga analogicznych parametrów wejściowych jak ta zaprezentowana w podrozdziale 5.2.



Rys. 5.23 Schemat blokowy systemu AutoSIO_SD

Rozpatrywany system, wykorzystujący sieci neuronowe, charakteryzuje się liniową oraz rozgałęzioną budową, w której informacja wejściowa jest przesyłana przez sieć w jednym kierunku aż do warstw wyjściowych po których następuje określenie błędu odtworzenia funkcji. W systemie dodano nowe elementy składowe sieci, które nie zostały zastosowane w podpunkcie 5.2.1.

1. **Conv1DTranspose(filters, kernel_size, strides, padding, activation)**: Odwrotna warstwa splotowa (dekonwolucyjna), której zadaniem jest "rozszerzanie" danych w kierunku przeciwnym do klasycznej warstwy Conv1D. Stosowana jest często w generatywnych sieciach neuronowych lub tam, gdzie potrzebujemy przywrócić pierwotne wymiary danych po operacjach takich jak próbkowanie w dół (ang. *downsampling*). Poniżej przedstawiono opis parametrów:
    1. **filters**: Liczba filtrów, które określają liczbę cech wyodrębnionych w danym kroku.
    2. **kernel_size**: Rozmiar okna splotowego, czyli liczba sąsiednich punktów analizowanych jednocześnie.
    3. **strides**: Krok przesunięcia, czyli jak daleko przesuwa się okno w danych wejściowych podczas splotu.
    4. **padding**: Parametr zapewnienia, że wyjście ma taki sam rozmiar jak wejście.
    5. **activation**: Funkcja aktywacji, która wprowadza nieliniowość do sieci neuronowej, zamieniając wszystkie wartości ujemne na zero.
2. **Reshape**: Jest to funkcja służąca do zmiany kształtu tensora (np. wektora lub macierzy) bez zmiany jego danych. Jest niezbędna, gdy wymagane jest przekształcenie danych do wymiarów odpowiednich dla kolejnych warstw sieci, zwłaszcza przy przejściach między warstwami splotowymi a w pełni połączonymi (Dense).

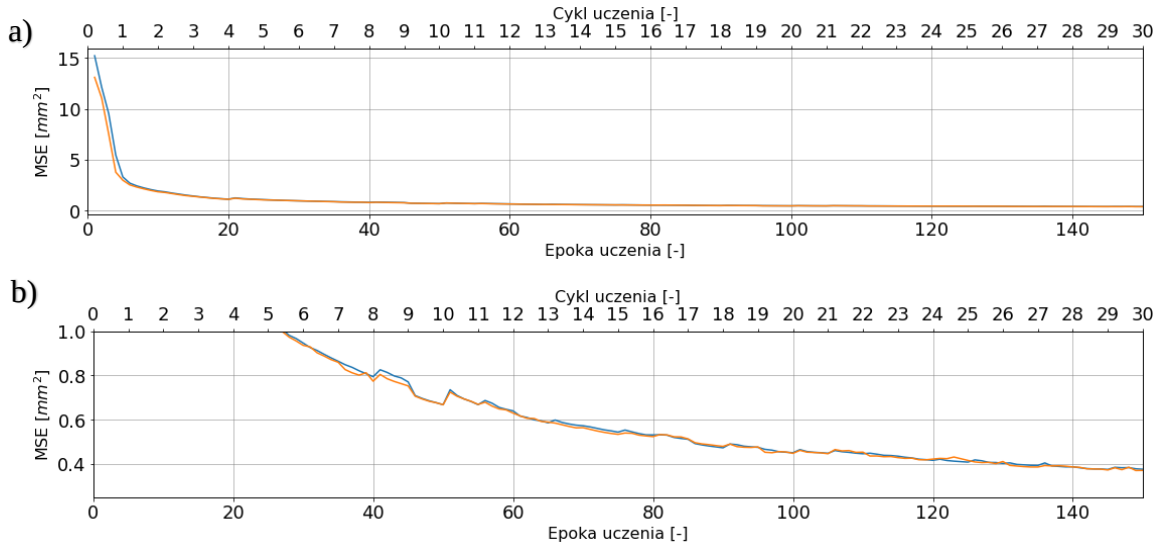
Schemat na rys. 5.23 przedstawia następujący proces działania systemu AutoSIO_SD:

- Wejściem do bloku Conv1D (warstwy CNN) jest sygnał odpowiedzi konstrukcji mostowej. Zadaniem warstw CNN jest ekstrakcja cech poprzez wykorzystanie odpowiednich filtrów. Analizowany sygnał jest wektorem, dlatego wykorzystano 1-wymiarowe warstwy splotowe. Jako funkcję aktywacji zastosowano funkcję ReLU, a następnie redukcję wymiarowości sygnału.
- Do uzyskanego wektora cech dodawane są informacje o konfiguracji pojazdu oraz prędkości przejazdu. Tak przygotowany wektor cech jest ze sobą łączony, a następnie analizowany przez warstwy gęste. W najwęższym miejscu, sieć neuronowa posiada tylko 5 neuronów, które odpowiadają za wartości sił poszczególnych osi.

- Wartości wyznaczone przez neurony są mnożone przez Funkcje Wpływu dla odpowiednich osi w funkcji. Produkty są następnie dodawane i efektem wyjściowym jest teoretycznie sygnał wejściowy.
- Warstwy DeCNN (Conv1DTranspose) przeprowadzają proces odwrotny do procesu ekstrakcji cech. Na podstawie wektora informacji, składającego się z wektora nacisków oraz konfiguracji pojazdu oraz nacisków odtwarzają sygnał wejściowy. Z uwagi na fakt, że nie ma wymuszenia kształtu w postaci funkcji statycznej, sygnał ten może jak najdokładniej odzwierciedlać sygnał wejściowy, zawierający zarówno statyczne jak i dynamiczne składowe.
- System porównuje różnice pomiędzy sygnałem wejściowym, a sygnałami wyjściowymi i dążąc do minimalizacji różnic zmienia wartości połączeń między neuronami.

5.3.2 Proces treningu sieci

Przeprowadzono analogiczny schemat treningu jak w podpunkcie 5.2.2 oraz wykorzystano te same dane uczące oraz testowe. Wykresy krzywych uczenia przedstawionych na rys. 5.24 prezentują zmianę błędu średniokwadratowego w trakcie procesu trenowania systemu.

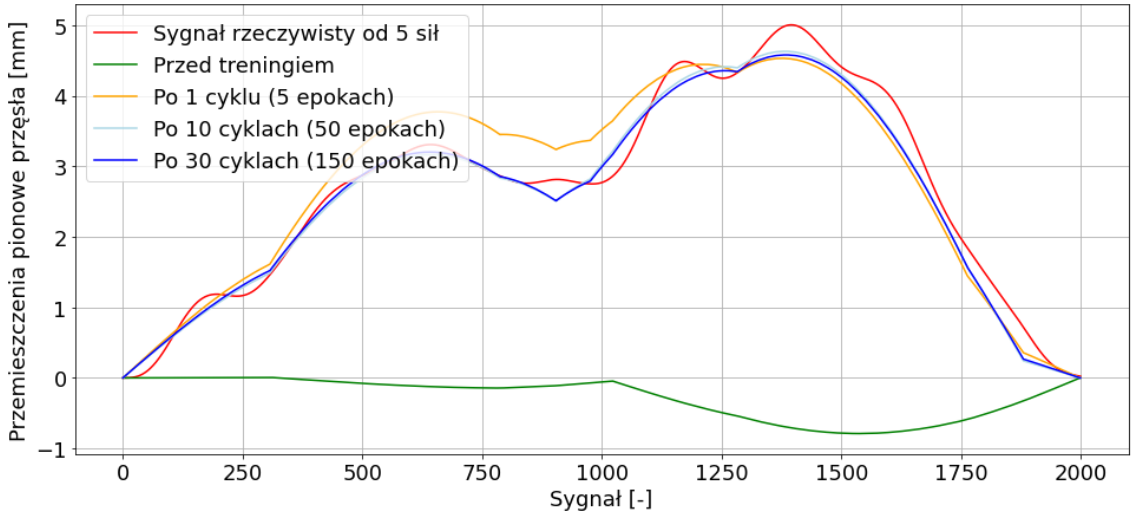


Rys. 5.24 Wartości krzywych uczenia w trakcie uczenia systemu AutoSIO_SD
a) Wykres przedstawia pełny zakres wartości MSE;
b) Wykres przedstawia zakres 0.25-1.0 wartości MSE

Dzięki zastosowanemu podejściu, sieć neuronowa osiągnęła wysoką dokładność w generalizacji wyników dla różnorodnych typów pojazdów, nie ograniczając się jedynie do wąskiego zakresu danych treningowych.

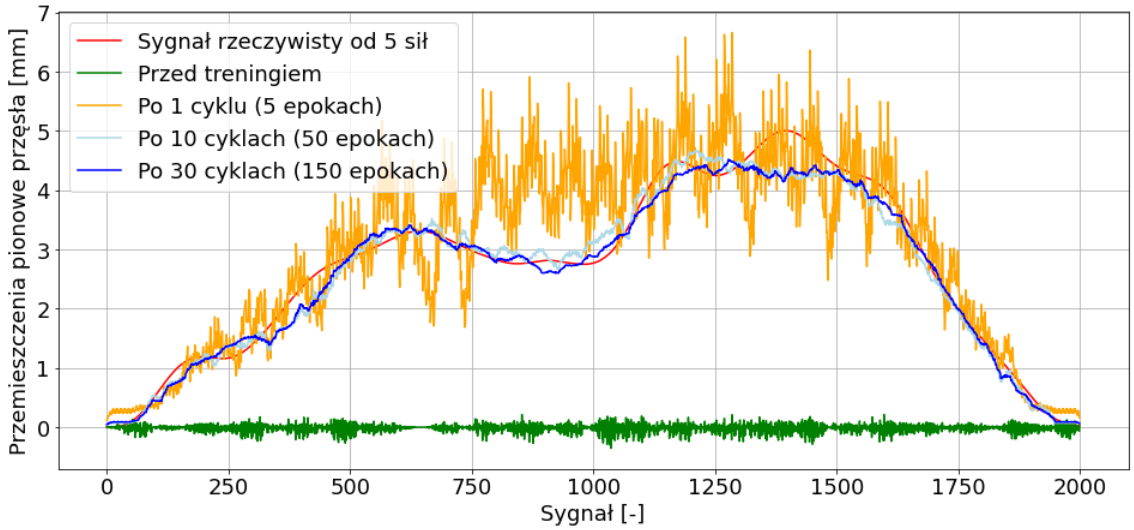
Analiza wykresów pozwala na szczegółowe zrozumienie efektywności procesu nauczania. Na rys. 5.25 przedstawiono porównanie sygnału statycznego, generowanego przez system, z oryginalnym sygnałem uzyskanym z symulatora, w różnych fazach treningu. Dodatkowo, zaprezentowano odpowiedzi systemu

przed treningiem, po pierwszym, a także po 10 (50 epoka treningu) i 30 (150 epoka treningu) cyklu treningowym.



Rys. 5.25 Porównanie odtworzonego sygnału nr 1 przez wariant systemu AutoSIO_SD z sygnałem wejściowym dla różnych faz treningu

Na podstawie analizy odpowiedzi statycznej można zauważyć, że system nauczył się efektywnie eliminować efekty dynamiczne i dokładnie rekonstruować statyczny przebieg przejazdu na podstawie dostarczonych danych wejściowych, co pozwoliło na precyzyjne oszacowanie wartości sił i nacisków generowanych przez pojazdy. Jest to szczególnie widoczne po ostatnim cyklu treningowym (150 epoka treningu), gdzie dokładność systemu była wysoka. Na rys. 5.26 przedstawiono analizę uzyskanego sygnału dynamicznego, porównując go z oryginalnym sygnałem dla różnych etapów treningu.



Rys. 5.26 Porównanie odtworzonego sygnału nr 2 przez wariant systemu AutoSIO_SD z sygnałem wejściowym dla różnych faz treningu

Odtworzona odpowiedź dynamiczna pokazuje, że system zdołał z dużą dokładnością odtworzyć zachowanie konstrukcji, bazując jedynie na ograniczonych danych wejściowych, takich jak szacunkowy nacisk osi pojazdu i odległości między osiami.

Dodatkowo została przeprowadzona analiza dla losowo wybranego pojazdu o sylwetce 2C+3N (ciągnik siodłowy + naczepa) o pięciu osiach, gdzie porównano rzeczywisty nacisk pojazdu w symulatorze z efektami nacisków określonymi przez system w różnych fazach treningu, włączając w to fazę przed rozpoczęciem treningu. Wyniki przedstawiono w tab. 5.2.

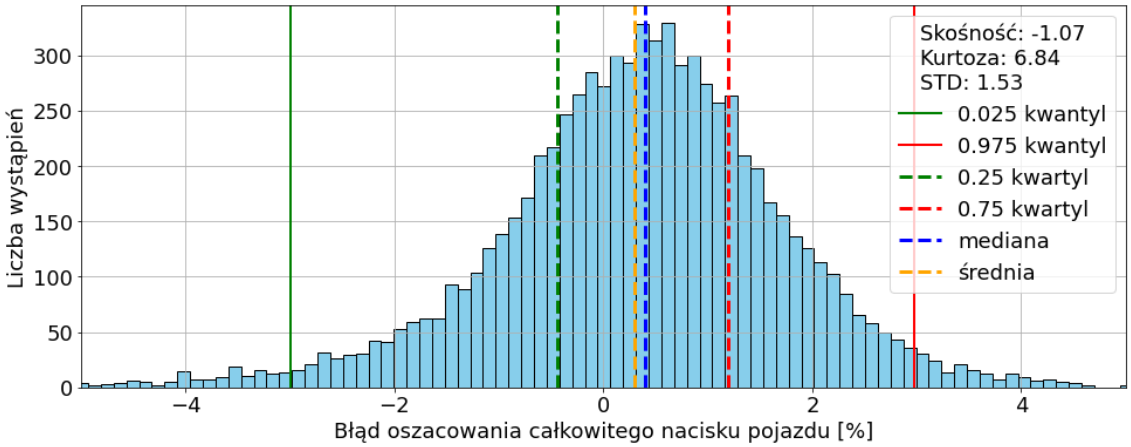
Tab. 5.2 Określone wartości nacisków statycznych na różnych etapach treningu systemu

Etap treningu	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
Rzeczywista wartość	54.82	54.26	48.57	47.96	53.87	109.07	150.40	259.47
Przed treningiem	0.30	-4.49	2.28	1.29	-26.42	-4.19	-22.85	-27.04
1 cykl	48.91	54.63	42.66	75.60	25.88	103.54	144.14	247.68
10 cykl	56.19	52.50	42.03	64.66	46.69	108.68	153.38	262.07
30 cykl	54.07	54.16	40.91	62.43	46.97	108.23	150.31	258.54

Prezentowane wartości stanowią wyniki działania poszczególnych neuronów zlokalizowanych na końcu jądra dekodującego sieci neuronowej i pokrywają się z rzeczywistymi naciskami poszczególnych osi pojazdów. Wartości przed treningiem przyjmują wartości losowe, ponieważ wszystkie połączenia w sieci zostały zainicjowane z losowymi wartościami. Zauważalne jest bardzo dobre oszacowanie całkowitego nacisku pojazdu oraz nacisków grup osi. Większymi różnicami cechują się natomiast oszacowane naciski na poszczególne osie.

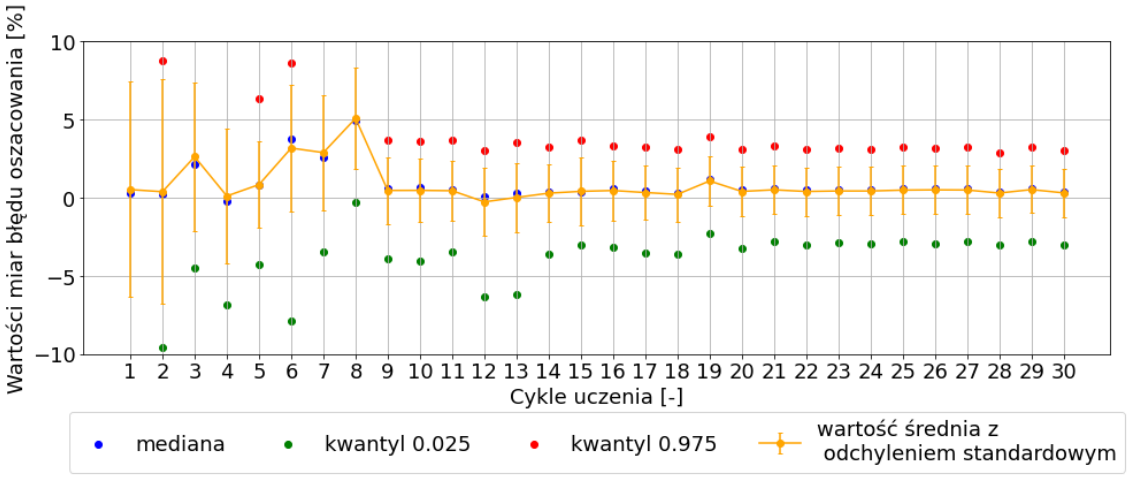
5.3.3 Analiza błędu określenia całkowitego nacisku pojazdu

Dokonano analogicznej ewaluacji błędu oszacowania całkowitego nacisku pojazdu jak w podpunkcie 5.2.3 oraz zastosowano te same miary do jej oceny.



Rys. 5.27 Rozkład wartości błędu oszacowania całkowitego nacisku pojazdów dla ostatniego cyklu treningu systemu AutoSIO_SD

Z histogramu rozkładu wartości błędu oszacowania całkowitego nacisku pojazdu wynika, że 95% błędów mieści się w przedziale od -2.99% do 2.98%. Mediana i średnia osiągnęły podobne wartości odpowiednio 0.40 % oraz 0.30% co wskazuje na symetryczny rozkład błędów wokół średniej wartości. Odchylenie standardowe na poziomie 1.53% od mediany również podkreśla koncentrację 50% wartości wokół średniej.

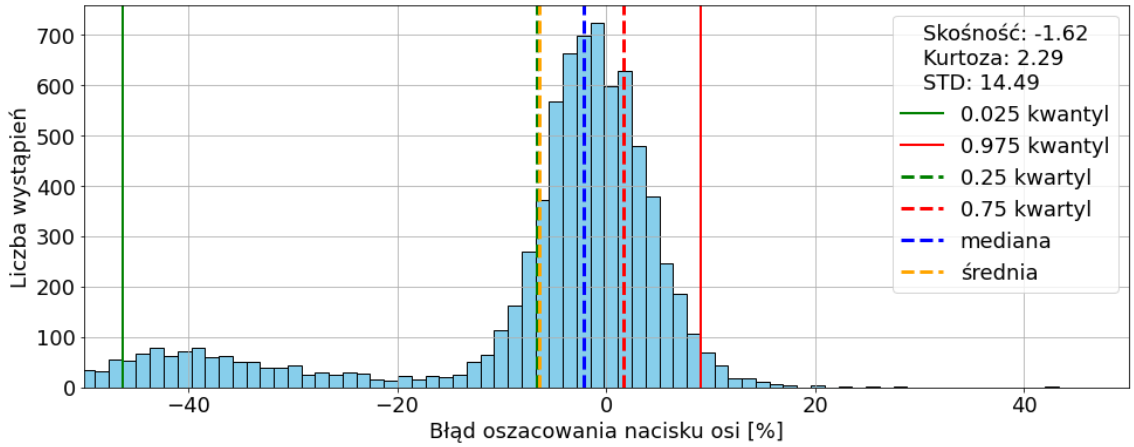


Rys. 5.28 Zmiany wartości miar błędu w trakcie uczenia systemu AutoSIO_SD

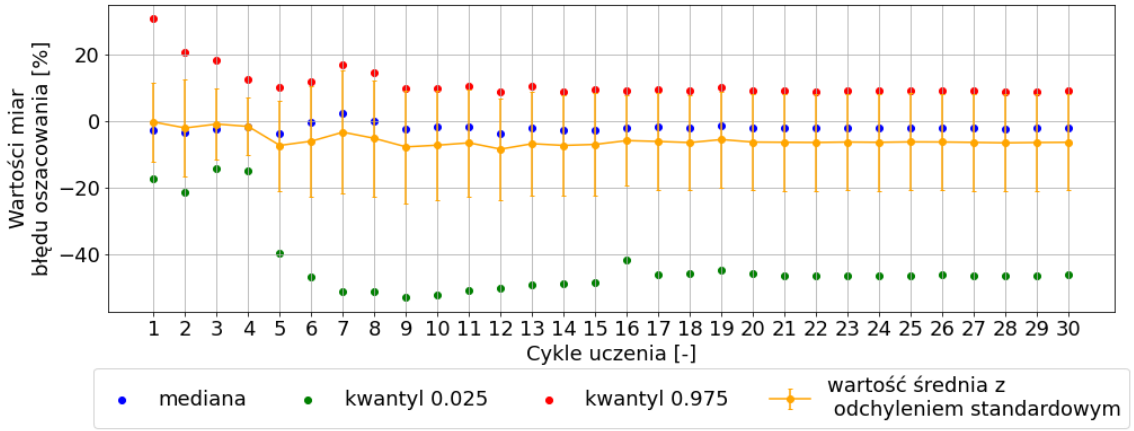
System, oparty o sieci neuronowe, osiąga stabilizację po dziesiątym cyklu nauczania, co oznacza, że jego dokładność nie ulega dalszym zmianom. W rezultacie, dla różnorodnych zestawów danych treningowych, system zapewnia stabilne rozwiązania.

5.3.4 Analiza błędu nacisków na poszczególne osie

Dokonano analogicznej ewaluacji błędu oszacowania nacisków poszczególnych osi pojazdu jak w podpunkcie 5.2.4 oraz zastosowane te same miary do oceny.



Rys. 5.29 Rozkład błędów oszacowania nacisku 1. osi dla systemu AutoSIO_SD

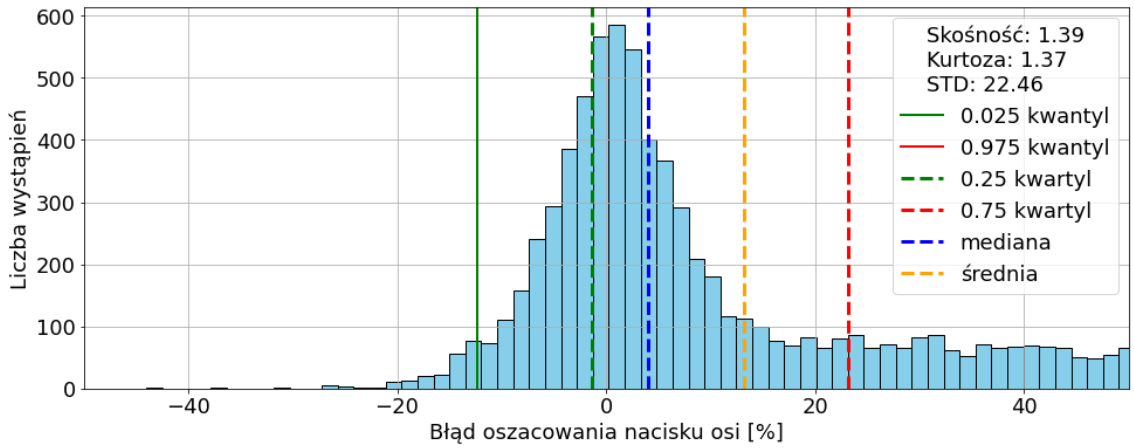


Rys. 5.30 Zmiany wartości miar błędu oszacowania nacisków 1.osi w trakcie uczenia

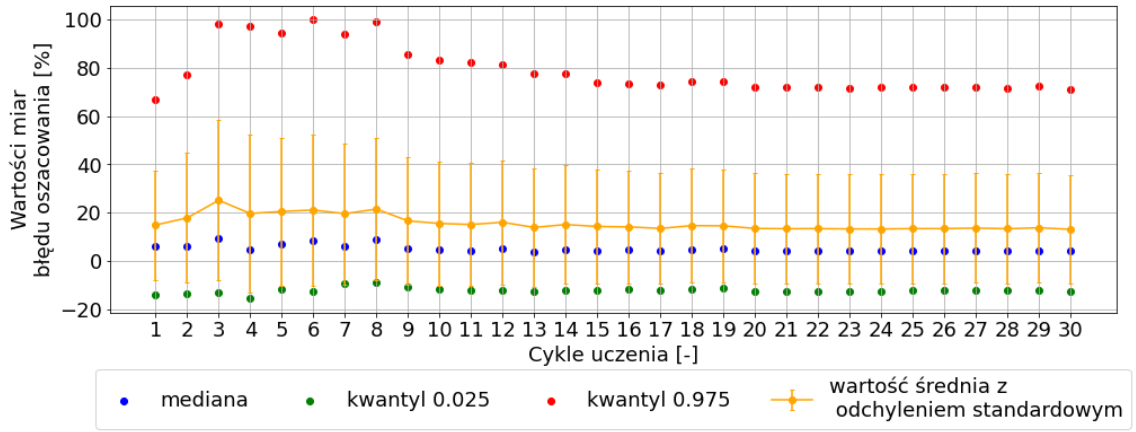
Z rys. 5.29 możemy odczytać średnią wartość błędu oszacowania nacisku pierwszej osi równą -6.38%, co sugeruje, że system w uśrednieniu dokonuje precyzyjnych oszacowań. Rozpiętość błędów dla 95% próbek, mieszcząca się w przedziale od -46.28% do 9.02%, jest większa niż w przypadku całkowitego nacisku pojazdu.

Rys. 5.30 ukazuje ewolucję kluczowych statystyk błędu oszacowania w trakcie procesu nauczania, wskazując na znaczące początkowe odchylenia, które stabilizują się po 15 cyklach nauczania. Od cyklu 15 do 30 utrzymują się na relatywnie stałym poziomie, co świadczy o stabilizacji sieci neuronowej i osiągnięciu pewnego poziomu błędu.

Na rys. 5.31 oraz rys. 5.32 przedstawiono rozkład błędów oszacowania oraz ewolucję miar błędu w trakcie uczenia dla 2. osi.



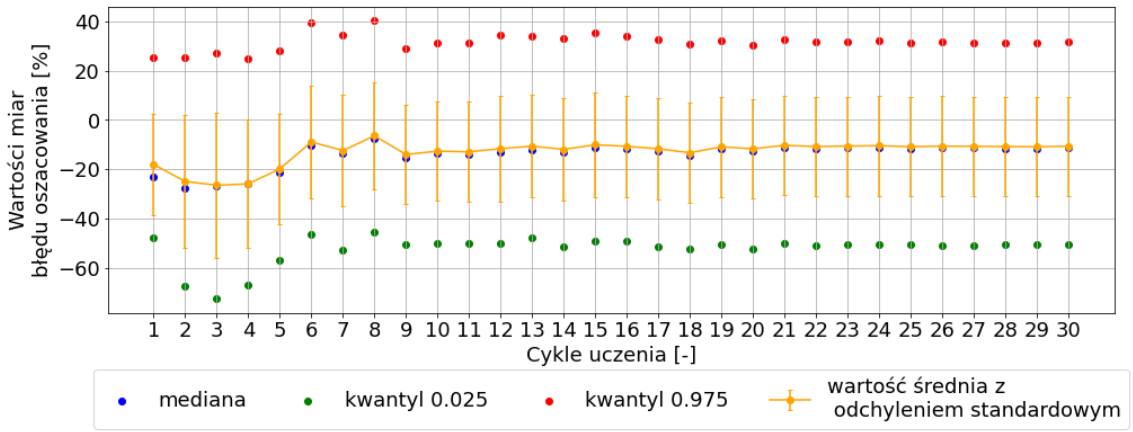
Rys. 5.31 Rozkład błędów oszacowania nacisku 2. osi



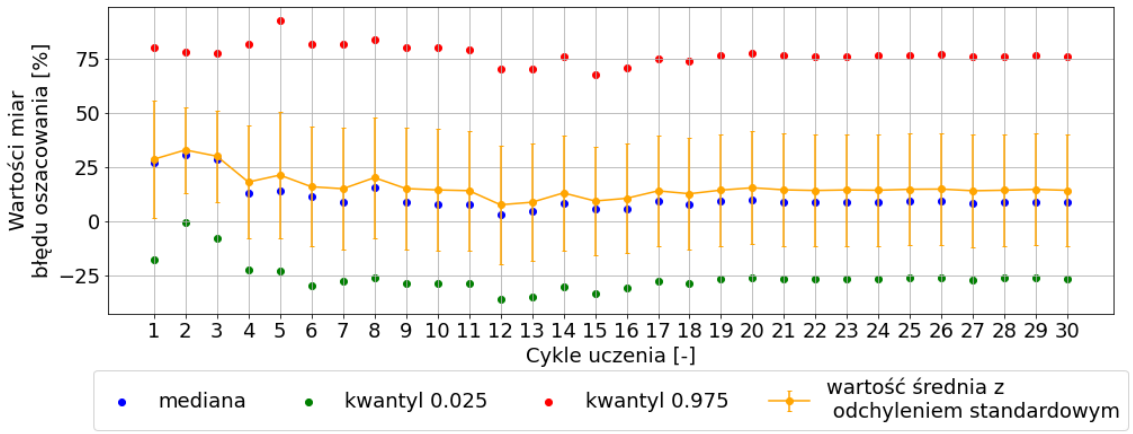
Rys. 5.32 Zmiany wartości miar błędu oszacowania nacisku 2. osi w trakcie uczenia

Analiza rozkładu błędów oszacowania nacisku na 2. oś ukazuje znacznie szerszy zakres wahań 95% próbek mieszczący się w przedziale od -12.38% do 71.00%. Średnia wartość błędu wynosi około 13.16%, co świadczy o tendencji systemu do przeszacowania nacisku drugiej osi, podczas gdy mediana, osiągająca poziom 4.07%, może sugerować mniejszą skłonność do takiego przeszacowania, lecz nadal podkreśla istotną różnicę pomiędzy oszacowaniami a wartościami rzeczywistymi.

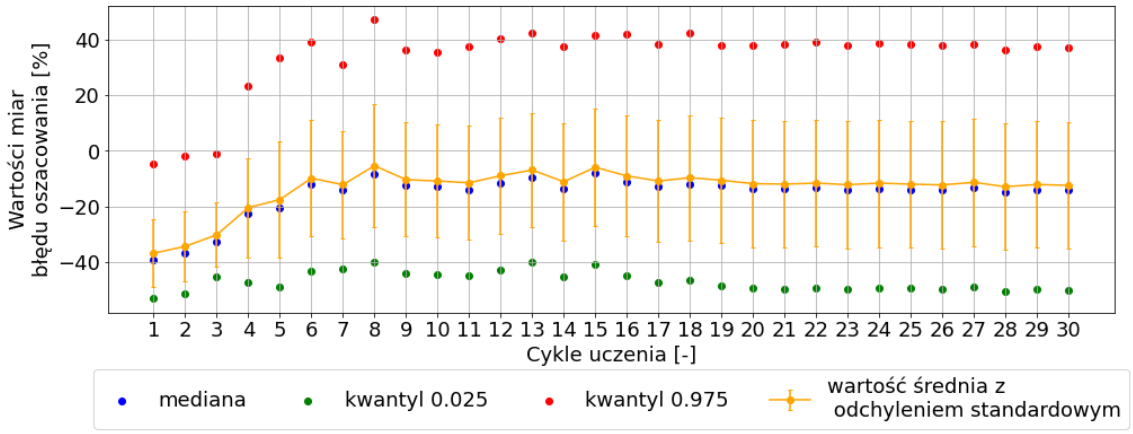
Na podstawie analizy wykresu dotyczącego zmian w miarach błędu estymacji dla drugiej osi, obserwuje się analogiczną tendencję, jak w przypadku pierwszej osi, gdzie do 15 cyklu następuje zmienność błędu oszacowania, a od 15 do 30 cyklu uczenia błąd oszacowania stabilizuje się. Wykresy zmian w miarach błędu estymacji dla trzeciej, czwartej i piątej osi w trakcie całego procesu uczenia zostały przedstawione na rys. 5.33, rys. 5.34 oraz rys. 5.35.



Rys. 5.33 Zmiany wartości miar błędu oszacowania nacisku 3. osi w trakcie uczenia



Rys. 5.34 Zmiany wartości miar błędu oszacowania nacisku 4. osi w trakcie uczenia

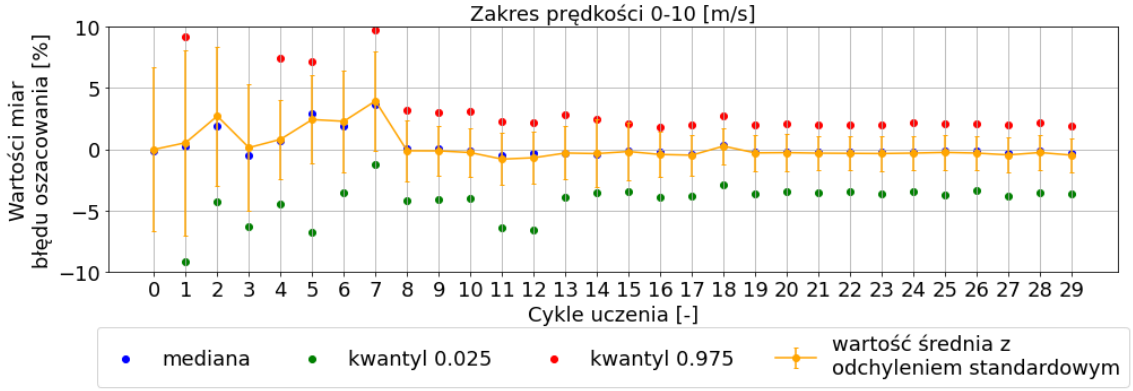


Rys. 5.35 Zmiany wartości miar błędu oszacowania nacisku 5. osi w trakcie uczenia

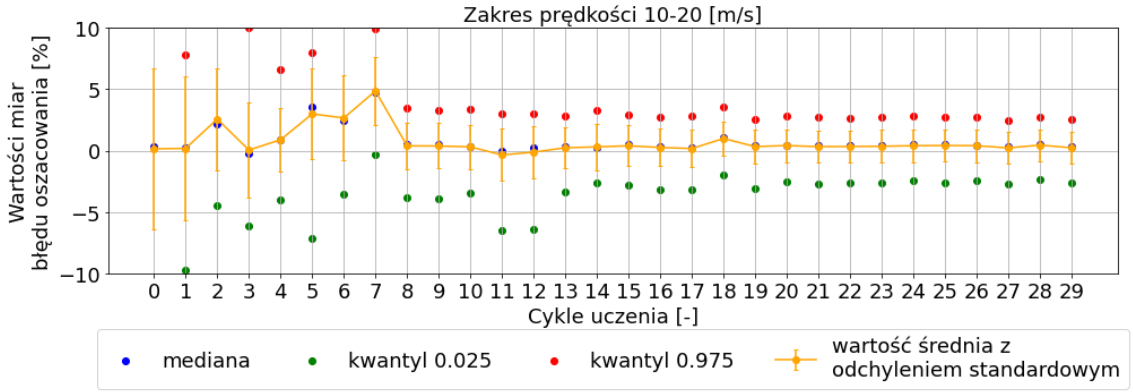
Podobnie jak dla pierwszej i drugiej osi, w początkowej fazie procesu uczenia dla tych osi obserwuje się znaczący błąd, który szybko maleje w ciągu pierwszych 15 cyklach uczenia, a następnie osiąga relatywnie stały poziom stabilizacji. Dodatkowo, zauważalna jest duża rozpiętość przedziałów ufności, w których zawiera się 95% błędów oszacowań dla poszczególnych osi, co wskazuje na stosunkowo wysoką niepewność estymacji dla tych pomiarów.

5.3.5 Analiza wpływu prędkości na błąd odtworzenia

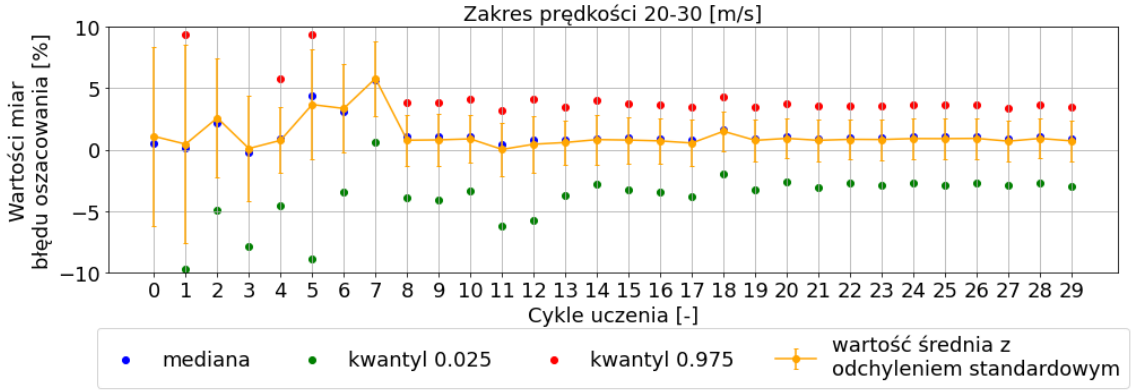
Dokonano analizy błędu oszacowania całkowitego nacisku pojazdu w zależności od prędkości, analogicznie jak w podpunkcie 5.2.5. Na rys. 5.36, rys. 5.37, rys. 5.38 przedstawiono zmiany wartości miar błędu oszacowania nacisku całkowitego pojazdu w trakcie uczenia dla różnych zakresów prędkości.



Rys. 5.36 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o prędkości od 0 do 10 m/s



Rys. 5.37 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o prędkości od 10 do 20 m/s



Rys. 5.38 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o prędkości od 20 do 30 m/s

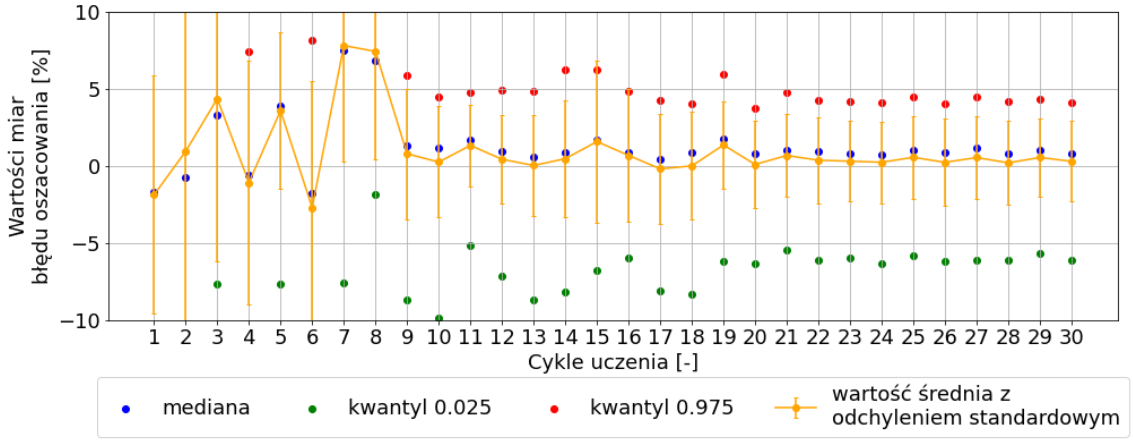
Analiza wykazała, że średnie błędy w oszacowaniu całkowitego nacisku pojazdu dla końcowego cyklu uczenia są porównywalne w obrębie wszystkich analizowanych przedziałów prędkości, co potwierdza

brak znaczących różnic w precyzji systemu w zależności od prędkości pojazdu. Takie obserwacje wskazują na to, że zaimplementowany system, oparty o sieci neuronowe, charakteryzuje się stabilnością w kontekście różnych prędkości przejazdu, co potwierdza jego przydatność w precyzyjnym określaniu nacisków.

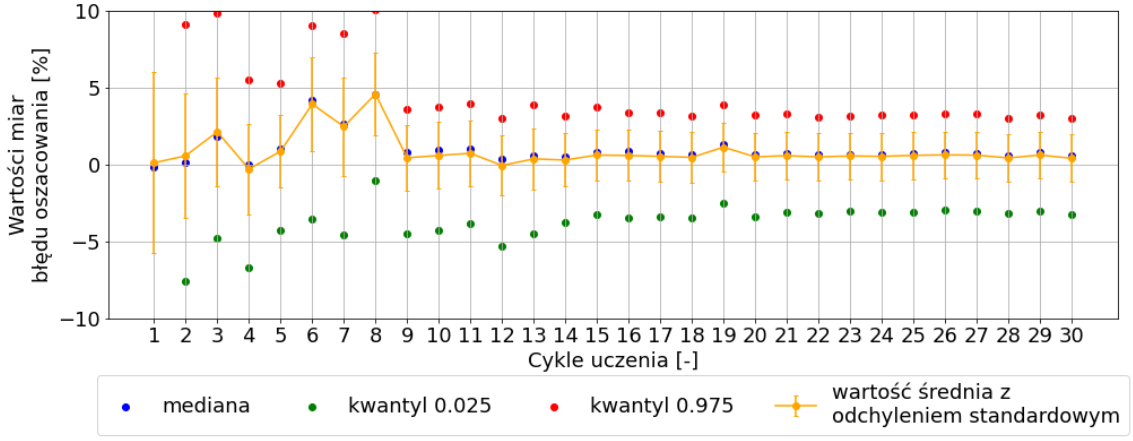
5.3.6 Analiza wpływu masy pojazdu na błąd odtworzenia

Dokonano analogicznej ewaluacji błędu oszacowania całkowitego nacisku pojazdu w zależności od masy całkowitej jak w podpunkcie 5.2.6 oraz zastosowane te same miary błędu oszacowania całkowitego nacisku pojazdu do oceny systemu.

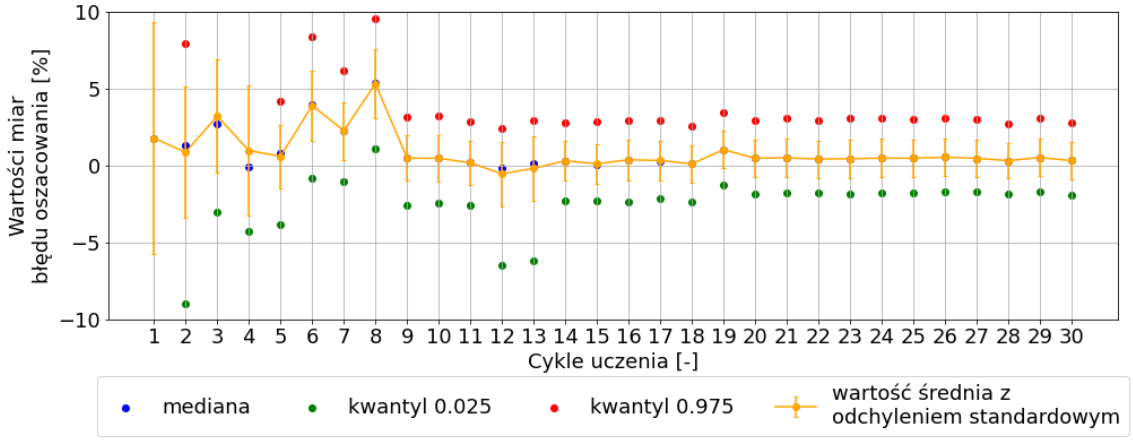
Przedstawiono na wykresach od rys. 5.39 do rys. 5.42 zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu dla przyjętych zakresów masy pojazdów.



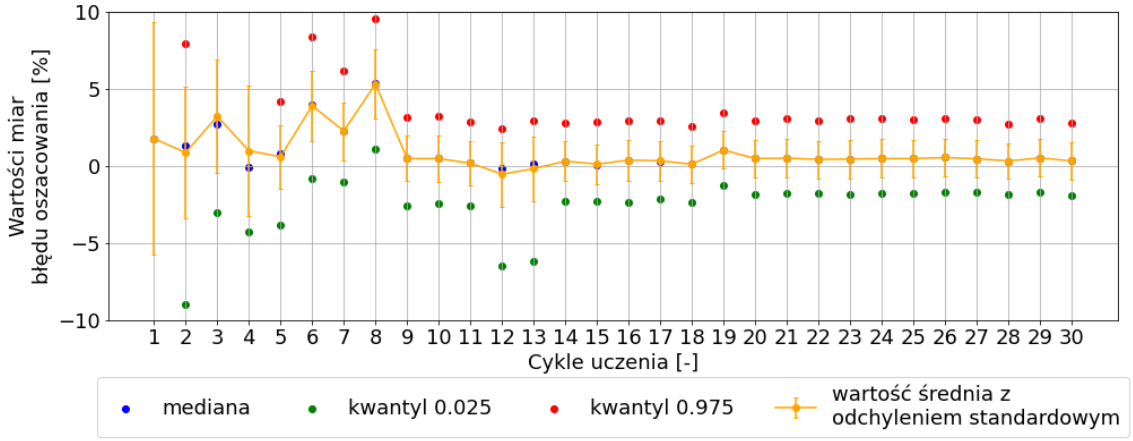
Rys. 5.39 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o masie całkowitej od 0 do 10 000 kg



Rys. 5.40 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o masie całkowitej od 10 0000 do 20 000 kg



Rys. 5.41 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o masie całkowitej od 20 0000 do 30 000 kg



Rys. 5.42 Zmiany wartości miar błędu oszacowania całkowitego nacisku pojazdu w trakcie uczenia systemu AutoSIO_SD przy uwzględnieniu pojazdów o masie całkowitej powyżej 30 000 kg

Analiza wykazała, że dla pojazdów lekkich (do 10 000 kg) średni błąd oszacowania całkowitego nacisku pojazdu może wynosić około 0.84%, przy czym 95% przedział błędu dla tej populacji mieści się w zakresie od -6.14% do 4.14%, co podkreśla stosunkowo znaczną niepewność oszacowań w tej kategorii.

Dla cięższych pojazdów, zwłaszcza tych o masie przekraczającej 30 ton, typowej dla ciężarówek na drogach, błąd oszacowania jest znacząco niższy, a 95% wartości błędów zawiera się w bardziej ograniczonym przedziale od -2.25% do 2.12%, co wskazuje na wyższą precyzję systemu w oszacowaniu całkowitych nacisków cięższych pojazdów.

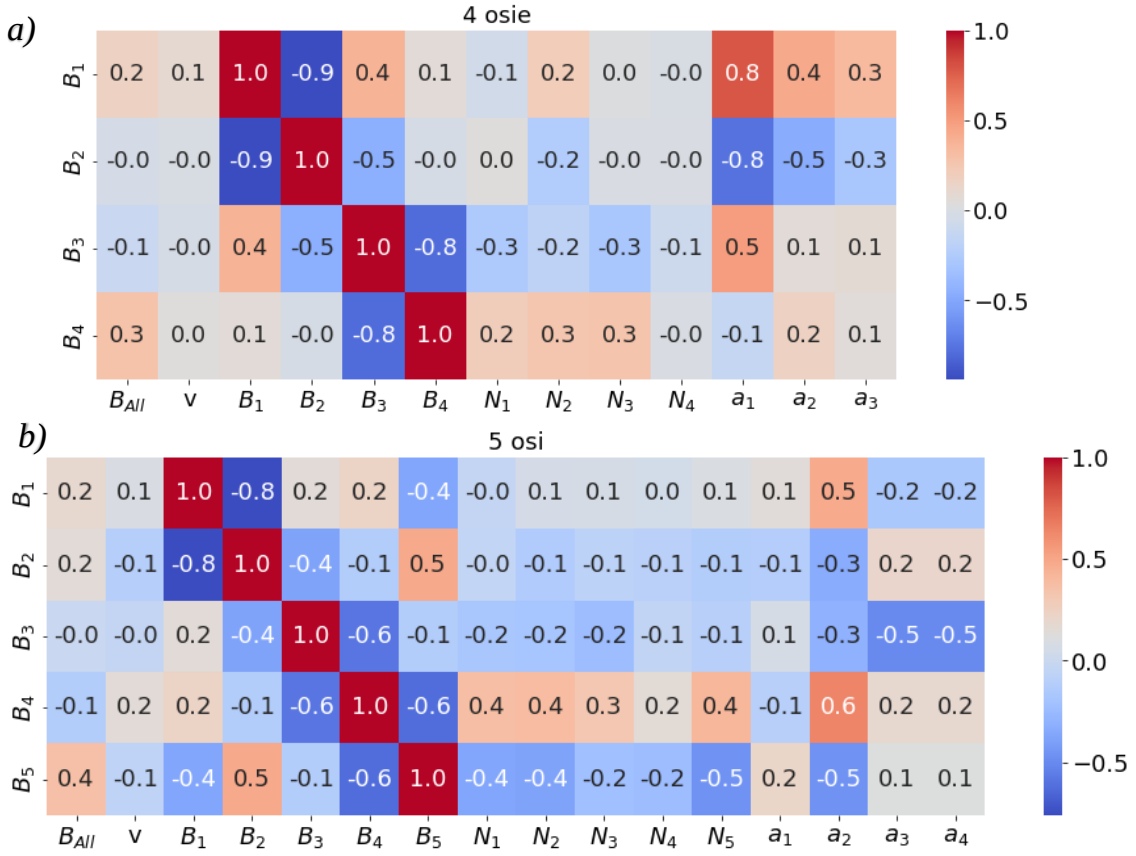
Takie obserwacje sugerują, że system, oparty o sieci neuronowe, jest bardziej efektywny w estymacji nacisków ciężkich pojazdów, co ma kluczowe znaczenie dla ochrony infrastruktury drogowej przed uszkodzeniami wynikającymi z nadmiernych obciążeń.

5.3.7 Analiza korelacji pomiędzy parametrami wejściowymi

Dokonano analogicznej ewaluacji korelacji pomiędzy parametrami jak w podpunkcie 5.2.7 oraz zastosowane te same miary do oceny.

Rozpatrzono zależność błędów oszacowania nacisku i-tej osi (B_i) od:

- B_{All} – błąd w określeniu całkowitego nacisku pojazdu,
- v – prędkość,
- B_i – błąd oszacowania nacisku i-tej,
- N_i – nacisk i-tej osi,
- a_i – rozstaw i między osią i -tą, a osią $i+1$.



Rys. 5.43 Wybrane korelacje Spearmana, odpowiednio dla pojazdów: a) 4-osiowych; b) 5-osiowych

Analiza nie wykazała znaczącej korelacji liniowej między błędami oszacowania, a prędkością pojazdów, co potwierdza wcześniejsze obserwacje, iż prędkość nie ma istotnego wpływu na precyzję oszacowań dokonywanych przez system. Pomimo istotnych błędów na poszczególnych osiach, ogólna dokładność oszacowania całkowitego nacisku pojazdu pozostaje na wysokim poziomie, co wskazuje na możliwość kompensacji błędów oszacowania między osiami.

Dodatkowo, analiza związku między błędami, a rozstawem osi ujawniła, że zmiany w rozstawie osi mogą mieć wpływ na dokładność oszacowań nacisków na poszczególne osie, co podkreśla złożoność interakcji między parametrami pojazdu a odpowiedzią konstrukcji mostowej. Zostało zauważone, że

zwiększenie rozstawu osi może przyczynić się do poprawy oszacowania nacisku na niektóre osie, podczas gdy na inne może mieć wpływ negatywny.

5.3.8 Podsumowanie

Podsumowując wyniki badań systemu AutoSIO_SD, można stwierdzić, że zastosowana sieć neuronowa, opierająca się na architekturze autodekodera z wykorzystaniem splotowych sieci neuronowych, efektywnie radzi sobie z zadaniem oceny całkowitych nacisków pojazdu. System wykazał się odpornością na zmiany prędkości pojazdów, co jest kluczową cechą dla systemów monitorowania ruchu drogowego.

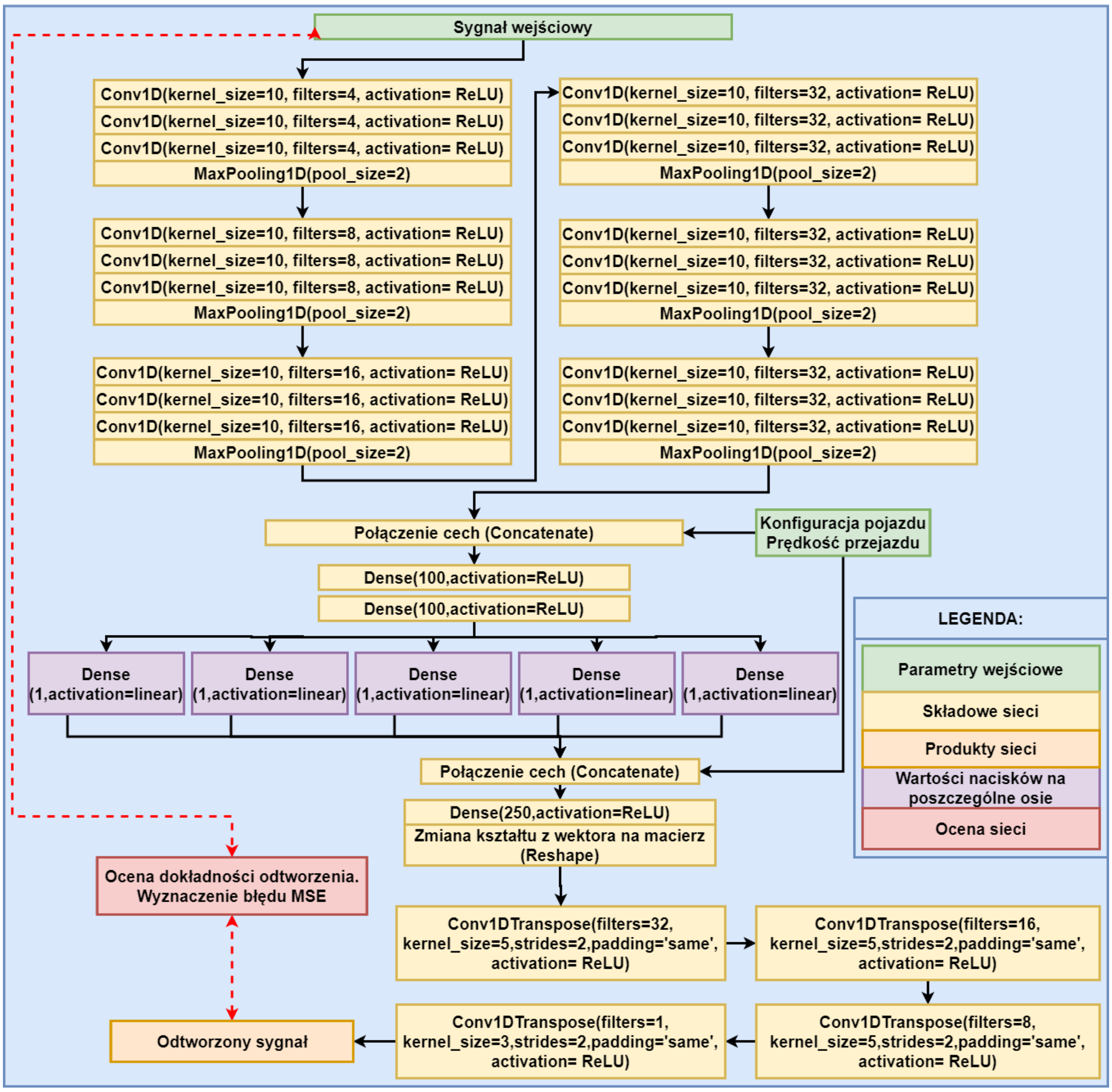
Osiągnięto satysfakcjonujące wyniki w oszacowaniu całkowitego nacisku pojazdu, co jest istotne z punktu widzenia oceny obciążenia infrastruktury drogowej. 95% błędów oszacowania całkowitego nacisku pojazdu mieści się w przedziale od -2.99% do 2.98%. System jednak wykazuje skłonność do znacznych błędów w oszacowaniach nacisków na indywidualne osie, co prowadzi do relatywnie wysokiej niepewności tych oszacowań. Niemniej jednak, system wykazuje zdolność do kompensacji błędów między osiami, co umożliwia uzyskanie wiarygodnych wyników dla całkowitego nacisku pojazdów oraz grup osi. Tak samo jak w przypadku systemu opisywanego w podrozdziale 5.2 możliwe jest zastosowanie systemu do zadowalająco precyzyjnej oceny nacisków pojazdów ciężkich. Minimalne błędy oszacowań sugerują potencjał systemu w identyfikacji i monitorowaniu pojazdów generujących największe obciążenie dla dróg i mostów.

Prezentowany system mimo pewnych ograniczeń, przedstawia się jako obiecujące narzędzie, które po dalszym udoskonaleniu może znaleźć szerokie zastosowanie w dziedzinie inżynierii drogowej i mostowej.

5.4 System AutoSIO_D odtwarzający dynamiczną odpowiedź obiektu mostowego

5.4.1 Struktura systemu

Przedstawione wyżej koncepcje systemu (AutoSIO_S oraz AutoSIO_SD), mimo różnic w architekturze i podejściu - statycznym czy dynamicznym – wykorzystywały wspólny element charakteryzujący konstrukcję mostową, a mianowicie funkcję wpływu. Jest to kluczowy element, ponieważ potrzebujemy stosować pewną miarę reakcji naszej konstrukcji mostowej, aby móc skalować sieć i na tej podstawie wyznaczać odpowiednie wartości sił obciążających. W systemie AutoSIO_D podjęto próbę przedstawienia rozwiązania, w którym nie dysponujemy wiedzą o konstrukcji mostowej. Znana jest jedynie odpowiedź konstrukcji obiektu mostowego oraz wybrane parametry pojazdu, takie jak odległości między osiami, prędkość pojazdu i liczba osi na obiekcie. Na rys. 5.44 przedstawiono ogólny schemat blokowy systemu opartego o sieci neuronowe. Dokładny opis implementacji został przedstawiony w załączniku C.3.



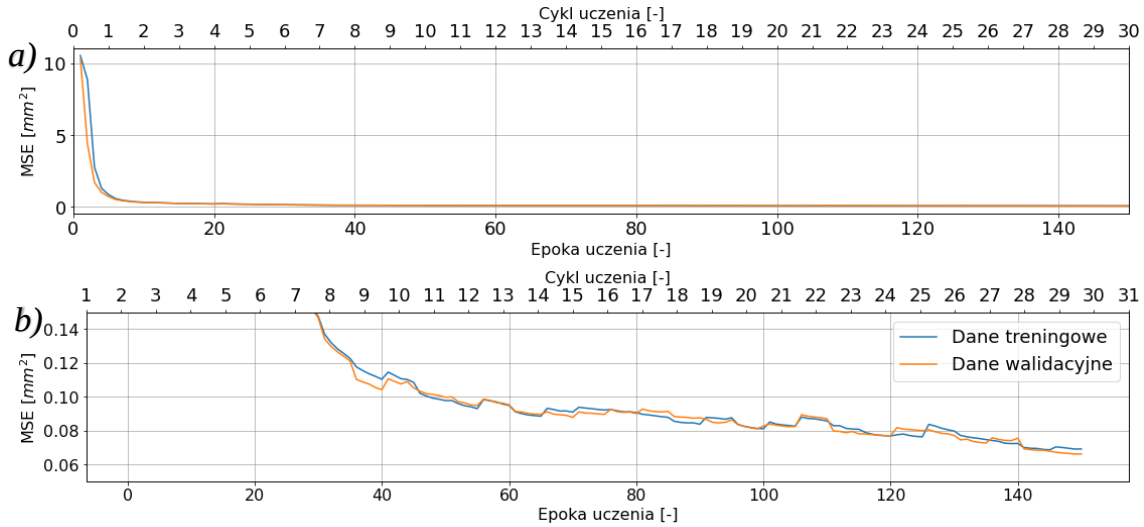
Rys. 5.44 Schemat blokowy systemu AutoSIO_D

Prezentowana sieć bazuje na autodekoderze wykorzystującym warstwy splotowe, gdzie sygnał odpowiedzi konstrukcji mostowej oraz dodatkowe parametry pojazdu, takie jak odległości między osiami i prędkość, stanowią dane wejściowe. System, korzystając z warstw splotowych, dokonuje ekstrakcji niezbędnych cech, które następnie są przekazywane do jądra logicznego. Zadaniem tego jądra jest określenie pięciu sił nacisku osi i odtworzenie z nich kluczowych elementów sygnału wejściowego.

Teoretycznie system będzie zobowiązany do ekstrakcji cech, co oznacza stworzenie pewnej logiki w jądrze autodekodera, tworząc ukrytą reprezentację zjawiska. Na podstawie tej ukrytej reprezentacji nastąpi odtworzenie sygnału. Przyjmując specjalną architekturę, poprzez ograniczenie warstw wyjściowych w jądrze autodekodera, zakłada się, że system powinien być w stanie oszacować całkowity nacisk pojazdu. Każdy neuron wyjściowy (fioletowy kwadracik) odpowiada za nacisk i-tej osi.

5.4.2 Proces treningu sieci

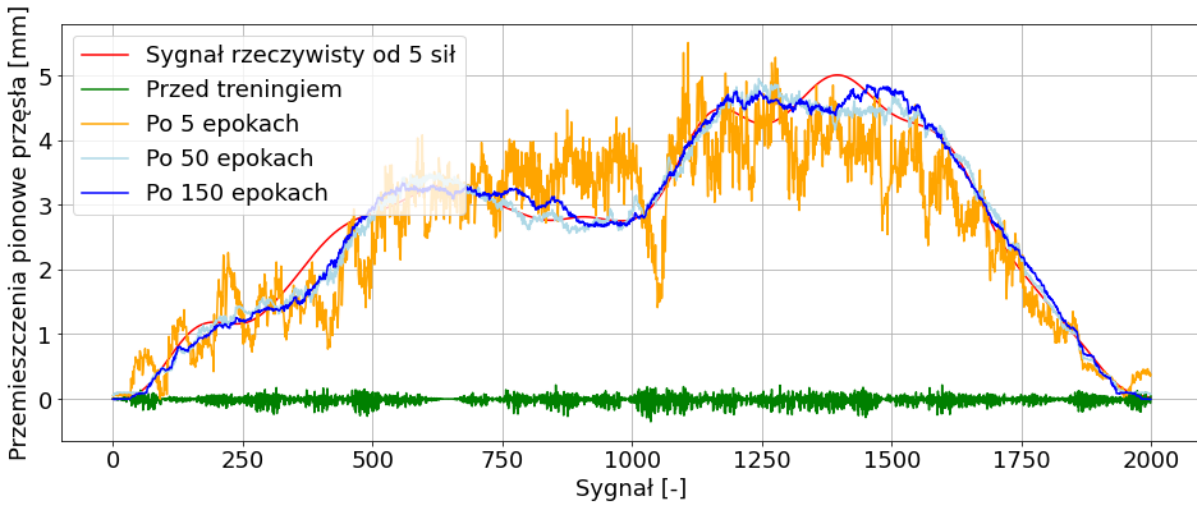
Przeprowadzono analogiczny schemat treningu jak w podpunkcie 5.2.2 oraz wykorzystano te same dane uczące oraz testowe. Wykresy krzywych uczenia przedstawionych na rys. 5.45 prezentują zmianę błędu średniokwadratowego w trakcie procesu trenowania sieci neuronowej.



Rys. 5.45 Wartości krzywych uczenia w trakcie uczenia systemu AutoSIO_D
a) Wykres przedstawia pełny zakres wartości MSE;
b) Wykres przedstawia zakres 0.05-0.15 wartości MSE

Analiza wykresów pokazuje, że błąd wartości funkcji kosztu szybko spadł do wartości stacjonarnych i utrzymywał się na stabilnym poziomie, co wskazuje na to, że sieć nie uległa przetrenowaniu.

Na rys. 5.46 przedstawiono efekty działania sieci neuronowej w trakcie uczenia się. Wybrano pojazd o sylwetce 2C+3N (ciągnik siodłowy + naczepa) z populacji testowej, a następnie w różnych fazach treningu sprawdzano dokładność odtworzenia sygnału wejściowego.



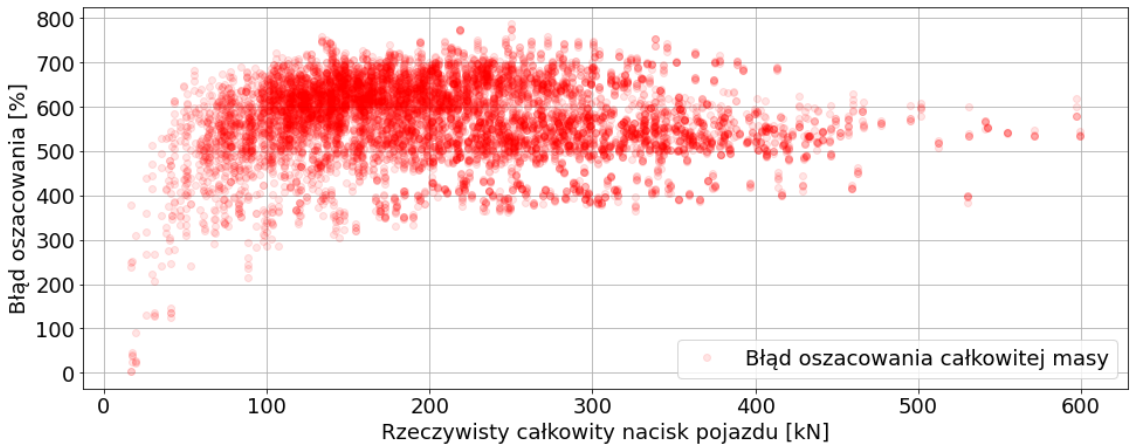
Rys. 5.46 Porównanie odtworzonego sygnału przez wariant systemu AutoSIO_D z sygnałem wejściowym dla różnych faz treningu

Prezentacja sygnałów z symulatora dla pojazdów oraz odtworzonych sygnałów przez system wskazuje na skuteczność programu w tworzeniu logicznej struktury. W tab. 5.3 przedstawiono wybrane wartości nacisków statycznych określone przez sieć na różnych etapach treningu, jak również zasymulowane wartości, która oddziaływały na konstrukcję mostową w symulatorze konstrukcji mostowej.

Tab. 5.3 Określone wartości nacisków statycznych na różnych etapach treningu systemu

Etap treningu	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
Rzeczywista wartość	54.81	54.25	48.56	47.95	53.87	109.06	150.38	259.44
Przed treningiem	0.29	-4.49	2.28	1.29	-26.42	-4.20	-22.85	-27.05
1 cykl	1125.90	1135.30	608.58	575.95	520.78	2261.20	1705.31	3966.51
10 cykl	781.79	776.90	409.58	464.16	182.43	1558.69	1056.17	2614.86
30 cykl	477.65	515.50	323.56	346.91	25.51	993.15	695.98	1689.13

Wartości oszacowanych nacisków osi są liczbami odbiegającymi od oczekiwanego wyniku. Na rys. 5.47 zaprezentowano błąd oszacowania całkowitego nacisku pojazdu dla populacji testowej.



Rys. 5.47 Błąd oszacowania całkowitego nacisku pojazdu przez system AutoSIO_D dla populacji testowej

Analiza błędów oszacowania całkowitego nacisku pojazdu na podstawie wartości neuronów, teoretycznie odpowiedzialnych za określenie nacisków osi, wykazuje, że błąd oszacowania całkowitego nacisku pojazdu przedziale do około 800%. Wartości błędu wykazują charakter losowy.

Można wyciągnąć wniosek, że wartości w jądrze dekodera niekoniecznie odpowiadają naciskom osi. Sieć nauczyła się odtwarzać sygnał wejściowy, jednak cechy wyszczególnione w jądrze nie są szukanymi wartościami.

5.4.3 Podsumowanie

Przedstawiona w podrozdziale 5.4 próba identyfikacji obciążeń przy użyciu systemu AutoSIO_D stanowi przykład problemu odwrotnego, gdzie na podstawie obserwowanego efektu staramy się zidentyfikować przyczyny i logikę, która mogła do niego doprowadzić. W tym kontekście, mając dany sygnał, próbujemy odtworzyć proces, który go wygenerował. Jednakże, istnieje nieskończenie wiele metod i ścieżek, które mogą prowadzić do uzyskania identycznego sygnału, co sprawia, że rzeczywista użyteczność wartości uzyskanych w jądrze i reprezentacji ukrytej jest niepewna.

Problem ten wynika z próby rozwiązania zagadnienia, które w pewnym sensie może wydawać się niemożliwe do rozwiązania bez wprowadzenia dodatkowych parametrów fizycznych, stałych czy miar, które pozwolą sieci nawiązać do rzeczywistych nacisków. Konieczne jest wprowadzenie dodatkowych elementów, takich jak funkcja wpływu, co stworzyłoby skalę umożliwiającą skorelowanie wartości sygnału z mierzalnymi wielkościami rzeczywistymi. To podkreśla znaczenie wprowadzenia określonej architektury i parametrów do sieci, aby na końcu uzyskać dokładne wartości, co jest kluczowe dla wiarygodności i praktycznej użyteczności systemu.

5.5 Podsumowanie i ogólna ocena opracowanych systemów

Ocena efektywności działania systemu jest zagadnieniem bardzo złożonym. Istnieje kilka aspektów, które można uwzględnić przy ocenie i porównywaniu systemów opartych o sieci neuronowe:

- rozkład błędów,
- interpretowalność działania systemu
- uogólnianie i zbieżność,
- złożoność systemu.

W tab. 5.4 przedstawiono zestawienie wybranych miar oceny dokładności działania systemów AutoSIO_S, AutoSIO_SD oraz AutoSIO_D – opracowanych w ramach niniejszej rozprawy. Dokładne analizy, uwzględniające kształt rozkładu błędu, historię jego zmian oraz dodatkowe parametry oceny zostały opisane w odpowiednich podpunktach.

Tab. 5.4 Zestawienie wybranych miar oceny dokładności działania wariantów systemu.

Parametry		Wariant systemu		
		AutoSIN_S (patrz 5.2._)	AutoSIN_SD (patrz 5.3._)	AutoSIN_D (patrz 5.4._)
Błąd oszacowania całkowitego nacisku [%] (patrz _._.3)		$< -2.41, 3.32 >$	$< -2.99, 2.98 >$	$< 500, 700 >$
Błąd oszacowania nacisku na osie [%] (patrz _._.4)		$< -18.55, 7.96 >$	$< -46.28, 9.02 >$	-
Błąd oszacowania całkowitego nacisku dla zakresu prędkość [%] (patrz _._.5)	$< 0{,}10 >$ (m/s)	$< -2.93, 2.75 >$	$< -3.57, 1.88 >$	-
	$< 10{,}20 >$ (m/s)	$< -2.50, 2.79 >$	$< -2.64, 2.57 >$	-
	$< 20{,}30 >$ (m/s)	$< -1.74, 4.05 >$	$< -2.93, 3.43 >$	-
Błąd oszacowania całkowitego nacisku dla zakresu masy pojazdów [%] (patrz _._.6)	$< 0{,}10 >$ (t)	$< -4.94, 7.77 >$	$< -6.14, 4.14 >$	-
	$< 10{,}20 >$ (t)	$< -2.80, 3.60 >$	$< -3.25, 3.02 >$	-
	$< 20{,}30 >$ (t)	$< -1.56, 2.60 >$	$< -1.89, 2.82 >$	-
	$< 30, +\infty >$ (t)	$< -0.91, 2.22 >$	$< -2.25, 2.11 >$	-

Na podstawie przeprowadzonych analiz, można stwierdzić, że zarówno proponowany system z wykorzystaniem podejścia quasi-statycznego oraz system wykorzystujący podejście mieszane poprawnie określają naciski całkowite pojazdu, dobrze radzą sobie z wyznaczaniem nacisków na grupy osi, jednak mają problem z określeniem nacisków na poszczególne osie w grupie. Pamiętać należy, że sieci bazują na globalnej odpowiedzi konstrukcji mostowej. System wykorzystujący tylko globalną odpowiedź konstrukcji mostowej, co prawda potrafi odtworzyć sygnał, jednak brak parametru fizycznego wiążącego go z konstrukcją mostową, sprawia, że wygenerowane parametry nie reprezentują nacisków na poszczególne osie. Zarówno system quasi-statycznej jak i system mieszany są interpretowalne. Jesteśmy w stanie dokładnie opisać i zrozumieć poszczególne zadania poszczególnych

elementów składowych sieci w systemie. Systemy dobrze radzą sobie z nowymi danymi, co zostało sprawdzone na podstawie populacji testowej, która nie brała udziału w treningu.

Sieci neuronowe osiągają zbieżność już dla populacji 50 tys. pojazdów. Zarówno system w wariancie quasi-statycznym, jak i mieszanym, charakteryzują się podobnymi czasami trenowania, inferencji oraz zasobności obliczeniowej.
Sieci neuronowe w opracowanych na potrzeby niniejszej rozprawy systemach mają odpowiednio 119 777 parametrów oraz 126 128. W porównaniu do popularnego ChatGPT 4.0, który posiada wg szacunków około 100 bilionów parametrów, są to niewielkie sieci, do których treningu można wykorzystać biurowy komputer stacjonarny wyposażony w komercyjną kartę graficzną.
Do dalszej analizy wybrano system oparty na podejściu mieszanym. Rozwiązanie takie dostarcza nam nie tylko przewidywaną odpowiedź statyczną konstrukcji, ale także odpowiedź dynamiczną, mając przy tym tylko niewiele zwiększoną złożoność obliczeniową.

6 Analiza stabilności systemu AutoSIO_SD

6.1 Podstawowe założenia i cele rozdziału

W rozdziale 5 przedstawiono wyniki działania rozpatrywanych systemów przy wykorzystaniu wygenerowanej populacji pojazdów poruszających się po obiekcie mostowym. Sprawdzono mechanizm zdolności do samodoskonalenia (auto-adaptacji) się opisanych systemów. Określono błędy oszacowania nacisków pojazdów na podstawie wygenerowanej populacji testowej. Sprawdzono wpływ wybranych parametrów na dokładność działania sieci.

Celem tego rozdziału jest określenie i zbadanie warunków brzegowych działania systemu AutoSIO_SD, opisanego w podrozdziale 5.3, a także ocena jego niezawodności i odporności na zmienne warunki w realnym środowisku.

W kontekście fizycznych aspektów konstrukcji mostowych, rozpatrzono wpływ następujących parametrów konstrukcji:

- rozpiętość konstrukcji,
- masa własna konstrukcji,
- nierówność nawierzchni.

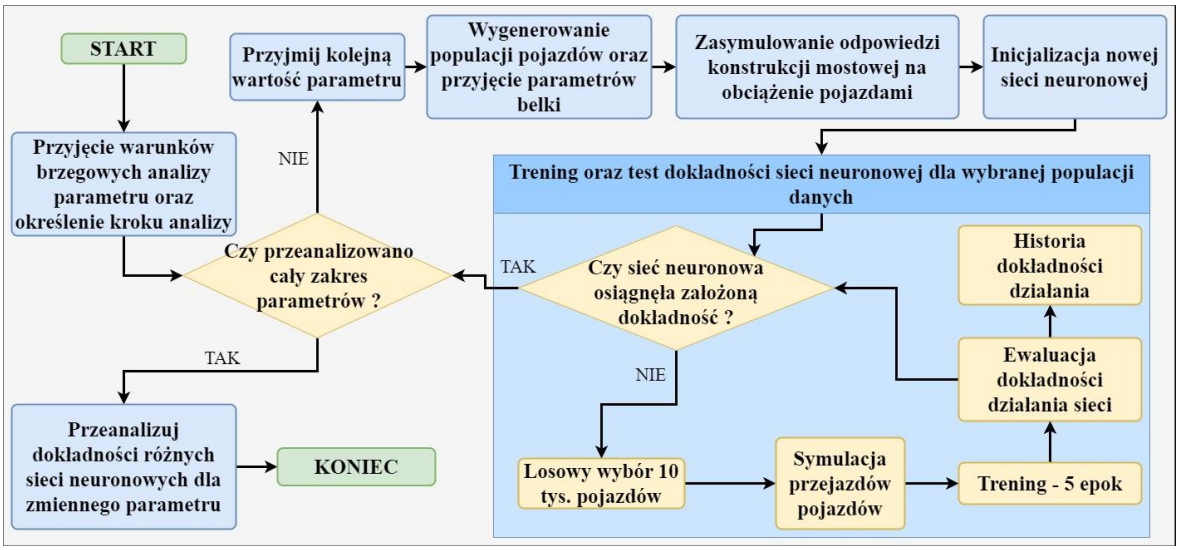
Ogólny schemat analizy wpływu parametrów konstrukcji na działanie sieci przedstawiono na rys. 6.1.

W rozdziale analizowano także stabilność systemu oraz odporności na błędy pomiarowe, biorąc pod uwagę czynniki takie jak:

- szum pomiarowy odpowiedzi konstrukcji mostowej,
- niepewności pomiarowe przy identyfikacji pojazdów,
- wpływ liczby danych uczących na dokładność systemu,
- wpływ dokładności określenia funkcji wpływu osi pojazdu.

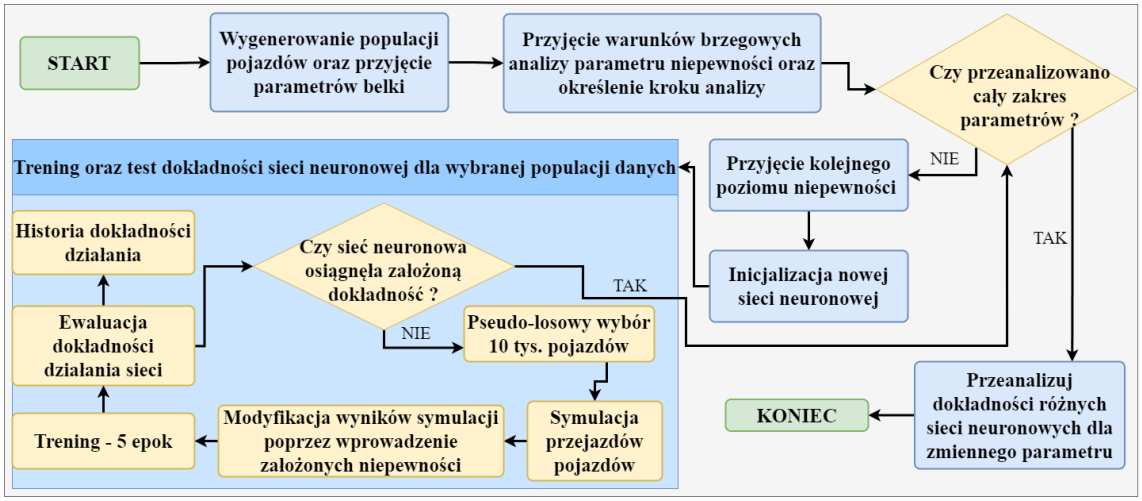
Na rys. 6.2 przedstawiono schemat blokowy analizy wpływu niepewności pomiarowych na dokładność sieci.

Należy wyraźne zaznaczyć, że testowanie systemu odbywa się poprzez wykorzystanie metody atrap, czyli obiektów, które symulują zachowanie się komponentów systemu i pozwalają na pełną kontrolę. W przypadku pracy, mowa jest o symulatorze dynamicznej odpowiedzi konstrukcji mostowej, który zawiera w sobie generator populacji pojazdów oraz wirtualny model mostu (patrz podrozdział 3.3 oraz rys. 3.14). Dzięki takiemu podejściu dysponujemy pełną kontrolą nad parametrami analiz i możemy porównać wartości oszacowane przez system, do wartości zasymulowanych.



Rys. 6.1 Schemat blokowy weryfikacji wpływu parametrów konstrukcji oraz pojazdów na dokładność systemu

Zgodnie ze schematem przyjmowano iteracyjnie kolejne wartości parametrów symulacji do analizy. Dla każdej wartości generowano populację pojazdów oraz przeprowadzano symulacje odpowiedzi konstrukcji. Następnie na otrzymanych danych trenowano sieć neuronową i zapisywano historię efektów uczenia. Po przeanalizowaniu pełnego zakresu parametrów, dokonywano analizy otrzymanych wyników.



Rys. 6.2 Schemat blokowy weryfikacji wpływu niepewności pomiarowych na dokładność systemu

W odróżnieniu od procedury przedstawionej na rys. 6.1, zgodnie ze schematem na rys. 6.2, tworzona jest jedna populacja pojazdów Następnie definiowane są warunki brzegowe analizy parametru oraz krok analizy. Główna pętla zapewnia, że analiza zostanie przeprowadzona dla całego zakresu parametrów. Dla analizowanego parametru inicjalizowana jest nowa sieć neuronowa, która następnie jest trenowana w cyklach po 5 epok. Warto zaznaczyć, że komputery nie generują rzeczywistych liczb losowych, ponieważ proces ten opiera się na deterministycznym algorytmie, takim jak Mersenne Twister (stosowanym m.in. w Pythonie). Możliwe jest jednak zablokowanie stanu początkowego generatora liczb losowych (domyślnie wykorzystywany jest czas systemowy, czyli liczba sekund od początku 1970

roku, będąca parametrem zmiennym i niepowtarzalnym). Dzięki temu można uzyskać powtarzalne sekwencje liczb losowych. Oznacza to, że w procesie treningu, niezależnie od liczby iteracji, dane wejściowe mogą być podawane w tej samej kolejności, co pozwala wyeliminować wpływ losowości na eksperyment i ocenić wyłącznie wpływ analizowanego parametru.

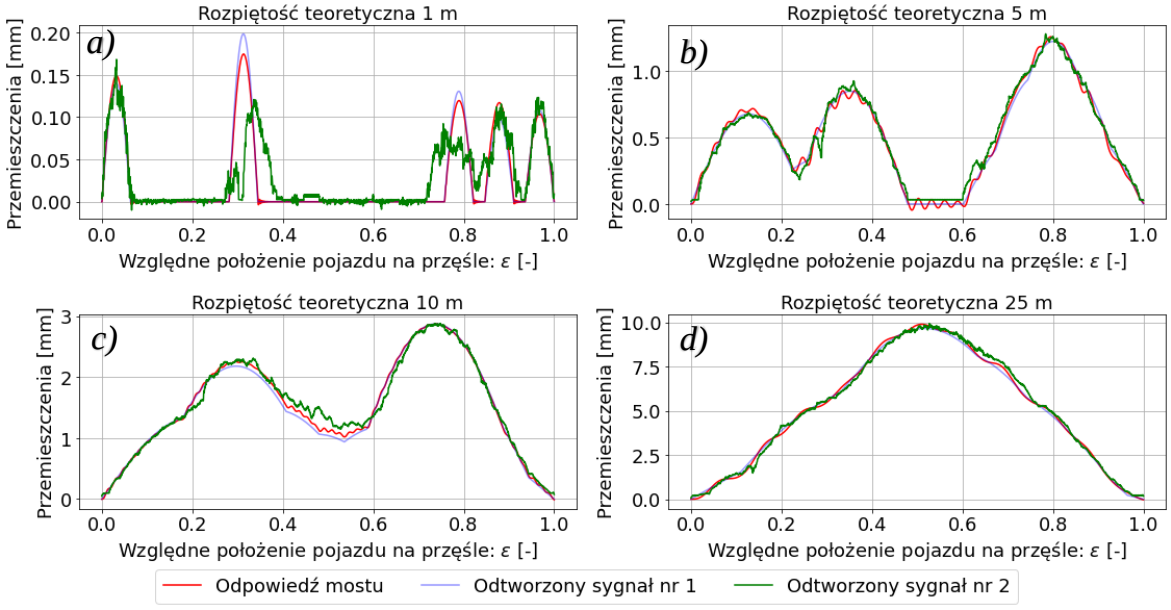
6.2 Analiza wpływu rozpiętości teoretycznej obiektu mostowego

6.2.1 Informacje ogólne

W podrozdziale 6.2 sprawdzono wpływ rozpiętości konstrukcji mostowej na dokładność działania systemu. W ramach analizy przyjęto wartości graniczne rozpiętości konstrukcji mostowych w przedziale od 1 metra do 25 metrów. Proces obliczeniowy został przeprowadzony metodą iteracyjną, co umożliwiło sprawdzenie dokładności sieci neuronowej po zastosowaniu jednolitego cyklu uczenia się. Efektywność oraz dokładność sieci oceniano pod kątem zdolności do określenia całkowitego nacisku pojazdu. Założono, że model konstrukcji przęseł będzie belką jednoprzęsłową, swobodnie podpartą. Przyjęto analogiczne założenie jak w podpunkcie 4.4.5, czyli założono wartości dopuszczalne ugięcia od pojazdu referencyjnego K, dla którego wraz ze wzrostem rozpiętości, ugięcia dopuszczalne zwiększają się liniowo. Na tej podstawie dobierano parametr sztywności konstrukcji, aby stosunek przemieszczenia do przemieszczenia dopuszczalnego był zawsze ten sam niezależnie od rozpiętości.

6.2.2 Wybrane efekty działania sieci

Na rys. 6.3 zilustrowano działanie systemu dla losowo wybranego przejazdu pojazdu pięcioosiowego. Przedstawiono odpowiedzi symulacji konstrukcji mostowej dla różnych rozpiętości oraz wartości odtworzone przez system. Wybrano 4 rozpiętości teoretyczne tj. 1,5,10 i 25 metrów.



Rys. 6.3 Zasymulowane i odtworzone przez system odpowiedzi mostu o różnych rozpiętościach teoretycznych, odpowiednio: a) 1 m; b) 5 m; c) 10 m; d) 25 m

Zauważany jest fakt, że im dłuższy obiekt, tym mniejsza możliwość odseparowania poszczególnych osi, lub grup osi z sygnału odpowiedzi konstrukcji. Obiekt staje się stosunkowo długi w porównaniu do długości pojazdu. Opracowany system dobrze radzi sobie z odtworzeniem sygnału wejściowego.

W tab. 6.1 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.2 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych rozpiętości konstrukcji mostowej. W tab. 6.3 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.1 Zamodelowane wartości nacisków losowo wybranego pojazdu

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.2 Wartości nacisków określonych przez system dla różnych rozpiętości teoretycznych konstrukcji

Rozpiętość teoretyczna [m]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
1	51.59	73.11	48.09	35.96	39.84	124.69	123.89	248.58
5	52.67	65.95	41.33	41.56	42.21	118.63	125.11	243.73
10	55.55	60.18	38.62	51.37	34.79	115.73	124.78	240.51
25	56.95	61.97	28.98	71.15	24.14	118.91	124.27	243.18

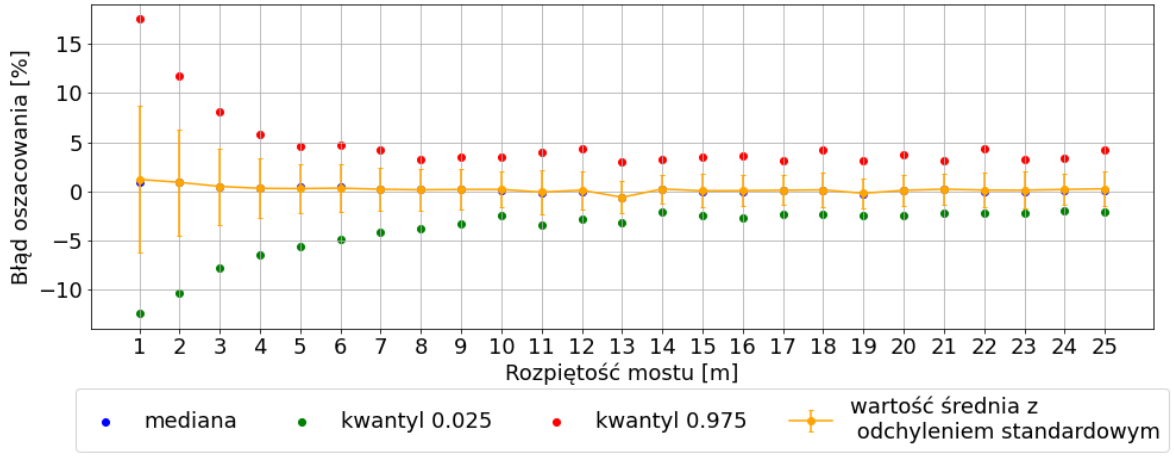
Tab. 6.3 Błąd określenia nacisków statycznych dla losowo wybranego pojazdu

Rozpiętość teoretyczna [m]	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
1	-5.94	13.76	9.26	-16.57	4.09	4.69	-1.20	1.67
5	-3.96	2.63	-6.10	-3.59	10.30	-0.41	-0.23	-0.32
10	1.28	-6.35	-12.27	19.15	-9.09	-2.84	-0.49	-1.64
25	3.83	-3.57	-34.16	65.05	-36.93	-0.16	-0.90	-0.54

Analizując wyniki uzyskane przez system, można zaobserwować, że niezależnie od rozpiętości obiektu mostowego, sieć była zdolna do precyzyjnego wyznaczania całkowitego nacisku pojazdu oraz nacisków poszczególnych grup osi.

6.2.3 Dokładność określenia całkowitego nacisku pojazdu

W poniższym podpunkcie dokonano syntetycznej analizy wyników uzyskanych dla 25 niezależnych sieci neuronowych, z których każda została skonfigurowana dla specyficznej rozpiętości konstrukcji mostowej. Badanie skupiało się na charakterystyce rozkładu błędu w kontekście ostatniego cyklu uczenia, wykorzystując do tego celu testową populację danych, która nie była wcześniej eksponowana na działanie sieci, a zatem nie uczestniczyła w procesie jej uczenia. Populacja ta służyła wyłącznie do oceny precyzji predykcyjnej wariantów sieci. Na rys. 6.4 przedstawiono rozkład błędu dla ostatniego cyklu uczenia systemu.



Rys. 6.4 Rozkład błędu oszacowania całkowitego nacisku pojazdu dla ostatniego cyklu uczenia systemu, dla różnych rozpiętości teoretycznych

Analiza danych przedstawionych na wykresie wskazuje, że zwiększająca się rozpiętość konstrukcyjna obiektu mostowego koreluje z poprawą dokładności oszacowań całkowitego nacisku pojazdów. Wnioskować można, że większe rozpiętości umożliwiają precyzyjniejsze określanie całkowitego nacisku pojazdu. Im dłuższy jest most, tym dłużej pojazd oddziaływał na niego i tym większą liczbę informacji opisujących tę zależność jesteśmy w stanie przekazać do sieci. Ponadto, obserwuje się, że przy rozpiętościach przekraczających 10 metrów dokładność sieci neuronowych ulega stabilizacji, osiągając zakładany poziom precyzji w granicach od -5% do +5%.

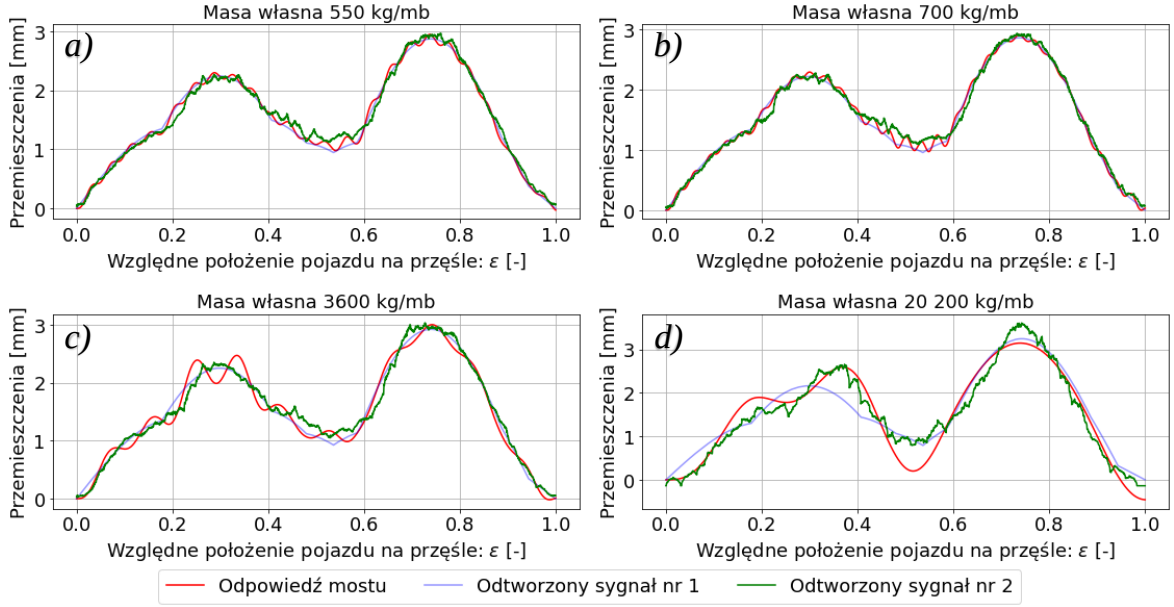
6.3 Analiza wpływu masy obiektu mostowego

6.3.1 Informacje ogólne

W podrozdziale 6.3 sprawdzono wpływ masy własnej konstrukcji belki na dokładność działania proponowanego systemu. W ramach analizy przyjęto graniczne wartości masy własnej w zakresie od $545\,\frac{kg}{mb}$ do $20200\,\frac{kg}{mb}$. Zastosowano nieregularny krok iteracyjny, gdzie każdy kolejna masa jest o $x\%$ większa od poprzedniej, przy założeniu, że $x\%$ zwiększa się liniowo od 3.5% do 20 % dla ostatniej iteracji. Parametry takie jak sztywność belki, dekrement tłumienia, rozpiętość pozostały takie same. Zwiększano więc wpływ bezwładności belki na działanie systemu. Efektywność oraz dokładność sieci oceniano pod kątem zdolności do określenia całkowitego nacisku pojazdu.

6.3.2 Wybrane efekty działania sieci

Na rys. 6.5 zilustrowano działanie systemu dla wybranego przejazdu pojazdu pięcioosiowego. Przedstawiono zasymulowane odpowiedzi konstrukcji mostowej dla różnych mas własnych oraz wartości odtworzone przez opracowany system. Wybrano 4 wartości masy własnej belki.



Rys. 6.5 Zasymulowane i odtworzone przez system odpowiedzi mostu o różnych wartościach masy własnej, odpowiednio: a) 550 kg/mb; b) 700 kg/mb; c) 3600 kg/mb; d) 20 200 kg/mb

W tab. 6.4 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.5 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych rozpiętości konstrukcji mostowej. W tab. 6.6 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.4 Zamodelowane wartości nacisków losowo wybranego pojazdu

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.5 Wartości nacisków określonych przez system dla różnych wartości masy własnej belki

Wartość parametru $[\frac{kg}{mb}]$	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
550	56.51	62.61	37.35	50.54	36.97	119.11	124.87	243.98
700	56.14	61.72	39.06	50.21	35.32	117.86	124.60	242.46
3600	56.23	63.24	34.59	57.91	33.79	119.48	126.29	245.76
20200	54.50	60.35	24.27	82.60	31.70	114.84	138.58	253.42

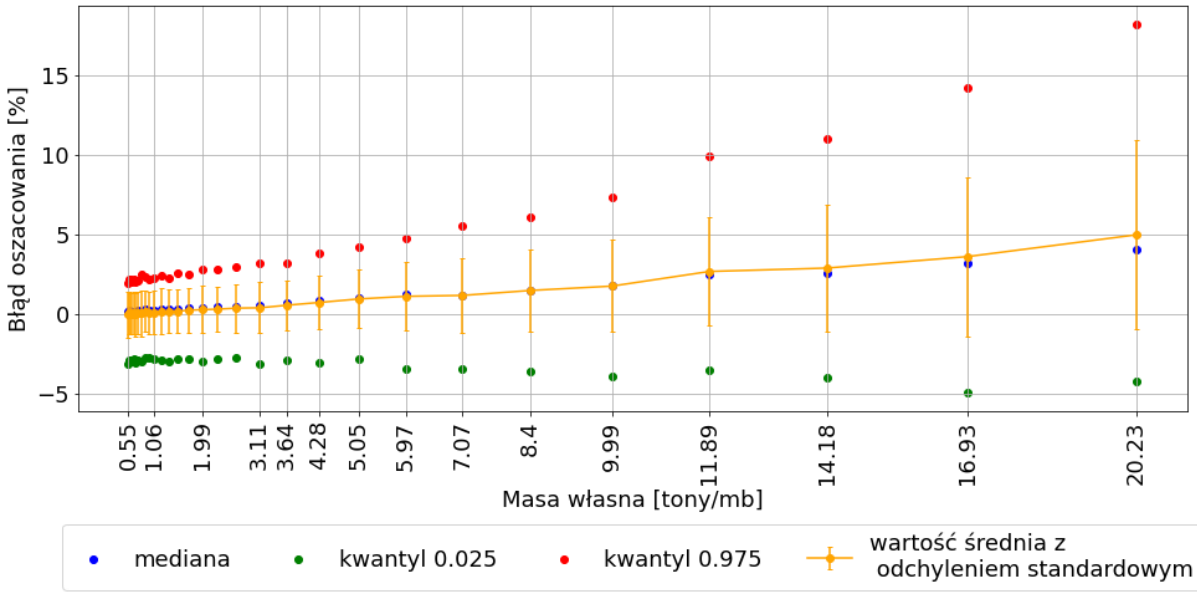
Tab. 6.6 Błąd określenia nacisków statycznych dla losowo wybranego pojazdu

Wartość parametru $[\frac{kg}{mb}]$	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
550	3.03	-2.58	-15.13	17.25	-3.40	0.00	-0.42	-0.21
700	2.37	-3.96	-11.26	16.49	-7.70	-1.05	-0.64	-0.84
3600	2.53	-1.59	-21.41	34.34	-11.72	0.31	0.71	0.51
20200	-0.63	-6.10	-44.86	91.62	-17.16	-3.58	10.51	3.65

Uzyskane wyniki potwierdzają, że sieć była zdolna do precyzyjnego wyznaczania całkowitego nacisku pojazdu oraz nacisków poszczególnych grup osi. Jednak, aby wyciągnąć ogólne wnioski, należy przeanalizować zachowanie sieci dla całej populacji pojazdów testowych.

6.3.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.6 przedstawiono rozkład błędu przy oszacowaniu całkowitego nacisku pojazdu w ostatnim cyklu uczenia systemu w zależności od masy własnej belki.



Rys. 6.6 Wykres błędów oszacowania całkowitego nacisku pojazdu dla ostatniej epoki uczenia systemu w zależności od masy własnej

Analiza wyników pozwala nam określić, że im większa bezwładność konstrukcji mostowej, tym większe niedokładności oszacowania całkowitego nacisku pojazdu. Wynika to z większej bezwładności układu i spowodowanej tym opóźnionej reakcji konstrukcji mostowej na przejazd pojazdu.

6.4 Analiza wpływu nierówności nawierzchni

6.4.1 Informacje ogólne

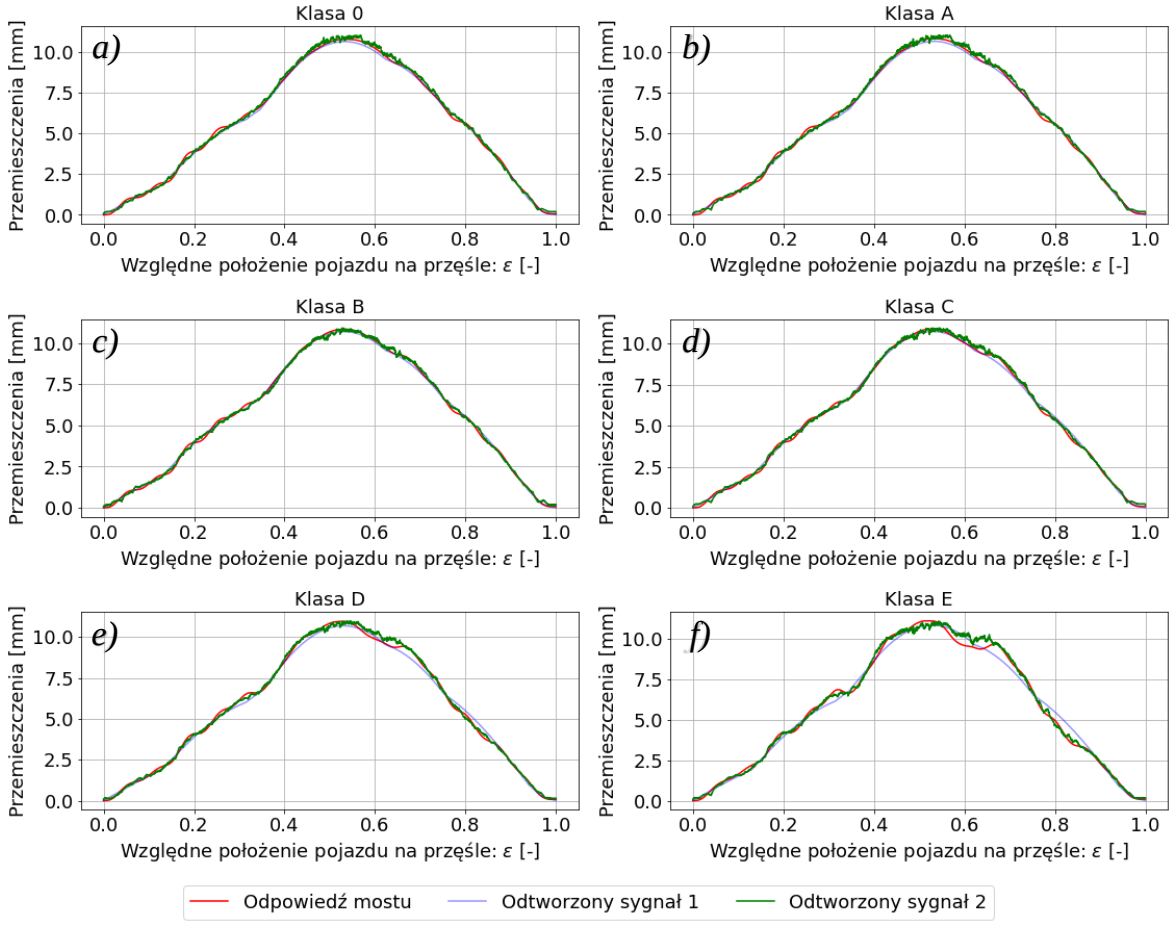
W podrozdziale 6.4 sprawdzono wpływ nierówności nawierzchni na dokładność działania systemu. W ramach analizy przyjęto profil drogi idealnej (bez nierówności) oraz profile klasy od A do E. Nierówności nawierzchni modelowano zgodnie z podpunktem 4.2.1.

Proces obliczeniowy został przeprowadzony metodą iteracyjną, co umożliwiło sprawdzenie dokładności sieci neuronowej po zastosowaniu jednolitego cyklu uczenia się.

Efektywność oraz dokładność sieci oceniano pod kątem zdolności do określenia całkowitego nacisku pojazdu.

6.4.2 Wybrane efekty działania sieci

Na rys. 6.7 zilustrowano działanie systemu dla przejazdu wybranego pojazdu pięcioosiowego. Przedstawiono zamodelowane odpowiedzi konstrukcji mostowej dla różnych rozpiętości oraz wartości odtworzone przez opracowany system. Wybrano 6 klas jakości drogi.



Rys. 6.7 Zasymulowane i odtworzone przez system odpowiedzi mostu dla różnych klas nierówności nawierzchni, odpowiednio: a) klasa 0; b) klasa A; c) klasa B; d) klasa C; e) klasa D; f) klasa E

W tab. 6.7 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.8 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych klas nierówności nawierzchni. W tab. 6.9 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.7 Zamodelowane wartości nacisków losowo wybranego pojazdu referencyjnego

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
59.35	70.30	50.06	49.42	42.75	129.65	142.23	271.87

Tab. 6.8 Zasymulowane naciski osi wybranego pojazdu oraz wartości nacisków określonych przez sieć neuronową dla różnych klas nierówności nawierzchni

Klasy nierówności nawierzchni [-]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	58.05	67.34	44.89	74.00	25.80	125.39	144.69	270.09
A	58.85	67.16	44.08	70.42	30.52	126.01	145.02	271.03
B	59.84	69.52	38.62	76.78	27.91	129.36	143.31	272.67
C	60.43	69.82	40.85	74.99	27.54	130.25	143.38	273.63
D	59.48	69.38	45.70	68.74	28.71	128.86	143.14	272.00
E	60.02	72.17	46.36	70.79	26.21	132.19	143.36	275.55

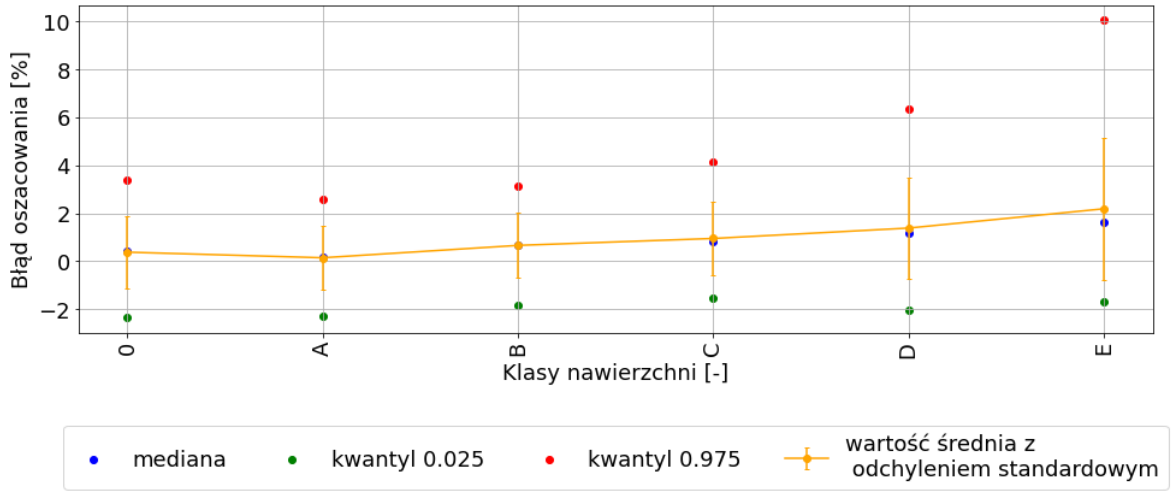
Tab. 6.9 Błąd określenia nacisków dla losowo wybranego pojazdu

Klasy nierówności nawierzchni [-]	Błąd określenia nacisków [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	-2.19	-4.20	-10.32	49.74	-39.64	-3.28	1.73	-0.66
A	-0.85	-4.46	-11.93	42.49	-28.61	-2.81	1.97	-0.31
B	0.83	-1.11	-22.84	55.36	-34.73	-0.22	0.76	0.29
C	1.82	-0.68	-18.40	51.73	-35.57	0.47	0.81	0.65
D	0.22	-1.30	-8.71	39.08	-32.85	-0.61	0.64	0.05
E	1.13	2.67	-7.38	43.24	-38.69	1.96	0.80	1.35

Analizując wyniki uzyskane przez sieć, można zaobserwować, że sieć była zdolna do precyzyjnego wyznaczania całkowitego nacisku pojazdu oraz nacisku grup osi. Jednak, aby wyciągnąć ogólne wnioski, należy przeanalizować zachowanie sieci dla całej populacji pojazdów testowych.

6.4.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.8 przedstawiono rozkład błędu przy oszacowaniu całkowitego nacisku pojazdu w ostatnim cyklu uczenia sieci neuronowej, biorąc pod uwagę stopień nierówności nawierzchni.



Rys. 6.8 Rozkład błędu oszacowania całkowitego nacisku pojazdu dla ostatniego cyklu uczenia systemu, dla różnych klas nierówności nawierzchni

Analiza wyników pozwala nam stwierdzić, że im większe nierówności nawierzchni, tym gorsze oszacowania całkowitego nacisku pojazdu. Wartości dla kategorii nawierzchni od 0 do C mieszczą się w przedziale błędu od -2% do 4 %. Można założyć, że proponowany system będzie montowany na drogach charakteryzujących się bardzo dobrą lub dobrą jakością nawierzchni drogowej.

6.5 Analiza wpływu liczby danych uczących

6.5.1 Informacje ogólne

W podrozdziale 6.5 zbadano wpływ liczby danych uczących na dokładność działania systemu. Celem było ustalenie, ile danych należy dostarczyć, aby sieć neuronowa, będąca elementem składowym systemu, osiągnęła stabilną dokładność w szacowaniu nacisków.

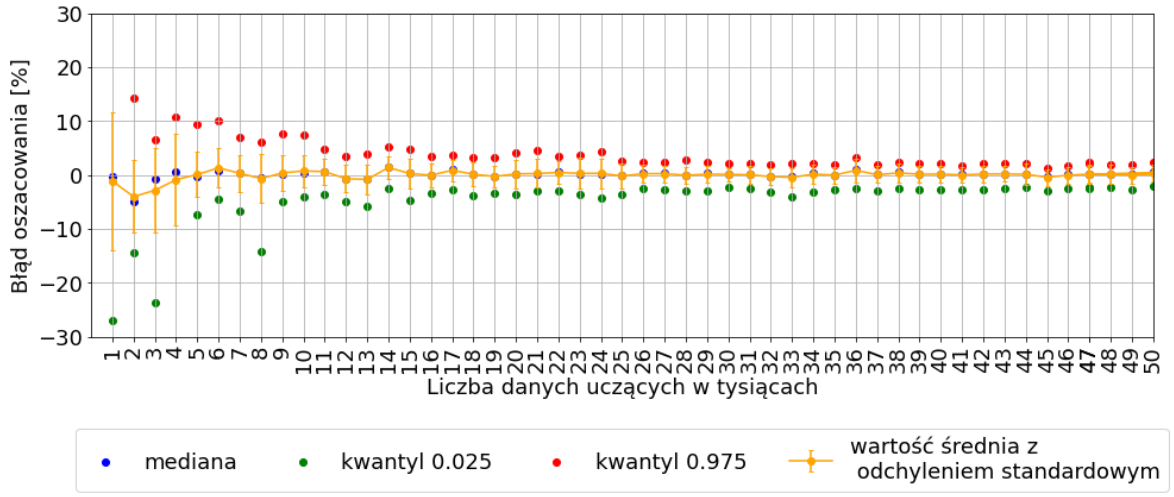
W ramach eksperymentu określono następującą procedurę:

1. Przyjmij populację testową, na której będzie ewaluowana sieć.
2. Zainicjuj nową sieć neuronową oraz zwiększ liczbę danych uczących, na których może uczyć się sieć.
3. Po określonej liczbie iteracji cykli treningowych określ efektywność działania sieci oraz zapisz wynik.
4. Wróć do kroku 2.

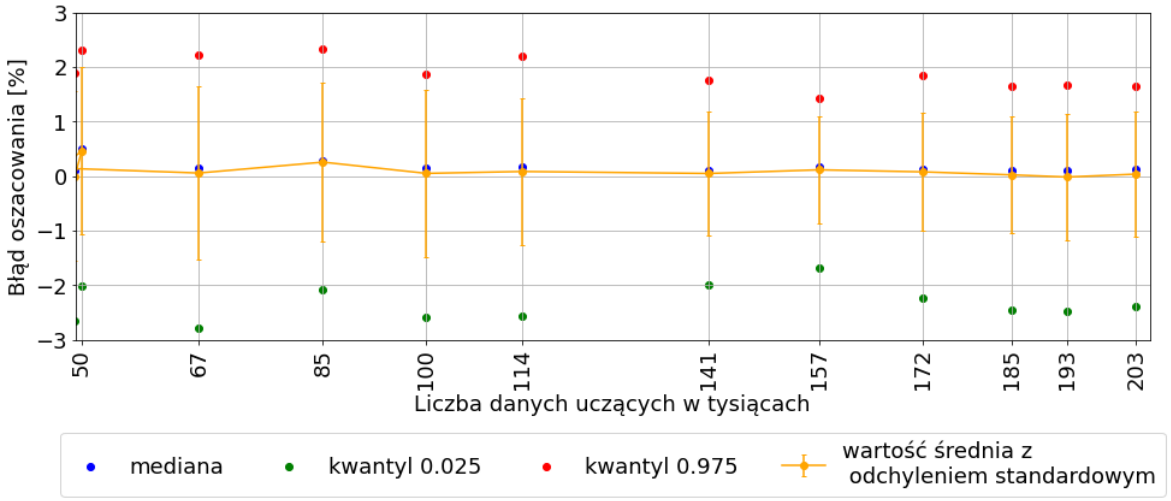
Analizę przeprowadzono w zakresie od 1 tys. danych uczących, aż do 203 tys. danych uczących.

6.5.2 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.9 oraz rys. 6.10 przedstawiono rozkład błędu przy określaniu całkowitego nacisku pojazdu dla ostatniego cyklu uczenia w zależności od liczby danych uczących. Dane te nie odnoszą się do pojedynczego pojazdu, ale do rozkładu błędów wśród testowej populacji.



Rys. 6.9 Wykres błędów oszacowania całkowitego nacisku pojazdu dla ostatniej epoki uczenia sieci neuronowej w zależności od rozmiaru danych uczących



Rys. 6.10 Wykres błędów oszacowania całkowitego nacisku pojazdu dla ostatniej epoki uczenia sieci neuronowej w zależności od rozmiaru danych uczących

Na podstawie wykresów można stwierdzić, że analizowana sieć neuronowa osiągnęła stabilną dokładność już przy 30 tys. danych uczących. Zwiększanie danych ponad ten poziom nie przynosiło wymiernych korzyści w dokładności działania sieci.

Należy pamiętać, że podana liczba danych uczących nie jest wartością uniwersalną dla wszystkich systemów opartych na sieciach neuronowych. Dla każdego przypadku, należy wykonać indywidualną analizę stabilności.

6.6 Analiza wpływu szumu pomiarowego w odpowiedzi konstrukcji

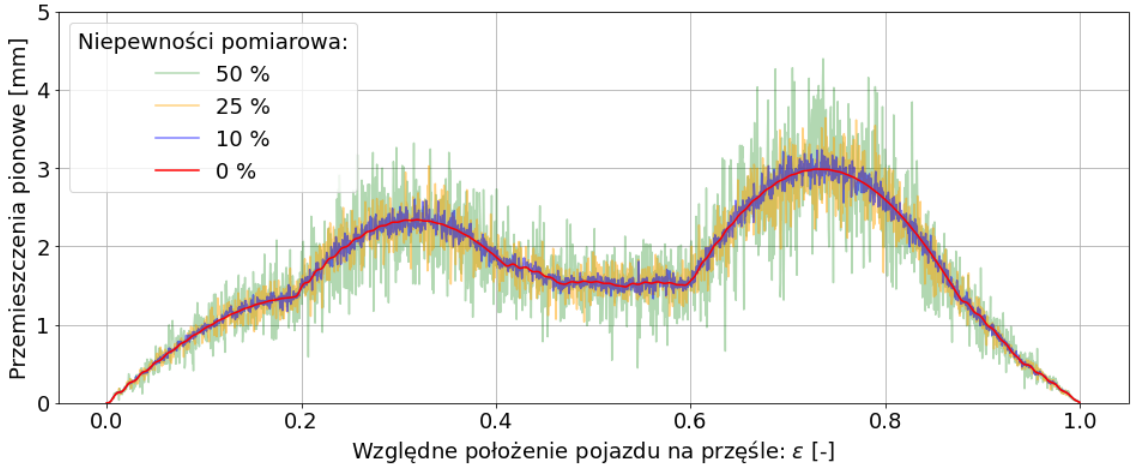
6.6.1 Informacje ogólne

W podrozdziale 6.6 sprawdzono, czy szum pomiarowy sygnału odpowiedzi konstrukcji mostowej ma znaczący wpływ na dokładność działania systemu identyfikacji obciążeń.
Szum pomiarowy został zdefiniowany jako zbiór niepewności pomiarowych wynikających z działania czujników oraz dodatkowych czynników zewnętrznych, które mogą negatywnie wpływać na precyzję uzyskanych danych.

Szum wygenerowano na podstawie rozkładu normalnego o średniej równej 0, co oznacza, że nie wprowadza on systematycznego przesunięcia sygnału, a jedynie losowe odchylenia wokół teoretycznej wartości. Wartość odchylenia standardowego dla szumu zależy od zdefiniowanego poziomu niepewności pomiarowej i jest proporcjonalna do sygnału. Dokładniej, dla każdego sygnału szum wygenerowano za pomocą funkcji normalnej $N(0, \sigma)$, gdzie odchylenie standardowe σ było równe:

$$\sigma = \frac{|niepewność\ pomiarowa\ \times sygnał|}{2}$$

Przyjęto, że 95% pomierzonych wartości mieści się w zakresie niepewności pomiarowej, co odpowiada dwóm odchyleniom standardowym rozkładu normalnego.
Analizę przeprowadzono dla różnych poziomów niepewności, od 0% (brak szumu, idealne warunki pomiarowe) do +/-50% niepewności pomiarowej, przy założeniu kroku iteracyjnego wynoszącego 2.5%. W każdym scenariuszu szum dodawano bezpośrednio do zasymulowanego sygnału odpowiedzi obiektu mostowego.
Na rys. 6.11 przedstawiono sygnał odpowiedzi konstrukcji mostowej przy różnych poziomach niepewności pomiarowej dla wybranego pojazdu 5 osiowego.



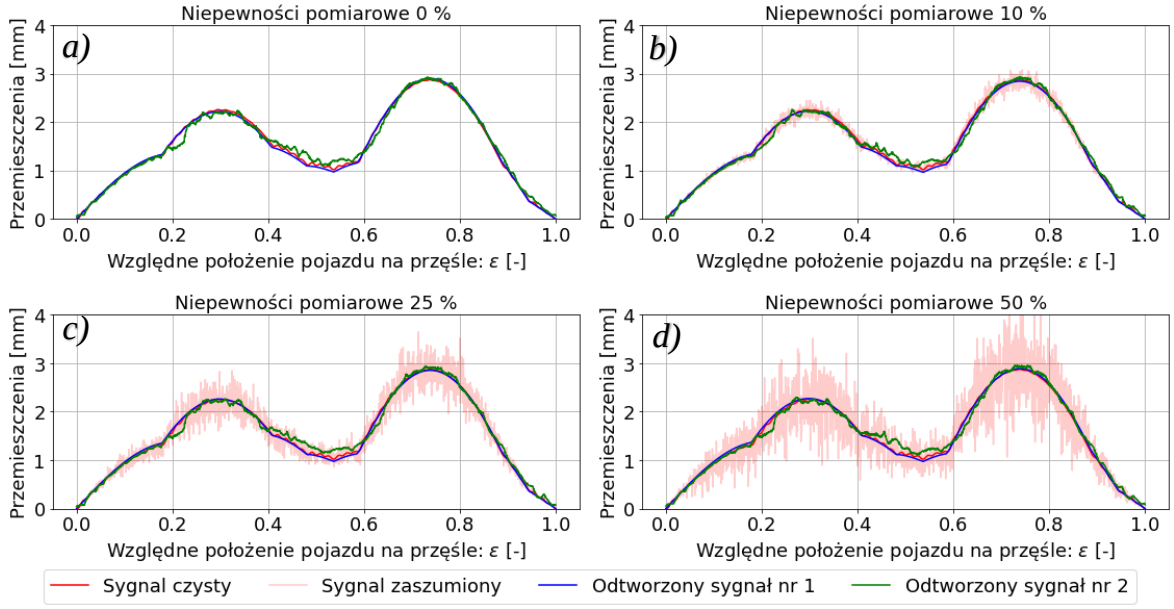
Rys. 6.11 Sygnał odpowiedzi konstrukcji mostowej z różnymi poziomami niepewności pomiarowej

Głównym kryterium oceny wpływu szumu pomiarowego na dokładność działania sieci była precyzja w estymacji całkowitego nacisku pojazdu dla ostatniej epoki uczenia, z uwzględnieniem różnych

poziomów zakłóceń. Skoncentrowano się głównie na całkowitym nacisku pojazdu jako kluczowym wskaźniku dokładności.

6.6.2 Wybrane efekty działania sieci

Na rys. 6.12 zilustrowano działanie systemu opartego o sieci neuronowe dla przejazdu wybranego pojazdu pięcioosiowego. Demonstruje on efektywność sieci w różnych warunkach, uwzględniając szum na poziomie 0%, 10%, 25% oraz 50%. Wykres zawiera czysty sygnał, który przedstawia zamodelowaną odpowiedź konstrukcji bez wprowadzenia jakichkolwiek niepewności pomiarowych, służąc tym samym jako punkt odniesienia. Dodatkowo, dla każdego poziomu szumu, przedstawiono również sygnał obarczony przyjętymi niepewnościami pomiarowymi. Odtworzone sygnały zostały również uwzględnione i zaprezentowane na rys. 6.12 dla wybranych poziomów niepewności pomiarowych, co pozwoliło na ocenę zdolności systemu identyfikacji obciążeń do adaptacji i korekcji w obliczu zakłóceń.



Rys. 6.12 Zasymulowane i odtworzone przez system odpowiedzi mostu przy różnych poziomach niepewności pomiarowej sygnału odpowiedzi, odpowiednio: a) 0%; b) 10%; c) 25%; d) 50%

Na podstawie analizy przejazdu wybranego pojazdu 5 osiowego przy różnych poziomach szumu pomiarowego, obserwujemy, że sieć neuronowa efektywnie identyfikowała i odtwarzała zarówno sygnał statyczny, jak i dynamiczny. Co istotne, odtworzone sygnały nie wykazywały cech zaszumienia, sugerując, że sieć nie tylko odtworzyła sygnały, ale również efektywnie je odfiltrowała z niepewności pomiarowych. Innymi słowy, odtworzone sygnały nie były prostym klonem pierwotnych sygnałów zaszumionych, lecz przypominały sygnały pozbawione dodatkowych niepewności. Wskazuje to na zdolność sieci do wyodrębnienia istotnych informacji z zaszumionych danych, co jest kluczowe dla dokładnej rekonstrukcji odpowiedzi konstrukcji mostowej.

Zdolność systemu do odszumiania danych jest efektem zastosowania autodekoderów, które poprzez swoją strukturę i mechanizm działania, są w stanie wydobyć istotne cechy z zaszumionych danych, jednocześnie eliminując zakłócenia, co czyni je niezwykle użytecznymi w aplikacjach wymagających precyzyjnego odtwarzania i analizy sygnałów.

W tab. 6.10 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.11 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych poziomów niepewności pomiarowej sygnału odpowiedzi konstrukcji. W tab. 6.12 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.10 Zamodelowane wartości nacisków losowo wybranego pojazdu referencyjnego

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.11 Wartości nacisków określonych przez system z różnymi poziomami niepewności pomiarowej sygnału odpowiedzi

Poziom szumu [%]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	55.73	61.91	39.98	49.73	36.48	117.63	126.20	243.83
10	55.92	62.26	39.07	48.53	36.13	118.18	123.74	241.91
25	56.51	63.45	38.98	49.03	36.13	119.97	124.14	244.10
50	57.20	63.34	38.96	49.24	37.41	120.55	125.61	246.16

Tab. 6.12 Błąd określenia nacisków statycznych dla losowo wybranego pojazdu

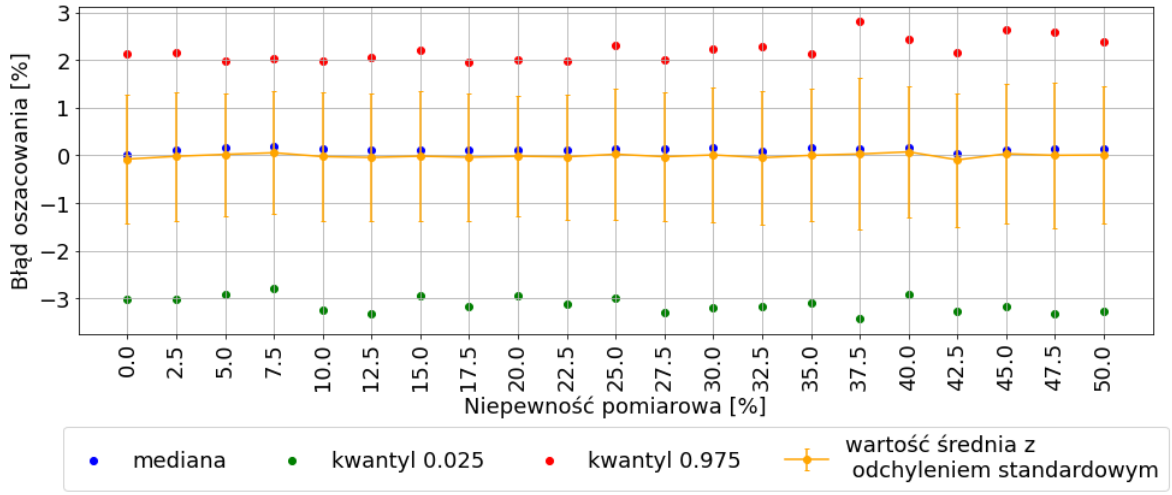
Poziom szumu [%]	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	1.61	-3.67	-9.16	15.37	-4.67	-1.24	0.64	-0.27
10	1.95	-3.12	-11.24	12.59	-5.59	-0.78	-1.32	-1.06
25	3.04	-1.26	-11.45	13.73	-5.59	0.72	-1.00	-0.16
50	4.30	-1.44	-11.49	14.23	-2.25	1.21	0.17	0.68

Analizując wyniki uzyskane przez system, można zaobserwować, że niezależnie od poziomu szumu, był on zdolny nie tylko do efektywnego oczyszczania danych, ale również do precyzyjnego wyznaczania całkowitego nacisku pojazdu oraz nacisków poszczególnych grup osi.

To wskazuje, że mimo pewnych ograniczeń, system zachowuje użyteczność w praktycznych zastosowaniach, gdzie pewien poziom niepewności pomiarowej jest nieunikniony.

6.6.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.13 przedstawiono rozkład błędu przy określaniu całkowitego nacisku pojazdu dla ostatniego cyklu uczenia, uwzględniając różne poziomy niepewności pomiarowej. Dane te nie odnoszą się do pojedynczego pojazdu, ale do rozkładu błędów wśród testowej populacji.



Rys. 6.13 Wykres błędów oszacowania całkowitego nacisku pojazdu dla ostatniej epoki uczenia sieci neuronowe w zależności od wartości niepewności pomiarowej sygnału odpowiedzi konstrukcji mostowej

Analiza wyników pozwala stwierdzić, że sieć neuronowa zachowała stabilną dokładność w estymacji całkowitego nacisku pojazdu dla całej populacji, szczególnie dla poziomów szumu poniżej 35%. Jednakże, zauważalne jest rozszerzenie wartości odpowiednich kwantyli, reprezentujących 95%-owy przedział ufności, co sugeruje wzrost błędu przy wyższych poziomach szumu.

Mimo że istnieje ryzyko, iż sieć neuronowa może stracić zdolność do efektywnego odszumiania i wyodrębniania kluczowych cech przy bardzo wysokich poziomach szumu, wyniki wskazują na jej skuteczność aż do poziomu około 30%. Jest to poziom zadowalający, biorąc pod uwagę, że większość systemów pomiarowych operuje z mniejszą niepewnością pomiarową.

Oznacza to, że w typowych warunkach pomiarowych sieć neuronowa wykazuje efektywność i jest w stanie prawidłowo funkcjonować, co potwierdza jej przydatność w realnych zastosowaniach.

6.7 Analiza wpływu niepewności określenia prędkości pojazdu

6.7.1 Informacje ogólne

W podrozdziale 6.7 zbadano wpływ niepewności pomiaru prędkości pojazdu na dokładność działania sieci neuronowej w określaniu nacisków osi oraz całkowitego nacisku pojazdu. Celem było ustalenie czy błąd w pomiarze prędkości jest czynnikiem istotnie wpływającym na wyniki systemu.

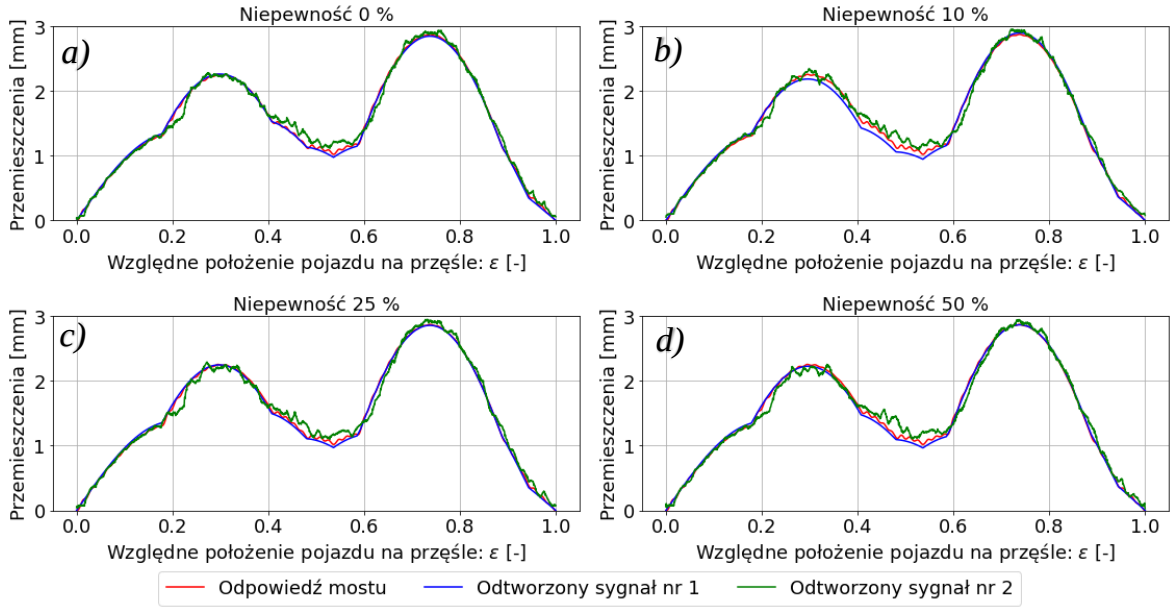
Analiza objęła różne scenariusze niepewności pomiaru, obejmujące zakres od 0% (idealna dokładność pomiaru prędkości) do 50% niepewności pomiarowej. Przyjęto normalny rozkład błędów pomiarowych, z założeniem, że zmierzona wartość z prawdopodobieństwem 95% mieści się w zakresie plus/minus od wartości rzeczywistej. W praktyce oznacza to, że błędy w pomiarze prędkości są modelowane za pomocą funkcji rozkładu normalnego $N(0,\sigma)$, gdzie odchylenie standardowe σ zależy od zdefiniowanego poziomu niepewności pomiarowej. Wartość szumu dodawana do prędkości jest proporcjonalna do oryginalnej wartości prędkości pojazdu i obliczana według wzoru:

$$szum = random.normal(0, niepewność \times \frac{prędkość\ rzeczywista}{2})$$

Szum jest iteracyjnie dodawany do pomiaru prędkości, zwiększając poziom szumu o krok 2.5%, aby zbadać, jak różne poziomy niepewności pomiarowej wpływają na dokładność systemu identyfikacji obciążeń.

6.7.2 Wybrane efekty działania sieci

Na wykresie rys. 6.14 ukazano wyniki działania sieci neuronowej dla pojazdu pięcioosiowego, poruszającego się ze stałą prędkością po konstrukcji mostowej, przy różnych poziomach niepewności pomiarowych prędkości, czyli przy poziomie: 0%, 10%, 25% oraz 50%.



Rys. 6.14 Zasymulowane i odtworzone przez system odpowiedzi mostu przy różnych poziomach niepewności określenia prędkości pojazdu, odpowiednio: a) 0%; b) 10%; c) 25%; d) 50%

W tab. 6.13 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.14 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych poziomów niepewności pomiarowej prędkości pojazdu. W tab. 6.15 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.13 Zamodelowane wartości nacisków losowo wybranego pojazdu referencyjnego

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.14 Wartości nacisków określonych przez system dla z różnymi poziomami niepewności pomiarowej prędkości pojazdu

Poziom niepewności pomiarowej prędkości [%]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	55.96	63.92	38.82	50.93	33.88	119.88	123.63	243.51
10	56.23	59.76	39.55	50.68	35.68	115.99	125.91	241.89
25	56.72	62.64	39.61	50.10	34.61	119.36	124.32	243.68
50	56.44	61.84	39.83	48.40	36.64	118.28	124.88	243.16

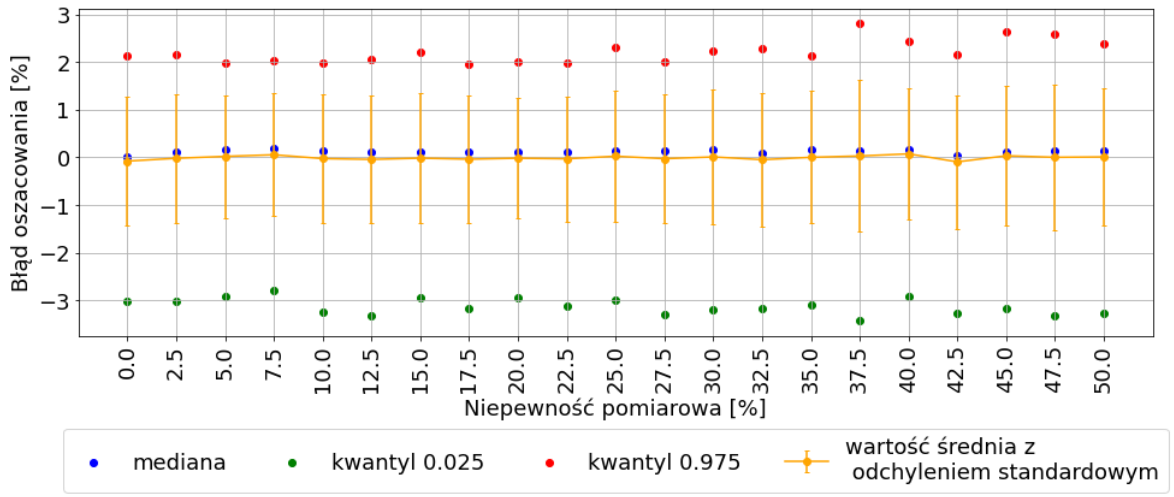
Tab. 6.15 Błąd określenia nacisków statycznych dla losowo wybranego pojazdu

Poziom niepewności pomiarowej prędkości [%]	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	2.03	-0.54	-11.81	18.14	-11.46	0.64	-1.41	-0.41
10	2.53	-7.02	-10.14	17.57	-6.78	-2.62	0.41	-1.07
25	3.42	-2.52	-10.01	16.23	-9.58	0.21	-0.86	-0.34
50	2.92	-3.78	-9.50	12.28	-4.25	-0.70	-0.41	-0.55

Wyniki te wskazują, że system był zdolny do precyzyjnego określenia wartości dla pojazdu, niezależnie od błędu pomiaru prędkości.

6.7.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.15 przedstawiono rozkład błędu przy oszacowaniu całkowitego nacisku pojazdu w ostatnim cyklu uczenia sieci neuronowej, biorąc pod uwagę różne poziomy dokładności w pomiarze prędkości pojazdu.



Rys. 6.15 Wykres błędów oszacowania całkowitego nacisku pojazdu dla ostatniej epoki systemu w zależności od niepewności pomiarowej prędkości pojazdu

Z przeprowadzonej analizy wynika, że dokładność systemu w określaniu całkowitego nacisku pojazdu pozostawała stabilna, niezależnie od poziomu niepewności pomiaru prędkości. Ten fakt sugeruje, że sieć neuronowa jest odporna na potencjalne błędy w określeniu prędkości przejazdu pojazdu.
W obliczu zmiennej dokładności pomiaru prędkości, obserwowana stabilność sieci wskazuje na to, że zastosowany system ma zdolność do ekstrakcji kluczowych cech z dostarczanych danych, które są istotne dla dokładnego modelowania. Zmiana jednego parametru, takiego jak prędkość, może wpłynąć na wyniki, lecz nie stanowi kluczowego aspektu, który mógłby zasadniczo zakłócić dokładność i skuteczność działania sieci neuronowej w kontekście określania całkowitego nacisku pojazdu.

6.8 Analiza wpływu niepewności określenia konfiguracji osi pojazdu

6.8.1 Informacje ogólne

W podrozdziale 6.8 podjęto próbę oceny wpływu niepewności pomiaru konfiguracji osi pojazdu na efektywność systemu w precyzyjnym określaniu nacisków osi oraz całkowitego nacisku pojazdu. Analizowany jest błąd w pomiarze odległości między osiami pojazdu, co bezpośrednio przekłada się na potencjalne błędy w określeniu pozycji podłużnej osi w czasie.

System identyfikacji pojazdów, który określa liczbę osi i odległości między nimi, wykorzystuje różnorodne technologie i każda z nich charakteryzuje się określoną dokładnością. W związku z tym, założono, że każdy pomiar odległości między osiami – na przykład od osi pierwszej do drugiej – może być obarczony pewnym błędem wynikającym z niepewności inherentnych dla danego systemu pomiarowego.

Przyjęto, że błąd pomiaru odległości między osiami ma rozkład normalny, z zakresem niepewności, w którym 95% pomiarów znajduje się w przedziale plus/minus od wartości rzeczywistej. Innymi słowy, błędy te są modelowane na podstawie rozkładu normalnego $N(0, \sigma)$, gdzie odchylenie standardowe σ zależy od zdefiniowanego poziomu niepewności pomiarowej.

Wartość szumu pomiarowego dodawana do odległości między osiami zmienia się proporcjonalnie do pierwotnych wartości i jest opisana pseudo-kodem:

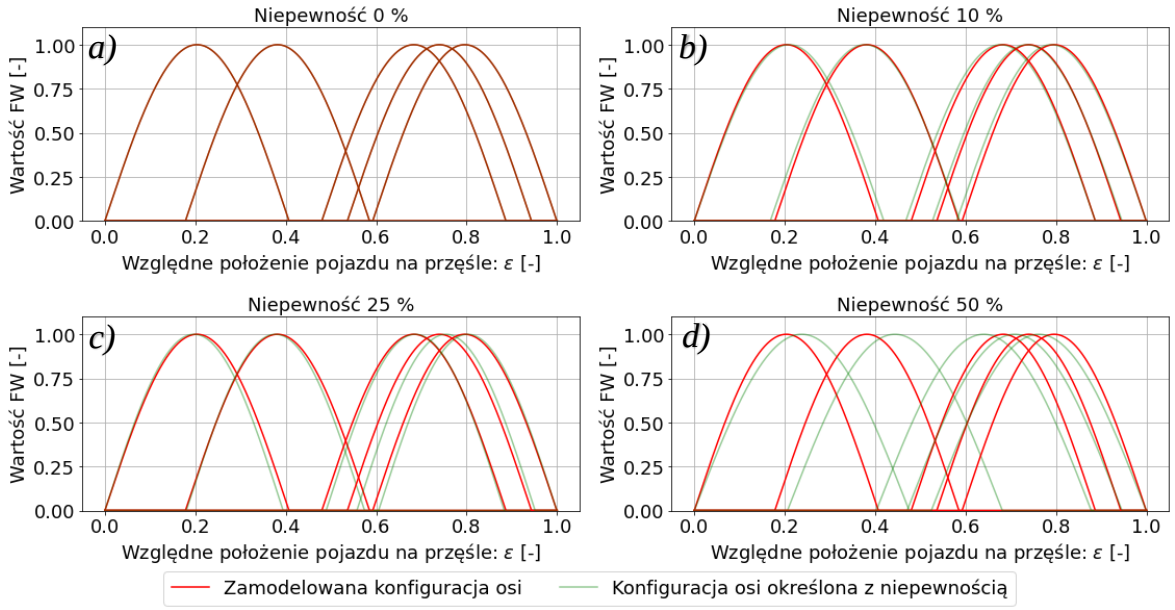
$$a_i = random.uniform(1 - A_{szum}, 1 + A_{szum}) \times a_i$$

gdzie:

- A_{szum} – to określony poziom niepewności dla odległości między osiami,
- a_i - to odległość między kolejnymi osiami pojazdu.

Analizę przeprowadzono dla różnych scenariuszy, z zakresem szumu pomiarowego od 0% (idealne warunki pomiarowe) do 50% niepewności, przy założeniu kroku iteracyjnego wynoszącego 2,5%. Dla każdego scenariusza, z każdym krokiem iteracji, zmieniano poziom niepewności w celu zbadania, jak różne poziomy błędu pomiarowego wpływają na wydajność systemu w zadaniu oszacowania nacisków osi i całkowitego nacisku pojazdu.

Na rys. 6.16 przedstawiono porównanie pomiędzy Funkcjami Wpływu dla osi przy prawidłowym oraz obarczonym niepewnością pomiarową określeniu konfiguracji osi pojazdu.

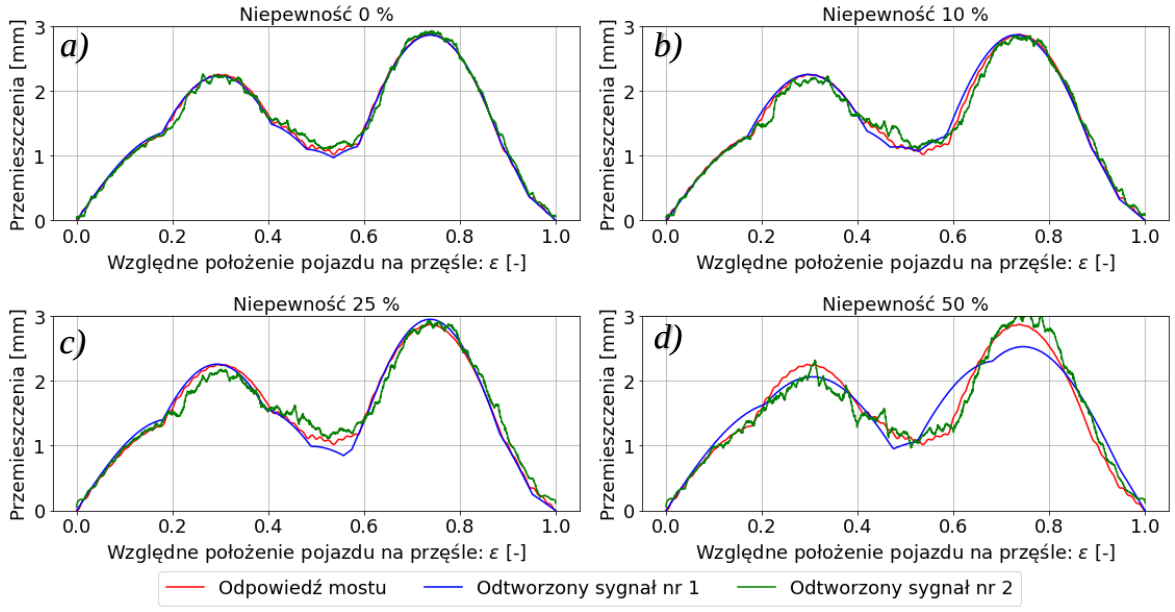


Rys. 6.16 Zamodelowane oraz obarczone błędem określenia konfiguracji osi funkcje wpływu, odpowiednio: a) 0%; b) 10%; c) 25%; d) 50%

Efektem tych niepewności jest fakt, że sieć neuronowa ma błędną wiedzę co do lokalizacji osi na obiekcie w czasie przejazdu.

6.8.2 Wybrane efekty działania sieci

Na rys. 6.17 ukazano wyniki działania systemu dla pojazdu pięcioosiowego, poruszającego się ze stałą prędkością po konstrukcji mostowej, przy różnych poziomach niedokładności określenia odległości między osiami, czyli przy błędach: 0%, 10%, 25% oraz 50%.



Rys. 6.17 Zasymulowane i odtworzone przez system odpowiedzi mostu przy różnych poziomach niepewności określenia konfiguracji pojazdu, odpowiednio: a) 0%; b) 10%; c) 25%; d) 50%

Analiza wykresów ujawnia, że wydajność sieci zaczyna obniżać się wraz ze wzrostem niepewności określenia konfiguracji osi. Szczególnie przy wyższych poziomach niepewności, pojawiają się przesunięcia w odtworzonym sygnale statycznym oraz rozbieżności między estymowanymi, a zasymulowanymi wartościami.

Te przesunięcia i rozbieżności wskazują na to, że precyzyjne określenie lokalizacji podłużnej osi w czasie jest kluczowe dla opracowanego systemu identyfikacji obciążeń.
W tab. 6.16 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.17 przedstawiono oszacowane przez system całkowity nacisk pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych poziomów niepewności określenia konfiguracji pojazdu. W tab. 6.18 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.16 Zamodelowane wartości nacisków losowo wybranego pojazdu referencyjnego

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.17 Wartości nacisków określonych przez sieć neuronową, z różnymi poziomami niepewności określenia konfiguracji pojazdu

Poziom niepewności [%]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	56.53	62.13	39.74	48.32	36.63	118.66	124.69	243.35
10	54.93	59.57	41.29	48.62	34.46	114.49	124.37	238.86
25	58.13	64.90	50.39	51.82	27.23	123.03	129.44	252.47
50	68.67	37.49	3.66	25.07	77.24	106.17	105.96	212.13

Tab. 6.18 Błąd określenia nacisków statycznych dla wybranego pojazdu

Poziom niepewności [%]	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	3.07	-3.32	-9.71	12.08	-4.28	-0.38	-0.56	-0.47
10	0.15	-7.31	-6.19	12.77	-9.96	-3.88	-0.82	-2.31
25	5.99	0.99	14.48	20.20	-28.84	3.29	3.22	3.26
50	25.22	-41.66	-91.69	-41.85	101.81	-10.87	-15.50	-13.24

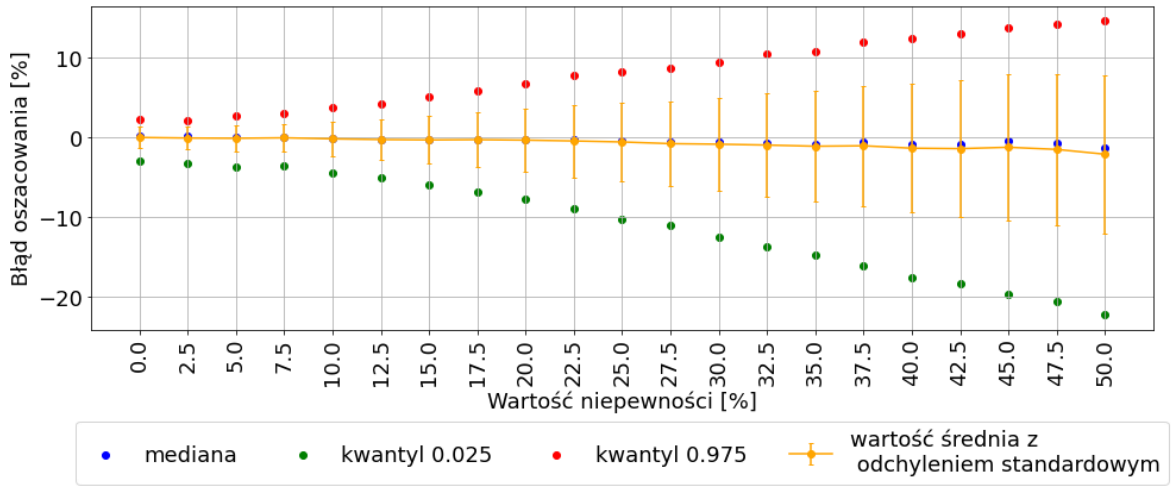
Odnotowane odchyłki między wartościami estymowanymi przez sieć a danymi z symulatora wydają się mieć charakter losowy. Warto zauważyć, że w niektórych przypadkach, jak przy nacisku trzeciej osi dla

niepewności pomiarowej 50%, doszło do znacznego niedoszacowania – oszacowanie systemu wynosiło 3.66 kN, przy modelowanym nacisku 44.02 kN. Z kolei nacisk ostatniej osi został przeszacowany przez system aż dwukrotnie.

Te różnice pokazują, że niepewności pomiarowe, szczególnie te związane z konfiguracją osi, mogą znacząco wpłynąć na dokładność sieci neuronowej w określeniu kluczowych parametrów pojazdu.

6.8.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.18 przedstawiono rozkład błędu przy oszacowaniu całkowitego nacisku pojazdu w ostatnim cyklu uczenia sieci neuronowej, biorąc pod uwagę różne poziomy niepewności w pomiarze konfiguracji pojazdu.



Rys. 6.18 Wykres błędów oszacowania masy całkowitej pojazdu dla ostatniej epoki uczenia sieci neuronowe w zależności od wartości niepewności określenia konfiguracji osi dla populacji pojazdów

Analiza rozkładu błędu przy oszacowaniu całkowitego nacisku pojazdu przez sieć neuronową, uwzględniająca różne poziomy niepewności pomiaru konfiguracji osi pojazdu, pokazuje istotny związek między dokładnością określenia odległości między osiami, a skutecznością sieci. Z wykresu wynika, że błąd oszacowania całkowitego nacisku pojazdu maleje w miarę wzrostu dokładności pomiarowej konfiguracji osi.

Tym samym, systemy określające te parametry muszą dostarczać dane z wysoką precyzją, aby sieć mogła efektywnie szacować całkowity nacisk pojazdu.

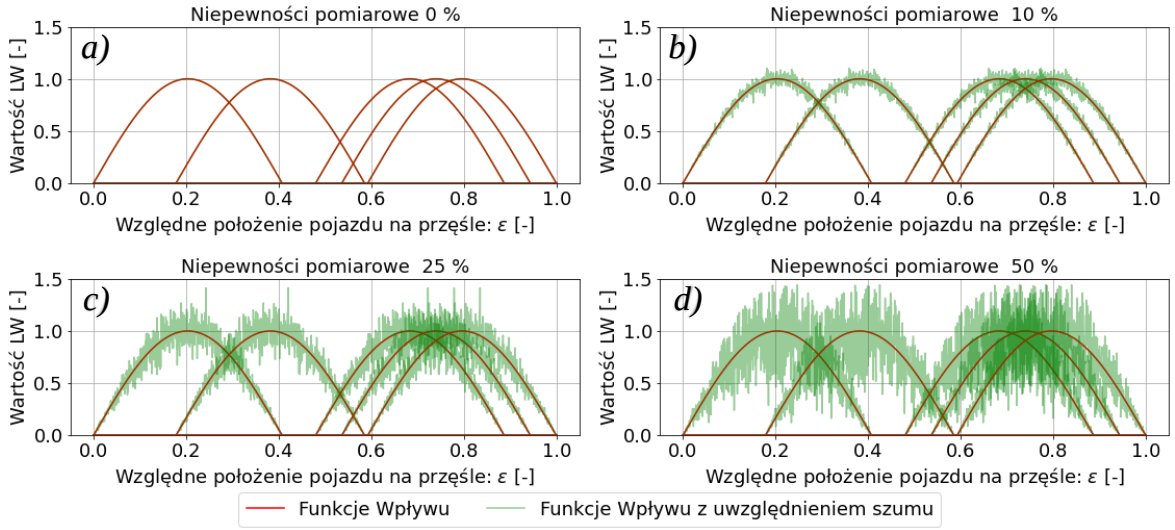
6.9 Analiza wpływu szumu pomiarowego przy określaniu funkcji wpływu

6.9.1 Informacje ogólne

W podrozdziale 6.9 zbadano, jak szum pomiarowy funkcji wpływu oddziałuje na działanie systemu przy określaniu nacisków osi oraz całkowity nacisk pojazdu. Funkcje wpływu, będące kluczowym elementem przy analizie oddziaływania pojazdów na konstrukcję mostową, zazwyczaj są wyznaczane na podstawie próbnych obciążeń obiektu mostowego. Aby zbadać wpływ tych niepewności na system, rozważono różne scenariusze z poziomami szumu pomiarowego od 0% (idealna dokładność) do 50%. Analiza została przeprowadzona iteracyjnie, zwiększając poziom szumu w krokach co 2.5%, co pozwoliło ocenić, jak różne poziomy niepewności wpływają na skuteczność działania systemu identyfikacji obciążeń. Założono, że niepewności te mają rozkład normalny, gdzie średnia wartość szumu wynosi 0, a odchylenie standardowe jest proporcjonalne do wartości funkcji wpływu co oznacza, że szum jest generowany z rozkładu normalnego $N(0,\sigma)$ jest równe połowie wartości funkcji wpływu pomnożonej przez współczynnik niepewności. Poziom szumu dostosowuje się do wartości funkcji wpływu – większe wartości funkcji wpływu powodują większe odchylenia szumu. Mechanizm narzucania szumu można zapisać pseudokodem:

$$szum = random.normal(0, \frac{|poziom\ szumu \times funkcja\ wpływu|}{2})$$

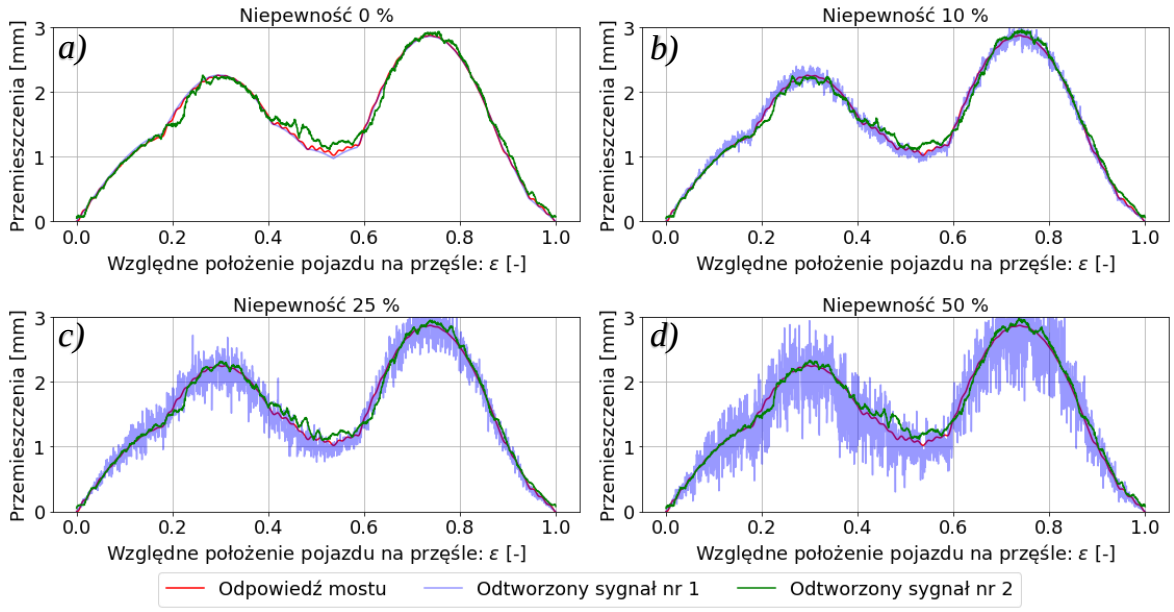
Dodawanie szumu miało na celu symulację niepewności pomiarowych, gdzie funkcja wpływu z szumem odpowiada sytuacji, w której 95% wartości funkcji mieści się w zakresie wyznaczonym przez poziom niepewności. Szum jest proporcjonalny do wartości funkcji, co odzwierciedla warunki pomiarowe, w których większe wartości są bardziej podatne na wpływ niepewności. Na rys. 6.19 przedstawiono funkcje wpływu z nałożonym szumem wynikającym z różnych poziomów niepewności pomiarowej czujnika określającego funkcję wpływu.



Rys. 6.19 Zasymulowane oraz obarczone niepewnością pomiarową funkcje wpływu, odpowiednio: a) 0%; b) 10 %; c) 25 %; d) 50%

6.9.2 Wybrane efekty działania sieci

Na rys. 6.20 ukazano wyniki działania sieci neuronowej dla pojazdu pięcioosiowego, poruszającego się ze stałą prędkością po konstrukcji mostowej, przy różnych poziomach dokładności w określeniu funkcji wpływu, czyli przy niepewnościach: 0%, 10%, 25% oraz 50%.



Rys. 6.20 Zasymulowane i odtworzone przez system odpowiedzi mostu przy różnych poziomach szumu nałożonej na funkcję wpływu, odpowiednio: a) 0%; b) 10%; c) 25%; d) 50%

Na analizowanych wykresach przedstawiono, jak sieć neuronowa radzi sobie z odtwarzaniem sygnałów dynamicznych i statycznych przy różnych poziomach niedoskonałości określenia funkcji wpływu. Sieć neuronowa wykazuje dość dobrą zdolność do odtwarzania sygnału dynamicznego, jednak napotyka trudności przy precyzyjnym określeniu sygnału statycznego. Wynika to z tego, że sygnał statyczny w dużej mierze opiera się na funkcji wpływu, która definiuje jak dana wielkość statyczna, np. nacisk, wpływa na analizowany przekrój obiektu mostowego w zależności od lokalizacji siły wymuszającej – w tym przypadku, pozycji pojazdu testowego na moście.

Błędy w określeniu funkcji wpływu, szczególnie te wynikające z dużych niepewności pomiarowych, mogą prowadzić do problemów z odtworzeniem dokładnego sygnału quasi-statycznego przez system. Niemniej jednak, zachowanie ogólnej formy i kształtu odtworzonych sygnałów sugeruje, że mimo zaszumienia funkcji wpływu błędami pomiarowymi, system może być zdolny do poprawnego odtworzenia niektórych kluczowych wartości, takich jak naciski wywierane przez poszczególne elementy pojazdu.

W tab. 6.19 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.20 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych poziomów niepewności szumu funkcji wpływu. W tab. 6.21 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.19 Zamodelowane wartości nacisków losowo wybranego pojazdu referencyjnego

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.20 Zasymulowane naciski statyczne osi losowo wybranego pojazdu oraz wartości nacisków określonych przez sieć neuronową, z różnymi poziomami szumu dla funkcji wpływu

Poziom szumu [%]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	56.68	63.22	39.12	50.02	35.17	119.90	124.31	244.21
10	57.23	62.99	39.97	48.91	35.30	120.22	124.17	244.39
25	56.43	59.82	38.78	50.26	34.43	116.25	123.47	239.72
50	54.45	58.87	37.67	49.87	34.64	113.32	122.18	235.50

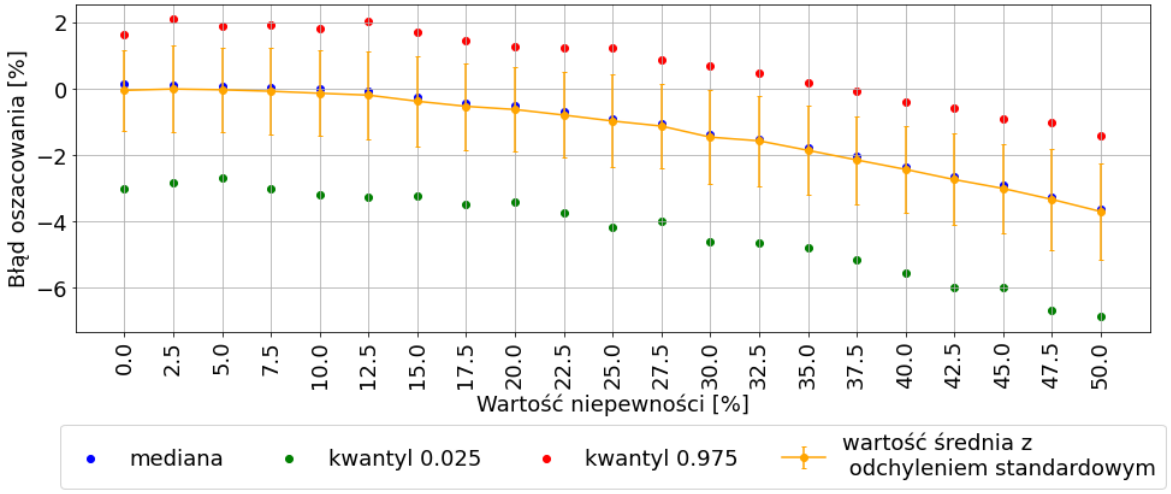
Tab. 6.21 Błąd określenia nacisków statycznych dla losowo wybranego pojazdu

Poziom szumu [%]	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
0	3.35	-1.63	-11.12	16.04	-8.12	0.67	-0.87	-0.12
10	4.35	-1.99	-9.20	13.45	-7.77	0.93	-0.98	-0.05
25	2.89	-6.92	-11.89	16.58	-10.03	-2.40	-1.54	-1.96
50	-0.73	-8.39	-14.42	15.68	-9.48	-4.86	-2.56	-3.68

Z analizy tabeli wynika, że w miarę wzrostu niepewności w określeniu funkcji wpływu, wartości estymowane przez sieć neuronową wykazują tendencję do niedoszacowania, czyli do określania nacisków niższych niż zasymulowane.

6.9.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.21 zaprezentowano rozkład błędu przy estymacji całkowitego nacisku pojazdu przez sieć neuronową w ostatnim cyklu uczenia dla populacji testowej, uwzględniając różne poziomy dokładności w określeniu funkcji wpływu.



Rys. 6.21 Rozkład błędu dla ostatniego cyklu uczenia w zależności od przyjętego poziomu szumu przy określaniu funkcji wpływu

Z wykresu można wnioskować, że im większa jest niepewność określenia funkcji wpływu, tym wyniki stają się bardziej niepewne. Co interesujące, mimo że ogólny rozkład błędu pozostaje na zbliżonym poziomie, obserwuje się tendencję do przesuwania się średniej i mediany rozkładu w stronę ujemnych wartości błędu, co sugeruje, że system ma tendencję do niedoszacowania całkowitego nacisku pojazdu. Z tej analizy wynika kluczowe znaczenie funkcji wpływu dla dokładności estymacji całkowitego nacisku pojazdu przez sieć neuronową. W związku z tym, gdy funkcja wpływu jest określana metodą doświadczalną, na przykład poprzez próbne obciążenia, istotne jest, aby uzyskana funkcja była jak najbardziej precyzyjna i pozbawiona zakłóceń pomiarowych.

6.10 Analiza wpływu błędu określenia funkcji wpływu

6.10.1 Informacje ogólne

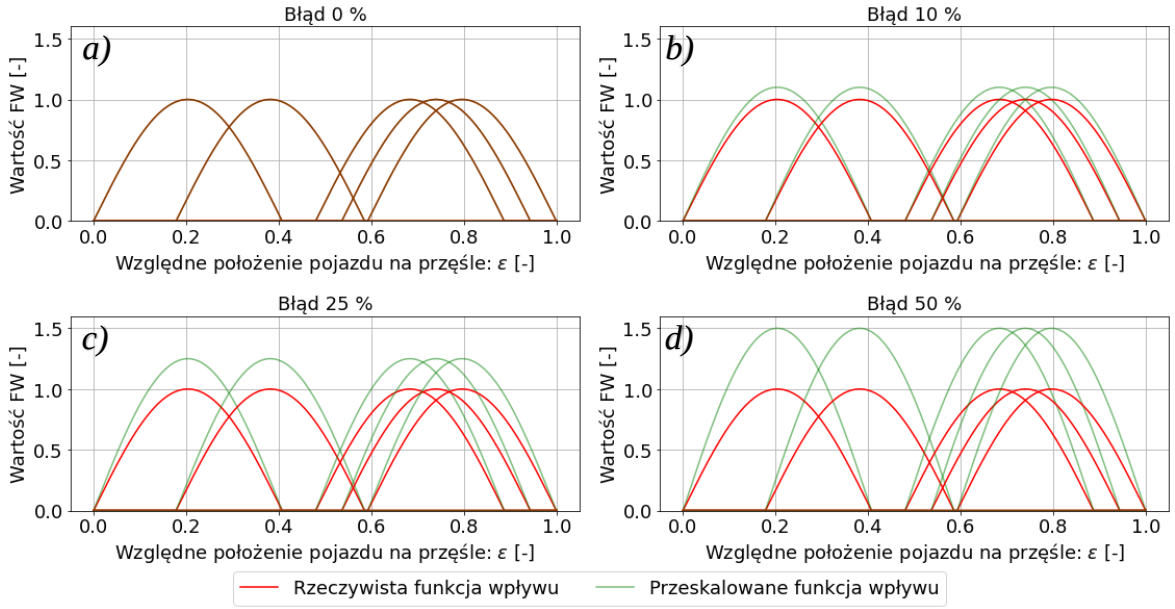
W podrozdziale 6.10 zbadano wpływ przeskalowania funkcji wpływu na efektywność działania systemu oraz na dokładność estymacji całkowitego nacisku pojazdu. Założono możliwość błędnego określenia funkcji wpływu dla danego obiektu mostowego i zbadano konsekwencje jej modyfikacji. Każdą wartość funkcji wpływu skalowano w zakresie od 50% do 150% jej rzeczywistej wartości, tj. analizowano przypadki, w których funkcja wpływu była mnożona przez współczynniki 0.5 oraz 1.5 względem wartości odniesienia.

Analizę przeprowadzono w sposób iteracyjny, sprawdzając wartości funkcji wpływu w podanym zakresie, z zastosowaniem kroku iteracyjnego wynoszącego 5%. Przeskalowanie funkcji wpływu zostało wprowadzone poprzez przemnożenie jej rzędnych przez współczynnik skali, który odpowiadał procentowemu przeszacowaniu lub niedoszacowaniu funkcji wpływu.

Skalowanie funkcji wpływu zostało zaimplementowane w pseudokodzie w następujący sposób:

$$Przeskalowana\ funkcja\ wpływu = Funkcja\ wpływu\ * skala$$

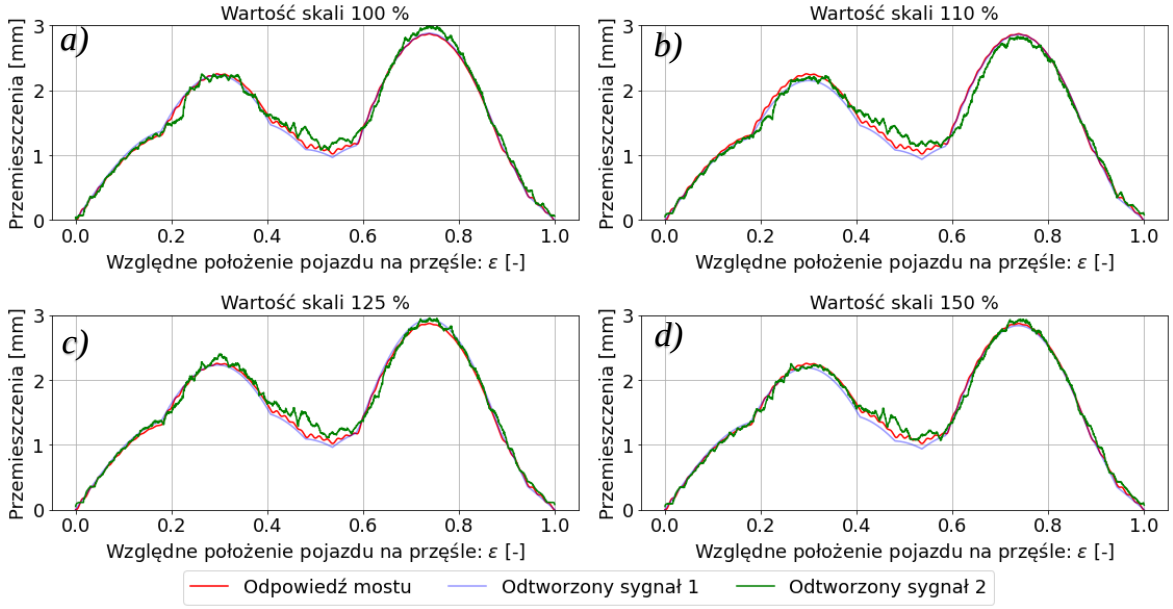
Każda iteracja zwiększała lub zmniejszała skalę funkcji wpływu w celu analizy, jak takie błędy wpływają na dokładność estymacji całkowitego nacisku pojazdu i nacisków osi. Na rys. 6.22 zaprezentowano funkcje wpływu dla poszczególnych kół pięcioosiowego pojazdu, przyjmując założenie o przeszacowaniu tych funkcji o 10%, 25% i 50%. Ta wizualizacja pokazuje, jak zmienia się postać funkcji wpływu w zależności od stopnia przeszacowania, co ma kluczowe znaczenie dla zrozumienia, jak błędne określenie funkcji wpływu może wpływać na analizę nacisków wywieranych przez pojazd na konstrukcję mostową.



Rys. 6.22 Poziom odniesienia funkcji wpływu oraz wybrane wartości przeskalowania funkcji wpływu, odpowiednio: a) 100%; b) 110 %; c) 125 %; d) 150 %

6.10.2 Wybrane efekty działania sieci

Na rys. 6.23 przedstawiono efekty działania sieci neuronowej dla wybranego pojazdu pięcioosiowego przy różnych założeniach dotyczących funkcji wpływu: przy jej bazowej wartości (100%, bez błędu), a także przy przeszacowaniu o 10% (110%), 25% (125%) oraz 50% (150%).



Rys. 6.23 Zasymulowane i odtworzone przez system odpowiedzi mostu dla różnych wartości skali funkcji wpływu, odpowiednio: a) 100%; b) 110%; c) 125 %; d) 150 %

Z analizy wynika, że sieć neuronowa skutecznie odtworzyła sygnał statyczny i dynamiczny, wiernie odwzorowując rzeczywistą odpowiedź obiektu mostowego, bez widocznych błędów wstępnych czy nieścisłości.

W tab. 6.22 przedstawiono zamodelowane wartości nacisków pojazdu, natomiast w tab. 6.23 przedstawiono oszacowane przez system całkowite naciski pojazdu, naciski na poszczególne osie oraz grupy osi z uwzględnieniem różnych poziomów wartości skali funkcji wpływu. W tab. 6.24 przedstawiono błędy określenia nacisków w stosunku do zamodelowanego pojazdu.

Tab. 6.22 Zamodelowane wartości nacisków losowo wybranego pojazdu referencyjnego

Zamodelowane wartości nacisków pojazdu referencyjnego [kN]							
N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
54.84	64.27	44.02	43.11	38.27	119.11	125.40	244.50

Tab. 6.23 Zasymulowane naciski statyczne osi losowo wybranego pojazdu oraz wartości nacisków określonych przez system, z różnymi wartościami skali funkcji wpływu

Wartość skali [%]	Naciski statyczne [kN]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
100 %	57.26	61.47	40.07	48.73	36.62	118.73	125.43	244.16
110 %	48.23	55.50	34.18	47.71	31.03	103.72	112.92	216.65
125 %	45.54	49.29	31.70	42.28	27.99	94.83	101.98	196.80
150 %	37.56	39.90	25.81	33.78	22.67	77.46	82.26	159.72

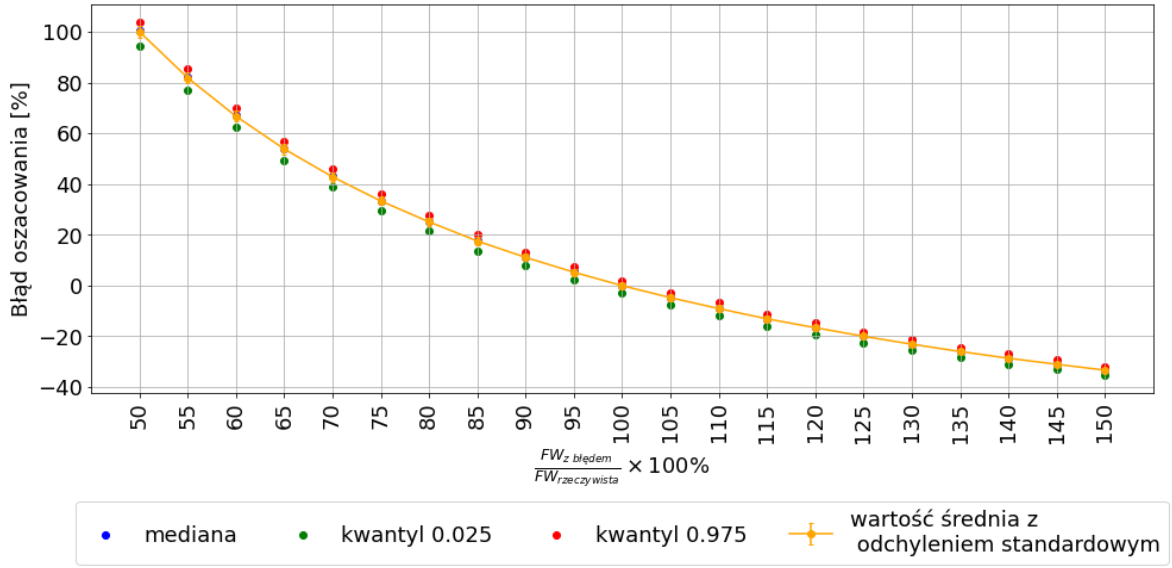
Tab. 6.24 Błąd określenia nacisków statycznych dla losowo wybranego pojazdu

Wartość skali [%]	Błąd określenia nacisków statycznych [%]							
	N_1	N_2	N_3	N_4	N_5	$\sum_{i=1}^{2} N_i$	$\sum_{i=3}^{5} N_i$	$\sum_{i=1}^{5} N_i$
100	4.41	-4.35	-8.96	13.05	-4.31	-0.32	0.03	-0.14
110	-12.07	-13.64	-22.34	10.68	-18.92	-12.92	-9.95	-11.39
125	-16.96	-23.30	-27.98	-1.91	-26.86	-20.38	-18.68	-19.51
150	-31.52	-37.91	-41.36	-21.63	-40.77	-34.97	-34.40	-34.68

W miarę wzrostu przeszacowania funkcji wpływu, określone naciski wykazują tendencję do liniowego spadku. Funkcja wpływu stanowi jedyną metrykę dostarczającą informacje o obiekcie niezbędną do szacowania nacisków wywieranych przez pojazd. Nie można oczekiwać, że sieć neuronowa, nie dysponując wiedzą o fizycznych parametrach rzeczywistego obiektu mostowego, będzie w stanie określić precyzyjnie całkowity nacisk pojazdu i naciski na grupy osi i osie bez właściwie określonej funkcji wpływu. Stąd przeszacowanie funkcji wpływu względem jej rzeczywistej wartości nieuchronnie prowadzi do pojawienia się błędów w estymacji.

6.10.3 Dokładność określenia całkowitego nacisku pojazdu

Na rys. 6.24 zaprezentowano rozkład błędu estymacji całkowitego nacisku pojazdu w ostatnim cyklu uczenia sieci neuronowej na populacji testowej. Analiza ta koncentruje się na wpływie różnych wartości funkcji wpływu, wyrażonej jako procent rzeczywistej wartości funkcji wpływu, na dokładność sieci.



Rys. 6.24 Rozkład błędu oszacowania całkowitego nacisku pojazdu dla ostatniego cyklu uczenia w zależności od stosunku zadanej funkcji wpływu do rzeczywistej funkcji wpływu

Kiedy funkcja wpływu zostaje zredukowana do 50% swojej rzeczywistej wartości, obserwujemy, że błąd w określeniu całkowitego nacisku pojazdu osiąga 100%. Jest to rezultat spodziewany, ponieważ przy zaniżonej wartości funkcji wpływu, wymagana siła musi być dwukrotnie większa, aby wywołać identyczną reakcję konstrukcji, co przy rzeczywistej wartości funkcji wpływu. Wynika to z bezpośredniej zależności między siłą działającą na konstrukcję, a jej odpowiedzią. Dalsza analiza błędów określenia całkowitego nacisku pojazdu wskazuje na istnienie zależności odwrotnej, wyrażonej jako funkcja $(\frac{100}{x})$, gdzie x reprezentuje procentową wartość wprowadzonej funkcji wpływu.

Najważniejszym wnioskiem jest to, że dokładne określenie funkcji wpływu ma kluczowe znaczenie dla precyzji estymacji całkowitego nacisku pojazdów przejeżdżających przez obiekt mostowy.

6.11 Podsumowanie

W rozdziale 6 przeprowadzono analizę wpływu poszczególnych czynników na dokładność działania systemu AutoSIN_SD. Na podstawie przeprowadzonych badań oraz analiz, wyciągnięto szereg wniosków dotyczących wpływu kluczowych czynników na działanie systemu w procesie identyfikacji obciążeń konstrukcji mostowych. Wszelkie sprawdzenia wykonywano dla licznej populacji testowej (10 tysięcy) która nie brała udziału w procesie uczenia się systemu.

Poniżej przedstawiono najważniejsze obserwacje i rekomendacje wynikające z tych badań:

- Na podstawie analizy wpływu rozpiętości teoretycznej obiektu na dokładność działania systemu (patrz podrozdział 6.2) określono, że największy błąd oszacowania nacisku całkowitego pojazdu występuje dla krótkich obiektów mostowych (poniżej 8 m). W zakresie do 12 m rozpiętości teoretycznej, zauważono wyraźną zależność, że im większa rozpiętość przęsła tym dłuższa interakcja pojazd-most, co skutkuje większą liczbą informacji dla systemu. Dla konstrukcji o rozpiętości 1 m, wartość błędu dla 95 % oszacowań mieści się w przedziale od - 12.5 do 14 %, gdzie dla wartości 12 m i powyżej jest już to przedział ok. -3 do 3 %. Wyraźnie widać również, że przy rozpiętości przekraczającej 12 m nie odnotowuje się wzrostu dokładności działania systemu.
- Na podstawie analizy wpływu masy własnej obiektu (patrz podrozdział 6.3), zauważono, że duża masa własna obiektu mostowego wpływa negatywnie na dokładność działania systemu, poprzez zwiększenie bezwładności układu. Konstrukcje mostowe o dużej sztywności w stosunku do masy wykazują większą efektywność w identyfikacji obciążeń użytkowych. Dziewięćdziesiąt pięć procent wartości błędu oszacowania całkowitych nacisków pojazdów dla populacji testowych zawierało się w przedziale -2.5 do 2.5 % dla masy własnej poniżej $3.11\,\frac{tony}{mb}$. Dla masy własnej na poziomie $20\,\frac{ton}{mb}$ dziewięćdziesiąt pięć procent wartości błędu oszacowania całkowitego nacisku znajdowało się w przedziale od -5 do 17.5 %.
- W podrozdziale 6.4 sprawdzono wpływ nierówności nawierzchni na dokładność działania systemu. Znaczne nierówności nawierzchni mają niekorzystny wpływ na dokładność działania systemu. Ich wpływ w przypadku nawierzchni o bardzo dobrej i dobrej jakości jest jednak stosunkowo niewielki, a 95% wartości błędu oszacowania całkowitego nacisku mieści się w przedziale od -2% do 4%.
- W podrozdziale 6.5 przeanalizowano wpływ liczby danych uczących, a najważniejszy wniosek wynikający z analiz brzmi: im więcej danych uczących tym lepsza dokładność działania opracowanego systemu wykorzystującego sieci neuronowe. W przypadku systemu opartego na uproszczonym modelu konstrukcji mostowej, w którym most został przedstawiony jako belka swobodnie podparta, wystarczyło 30 tys. przejazdów, aby sieć uzyskała stabilne rozwiązanie. Dla każdego systemu liczba wymaganych danych będzie jednak indywidualna i zależna od złożoności analizowanego modelu. Drugi istotny wniosek to fakt, że powyżej około 30 tys.

danych nowe próbki nie wpływają na dalsze zwiększenie skuteczności działania sieci — system osiąga wówczas swoją optymalną dokładność.

- W podrozdziale 6.6 analizowano wpływ szumu pomiarowego i wykazano, że system jest mało wrażliwy na efekty typowego szumu pomiarowego sygnału odpowiedzi konstrukcji wynikającego z czynników zewnętrznych. Jest to wynikiem naturalnej zdolności autodekoderów do filtrowania i odszumiania danych. W analizach sprawdzono różne poziomy szumu, które znacząco przekraczają poziom zakłóceń w stosowanych systemach pomiarowych. Rozkład błędów oszacowania całkowitego nacisku na osie pojazdu jest stabilny dla różnych poziomów niedoskonałości sygnału, a 95 % wartości błędu oszacowania mieści się w przedziale -3.5 % do 3 %.
- W podrozdziale 6.7 przeanalizowano wpływ niepewności określenia prędkości na dokładność systemu. System oparty na sieciach neuronowych wykazuje wysoką stabilność w określaniu całkowitego nacisku pojazdu, nawet w warunkach zwiększonej niepewności pomiarowej prędkości. Fakt, że system jest odporny na potencjalne błędy w pomiarze prędkości, wskazuje na skuteczność ekstrakcji kluczowych cech z danych, co ma znaczenie dla dokładności i niezawodności jego działania.
- W podrozdziale 6.8 wykazano, że proponowany system identyfikacji obciążeń jest wrażliwy na informacje o konfiguracji osi pojazdu. Wraz ze wzrostem niepewności oszacowania odległości między osiami pojazdu, zwiększa się błąd oszacowania całkowitego nacisku pojazdu. Dla idealnie oszacowanej konfiguracji, 95% wartości błędu oszacowania mieści się w przedziale -3 do 3 %. Wraz ze wzrostem poziomu błędu określenia konfiguracji, rosną również błędy oszacowania całkowitego nacisku pojazdu. Ważne jest, aby system identyfikacji pojazdów, którego zadaniem jest określenie liczby osi oraz lokalizacji osi na moście w czasie przejazdu, cechował się dużą dokładnością. W tym celu można zastosować na przykład kamery wizyjne o wysokiej rozdzielczości lub nowoczesne radary.
- Ważnym czynnikiem jest również poprawna kalibracja funkcji wpływu dla systemu. Analizowano zarówno błąd przeskalowania funkcji wpływu (zaniżenie lub zawyżenie wartości), jak i błąd wynikający z szumu oraz niedoskonałości pomiarowych. Przy niewielkich poziomach szumu (do 10%) rozkład błędów oszacowania całkowitego nacisku pojazdu pozostaje stabilny, a 95% błędów mieści się w przedziale od -3% do 3%. Powyżej tego poziomu obserwuje się spadek dokładności oszacowania. Na podstawie wyników przeprowadzonych analiz można przyjąć, że w rzeczywistych warunkach błąd pomiarowy czujników określających funkcję wpływu nie przekroczy 10%.
- Analizy wskazują również, że o ile system jest odporny na niewielki szum w funkcji wpływu, o tyle jej przeskalowanie ma kluczowe znaczenie. Funkcja wpływu jest jedynym elementem wiążącym rzeczywistą konstrukcję z logiką tworzoną przez sieć neuronową systemu, dlatego powinna być określona z jak największą precyzją.

7 Podsumowanie, wnioski i kierunki dalszych badań

7.1 Podsumowanie i wnioski

W rozprawie stworzono system AutoSIN_SD pozwalający na identyfikację obciążeń eksploatacyjnych mostów poprzez określanie całkowitego nacisku pojazdu na podstawie dynamicznej odpowiedzi konstrukcji mostowej z wykorzystaniem uczenia maszynowego, realizując w ten sposób cel główny. W trakcie przeprowadzonych prac naukowo-badawczych przygotowano także wariantowe opracowania w postaci systemów AutoSIO_S (rozdział 5.2), AutoSIO_SD (rozdział 5.3), AutoSIO_D (rozdział 5.4), które wykorzystano w analizach porównawczych.

W pracy zrealizowano również cele dodatkowe poprzez stworzenie i przedstawienie architektury systemu identyfikacji obciążeń eksploatacyjnych oraz opracowanie metodyki implementacji i weryfikacji. Następnie zaimplementowano omawianą koncepcję systemu, opracowano metodę weryfikacji poprawności jego działania oraz przeprowadzono analizy stabilności i niezawodności.

Przeprowadzone analizy w trakcie pracy nad auto-adaptacyjnym systemem identyfikacji obciążeń obiektów mostowych pojazdami z wykorzystaniem uczenia maszynowego pozwalają na sformułowanie następujących wniosków:

1. Zaproponowana w podrozdziale 3.2 koncepcja budowy i działania systemu, a także przedstawiona w podrozdziale 3.3 metoda implementacji oraz walidacji, okazały się skuteczne i umożliwiły przeprowadzenie analiz błędu oszacowania całkowitych nacisków pojazdów oraz wpływu parametrów konstrukcji i przejazdów na ten błąd.
2. Stworzony system jest auto-adaptacyjny, czyli poprawia swoją dokładność w trakcie działania, co eliminuje konieczność ingerencji człowieka w proces uczenia systemu. Uzyskano to za pomocą zastosowania techniki uczenia nienadzorowanego sieci neuronowej. Zdolność do poprawy działania została przedstawiona w podpunktach m.in. 5.2.2, 5.2.3, 5.3.2, 5.3.3, gdzie zaprezentowano oszacowania całkowitego nacisku pojazdu oraz błąd oszacowania całkowitych nacisków pojazdów populacji testowej na różnych etapach działania systemu.
3. System identyfikacji obciążeń poprawnie określa całkowite naciski pojazdu. Błędy oszacowania całkowitego nacisku pojazdów zarówno dla wariantów systemu przedstawionych w podrozdziale 5.2 oraz 5.3 mieszczą się akceptowanych granicach. W podpunktach 5.2.3, 5.3.3 sprawdzono zmiany rozkładu błędu oszacowania całkowitego nacisku w trakcie uczenia i wykazano, że sieć po odpowiednim czasie uczenia się, osiąga stabilizację, a 95 % błędów oszacowania mieści się w zakresie -3 do 3% od wartości oczekiwanej.
4. W systemie sprawdzono również naciski na poszczególne osie pojazdów. System ma trudności z dokładnym oszacowaniem nacisków na poszczególne osie, zwłaszcza przy niewielkich odległościach między nimi. System ma tendencję do niedoszacowania i-tej osi grupy osi oraz

przeszacowania j-tej osi grupy. Zostało to pokazane m. in. w podpunkcie 5.2.7, gdzie dla korelacji parametrów pojazdu 4-osiowego dla błędu oszacowania nacisku 1 i 2. osi od rozstawu pomiędzy osiami widać odpowiednio korelację 0.7 i -0.7.

5. System wykorzystujący modele uczenia maszynowego charakteryzuje się stosunkowo niską złożonością obliczeniową, co umożliwia jego trenowanie i obsługę na komputerach o przeciętnej specyfikacji, zwiększając możliwości implementacji w warunkach rzeczywistych. Nie jest potrzebne budowanie super-komputera, aby na nim budować i trenować sieci neuronowe, chociaż wskazane jest, aby wyposażony był w kartę graficzną. Uczenie sieci neuronowych opiera się na operacjach na tensorach i tablicach wielowymiarowych, a karty graficzne, które przeliczają wartości pikseli ekranów komputerowych (tablice punktów np. 3840x2160 pikseli – 60 klatek na sekundę) są dedykowane do tego typu operacji.
6. Architektura systemu jest interpretowalna, co oznacza, że wszystkie poszczególne zadania sieci są znane i opisane. O ile sieć neuronowa jest „czarną skrzynką” i nie mamy bezpośrednio kontroli nad uczeniem się, o tyle tworząc odpowiednią strukturę całego systemu, parametry wejściowe oraz wejściowe, funkcje optymalizacyjne oraz funkcje aktywacji możemy zapanować na jego działaniem. Przykład stanowią warianty systemu przedstawione w podrozdziałach 5.2 oraz 5.3, które poprawnie szacują naciski.
7. Przeprowadzone analizy wykazały, że system wykazuje dużą stabilność i odporność na szum pomiarowy, dzięki zastosowaniu autodekoderów odszumiających dane i wyodrębniających kluczowe cechy. Dokładne wyniki zostały przedstawione w podrozdziale 6.6. Błędy oszacowania dla szerokiego spektrum poziomu szumu, były na stałym poziomie.
8. System identyfikacji obciążeń jest podatny na błędy określenia parametrów pojazdu takich jak np. konfiguracja osi. System wymaga precyzyjnego systemu identyfikacji pojazdów, który określi sylwetkę pojazdu, jego lokalizację oraz prędkość – można to uzyskać przez zastosowanie np. systemów wizyjnych.
9. Przy implementacji systemu, należy z jak największą dokładnością określić funkcję wpływu dla mierzonej odpowiedzi konstrukcji mostowej. System wykorzystujący sieci neuronowe, nawet najbardziej złożony, nie jest w stanie samodzielnie jednoznacznie określić całkowitego nacisku pojazdu bez uwzględnienia fizycznych parametrów konstrukcji. Zostało to wyraźnie pokazane zarówno w podrozdziale 5.4, gdzie przedstawiono wariant systemu, który bazuje tylko na odpowiedzi konstrukcji mostowej, bez znajomości funkcji wpływu oraz w podrozdziale 6.10, gdzie celowo przeskalowywano funkcje wpływu.
10. Funkcję wpływu można uzyskać na podstawie teoretycznych rozważań bądź też obciążeń próbnych. Uzyskana metodą doświadczalną funkcja wpływu najdokładniej opisuje zachowanie konstrukcji, ponieważ uzyskiwana jest rzeczywista odpowiedź obiektu na siły wymuszające od pojazdu kalibracyjnego.

11. Opracowane rozwiązanie wykazuje niewielką wrażliwość na prędkość przejazdu pojazdu. Jest to szczególnie przydatna cecha, która będzie umożliwiać monitorowanie obciążeń na obiektach mostowych w ciągu dróg, bez wymuszania ograniczenia prędkości. Zostało to przedstawione w podrozdziałach 5.2.5 oraz 5.3.5, gdzie dla różnych zakresów prędkości, rozkład błędu oszacowania mieścił się w przedziale -4% do 4%.
12. W trakcie analiz zauważono także, że im większa jest masa pojazdu, tym dokładniejsze otrzymujemy oszacowania całkowitego nacisku pojazdu. Wynika to z faktu, że cięższe pojazdy generują wyraźniejszą odpowiedź konstrukcji mostowej. Jest to przydatna cecha, przy rozwiązaniu problemu monitorowania przeciążonych pojazdów.
13. Sprawdzono również, że nawierzchnia drogowa na obiekcie mostowym powinna charakteryzować się dobrym poziomem jakości. Nierówności drogowe wprowadzają dodatkowe przemieszczenia pionowe pojazdów co sprawia, że odpowiedź konstrukcji zawiera więcej szumu pochodzenia dynamicznego. Sieć poprawnie wyznacza naciski dla nierówności nawierzchni do klasy C, ale dla klasy D i niższych dokładność działania systemu spada.

W pracy zawarto też w załącznikach implementacje stworzonych na potrzeby systemu programów komputerowych. Podzielono je na 3 tematyczne załączniki:

- Załącznik A: Przedstawia implementacje 2 autodekoderów (A.1 oraz A.2). Implementacje powstały na potrzeby podrozdziału 3.1, w którym oprócz ogólnego przedstawienia uczenia maszynowego, przedstawiono również mechanizm działania sieci na podstawie obrazów uzyskanych z napisanych programów.
- Załącznik B: Przedstawia Symulator dynamicznej odpowiedzi konstrukcji, zgodnie z opisem z podrozdziału 3.3 Metodyka implementacji systemu identyfikacji obciążeń. Załącznik zawiera trzy podprogramy:
    - Generator populacji pojazdów,
    - Wirtualny model mostu,
    - Program zarządzający.
- Załącznik C: Przedstawia system AutoSIO, który jest modelem sieci neuronowej określającej całkowite naciski pojazdu oraz program przygotowujący dane wejściowe do systemu.

7.2 Proponowane kierunki dalszych badań

Przeprowadzone rozpoznanie literatury i własne analizy poczynione na potrzeby niniejszej rozprawy doktorskiej pozwoliły zidentyfikować następujące kierunki dalszych badań:

1. W pracy przyjęto uproszczony model mostu jako belki swobodnie podpartej, po której porusza się pojazd zamodelowany jako zbiór sił i punktów masowych. W dalszych badaniach warto rozważyć model wielodźwigarowy z wieloma czujnikami mierzącymi dynamiczną odpowiedź konstrukcji na obciążenie pojazdem, co pozwoli uwzględnić zarówno lokalizację podłużną, jak i poprzeczną pojazdu.
2. W analizie rozpatrywano stałe i niezmienne parametry konstrukcji mostowej w czasie. W przyszłości warto zbadać wpływ uszkodzeń i degradacji konstrukcji na skuteczność systemu, w tym zastosowanie systemu do identyfikacji takich zmian na podstawie dynamicznej odpowiedzi konstrukcji na obciążenia pojazdami.
3. W pracy założono, że testowanie proponowanego pomysłu odbędzie się w warunkach izolowanych i kontrolowanych poprzez stworzenie symulatora odpowiedzi konstrukcji mostowej. W dalszych badaniach warto przeprowadzić pilotażowe wdrożenie na rzeczywistym obiekcie mostowym, aby potwierdzić skuteczność systemu.
4. W analizie rozpatrywano pojedynczy pojazd poruszający się po obiekcie mostowym usytuowany w ciągu jednej jezdni autostrady lub drogi ekspresowej. W przyszłości można uwzględnić ruch w obu kierunkach i zbadać skuteczność systemu w takich warunkach.
5. System zaprojektowano do identyfikacji obciążeń eksploatacyjnych. Warto sprawdzić, czy podobne rozwiązanie może być użyteczne w prognozowaniu nośności zmęczeniowej mostów w ramach monitoringu długoterminowego.

PIŚMIENNICTWO

[1] „Obciążenie dróg przez pojazdy ciężkie i ich wpływ na trwałość zmęczeniową konstrukcji nawierzchni podatnych i półsztywnych". Politechnika Gdańska Wydział Inżynierii Lądowej i Środowiska, Gdańsk, s. 201, 2015. [Online]. Dostępne na: https://pbc.gda.pl/Content/50418/phd_ry%C5%9B_dawid.pdf

[2] D. Ryś, J. Judycki, i P. Jaskuła, „Wpływ pojazdów przeciążonych na trwałość nawierzchni asfaltowych", *Logistyka*, t. 6, s. 9318–9328, 2014.

[3] T. Kula, „Pojazdy przeciążone w ruchu drogowych - skala problemu". Nawierzchnie Drogowe 2015, 26 listopad 2015.

[4] J. Staniszewski, „Ruch pojazdów przeciążonych na obszarach zurbanizowanych". Najwyższa Izba Kontroli, 2020.

[5] J. Zieliński, P. Tutka, P. Kunikowski, i A. Szyszło, *Synteza wyników GPR 2020/21 na zamiejskiej sieci dróg krajowych*. Generalna Dyrekcja Dróg Krajowych i Autostrad, 2021.

[6] ASTM Committee E17 on Vehicle - Pavement Systems, *ASTM E1318-09(2017) Standard Specification for Highway Weigh-In-Motion (WIM) Systems with User Requirements and Test Methods*. ASTM International, 2017.

[7] O. K. Normann i R. C. Hopkins, „Weighing Vehicles In Motion", *Highway Research Board*, t. 50, nr 221, 1952.

[8] P. Burnos, „Ważenie pojazdów samochodowych w ruchu. Część 1: Oddziaływanie pojazdów przeciążonych na nawierzchnię", *Drogownictwo*, nr 6/2014, 2014.

[9] P. Burnos, „Ważenie pojazdów samochodowych w ruchu. Część 2: Rodzaje i charakterystyka systemów Weigh In Motion (WIM)", *Drogownictwo*, nr 6/2014, 2014.

[10] P. Burnos, „Ważenie pojazdów samochodowych w ruchu. Część 3: Czujniki nacisku stosowane w systemach Weigh In Motion (WIM)", *Drogownictwo*, nr 6/2014, 2014.

[11] P. Burnos, „Ważenie pojazdów samochodowych w ruchu. Część 4: Ocena dokładności systemów Weigh In Motion (WIM)", *Drogownictwo*, nr 6/2014, 2014.

[12] Rada Unii Europejskiej, „Dyrektywa Rady nr 96/53/WE z dnia 25 lipca 1996 roku w sprawie określenia maksymalnych wymiarów poszczególnych pojazdów kołowych w ruchu krajowym i ponad granicznym na obszarze Wspólnoty oraz określenia maksymalnych ciężarów w ruchu ponad granicznym".

[13] Minister Infrastruktury, „Rozporządzenie Ministra Infrastruktury z dnia 31 grudnia 2002 r. w sprawie warunków technicznych pojazdów oraz zakresu ich niezbędnego wyposażenia." 2002.

[14] F. Moses, „Weigh-In-Motion System Using Instrumented Bridges", *Transportation Engineering Journal of ASCE*, t. 105, nr 3, s. 233–249, sty. 1979, doi: 10.1061/TPEJAN.0000783.

[15] R. Peters, „AXWAY - a system to obtain vehicle axle weights", zaprezentowano na 12th ARRB Conference, Hobart, Hobart, TAS, Australia: Vermont South, VIC, Australia: ARRB Group Limited, sie. 1984.

[16] R. Peters, „CULWAY—an unmanned and undetectable highway speed vehicle weighing system.", *Australian road research board proceedings*, t. 13, s. 70–83, 1986.

[17] B. Jacob, E. J. O. Brien, i S. Jehaes, „Weigh-in-motion of road vehicles. Final report of the COST 323 Action (WIM-LOAD)". Laboratoire Central des Ponts et Chausse´es, Paris, 2002. [Online]. Dostępne na: https://www.is-wim.org/doc/wim_eu_specs_cost323.pdf

[18] E. O'Brien i Aleš. Žnidarič, Red., *Weighing-in-motion of axles and vehicles for Europe (WAVE). Report of work package 1.2: bridge WIM systems (B-WIM)*. Ljubljana: Zavod za gradbeništvo Slovenije, 2001.

[19] A. J. Cardini i J. T. DeWolf, „Implementation of a Long-Term Bridge Weigh-In-Motion System for a Steel Girder Bridge in the Interstate Highway System", *J. Bridge Eng.*, t. 14, nr 6, s. 418–423, lis. 2009, doi: 10.1061/(ASCE)1084-0702(2009)14:6(418).

[20] R. Trafford, R. Linden, J. Donovan, E. Neville, W. Marroquin, i S. Tan, „Retrofitting Rural Infrastructure for Smart Parking and Traffic Monitoring", zaprezentowano na IEEE Sensors Applications Symposium 2017, mar. 2017.

[21] M. Quilligan, „Bridge weigh-in motion : development of a 2-D multi-vehicle algorithm", *Trita-BKN. Bulletin*, nr 69, s. 144, 2003.

[22] A. T. Dempsey, B. Jacob, i J. Carracilli, „Orthotropic bridge weigh-in-motion for determining axle and gross vehicle weights", zaprezentowano na SECOND EUROPEAN CONFERENCE ON WEIGH-IN-MOTION OF ROAD VEHICLES, LISBON, 1998, s. 435–44.

[23] E. O'Brien i Aleš. Žnidarič, Red., *Weighing-in-motion of axles and vehicles for Europe (WAVE). Report of work package 1.2: bridge WIM systems (B-WIM)*. Ljubljana: Zavod za gradbeništvo Slovenije, 2001.

[24] M. Lydon, S. E. Taylor, D. Robinson, A. Mufti, i E. J. O. Brien, „Recent developments in bridge weigh in motion (B-WIM)", *J Civil Struct Health Monit*, t. 6, nr 1, s. 69–81, luty 2016, doi: 10.1007/s13349-015-0119-6.

[25] T. Ojio i K. Yamada, „Bridge WIM by reaction force method", *Proceedings of the 4th international WIM conference*, luty 2005.

[26] J. Kalin i A. Žnidari, „Practical implementation of Nothing-On-the-Road Bridge Weigh-In-Motion system", 2019, [Online]. Dostępne na: https://hvttforum.org/wp-content/uploads/2019/11/Practical-Implementation-of-Nothing-on-the-Road-Bridge-Weigh-In-Motion-System-Kalin.pdf

[27] P. Chatterjee, E. J. O'Brien, Y. Li, i A. González, „Wavelet domain analysis for identification of vehicle axles from bridge measurements", *Computers & Structures*, t. 84, nr 28, s. 1792–1801, lis. 2006, doi: 10.1016/j.compstruc.2006.04.013.

[28] S.-S. Law, S. S. Law, S. Q. Wu, W. Shaoqing, i Z. Y. Shi, „Moving Load and Prestress Identification Using Wavelet-Based Method", *Journal of Applied Mechanics*, t. 75, nr 2, s. 021014, mar. 2008, doi: 10.1115/1.2793134.

[29] Y. Yu, C. S. Cai, C.S. Cai, C. S. Cai, i L. Deng, „Vehicle axle identification using wavelet analysis of bridge global responses", *Journal of Vibration and Control*, t. 23, nr 17, s. 1077546315623147, paź. 2017, doi: 10.1177/1077546315623147.

[30] J. T. Białasiewicz, *Falki i aproksymacje*, Wyd. 2. Warszawa: Wydawnictwa Naukowo-Techniczne, 2004.

[31] W. Ding, S. Li, G. Zhang, X. Lei, i H. Qian, „Vehicle Pose and Shape Estimation Through Multiple Monocular Vision", w *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Kuala Lumpur, Malaysia: IEEE, grudz. 2018, s. 709–715. doi: 10.1109/ROBIO.2018.8665155.

[32] P. Viola i M. Jones, „Rapid object detection using a boosted cascade of simple features", w *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, grudz. 2001, s. I–I. doi: 10.1109/CVPR.2001.990517.

[33] N. Dalal i B. Triggs, „Histograms of oriented gradients for human detection", w *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, cze. 2005, s. 886–893 t. 1. doi: 10.1109/CVPR.2005.177.

[34] R. B. Girshick, J. Donahue, T. Darrell, i J. Malik, „Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation", *2014 IEEE Conference on Computer Vision and Pattern Recognition*, s. 580–587, 2013.

[35] J. Redmon, S. Divvala, R. Girshick, i A. Farhadi, *You Only Look Once: Unified, Real-Time Object Detection*. 2016, s. 788. doi: 10.1109/CVPR.2016.91.

[36] W. Liu *i in.*, „SSD: Single Shot MultiBox Detector", zaprezentowano na Computer Vision – ECCV 2016, w Lecture Notes in Computer Science, vol. 9905. Springer, paź. 2016. doi: 10.1007/978-3-319-46448-0_2.

[37] J. Dai, Y. Li, K. He, i J. Sun, „R-FCN: Object Detection via Region-based Fully Convolutional Networks", w *Proceedings of the 30th International Conference on Neural Information Processing Systems*, w NIPS'16. Barcelona, Spain: Curran Associates Inc., 2023, s. 9. [Online]. Dostępne na: https://arxiv.org/abs/1605.06409

[38] L. Deng i Y. Zhou, „The Vehicle collision warning system detects the vehicle ahead", *J. Phys.: Conf. Ser.*, t. 1303, nr 1, s. 012081, sie. 2019, doi: 10.1088/1742-6596/1303/1/012081.

[39] Navneet Dalal, „Object Detection using Histograms of Oriented Gradients", zaprezentowano na Pascal VOC 2006 Workshop, Graz, Austria, 7 maj 2006. [Online]. Dostępne na: https://web.archive.org/web/20131205153059/http://pascallin.ecs.soton.ac.uk/challenges/VOC/voc2006/slides/dalal.pdf

[40] M. Hussain, „YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection”, *Machines*, t. 11, nr 7, 2023, doi: 10.3390/machines11070677.
[41] A. Znidaric, I. Lavroc, i J. Kalin, „Using strips to mitigate the multiple-presence problem of BWIM systems.”, *Proceedings of the 6th international WIM conference*, cze. 1998.
[42] H. Zhao, N. Uddin, E. J. O'Brien, X. Shao, i P. Zhu, „Identification of Vehicular Axle Weights with a Bridge Weigh-in-Motion System Considering Transverse Distribution of Wheel Loads”, *J. Bridge Eng.*, t. 19, nr 3, s. 04013008, mar. 2014, doi: 10.1061/(ASCE)BE.1943-5592.0000533.
[43] A. Tikhonov i Arsenin VY, *Solutions of ill-posed problems*. w Scripta series in mathematics. Winston, 1977. [Online]. Dostępne na: https://books.google.pl/books?id=ECrvAAAAMAAJ
[44] T. Ojio i K. Yamada, „Bridge weigh-in-motion systems using stringers of plate girder bridges.”, *Proceedings of the 3rd international WIM conference*, maj 2002.
[45] C. O'Connor i T. H. T. Chan, „Dynamic wheel loads from bridge strains”, *Journal of Structural Engineering-asce*, t. 114, nr 8, s. 1703–1723, wrz. 1988, doi: 10.1061/(asce)0733-9445(1988)114:8(1703).
[46] T. H. T. Chan, S.-S. Law, S.S. Law, T. H. Yung, i X. R. Yuan, „An interpretive method for moving force identification”, *Journal of Sound and Vibration*, t. 219, nr 3, s. 503–524, sty. 1999, doi: 10.1006/jsvi.1998.1904.
[47] S.-S. Law, S.S. Law, T. H. T. Chan, i Q. H. Zeng, „Moving force identification: A time domain method”, *Journal of Sound and Vibration*, t. 201, s. 1–22, 1997.
[48] S. S. Law, T. H. T. Chan, i Q. H. Zeng, „Moving Force Identification—A Frequency and Time Domains Analysis”, *Journal of Dynamic Systems, Measurement, and Control*, t. 121, nr 3, s. 394–401, wrz. 1999, doi: 10.1115/1.2802487.
[49] X. Zhu, S.-S. Law, i S.S. Law, „Moving forced identification on multi-span continous bridge”, *Journal of Sound and Vibration*, t. 228, nr 2, s. 377–396, lis. 1999, doi: 10.1006/jsvi.1999.2416.
[50] T. H. T. Chan i T. H. Yung, „A theoretical study of force identification using prestressed concrete bridges”, *Engineering Structures*, s. 9, 2000.
[51] T. H. T. Chan, S.-S. Law, i T. H. Yung, „Moving force identification using an existing prestressed concrete bridge”, *Engineering Structures*, t. 22, nr 10, s. 1261–1270, paź. 2000, doi: 10.1016/s0141-0296(99)00084-x.
[52] X. Zhu, S.S. Law, i S.-S. Law, „Identification of vehicle axle loads from bridge dynamic responses”, *Journal of Sound and Vibration*, t. 236, nr 4, s. 705–724, wrz. 2000, doi: 10.1006/jsvi.2000.3021.
[53] P. Asnachinda, T. Pinkaew, i J. A. Laman, „Multiple vehicle axle load identification from continuous bridge bending moment response”, *Engineering Structures*, t. 30, nr 10, s. 2800–2817, paź. 2008, doi: 10.1016/j.engstruct.2008.02.018.
[54] J. Dowling, E. J. O'Brien, i A. González, „Adaptation of Cross Entropy optimisation to a dynamic Bridge WIM calibration problem”, *Engineering Structures*, t. 44, nr 44, s. 13–22, lis. 2012, doi: 10.1016/j.engstruct.2012.05.047.
[55] S. Law, J. Bu, i X. Q. Zhu, „Vehicle axle loads identification using finite element method”, *Engineering Structures*, t. 26, nr 8, s. 1143–1153, 2004, doi: 10.1016/j.engstruct.2004.03.017.
[56] L. Deng, C. S. Cai, C.S. Cai, i C. S. Cai, „Identification of Dynamic Vehicular Axle Loads: Theory and Simulations”:, *Journal of Vibration and Control*, t. 16, nr 14, s. 2167–2194, maj 2010, doi: 10.1177/1077546309351221.
[57] S.-S. Law, S.S. Law, i Y. L. Fang, „Moving fotce identification: Optimal state estimation approach”, *Journal of Sound and Vibration*, t. 239, nr 2, s. 233–254, sty. 2001, doi: 10.1006/jsvi.2000.3118.
[58] S.-S. Law i X. Zhu, „Study on Different Beam Models in Moving Force Identification”, *Journal of Sound and Vibration*, t. 234, nr 4, s. 661–679, lip. 2000, doi: 10.1006/jsvi.2000.2867.
[59] X. Zhu, S.-S. Law, i S.S. Law, „Moving loads identification through regularization”, *Journal of Engineering Mechanics-asce*, t. 128, nr 9, s. 989–1000, wrz. 2002, doi: 10.1061/(asce)0733-9399(2002)128:9(989).
[60] S. Kim, J. Lee, M.-S. Park, i B.-W. Jo, „Vehicle Signal Analysis Using Artificial Neural Networks for a Bridge Weigh-in-Motion System”, *Sensors*, t. 9, nr 10, s. 7943–7956, paź. 2009, doi: 10.3390/s91007943.

[61] Y. Wu, L. Deng, i W. He, „BwimNet: A Novel Method for Identifying Moving Vehicles Utilizing a Modified Encoder-Decoder Architecture”, *Sensors*, t. 20, nr 24, s. 7170, grudz. 2020, doi: 10.3390/s20247170.
[62] T. Kawakatsu, K. Aihara, A. Takasu, J. Adachi, H. Wang, i T. Nagayama, „Fully-Neural Approach to Vehicle Weighing and Strain Prediction on Bridges Using Wireless Accelerometers”, w *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada: IEEE, cze. 2021, s. 8027–8031. doi: 10.1109/ICASSP39728.2021.9414433.
[63] Mieszko. Kużawa, *Monitoring sensoryczny obiektów mostowych w trakcie ich eksploatacji*. Wrocław: Oficyna Wydawnicza Politechniki Wrocławskiej, 2022. [Online]. Dostępne na: http://www.nukat.edu.pl/nukat/icov/GD015/xx005577754.jpg
[64] J. S. Wilson, *Sensor Technology Handbook*. w Electronics & Electrical, no. t. 1. Elsevier, 2005. [Online]. Dostępne na: https://books.google.pl/books?id=fdeToUK8edMC
[65] F. T. K. Au, F. Au, R. J. Jiang, i Y. K. Cheung, „Parameter identification of vehicles moving on continuous bridges”, *Journal of Sound and Vibration*, t. 269, nr 1, s. 91–111, sty. 2004, doi: 10.1016/s0022-460x(03)00005-1.
[66] D. Cantero, R. Karoumi, i A. González, „The Virtual Axle concept for detection of localised damage using Bridge Weigh-in-Motion data”, *Engineering Structures*, t. 89, s. 26–36, kwi. 2015, doi: 10.1016/j.engstruct.2015.02.001.
[67] Y. Yu, C. Cai, i L. Deng, „State-of-the-art review on bridge weigh-in-motion technology”, *Advances in Structural Engineering*, t. 19, nr 9, s. 1514–1530, cze. 2016, doi: 10.1177/1369433216655922.
[68] P. Burnos, J. Gajda, i R. Sroka, „Accuracy criteria for evaluation of Weigh-in-Motion systems”, *Metrology and Measurement Systems*, s. 743–743, paź. 2018, doi: 10.24425/mms.2018.124881.
[69] P. Burnos, J. Gajda, R. Sroka, M. Wasilewska, i C. Dolega, „High Accuracy Weigh-In-Motion Systems for Direct Enforcement”, *Sensors*, t. 21, nr 23, 2021, doi: 10.3390/s21238046.
[70] R. Pallas-Areny i J. Webster, *Sensors and signal conditioning, Second Edition*. 2001.
[71] C.-Y. Li, C. Wang, Q.-X. Yang, i T.-Y. Qi, „Identification of Vehicle Loads on an Orthotropic Deck Steel Box Beam Bridge Based on Optimal Combined Strain Influence Lines”, *Applied Sciences*, t. 12, nr 19, s. 9848, wrz. 2022, doi: 10.3390/app12199848.
[72] D. Dunne, E. J. O’Brien, B. Basu, i A. Gonzalez, „Bridge WIM systems with Nothing On the Road (NOR)”, w *Proceedings of the 4th international WIM conference*, Zurich: International Society for Weigh-in-Motion, sty. 2005, s. 109–117.
[73] S. C. K. Shek, J. Butterfield, A. Murphy, i I. Spence, „Current State of the Art in Object Detection for Autonomous Systems”, zaprezentowano na IMC37 - 37th International Manufacturing Conference, Athlone, 2021. [Online]. Dostępne na: https://www.manufacturingcouncil.ie/imc-conference-archive
[74] S. R. Lorenzen *i in.*, „Virtual Axle Detector Based on Analysis of Bridge Acceleration Measurements by Fully Convolutional Network”, *Sensors*, t. 22, nr 22, s. 8963, lis. 2022, doi: 10.3390/s22228963.
[75] Y. Zhu, H. Sekiya, T. Okatani, I. Yoshida, i S. Hirano, „Real-time vehicle identification using two-step LSTM method for acceleration-based bridge weigh-in-motion system”, *J Civil Struct Health Monit*, t. 12, nr 3, s. 689–703, cze. 2022, doi: 10.1007/s13349-022-00576-2.
[76] C. Rowley, A. Gonzalez, E. Obrien, i A. Žnidarič, „Comparison of conventional and regularized bridge weigh-in-motion algorithms”, w *International Conference on Heavy Vehicles HVParis 2008*, B. Jacob i E. O’Brien, Red., Hoboken, NJ, USA: John Wiley & Sons, Inc, 2013, s. 271–282. doi: 10.1002/9781118623305.ch21.
[77] L. Deng i C. S. Cai, „Identification of Dynamic Vehicular Axle Loads: Demonstration by a Field Study”, *Journal of Vibration and Control*, t. 17, nr 2, s. 183–195, luty 2011, doi: 10.1177/1077546309351222.
[78] A. L. Samuel, „Some Studies in Machine Learning Using the Game of Checkers”, *IBM Journal of Research and Development*, t. 3, nr 3, s. 210–229, 1959, doi: 10.1147/rd.33.0210.
[79] T. M. Mitchell, *Machine learning*, t. 1. McGraw-hill New York, 1997.
[80] Aurélien Géron, *Uczenie maszynowe z użyciem Scikit-Learn i TensorFlow*, 3. wyd. Helion, 2020.
[81] S. Guido i A. Müller, *Machine learning, Python i data science*. w Oreilly. Helion, 2023.

[82] G. Bonaccorso, *Algorytmy uczenia maszynowego. Zaawansowane techniki implementacji*. Helion, 2019.
[83] Chrystal R. China, „Five machine learning types to know", IBM Think. [Online]. Dostępne na: https://www.ibm.com/think/topics/machine-learning-types
[84] W. S. McCulloch i W. Pitts, „A logical calculus of the ideas immanent in nervous activity", *The bulletin of mathematical biophysics*, t. 5, nr 4, s. 115–133, grudz. 1943, doi: 10.1007/BF02478259.
[85] J. McCarthy, „Professor Sir James Lighthill, FRS. Artificial Intelligence: A General Survey.", *Artif. Intell.*, t. 5, nr 3, s. 317–322, 1974.
[86] D. E. Rumelhart, J. L. McClelland, i P. R. Group, *Parallel Distributed Processing, Volume 1: Explorations in the Microstructure of Cognition: Foundations*. The MIT Press, 1986. doi: 10.7551/mitpress/5236.001.0001.
[87] J. L. McClelland, D. E. Rumelhart, i P. R. Group, *Parallel Distributed Processing, Volume 2: Explorations in the Microstructure of Cognition: Psychological and Biological Models*. The MIT Press, 1986. doi: 10.7551/mitpress/5237.001.0001.
[88] J. Dean, „A Golden Decade of Deep Learning: Computing Systems & Applications", *Daedalus*, t. 151, nr 2, s. 58–74, maj 2022, doi: 10.1162/daed_a_01900.
[89] Y. Jiang, J. Luo, D. Huang, Y. Liu, i D. Li, „Machine Learning Advances in Microbiology: A Review of Methods and Applications", *Frontiers in Microbiology*, t. 13, 2022, doi: 10.3389/fmicb.2022.925454.
[90] G. Swirszcz, W. M. Czarnecki, i R. Pascanu, „Local minima in training of neural networks", 2017, [Online]. Dostępne na: https://arxiv.org/pdf/1611.06310
[91] I. of Medicine i N. A. of Sciences, *Discovering the Brain*. Washington, DC: The National Academies Press, 1992. doi: 10.17226/1785.
[92] F. Rosenblatt, „Perceptron Simulation Experiments", *Proceedings of the IRE*, t. 48, nr 3, s. 301–309, mar. 1960, doi: 10.1109/JRPROC.1960.287598.
[93] F. Rosenblatt, *The perceptron: a theory of statistical separability in cognitive systems (Project Para)*. Cornell Aeronautical Laboratory, 1958.
[94] D. O. Hebb, *The organization of behavior*. New York: Wiley, 1949.
[95] M. Minsky i S. Papert, *Perceptrons.* w Perceptrons. Oxford, England: M.I.T. Press, 1969.
[96] David E. Rumelhart i James L. McClelland, „Learning Internal Representations by Error Propagation", w *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, MIT Press, 1987, s. 318–362. [Online]. Dostępne na: http://ieeexplore.ieee.org/document/6302929
[97] R. Yamashita, M. Nishio, R. K. G. Do, i K. Togashi, „Convolutional neural networks: an overview and application in radiology", *Insights into Imaging*, t. 9, nr 4, s. 611–629, sie. 2018, doi: 10.1007/s13244-018-0639-9.
[98] A. Makone, *The Landscape of Deep Learning Based Object Detection Techniques in Computer Vision*. 2020. doi: 10.13140/RG.2.2.12679.42401.
[99] Y. Lecun, L. Bottou, Y. Bengio, i P. Haffner, „Gradient-Based Learning Applied to Document Recognition", *Proceedings of the IEEE*, t. 86, s. 2278–2324, grudz. 1998, doi: 10.1109/5.726791.
[100] S. Chen i W. Guo, „Auto-Encoders in Deep Learning—A Review with New Perspectives", *Mathematics*, t. 11, nr 8, 2023, doi: 10.3390/math11081777.
[101] X. Guo, X. Liu, E. Zhu, i J. Yin, „Deep Clustering with Convolutional Autoencoders", w *Lecture Notes in Computer Science*, Cham: Springer, paź. 2017. doi: 10.1007/978-3-319-70096-0_39.
[102] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, i P.-A. Manzagol, „Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion", *Journal of Machine Learning Research*, t. 11, nr 110, s. 3371–3408, 2010.
[103] P. Vincent, H. Larochelle, Y. Bengio, i P.-A. Manzagol, „Extracting and composing robust features with denoising autoencoders", w *Proceedings of the 25th International Conference on Machine Learning*, sty. 2008, s. 1096–1103. doi: 10.1145/1390156.1390294.
[104] A. Reyes-Muñoz i J. Guerrero-Ibáñez, „Vulnerable Road Users and Connected Autonomous Vehicles Interaction: A Survey", *Sensors*, t. 22, nr 12, 2022, doi: 10.3390/s22124614.

[105] J. Laconte, A. Kasmi, R. Aufrère, M. Vaidis, i R. Chapuis, „A Survey of Localization Methods for Autonomous Vehicles in Highway Scenarios", *Sensors*, t. 22, nr 1, 2022, doi: 10.3390/s22010247.
[106] G. Van Rossum i F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.
[107] J. Kmita, J. Bień, C. Machelski, i Wydawnictwa Komunikacji i Łączności, *Komputerowe wspomaganie projektowania mostów*. w Inżynieria Komunikacyjna. Warszawa: Wydawnictwa Komunikacji i Łączności, 1989.
[108] A. Madaj i W. Wołowicki, *Podstawy projektowania budowli mostowych*. Warszawa: Wydawnictwa Komunikacji i Łączności, 2003.
[109] „DYNA Manual, Service Pack 2022-3 Build 143", 2022.
[110] B. Lewicki, „PN-EN 1990:2004 Eurokod - Podstawy projektowania konstrukcji", 2004. [Online]. Dostępne na: https://api.semanticscholar.org/CorpusID:178864305
[111] Europejski Komitet Normalizacyjny, „Eurokod 1: Oddziaływania na konstrukcje Część 2: Obciążenia ruchome mostów". PKN, 2007.
[112] „PN-EN 1992-2:2010 – Eurokod 2: Projektowanie konstrukcji z betonu – Część 2: Mosty betonowe; EN 1992-2". Polski Komitet Normalizacyjny (PKN), 2010.
[113] „PN-EN 1993-2:2007 – Eurokod 3: Projektowanie konstrukcji stalowych – Część 2: Mosty stalowe". Polski Komitet Normalizacyjny (PKN), 2007.
[114] Polski Komitet Normalizacji, Miar i Jakości, „PN-85 S-10030 Obiekty Mostowe Obciążenia". PKN.
[115] „PN-82/S-10052: Obiekty mostowe. Konstrukcje stalowe. Projektowanie". Polski Komitet Normalizacji, Miar i Jakości, 1982.
[116] Polski Komitet Normalizacji, Miar i Jakości, „PN 91 S-10042 Obiekty Mostowe, Konstrukcje betonowe, żelbetowe i sprężone, Projektowanie". PKN.
[117] C. Westerkamp-Freitag i J.-D. Wörner, „Baudynamik", w *Beton-Kalender 2015*, John Wiley & Sons, Ltd, 2014, s. 461–493. doi: https://doi.org/10.1002/9783433603406.ch8.
[118] „ISO 8608: Mechanical Vibration–Road Surface Profiles–Reporting of Measured Data". International Standardization Organization, 1995.
[119] K. Bogsjö, K. Podgórski, i I. Rychlik, „Models for road surface roughness", *Vehicle System Dynamics*, t. 50, nr 5, s. 725–747, maj 2012, doi: 10.1080/00423114.2011.637566.
[120] K. M. A. Kamash i J. D. Robson, „Implications of isotropy in random surfaces", *Journal of Sound and Vibration*, t. 54, nr 1, s. 131–145, 1977, doi: https://doi.org/10.1016/0022-460X(77)90412-6.
[121] J. Gardulski, „Metody badań amortyzatorów samochodów osobowych", *Diagnostyka*, t. nr 3(51), s. 93–100, 2009.
[122] Robert Wardęga, „Wpływ struktury ruchu na nośność nawierzchni drogowych", *Praca doktorska, Wydawnictwo Wydziału Budownictwa Lądowego i Wodnego Politechniki Wrocławskiej*, 2006.
[123] K. Pearson i O. M. F. E. Henrici, „VII. Mathematical contributions to the theory of evolution.—III. Regression, heredity, and panmixia", *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, t. 187, s. 253–318, 1896, doi: 10.1098/rsta.1896.0007.
[124] C. Spearman, „The Proof and Measurement of Association between Two Things", *The American Journal of Psychology*, t. 15, nr 1, s. 72–101, 1904, doi: 10.2307/1412159.
[125] M. G. Kendall, „A New Measure of Rank Correlation", *Biometrika*, t. 30, nr 1/2, s. 81–93, 1938, doi: 10.2307/2332226.
[126] Y. LeCun i C. Cortes, „The mnist database of handwritten digits", 2005. [Online]. Dostępne na: https://api.semanticscholar.org/CorpusID:60282629

ŹRÓDŁA INTERNETOWE

[W 1.1] - https://www.gov.pl/web/gddkia/stacje-ciaglych-pomiarow-ruchu

[W 4.1] – https://cdn-7.motorsport.com/images/mgl/2eAa58K2/s1200/thierry-neuville-martijn-wydae-1.webp

[W 4.2] – https://www.zasadaauto.pl/files/filemanager/aneta/SKP_kontrola-zawieszenia-900x600.jpg

[W 4.3] – https://toc.mercedes-benz-trucks.com/pl_pl/modelseries

[W 4.4] – https://a.allegroimg.com/s1024/0cb054/cc0c9c2b44f68c140e7c98a7d4fd

[W 4.5] –https://archiwum.allegro.pl/oferta/mercedes-benz-arocs-2648-6x4-meiller-kipper-i12632070633.html

[W 4.6] – https://autoportal.hr/wp-content/uploads/2020/07/Arocs-4151-K-1.jpg

[W 4.7] – https://ireland.apollo.olxcdn.com/v1/files/3elyd6ujg0dm2-PL/image;s=1000x700

STRESZCZENIE

Praca dotyczy opracowania systemu identyfikacji obciążeń eksploatacyjnych mostów, umożliwiającego określenie masy pojazdu na podstawie dynamicznej odpowiedzi konstrukcji mostowej z wykorzystaniem metod uczenia maszynowego. Motywacją do pracy nad systemem jest dynamiczny wzrost ruchu oraz rosnąca liczba pojazdów przeciążonych na drogach.

Celem pracy jest opracowanie koncepcji auto-adaptacyjnego kompleksowego systemu identyfikacji obciążeń mostowych oraz metodyki jego implementacji, weryfikacji i oceny stabilności.

W pracy omówiono motywację, cele badawcze oraz przedstawiono przegląd treści pracy. Przeprowadzono analizę literatury obejmującą systemy identyfikacji obciążeń eksploatacyjnych, metody detekcji pojazdów oraz techniki pomiarowe, w tym sensoryczny monitoring mostów. Dokonano klasyfikacji istniejących rozwiązań identyfikacji obciążeń oraz dokonano ich oceny pod kątem użyteczności.

W odpowiedzi na istniejące wyzwania przedstawiono technologie, takie jak sieci neuronowe i autodekodery, a następnie przedstawiono koncepcję architektury systemu identyfikacji obciążeń opartego na uczeniu maszynowym. Oprócz ogólnej koncepcji systemu identyfikacji obciążeń przedstawiono także proponowaną metodę implementacji i walidacji systemu w oparciu o symulator odpowiedzi obiektu mostowego.

W pracy przedstawiono opracowany symulator dynamicznej odpowiedzi obiektu mostowego na obciążenia pojazdami, dokonano jego walidacji oraz przeanalizowano wpływ różnych parametrów na wyniki.

Przedstawiono warianty auto-adaptacyjnych systemów identyfikacji obciążeń oraz oceniono ich działanie przy użyciu symulatora, wybierając najlepszy wariant.

Ostatecznie na wybranym systemie przetestowano stabilność systemu, uwzględniając wpływ takich parametrów, jak rozpiętość obiektu mostowego, szum pomiarowy oraz niepewność parametrów pojazdu. W podsumowaniu przedstawiono główne wnioski z badań oraz kierunki dalszych badań.

Efektem pracy jest koncepcja systemu umożliwiającego monitorowanie i aktualizację obciążeń mostów, zarządzanie ruchem, dostosowanie wymagań infrastrukturalnych oraz monitorowanie procesów zmęczeniowych, co przyczynia się do zwiększenia trwałości i bezpieczeństwa mostów.

ABSTRACT

The work concerns the development of a system for the identification of operational loads of bridges, enabling the determination of vehicle weight based on the dynamic response of the bridge structure using machine learning methods. The motivation to work on the system is the dynamic increase in traffic and the growing number of vehicles overloaded on the roads.

The aim of the study is to develop the concept of an auto-adaptive comprehensive system for the identification of bridge loads and the methodology of its implementation, verification, and stability assessment.

The paper discusses motivation, research objectives and presents an overview of the content of the work. A literature analysis was conducted, including systems for identifying operating loads, vehicle detection methods and measurement techniques, including sensory monitoring of bridges. Existing load identification solutions were classified and evaluated in terms of usability.

In response to the existing challenges, technologies such as neural networks and autodecoders were presented, followed by the concept of a machine learning-based workload identification system architecture. In addition to the general concept of the load identification system, the proposed method of implementation and validation of the system based on the response simulator of the bridge object was also presented.

The paper presents the developed simulator of the dynamic response of a bridge structure to vehicle loads, its validation and analysis of the impact of various parameters on the results.

Variants of auto-adaptive load identification systems are presented, and their operation is evaluated using a simulator, choosing the best variant.

Finally, the stability of the system was evaluated on the selected system, considering the impact of parameters such as the span of the bridge structure, measurement noise and the uncertainty of the vehicle's parameters. The summary presents the main conclusions of the research and the directions of further research.

The result of the work is the concept of a system that enables monitoring and updating of bridge loads, traffic management, adjustment of infrastructure requirements and monitoring of fatigue processes, which contributes to increasing the durability and safety of bridges.

Załącznik A: Autodekodery – implementacja

A.1 Autodekoder stosowy

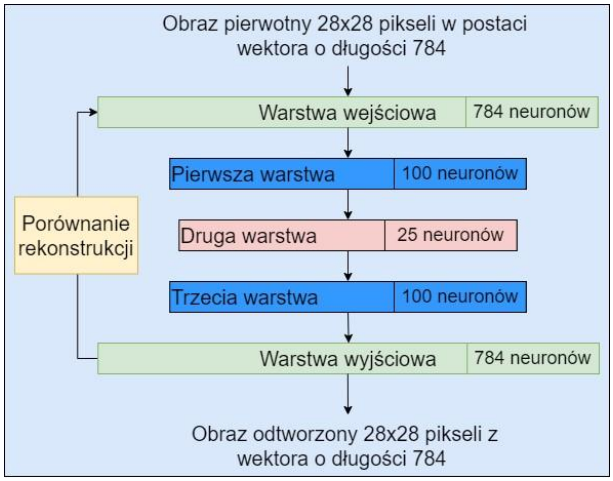
A.1.1 Opis zadania

W załączniku przedstawiono przykład implementacji autodekodera stosowego, którego zadaniem jest na podstawie zbioru obrazów odręcznego pisma nauczyć się klasyfikować liczbę, by następnie rekonstruować obraz wejściowy. W celu prezentacji mechanizmów działania sieci posłużono się bazą danych MNIST [126] (ang. *Modified National Institute of Standards and Technology database,* MNIST), która jest obszerną bazą danych ręcznie pisanych cyfr i jest powszechnie wykorzystywana do trenowania różnych systemów przetwarzania obrazów. Baza danych MNIST zawiera 60 000 obrazów przeznaczonych do trenowania oraz 10 000 obrazów używanych do testowania. Na rys. a.1 przedstawiono przykładowe obrazy zeskanowanych, odręcznie napisanych liczb.

Rys. A.1 Przykładowe wartości obrazów z bazy danych MNIST

Implementacja autodekodera stosowanego, przedstawionego na rys. a.2 obejmuje dwa główne komponenty: koder i dekoder. Kodowanie rozpoczyna się od transformacji danych wejściowych, takich jak obrazy monochromatyczne o wymiarach 28x28 pikseli, do wektora 784-elementowego. Następnie, te wektory przechodzą przez warstwy o malejącej liczbie jednostek, co umożliwia ekstrakcję i kompresję kluczowych cech. Dekoder dokonuje transformacji kodowań przez warstwy o rosnącej liczbie jednostek, co prowadzi do rekonstrukcji oryginalnych obrazów.

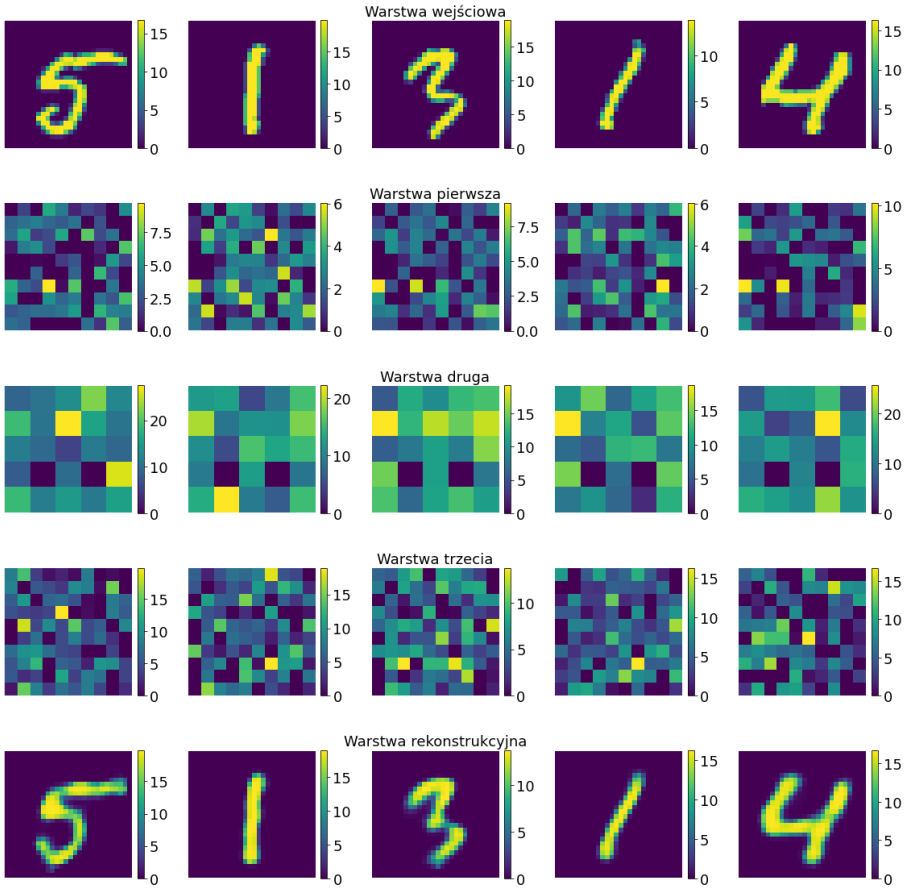
Na rys. a.2 przedstawiono wybrane obrazy wejściowe, aktywacje poszczególnych neuronów w koderze oraz dekoderze oraz odtworzony obraz wejściowy.



Rys. A.2 Autodekoder stosowy

A.1.2 Efekty działania

Na rys. a.3 przedstawiono produkty działania dla wytrenowanej sieci neuronowej. Zaprezentowano aktywacje poszczególnych neuronów w poszczególnych warstwach.



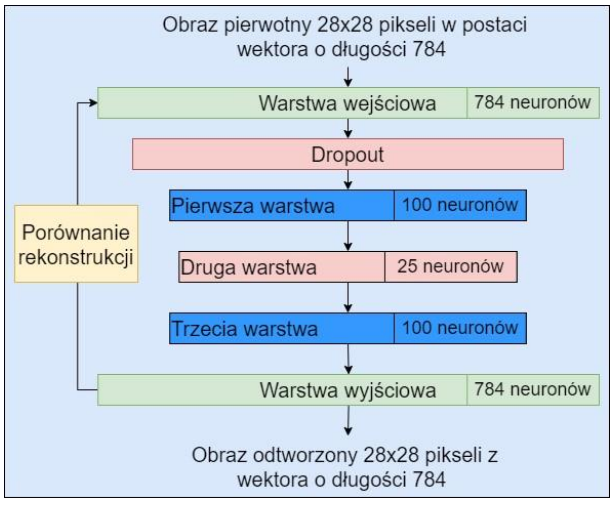
Rys. A.3 Przykłady działania autodekodera dla 5 wybranych liczb. w kolumnach są zaprezentowane poszczególne wartości w warstwach, odpowiednio: warstwa wejściowa, 1 warstwa ukryta, 2 warstwa ukryta, 3 warstwa ukryta, warstwa wyjściowa – rekonstrukcja

Zauważalny jest fakt, że sieć jest w stanie zredukować wartości o reprezentowanej liczbie do zaledwie 25 wartości, a następnie odtworzyć obraz z zachowaniem cech szczególnych jak np. charakter pisma, niedoskonałości nacisku długopisu na kartkę.

A.2 Odszumiający autodekoder stosowy

A.2.1 Opis zadania

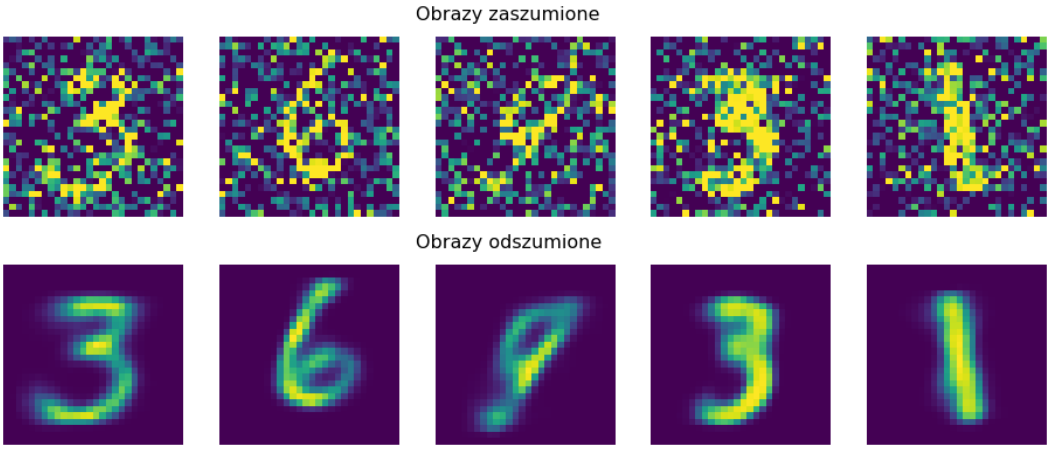
W załączniku przedstawiono przykład implementacji autodekodera stosowego, którego zadaniem jest na podstawie zbioru obrazów odręcznego pisma nauczyć się klasyfikować liczbę, wydobywać informacje charakterystyczne, a następnie rekonstruować obraz wejściowy. Wykorzystano bazę danych z podrozdziału A.1. Bazę MNIST zmodyfikowano poprzez celowe dodanie szumu gaussowskiego.
Implementacja autodekodera stosowanego, przedstawionego na rys. a.4 obejmuje dwa główne komponenty: koder i dekoder. Kodowanie rozpoczyna się od transformacji danych wejściowych, takich jak obrazy monochromatyczne o wymiarach 28x28 pikseli, do wektora 784-elementowego. Następnie, te wektory przechodzą przez warstwy o malejącej liczbie jednostek, co umożliwia ekstrakcję i kompresję kluczowych cech. Dodatkowo dodano warstwę Dropout, która redukuje liczbę połączeń pomiędzy neuronami, poprzez wyłączenie najsłabszych połączeń.
Dekoder dokonuje transformacji kodowań przez warstwy o rosnącej liczbie jednostek, co prowadzi do rekonstrukcji oryginalnych obrazów.



Rys. A.4 Autokoder stosowy odszumiający

A.2.2 Efekty działania

Na rys. a.5 przedstawiono produkty działania dla wytrenowanej sieci neuronowej.



Rys. A.5 Przykłady obrazów wejściowych oraz wyjściowych dla wytrenowanej sieci autodekodera odszumiającego

Zauważalny jest fakt, że sieć jest w stanie zinterpretować z silnie zaszumionego obrazu wyciągnąć najważniejsze cechy oraz oczyścić obraz.

Załącznik B: Symulator dynamicznej odpowiedzi konstrukcji mostowej

B.1 Założenia

Symulator dynamicznej odpowiedzi konstrukcji mostowej jest programem, który służy do rozwiązywania i modelowania przejazdu pojazdu po belce zgodnie z założeniami przyjętymi w podrozdziale 4.2. Stworzono 3 modele pojazdów i dla każdego z nich napisano program rozwiązujący zagadnienie. W ramach tego załącznika ograniczono się do prezentacji symulatora dla modelu szczegółowego, ponieważ jest najbardziej rozbudowany.

Program został napisany w środowisku Python 3.X. Symulator składa się z kilku podprogramów, które ze sobą współpracują. Wyszczególnić można następujące podprogramy:

1. **Generator populacji pojazdów:** Zadaniem programu jest przygotowanie populacji pojazdów poruszających się po analizowanej konstrukcji mostowej. Parametry obiektu mostowego są dobierane z góry, natomiast pojazdy są generowane losowo z zachowaniem jednak pewnych reguł i korelacji między danymi.
2. **Program zarządzający:** Program sterujący procesem symulowania tysięcy przejazdów. Program napisany z myślą o pracy wielowątkowej. Jego zadaniem jest rozdzielanie pojedynczych przejazdów i przypisywanie do odpowiednich wątków wykonawczych oraz zebranie i scalenie otrzymanych wyników.
3. **Wirtualny model mostu:** Program obliczeniowy. Na podstawie dostarczonych informacji opisujących parametry modelu konstrukcji mostowej oraz parametry przejazdu, oblicza odpowiedź konstrukcji.

B.2 Program – Generator populacji pojazdów

B.2.1 Struktura programu

Zadaniem programu jest generowanie populacji pojazdów. w pracy wykorzystywano wiele niezależnych populacji pojazdów oraz wyników symulacji odpowiedzi konstrukcji. Każda populacja pojazdów oraz seria analiz wymagała indywidualnego oprogramowana, mimo to w każdej wersji Generator zawierał kluczowe podprogramy takie jak:

- generator słowników,
- generator populacji,
- generator serii.

Zadaniem generatora słowników, jest generowanie na podstawie odgórnie narzuconych ograniczeń oraz jawnie zadeklarowanych funkcji rozkładu prawdopodobieństwa, stworzenie określonej populacji pojazdów. Program ten został opisany w podpunkcie 4.5.2 i stanowi on podstawę całego programu.

Parametry wejściowe:

- sylwetki pojazdów,
- liczebność populacji,
- udział procentowy sylwetek pojazdów w ruchu,
- udział pojazdów przeciążonych w ruchu,
- ograniczenia rozstawów osi,
- rozkłady funkcji gęstości dla poszczególnych osi sylwetek pojazdów.

Parametry wyjściowe:

- słownik pojazdów zawierających:
    - konfigurację osi,
    - masy statyczne przypadające na poszczególne osie.

Zadaniem generatora populacji jest podczytanie wygenerowanych sylwetek pojazdów oraz parametrów konstrukcji mostowej, wylosowanie prędkości ruchu oraz stworzenie nowego słownika, zawierającego już wszystkie parametry wejściowe, niezbędne do obliczenia dynamicznej odpowiedzi konstrukcji. Program na podstawie danych wejściowych od użytkownika, przygotowuje dane dla wybranego modelu.

Parametry wejściowe:

- lokalizacja słownika pojazdów,
- typ modelu obliczeniowego (podstawowy, rozszerzony, szczegółowy),
- parametry obiektu mostowego,
- docelowa ścieżka zapisu wygenerowanych danych.

Parametry wyjściowe:

- Słownik parametrów symulacji:
    - konfiguracja osi,
    - naciski statyczne na oś,
    - masy resorowane i nieresorowane,
    - charakterystyka zawieszenia pojazdu,
    - prędkość ruchu,
    - parametry konstrukcji mostowej.

Zadaniem generatora serii jest generowanie wielu niezależnych populacji o określonej liczebności. Szczególnie w trakcie analizy stabilności proponowanego rozwiązania konieczne było sprawdzenie wpływu czynników na dokładność systemu. z tego względu generowano niezależne populacje dynamicznej odpowiedzi konstrukcji.

Parametry wejściowe:

- wartości brzegowe zmieniającego się parametru (np. rozpiętość obiektu mostowego),
- ścieżki docelowe zapisu danych.

Parametry wyjściowe:

- wiele niezależnych populacji danych wejściowych.

Najważniejszy w kontekście omawianej pracy jest generator słowników, ponieważ wygenerowane naciski oraz rozstawy osi są kluczowe w systemie identyfikacji obciążeń. Dodatkowe programy są swego rodzaju dopełnieniem, które umożliwiają sprawne przygotowywanie danych wejściowych do symulatora. w B.2.2 opisano więc metody kluczowe wykorzystywane w Generatorze Słowników.

B.2.2 Opis metod

B.2.2.1 Metoda inicjująca: **`__init__(*params)`**

Metoda inicjalizująca jest konstruktorem klasy **Generator_slownikow**, który przygotowuje obiekt do pracy z danymi wejściowymi dotyczącymi rozstawów i nacisków osi pojazdów.
Parametry wejściowe:

- **`file1`**: ścieżka do pliku Excel zawierającego graniczne rozstawy osi.
- **`file2`**: ścieżka do pliku Excel zawierającego graniczne naciski osi.

Metoda nie zwraca bezpośrednio żadnych wartości, ale inicjalizuje atrybuty instancji klasy
Poszczególne kroki:

1. **Inicjalizacja atrybutów instancji**: Przypisanie wartości przekazanych parametrów **`file1`** i **`file2`** do atrybutów **`graniczne_rozstawy_osi`** i **`graniczne_naciski_osi`**. Inicjalizacja atrybutów **`katalog_pojazdow`** i **`naciski_graniczne`** jako **`None`**, przygotowując je do późniejszego wypełnienia danymi.

B.2.2.2 Metoda*:* **`Podczytaj_rozstawy_graniczne(*params)`**

Metoda została zaprojektowana do odczytu danych o rozstawach granicznych osi pojazdów z pliku Excel i zapisania ich w słowniku **`katalog_pojazdow`**.

Metoda nie zwraca bezpośrednio żadnych wartości, ale aktualizuje atrybut **`katalog_pojazdow`** instancji klasy.

Poszczególne kroki:

1. Odczytanie danych z pliku Excel.
2. Stworzenie pustego słownika **katalog_pojazdow**.
3. Wyciągnięcie kategorii pojazdu z odpowiednich komórek arkusza Excel.
4. Wyciągnięcie liczby osi dla każdego typu pojazdu.
5. Ustalenie wartości rozstawów osi (maksymalne, średnie i minimalne), upewniając się, że nie są **NaN** (ang. *Not a Number*).
6. Dodanie wartości do słownika **katalog_pojazdow**, używając kategorii pojazdu jako klucza.

B.2.2.3 Metoda: `Rozklad_normalny(*params)`

Metoda generuje rozkłady normalne dla rozstawów osi pojazdu na podstawie jego typu.
Parametry wejściowe:

- **`typ_pojazdu`**: typ pojazdu, dla którego mają być wygenerowane rozkłady.

Produkty wyjściowe:

- **`dict`**: Słownik zawierający rozkłady normalne dla rozstawów osi. Klucze w słowniku reprezentują kolejne osie (np. **`a1`**, **`a2`**), a wartości to obiekty rozkładów normalnych z biblioteki **`scipy.stats()`**.

Poszczególne kroki:

1. Sprawdzenie obecności typu pojazdu w słowniku **katalog_pojazdow**.
2. Pobranie odpowiednich parametrów dla danego typu pojazdu z **katalog_pojazdow**.
3. Utworzenie pustego słownika **rozklady_prawdopodobienstwa** do przechowywania rozkładów normalnych.
4. Wyznaczenie rozkładów normalnych: dla każdej osi wyznaczenie parametrów rozkładu normalnego (średnia, odchylenie standardowe) na podstawie wartości maksymalnych, średnich i minimalnych oraz dodanie obliczonych rozkładów do słownika **rozklady_prawdopodobienstwa**.

B.2.2.4 Metoda: `Wegeneruj_populacje_rozkladu_osi(*params)`

Metoda generuje populację rozstawów osi dla danego typu pojazdu.
Parametry wejściowe:

- **`typ_pojazdu`** : typ pojazdu, dla którego mają być wygenerowane rozstawy.
- **`liczba_pojazdow`** : liczba pojazdów do wygenerowania.

Produkty wyjściowe:

- **`list`** : Lista słowników reprezentujących wygenerowane pojazdy. Każdy słownik zawiera typ pojazdu oraz rozstawy osi.

Poszczególne kroki:

1. Generowanie rozkładów normalnych: wywołanie funkcji **generate_normal_distributions()**, aby uzyskać rozkłady normalne dla danego typu pojazdu.
2. Inicjalizacja listy populacji: stworzenie pustej listy **lista_populacji** do przechowywania wygenerowanych pojazdów.
3. Generowanie pojazdów: losowanie rozstawów osi dla każdego pojazdu zgodnie z wcześniej wygenerowanymi rozkładami normalnymi, sprawdzanie poprawności wygenerowanych rozstawów (czy mieszczą się w granicach określonych w **katalog_pojazdow**), dodanie poprawnych pojazdów do listy **lista_populacji**.

B.2.2.5 Metoda: `Podczytaj_naciski_graniczne(*params)`

Metoda odczytuje dane o naciskach granicznych osi z pliku Excel i zapisuje je w słowniku **`naciski_graniczne`**.

Metoda nie zwraca bezpośrednio żadnych wartości, ale aktualizuje atrybut **`naciski_graniczne`** instancji klasy.

Poszczególne kroki:

1. Odczytanie pliku Excel zawierającego dane o naciskach granicznych osi.
2. Stworzenie pustego słownika **naciski_graniczne**, który będzie przechowywał odczytane dane.
3. Odczytanie danych z arkuszy i zapisanie ich w słowniku **naciski_graniczne**.

B.2.2.6 Metoda: `Wyznacz_funkcje_gestosci(*params)`

Metoda wyznacza funkcje gęstości rozkładu dla nacisków osi na podstawie danych z arkusza Excel.

Parametry wejściowe:

- **`nazwa_arkusza`**: nazwa arkusza w pliku Excel zawierającego dane nacisków osi.

Produkty wyjściowe:

- **`x_wartosci`**: wartości x dla funkcji gęstości.
- **`wartosci_rozkladu_dla_osi`**: lista wartości funkcji gęstości dla każdej osi.

Poszczególne kroki:

1. Pobranie średnich nacisków i odchyleń standardowych dla każdej osi z arkusza Excel oraz pobranie typu rozkładu dla każdej osi (0 - rozkład normalny, 1 - rozkład wykładniczy, 2 - rozkład gamma).
2. Stworzenie pustej listy **wartosci_rozkladu_dla_osi**, która będzie przechowywała wartości funkcji gęstości dla każdej osi, oraz określenie minimalnej i maksymalnej wartości x na podstawie średnich nacisków i odchyleń standardowych.
3. Obliczenie wartości funkcji gęstości rozkładu dla każdej osi na podstawie odpowiedniego typu rozkładu (normalny, wykładniczy, gamma) oraz zwrócenie wartości x i listy wartości funkcji gęstości dla każdej osi.

B.2.2.7 Metoda: `Wygeneruj_populacje_naciskow_osi(*params)`

Metoda generuje populację nacisków osi dla danego typu pojazdu na podstawie wyznaczonych funkcji gęstości rozkładu.

Parametry wejściowe:

- **`typ_pojazdu`** : typ pojazdu, dla którego mają być wygenerowane naciski.
- **`liczba_pojazdow`** : liczba pojazdów do wygenerowania.

Produkty wyjściowe:

- **`populacja`**: lista słowników reprezentujących wygenerowane pojazdy z naciskami osi. Każdy słownik zawiera typ pojazdu oraz naciski na poszczególne osie.

Poszczególne kroki:

1. Obliczenie funkcji gęstości rozkładu poprzez wywołanie funkcji **wyznacz_funkcje_gestosci()**, aby uzyskać funkcje gęstości rozkładu dla danego typu pojazdu.
2. Stworzenie pustej listy **populacja**, która będzie przechowywała wygenerowane pojazdy.
3. Generowanie pojazdów poprzez losowanie nacisków osi zgodnie z wcześniej wyznaczonymi funkcjami gęstości rozkładu, normalizację rozkładów nacisków oraz dodanie wygenerowanych nacisków osi do słownika pojazdu.
4. Dodanie poprawnych pojazdów z wygenerowanymi naciskami osi do listy **populacja**.

B.3 Program zarządzający – Worker

B.3.1 Struktura programu

Program **Worker** jest klasą, która służy do zarządzania wieloma obiektami klasy PhysicsSolver wykonującymi równoległe, niezależne obliczenia. Głównym celem klasy **Worker** jest przydzielanie części populacji do poszczególnych procesów, a następnie agregacja wyników do jednego pliku wyjściowego.

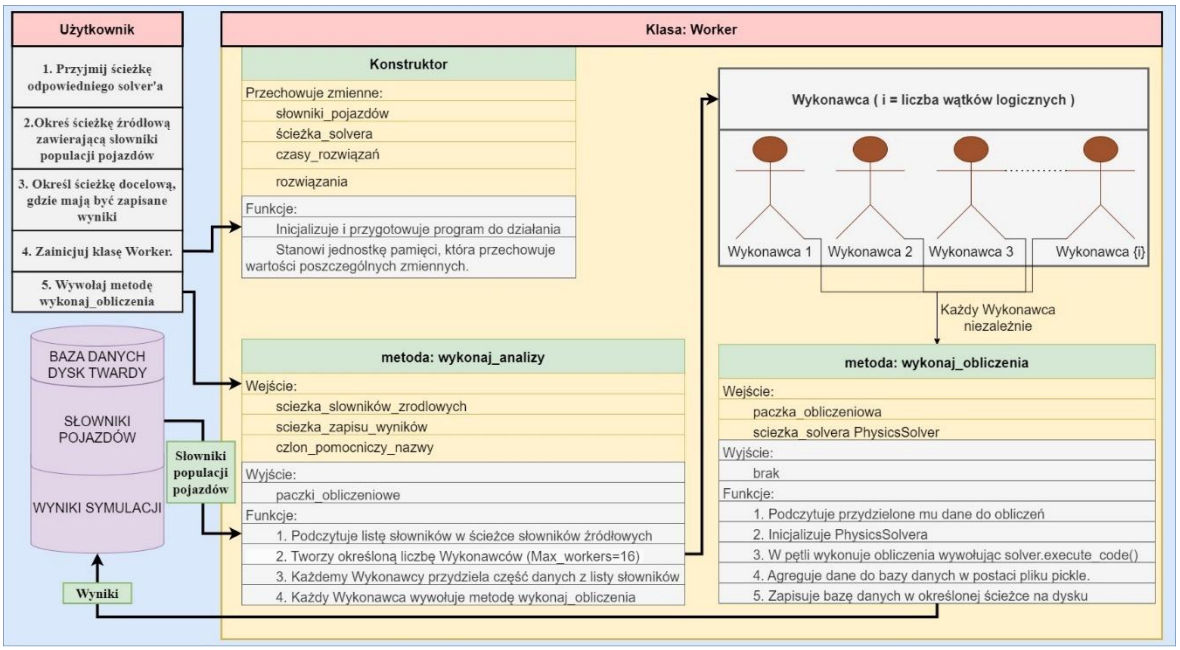
Klasa **Worker** zawiera następujące metody:

- konstruktor **`__init__()`**,
- metoda: **`wykonaj_analizy()`**,
- metoda: **`wykonaj_obliczenia()`**,
- metoda pomocnicza: **`podczytaj_solver_ze_sciezki()`**,
- metoda pomocnicza: **`set_solver_path()`**,
- setter i getter dla atrybutu **`sciezka_solvera`**,

Dodatkowo w strukturze programu znajdują się metody pomocnicze:

- **`podczytaj_parametry_mostu()`**,
- **`uruchom_model(),`**

Schemat działania programu Worker został zaprezentowany na schemacie blokowym rys. b.1.



Rys. B.1 Schemat blokowy działania programu Worker

Użytkownik określa model obliczeniowy, który zostanie do wykonania analiz oraz ścieżki, gdzie zapisane są wyniki wygenerowane przez Generator Populacji Pojazdów. Następnie inicjalizuje klasę Worker, która podczytuje listę słowników pojazdów do przeanalizowania, następnie inicjuje Wykonawców i przekazuje im pewien fragment słowników do przeanalizowania. Każdy Wykonawca,

tworzy swoją niezależną instancję klasy PhysicsSolver i następnie iteracyjnie wykonuje symulacje dla każdego pojazdu, że swojej części słownika. Wyniki są agregowane i zapisywane w bazie danych na dysku twardym

W poniższych podrozdziałach opisano dokładniej działanie metod głównych programu tj. metoda **`wykonaj_analizy()`** oraz metoda **`wykonaj_obliczenia()`**.

B.3.2 Opis metod

B.3.2.1 Metoda: **`wykonaj_analizy(*params)`**

Metoda została zaprojektowana do wykonywania analiz symulacyjnych na podstawie dostarczonych plików źródłowych. Metoda ta automatyzuje proces przetwarzania danych symulacyjnych i rozdziela zadania na wielowątkowy system, co umożliwia efektywne wykorzystanie zasobów procesora i przyspieszenie obliczeń.

Metoda przyjmuje następujące parametry:

- **`sciezka_slownikow_zrodlowych`**: ścieżka do folderu z plikami źródłowymi zawierającymi dane do analizy.
- **`sciezka_zapisu_wynikow`**: ścieżka do folderu, w którym będą zapisywane wyniki analiz.
- **`czlon_pomocniczy_nazwy`**: pomocniczy człon nazwy plików, który umożliwia identyfikację plików przeznaczonych do analizy.

Poszczególne kroki:

1. Stworzenie listy słowników poprzez sprawdzenie ścieżki słowników źródłowych, gdzie znajdują się wygenerowane słowniki populacji pojazdów, oraz utworzenie listy plików do przeanalizowania.
2. Generowanie ścieżek docelowych dla każdego słownika pojazdów, pod którymi zostaną zapisane wyniki obliczeń.
3. Rozdzielenie pracy przez stworzenie niezależnych procesów, z przypisaniem do każdego wykonawcy słownika do przeanalizowania. Po wykonaniu zadania przez wykonawcę zostaje mu przyporządkowana nowa paczka, aż do przeanalizowania wszystkich plików z folderu źródłowego.

B.3.2.2 Metoda: `wykonaj_obliczenia(*params)`

Metoda została zaprojektowana do wykonywania analiz symulacyjnych na podstawie dostarczonych plików źródłowych. Metoda ta automatyzuje proces przetwarzania danych symulacyjnych i rozdziela zadania na wielowątkowy system, co umożliwia efektywne wykorzystanie zasobów procesora i przyspieszenie obliczeń.

Metoda przyjmuje następujące parametry:

- **`dane_do_analizy`**: ścieżka do słownika pojazdów.
- **`solver_path`**: ścieżka dostępu do solvera, rozwiązującego zagadnienie.

Poszczególne kroki:

1. Podczytanie danych ze słownika pojazdów.
2. Inicjacja obiektu klasy **PhysicsSolver**.
3. Iteracyjne wykonywanie symulacji przez wywołanie w pętli metody **execute_code** z obiektu **PhysicsSolver**, przekazując kolejno każdy pojazd do symulacji i otrzymując wyniki.
4. Agregacja rozwiązań do pliku zapisywanego w bazie danych na dysku twardym w ścieżce docelowej.

B.4 Wirtualny model mostu – PhysicsSolver

B.4.1 Struktura programu

PhysicsSolver jest wg. programowania obiektowego abstrakcyjną klasą, która do poprawnego działania wymaga inicjalizacji obiektu, czyli deklaracji zmiennych oraz zarezerwowanie pamięci operacyjnej. Programowanie obiektowe, pozwala tworzyć wiele niezależnych obiektów, co jest niezwykle przydatne przy programowaniu wielowątkowym.

Celem symulatora dynamicznej odpowiedzi obiektu mostowego jest generowanie jak największej liczby danych. Omawiany program pozwala na stworzenie wielu niezależnych obiektów wykonawczych i przypisanie ich do poszczególnych wątków procesora wielordzeniowego. Każdy obiekt posiada te same funkcje i te same zmienne. Zakładając, że program jest uruchamiany na procesorze 16 wątkowym, a symulacje są od siebie niezależne, mogą być one wykonywane jednocześnie.

Każdy z programów wykonawczych przyjmuje następujące parametry wejściowe:

- sztywność belki,
- tłumienie belki,
- masa własna belki,
- rozpiętość teoretyczna,
- liczba osi,
- odległości między osiami,
- sztywność zawieszenia,
- tłumienie zawieszenia,
- prędkość ruchu,
- nierówność nawierzchni,

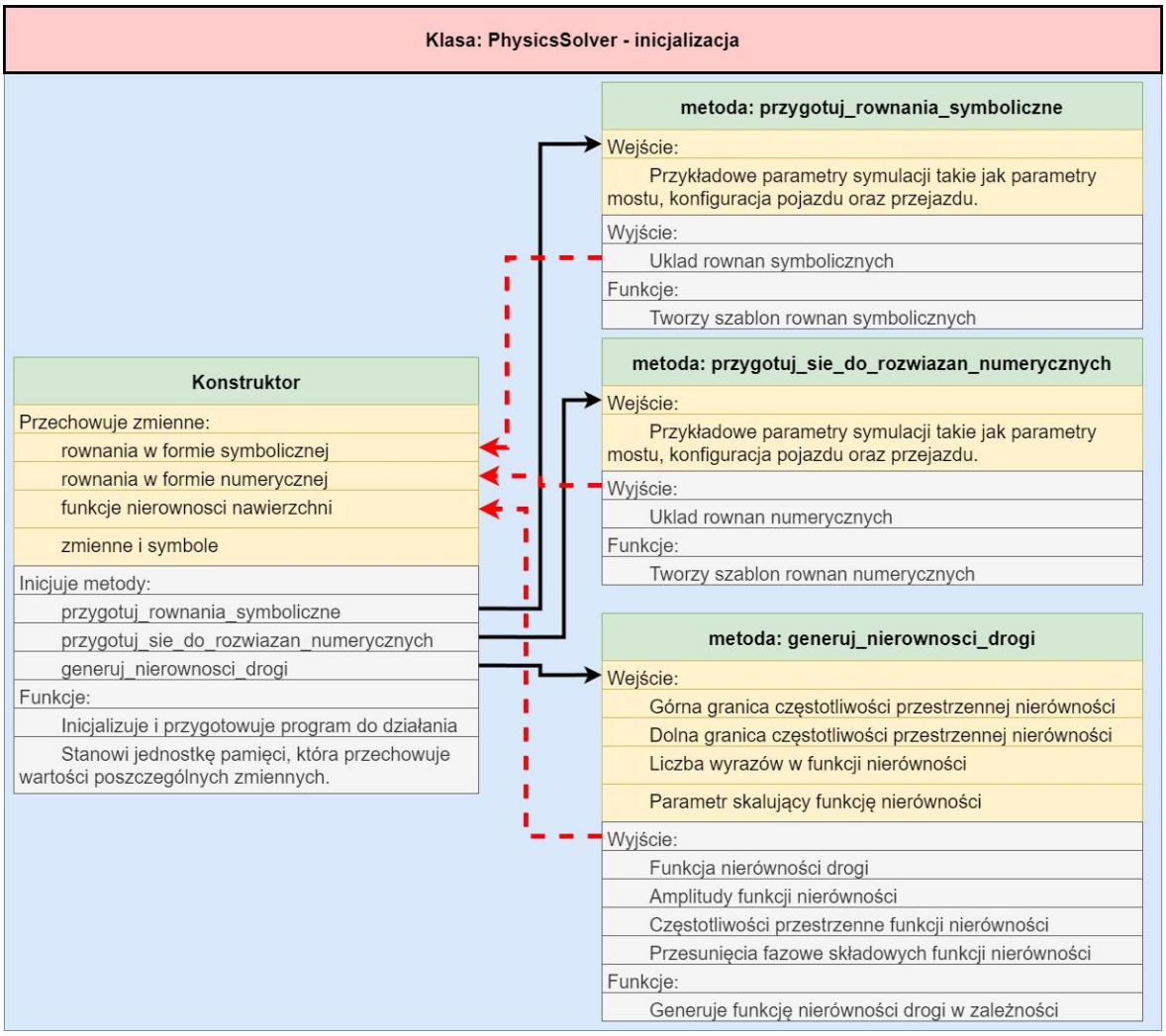
Wynikiem obliczeń jest:

- wektor czasu,
- lista wektorów rozwiązań tj. przemieszczeń oraz prędkości przemieszczeń belki oraz punktów masowych pojazdu.

PhysicsSolver zawiera następujące metody:

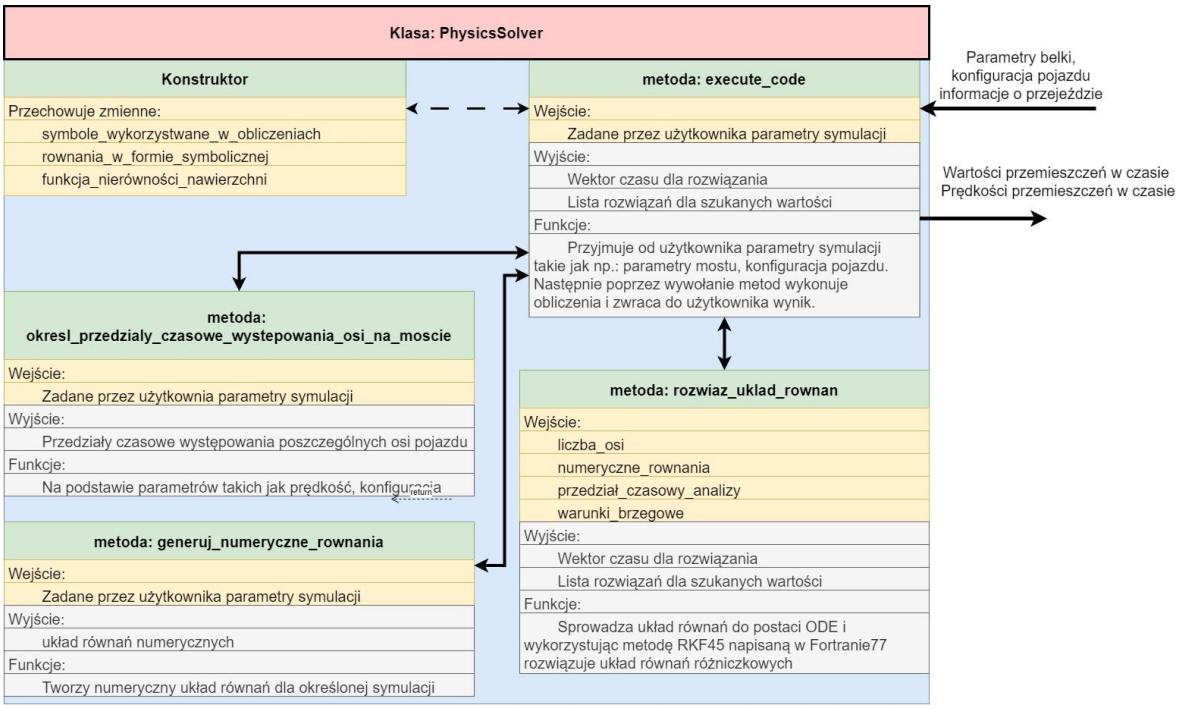
- konstruktor **`__init__()`**,
- metoda: **`przygotuj_rownania_symboliczne()`**,
- metoda: **`przygotuj_sie_do_rozwiazan_numerycznych()`**,
- metoda: **`genetuj_nierownosci_drogi()`**,
- metoda: **`okresl_przedzialy_czasowe_wystepowania_osi_na_moscie()`**,
- metoda: **`generuj_numeryczne_rownania()`**,
- metoda: **`rozwiąż_uklad_rownan()`**,
- metoda: **`execute_code()`**,

W pierwszej kolejności użytkownik albo program nadrzędny musi zainicjować obiekt. Odbywa się to poprzez wywołanie konstruktora, który rezerwuje miejsce w pamięci operacyjnej na zmienne oraz przeprowadza metody przygotowawcze. Inicjacja ma miejsce tylko raz, a jej schemat został przedstawiony na rys. b.2.



Rys. B.2 Inicjalizacja obiektu klasy PhysicsSolver

Następnie operator wywołuje metodę **`execute_code()`** przekazując parametry dot. przejazdu i w wyniku otrzymuje odpowiedź dynamiczną konstrukcji oraz pojazdu. Mechanizm działania został przedstawiony na schemacie rys. b.3.



Rys. B.3 Proces symulowania odpowiedzi konstrukcji mostowej

Poszczególne metody zostały opisane w odpowiednich podrozdziałach.

B.4.2 Opis metod

B.4.2.1 Metoda inicjująca: `__init__(*params)`

Konstruktor to specjalna metoda w programowaniu obiektowym, która jest wywoływana w momencie tworzenia nowej instancji (obiektu) klasy. Jej głównym zadaniem jest inicjalizacja nowo utworzonego obiektu, czyli przygotowanie go do użycia poprzez ustawienie początkowych wartości jego atrybutów (zmiennych instancji).

W tym przypadku konstruktor tworzy miejsce w pamięci operacyjnej na symbole oraz równania. Dodatkowo wywołuję funkcje pomocnicze w celu:

- wygenerowania funkcji nierówności nawierzchni,
- zbudowania układów równań w formie symbolicznej,
- częściowej transformacji układów równań z formy symbolicznej do formy numerycznej.

Celem powyższych procedur, jest przygotowanie części elementów rozwiązania, elementów które są ogólne i niezależne od parametrów przejazdu. Dzięki temu metoda **`execute_code()`** nie musi wykonywać wstępnych obliczeń i przygotowywać układów równań, ponieważ są już one zdefiniowane. Konstruktor może przyjąć parametry zewnętrzne, takie jak parametr opisujący nierówności nawierzchni. Dzięki temu wygenerowana funkcja nierówności nawierzchni może być identyczna dla wszystkich instancji.

B.4.2.2 Metoda: `generująca_nierownosci_drogi(*params)`

Metoda generuje profil nierówności drogi zgodnie z teorią opisaną w podpunkcie 4.2.1 , używając serii funkcji sinusowych. Profil ten jest kluczowy w symulacjach dynamiki pojazdów, umożliwiając realistyczne modelowanie interakcji między pojazdem a nawierzchnią drogi.
Parametry wejściowe funkcji obejmują:

- **`n_lower`:** dolną częstotliwość przestrzenną nierówności,
- **`n_upper`:** górną częstotliwość przestrzenną nierówności,
- **`iters`:** liczbę różnych nierówności,
- **`chropowatosc_val`:** wartość nierówności nawierzchni ().

Produkty wyjściowe programu to:

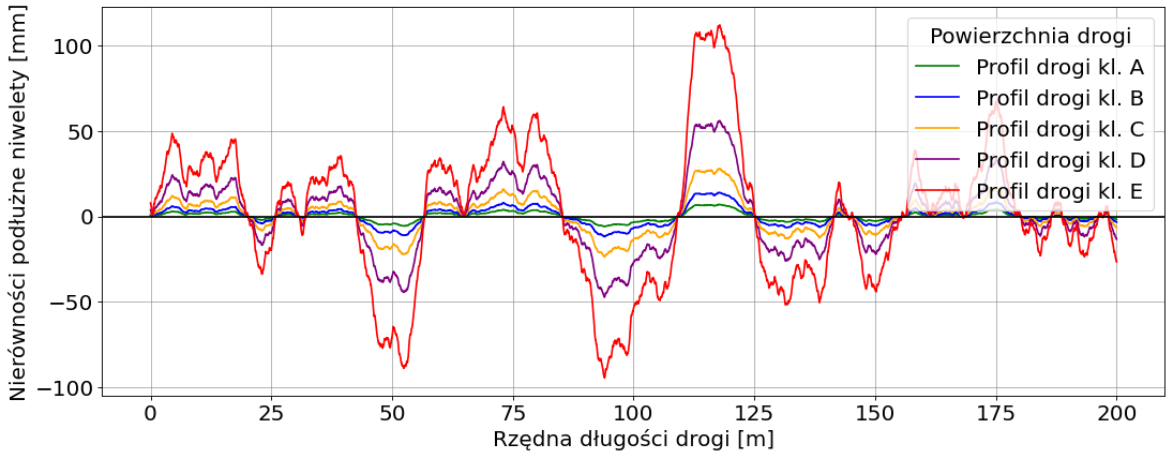
- symboliczne równanie profilu drogi,
- numeryczne wartości opisujące amplitudy, przesunięcia fazowe oraz częstotliwości poszczególnych nierówności.

Poszczególne kroki:

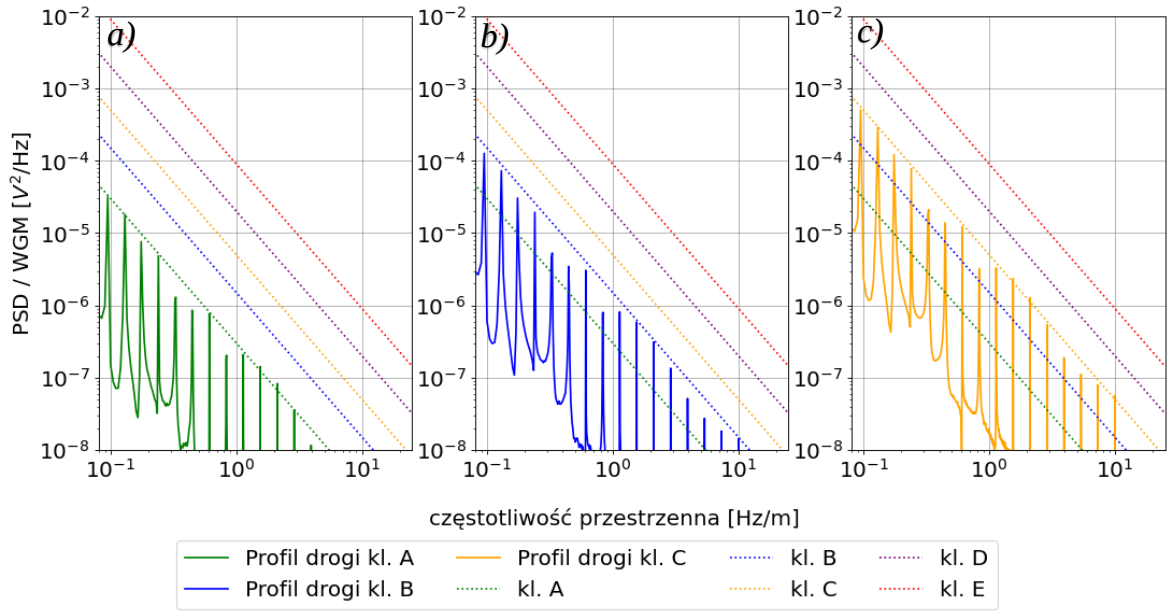
1. Inicjalizacja parametrów, które będą używane do generowania profilu drogi.
2. Generowanie profilu nierówności poprzez użycie metody **generate_rx()**, inicjalizację zmiennych przechowujących funkcję nierówności nawierzchni, amplitudy, częstotliwości i przesunięcia fazowe, oraz iterację przez liczbę terminów w serii, z obliczaniem dla każdej iteracji odpowiedniej częstotliwości, losowego przesunięcia fazowego oraz amplitudy. Dla każdej iteracji dodawanie składnika funkcji sinusoidalnej do całkowitego profilu drogi.
3. Obliczanie amplitudy przy użyciu metody **policz_wart_alpha()** dla danej częstotliwości, z wykorzystaniem funkcji **gdnk()**, która uwzględnia wartość referencyjną nierówności drogi.
4. Sumowanie wyników funkcji nierówności nawierzchni, amplitud, częstotliwości i przesunięć fazowych po zakończeniu iteracji.
5. Przypisanie wyników generowania profilu drogi do atrybutów klasy.

Jednym z kluczowych wymagań programu jest zapewnienie powtarzalności wyników przy zachowaniu elementów losowości w generowaniu profilu drogi. Realizacja tego założenia umożliwia symulację różnych pojazdów przejeżdżających po tym samym obiekcie mostowym, które napotykają identyczny profil nierówności. Taka funkcjonalność jest niezbędna, aby zapewnić realistyczne warunki testowe, które są jednocześnie powtarzalne i przewidywalne dla różnych scenariuszy ruchu drogowego.
Osiągnięcie powtarzalności i losowości w generowaniu profilu drogi zostało zrealizowane poprzez zastosowanie stałego ziarna losowości w funkcji **`random.seed(ziarno=numer_kroku_iteracyjnego)`**.
Oznacza to, że za każdym razem ciąg wylosowanych wartości jest ten sam. w ramach procesu weryfikacji profilu drogi przeprowadzono analizę widmową gęstości mocy (rys. b.4 oraz rys. b.5).

Celem tej analizy było sprawdzenie, czy uzyskane wyniki mieszczą się poniżej ustalonej krzywej odniesienia, która określa akceptowalne parametry dla profilu drogi. Analiza ta pozwala na ocenę charakterystyk widmowych generowanego profilu, co jest kluczowe dla zapewnienia zgodności z normami i oczekiwaniami dotyczącymi jakości nawierzchni drogowej.



Rys. B.4 Wygenerowane niwelety dróg dla klas o odpowiedniej klasie nierówności nawierzchni zgodnie z wytycznymi Eurokodów oraz normy ISO 8608



Rys. B.5 Analiza widmowa gęstości nocy niwelet wraz z ograniczeniami odpowiednich klas drogi

B.4.2.3 Metoda: `okresl_przedzialy_czasowe_wystepowania_osi_na_moscie(*params)`

Metoda została zaprojektowana do określania przedziałów czasowych, w których osie pojazdów znajdują się na moście, na podstawie dostarczonych parametrów symulacji.

Metoda przyjmuje słownik zawierający wartości i parametry wejściowe dla symulacji. Kluczowymi elementami tego słownika są:

- t_i: czasy wjazdu na most dla poszczególnych osi pojazdów,
- t_{ki}: czasy zjazdu z obiektu mostowego dla poszczególnych osi pojazdów,
- l_{osi}: liczba osi pojazdu.

Metoda zwraca listę krotek (*dop. uporządkowana kolekcja stałych wartości*), gdzie każda krotka reprezentuje przedział czasowy. Elementy składowe krotki to:

- `T0`: czas początkowy przedziału,
- `T1`: czas końcowy przedziału,
- `osie_na_przedziale`: Lista numerów osi, które znajdują się na moście w danym przedziale czasowym,

Poszczególne kroki:

1. Pobranie liczby osi z dostarczonego słownika **parametry_symulacji**.
2. Pobranie czasów wjazdu i zjazdu dla każdej osi oraz stworzenie dwóch list: **czasy_wjazdu** i czasy_zjazdu.
3. Kombinacja i sortowanie czasów poprzez połączenie list **czasy_wjazdu** i **czasy_zjazdu** w jedną listę i posortowanie jej rosnąco. Jeśli pierwszy czas w listach nie jest równy zeru, dodanie zera jako początek pierwszego przedziału czasowego.
4. Generowanie przedziałów czasowych poprzez iterację przez posortowaną listę czasów, tworzenie przedziałów czasowych (od **T0** do **T1**) i sprawdzanie, które osie znajdują się na moście w każdym przedziale czasowym.
5. Pomijanie przedziałów o zerowej długości poprzez usunięcie przedziałów o tej samej wartości początkowej i końcowej.
6. Tworzenie listy wynikowej zawierającej krotki, gdzie każda krotka składa się z czasu początkowego, czasu końcowego oraz listy numerów osi znajdujących się na moście w danym przedziale czasowym.

B.4.2.4 Metoda: `przygotuj_rownania_symboliczne(*params)`

Metoda służy do generowania ogólnego układu równań opisującego przejazd pojazdu o maksymalnie 5 osiach przez most. Jest to funkcja pomocnicza, której celem jest przyspieszenie późniejszych obliczeń poprzez przygotowanie niezbędnych równań tylko raz, podczas inicjalizacji klasy.

Parametry wejściowe:

- **`przykladowy_slownik`:** przykładowy, z góry zdefiniowany słownik danych. Służy jako podstawa do inicjalizacji i zawiera niezbędne parametry symulacji, takie jak liczba osi oraz czasy wjazdu i zjazdu osi z obiektu mostowego.

Produkty wyjściowe:

- **`przygotowany_uklad_rownan_ogolnych`:** lista zawierająca przygotowane układy równań dla przejazdu pojazdu. Każdy element listy reprezentuje zbiór równań, który uwzględnia siły działające na most oraz masy pojazdu w danym przedziale czasowym.

Poszczególne kroki:

1. Określenie przedziałów czasowych poprzez wywołanie funkcji **okresl_przedzialy_czasowe_wystepowania_osi_na_moscie()**, która generuje przedziały czasowe, w których poszczególne osie pojazdu znajdują się na moście.
2. Pobranie liczby osi z przykładowego słownika danych.
3. Inicjalizacja list równań dla obiektu mostowego oraz sił działających na most i na masy pojazdu, z przypisaniem odpowiednich równań opisujących zachowanie obiektu mostowego i siły kontaktowe działające na poszczególne osie pojazdu.
4. Stworzenie równań dla każdej osi oraz dla każdej masy (resorowanej i nieresorowanej) poprzez tworzenie kopii odpowiednich równań i ich modyfikację w zależności od obecności osi na moście w danym przedziale czasowym.
5. Generowanie układu równań opisującego interakcje między mostem a przejeżdżającym pojazdem na podstawie przygotowanych przedziałów czasowych oraz zidentyfikowanych sił kontaktowych.
6. Stworzenie listy wynikowej zawierającej pełny zestaw równań symbolicznych dla każdego przedziału czasowego, co umożliwia późniejsze efektywne obliczenia dynamiki przejazdu pojazdu przez most.

B.4.2.5 Metoda: `przygotuj_sie_do_rozwiazan_numerycznych(*params)`

Metoda służy do przygotowania składników równań, które będą wykorzystywane w dalszych obliczeniach numerycznych. Celem tej funkcji jest przeprowadzenie preprocessing'u, aby przyspieszyć późniejsze obliczenia, które są wykonywane podczas symulacji dynamicznej przejazdu pojazdu przez most.

Parametry wejściowe:

- **`ogolne_symboliczne_rownania`**: lista symbolicznych równań opisujących przejazd przykładowego pojazdu o 5 osiach. Są to równania ogólne, które zostaną przekształcone na formę numeryczną.

Metoda nie zwraca bezpośrednio żadnych wartości, ale przygotowuje składniki równań, które są przechowywane jako atrybuty klasy:

- **`lista_skladowych_rownania_mostu`**: lista składników równań dla obiektu mostowego,
- **`lista_skladowych_rownan_m_nieresorowanych`**: lista składników równań dla nieresorowanych mas pojazdu,
- **`lista_skladowych_rownan_m_resorowanych`**: lista składników równań dla resorowanych mas pojazdu.

Poszczególne kroki:

1. Przygotowanie listy symboli używanych w równaniach poprzez stworzenie krotki symbole, zawierającej zmienne wykorzystywane w równaniach.
2. Rozwiązanie układów równań symbolicznych dla przyspieszeń (drugi człon w równaniach ruchu) za pomocą funkcji **solve()**, przy uwzględnieniu wszystkich punktów masowych.
3. Wyciągnięcie składników równań dla obiektu mostowego, obejmujące przyspieszenie (**czlon_V**) oraz różnice między kolejnymi rozwiązaniami (**czlon_P1, czlon_P2, czlon_P3, czlon_P4, czlon_P5**).
4. Wyciągnięcie składników równań dla punktów masowych, zarówno dla nieresorowanych mas (**czlon_w11_0, czlon_w11_1, czlon_w21_0, itp.**), jak i resorowanych mas (**czlon_w12_0, czlon_w12_1, czlon_w22_0, itp.**).
5. Przygotowanie gotowych składników do budowy równań poprzez utworzenie list składników równań, które są przechowywane jako atrybuty klasy i będą wykorzystywane przez inne metody klasy do szybkiego tworzenia układów równań w trakcie symulacji.

B.4.2.6 Metoda: `rozwiaz_uklad_rownan(*params)`

Metoda służy do rozwiązywania układu równań różniczkowych dla systemu dynamicznego, jakim jest pojazd przejeżdżający przez most. Funkcja ta uwzględnia różną liczbę osi pojazdu i wykorzystuje odpowiednie modele matematyczne do przeprowadzenia analizy w zadanym przedziale czasowym.
Parametry wejściowe:

- **`liczba_osi`**: liczba osi w analizowanym układzie,
- **`numeryczne_rownania`**: lista równań różniczkowych w formie numerycznej,
- **`przedzialy_czasowe_analizy`**: przedział czasowy analizy, zawierający czas początkowy i końcowy,
- **`warunki_brzegowe`**: lista warunków brzegowych dla układu równań.

Produkty wyjściowe:

- **`czas_rozwiazania`**: wektor czasu.
- **`rozwiazania_ukladu_rownan`**: rozwiązania układu równań w kolejnych punktach czasowych.

Metoda zawiera kilka zagnieżdżonych funkcji pomocniczych, które są odpowiedzialne za obliczanie pochodnych dla poszczególnych układów osi. Każda z tych funkcji (**`system_dla_2_osi`, `system_dla_3_osi`, `system_dla_4_osi`, `system_dla_5_osi`**) oblicza wartości drugich pochodnych przy użyciu funkcji lambdyfikowanych *(dop. funkcja lambda to mała, funkcja zdefiniowana za pomocą słowa kluczowego lambda. Funkcje te mogą przyjmować dowolną liczbę argumentów, ale mogą mieć tylko jedno wyrażenie. Są bardzo przydatne, gdy potrzebujemy krótkiej funkcji na chwilę i nie chcemy definiować pełnej funkcji.)*
Poszczególne kroki:

1. Inicjalizacja przedziału czasowego poprzez stworzenie wektora czasu **czas_rozwiazania**, który dzieli przedział czasowy analizy na 1000 równych części.
2. Lambdyfikacja równań dla wybranej liczby osi, polegająca na zamianie symbolicznych równań różniczkowych na funkcje numeryczne przy użyciu **sympy.lambdify()**.
3. Definiowanie systemu równań poprzez stworzenie funkcji pomocniczych (**system_dla_2_osi, system_dla_3_osi, system_dla_4_osi, system_dla_5_osi**), które obliczają wartości pochodnych dla odpowiedniej liczby osi.
4. Wybór odpowiedniego systemu równań w zależności od liczby osi, aby rozwiązać układ różniczkowy.
5. Rozwiązanie układu równań różniczkowych dla zadanych warunków brzegowych i w przedziale czasowym przy użyciu **solve_ivp()** z metody **scipy.integrate.RK45()**, z wynikiem w postaci macierzy wartości reprezentującej stan układu w kolejnych punktach czasowych.

6. Zwracanie wyników, polegające na zwróceniu wektora czasu **czas_rozwiazania** oraz macierzy **rozwiazania_ukladu_rownan**, zawierającej rozwiązania układu równań dla każdego punktu czasowego.

B.4.2.7 Metoda: `generuj_numeryczne_rownania(*params)`

Metoda służy do generowania równań numerycznych na podstawie parametrów symulacji dla analizy dynamicznej pojazdu przejeżdżającego przez most. Funkcja ta przekształca symboliczne równania systemu na formę numeryczną, co pozwala na ich efektywne rozwiązywanie przy użyciu metod numerycznych.

Parametry wejściowe:

- **`parametry_symulacji`**: słownik zawierający wartości i parametry wejściowe potrzebne do symulacji. Parametry te mogą obejmować czasy wjazdu i zjazdu osi, siły działające na most oraz inne niezbędne wartości.

Produkty wyjściowe:

- **`układ_rownan_numerycznych`**: lista równań w formie numerycznej. Każdy element listy reprezentuje zestaw równań obowiązujących w danym przedziale czasowym analizy.

Poszczególne kroki:

1. Określenie przedziałów czasowych, w których poszczególne osie pojazdu znajdują się na moście, poprzez wywołanie funkcji:
   okresl_przedzialy_czasowe_wystepowania_osi_na_moscie().
2. Stworzenie równań obiektu mostowego każdego dla przedziału czasowego, opisujących dynamikę z uwzględnieniem liczby osi będących na moście, poprzez dodanie odpowiednich składników do listy składników równań obiektu mostowego.
3. Stworzenie równań dla punktów masowych pojazdu, zarówno resorowanych, jak i nieresorowanych, z modyfikacją listy równań w zależności od obecności osi na moście w każdym przedziale czasowym.
4. Transformacja równań do postaci numerycznej poprzez przekształcenie symbolicznych równań za pomocą metody xreplace(), która zamienia symbole na wartości numeryczne zgodnie z parametrami symulacji.
5. Zwrócanie układów równań numerycznych poprzez utworzenie końcowej listy równań numerycznych, zawierającej układy równań dla każdego przedziału czasowego analizy, która jest zwracana jako wynik metody.

B.4.2.8 Metoda: `execute_code(*params)`

Metoda wykonuje cały proces rozwiązywania zagadnienia fizycznego dotyczącego dynamiki obiektu mostowego i przejazdu pojazdu. Metoda ta przeprowadza kompleksową analizę, zaczynając od inicjalizacji równań ruchu, poprzez iteracyjne rozwiązywanie układów równań w różnych przedziałach czasowych, aż po końcowe rozwiązanie drgań swobodnych obiektu mostowego po przejeździe pojazdu.
Parametry wejściowe:

- **`parametry_symulacji`**: słownik zawierający wartości i parametry wejściowe symulacji. Parametry te obejmują m.in. czasy wjazdu i zjazdu osi pojazdu, liczba osi oraz inne niezbędne dane do przeprowadzenia analizy.

Produkty wyjściowe:

- **`wektor_czasu`**: wektor czasu, reprezentujący dyskretne punkty czasowe w trakcie symulacji.
- **`tablica_rozwiazan`**: tablica zawierająca przemieszczenia oraz prędkości przemieszczeń w kolejnych punktach czasowych.

Poszczególne kroki:

1. Inicjalizacja równań ruchu poprzez wygenerowanie numerycznych równań ruchu na podstawie parametrów symulacji, wywołując funkcję **generuj_numeryczne_rownania**.
2. Ustalenie liczby szukanych wielkości poprzez pobranie liczby osi pojazdu z parametrów symulacji, która określa liczbę szukanych wielkości w układzie równań.
3. Inicjalizacja warunków brzegowych poprzez stworzenie listy zer, której długość zależy od liczby osi pojazdu.
4. Inicjalizacja wektorów czasu i rozwiązań poprzez utworzenie pustych list **wektor_czasu** i **tablica_rozwiazan**, które będą przechowywać odpowiednio wektory czasu i rozwiązania układu równań.
5. Generowanie przedziałów czasowych za pomocą funkcji **okresl_przedzialy_czasowe_wystepowania_osi_na_moscie**.
6. Iteracja przez przedziały czasowe i rozwiązywanie układów równań poprzez przechodzenie przez wszystkie przedziały czasowe, rozwiązując układy równań dla każdego przedziału przy użyciu funkcji **rozwiaz_uklad_rownan**. Metoda aktualizuje warunki brzegowe na podstawie końcowych wartości poprzedniego rozwiązania i dodaje uzyskane wyniki do **wektora_czasu** i **tablicy_rozwiazan**.
7. Rozwiązanie dla drgań swobodnych obiektu mostowego po przejeździe pojazdu poprzez rozwiązanie układu równań opisujących drgania swobodne konstrukcji obiektu mostowego.
8. Agregacja danych i zwrócenie wartości poprzez utworzenie krotki zawierającej **wektor_czasu** oraz **tablice_rozwiazan** i zwrócenie tej krotki jako wynik.

Załącznik C: System AutoSIO

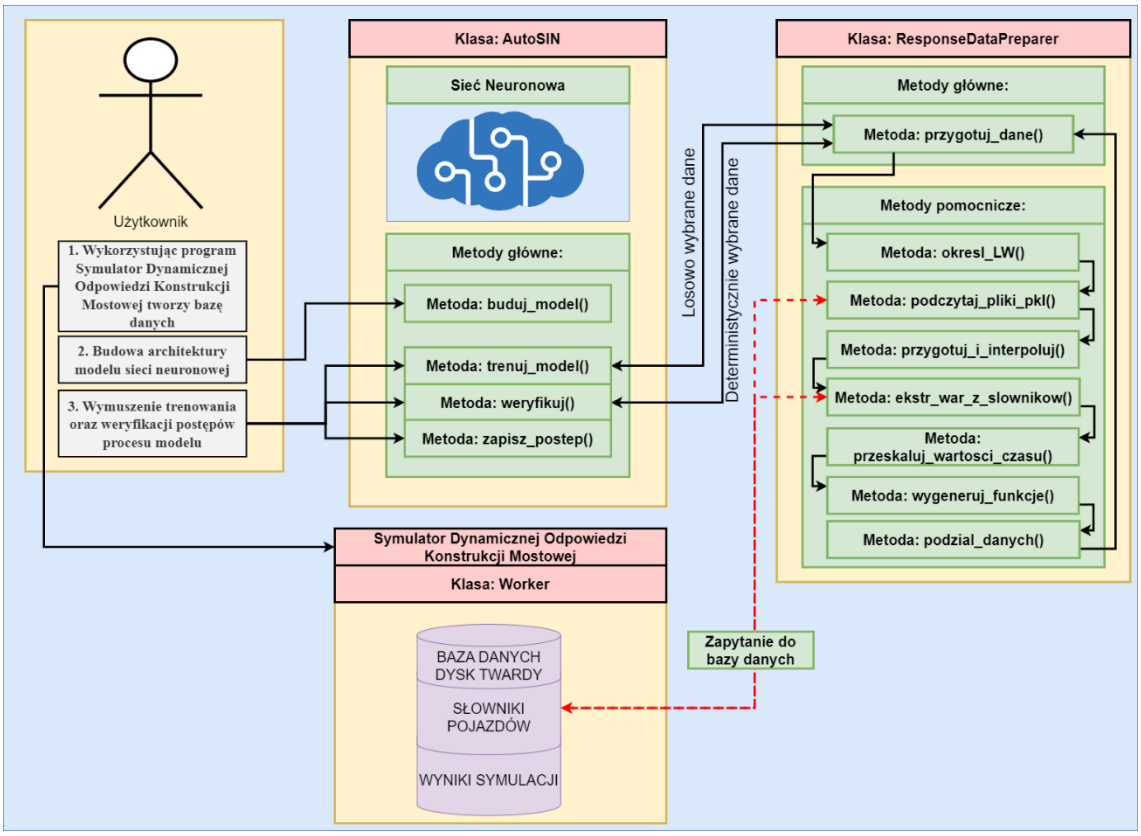
C.1 Założenia

System AutoSIO to system wykorzystujący sieci neuronowe do określania całkowitego nacisku pojazdu oraz nacisków na poszczególne osie. Podstawowym wymaganiem jest możliwość samodoskonalenia oraz zwiększania swojej precyzji szacunków w trakcie działania poprzez wykorzystanie nienadzorowanego uczenia maszynowego.

W ramach systemu AutoSIO stworzono 2 programy:

1. **AutoSIO:** zadaniem programu jest określenie całkowitego nacisku pojazdu oraz nacisków na poszczególne osie na podstawie sygnału dynamicznej odpowiedzi konstrukcji mostowej oraz walidacja dokładności oszacować na podstawie znanej testowej populacji pojazdów o znanych naciskach.
2. **ResponseDataPreparer:** zadaniem programu jest preprocessing danych dla programu AutoSIO, czyli interakcja z bazą danych stworzoną za pomocą programu Worker z pakietu Symulator Dynamicznej Odpowiedzi Konstrukcji Mostowej.

Na rys. c.1 przedstawiono schemat działania auto-adaptacyjnego systemu identyfikacji obciążeń eksploatacyjnych wraz z zaznaczeniem metod głównych oraz ogólnej procedury działania.



Rys. C.1 Schemat blokowy auto-adaptacyjnego systemu identyfikacji obciążeń z wykorzystaniem uczenia maszynowego

Użytkownik w pierwszej kolejności, korzystając z Symulatora Dynamicznej Odpowiedzi Konstrukcji Mostowej tworzy bazę danych zawierającą populację pojazdów oraz odpowiedzi konstrukcji mostowej od przejazdów pojazdów. w następnym kroku, użytkownik definiuje strukturę sieci neuronowej w odpowiedniej metodzie. Następnie trenuje, weryfikuje oraz zapisuje postęp treningu sieci neuronowej. w trakcie procesu treningu, system za pomocą dedykowanego ResponceDataPrepera, pobiera z bazy danych niezbędne informacje oraz dostosowuje je do zdefiniowanego formatu danych.
W pracy przedstawiono 3 warianty systemu:

- **Wariant 1:** System bazujący na odtworzeniu sygnału quasi-statycznego
- **Wariant 2:** System bazujący na odtworzeniu sygnału dynamicznego i quasi-statycznego
- **Wariant 3:** System bazujący na odtworzeniu tylko sygnału dynamicznego bez znajomości funkcji wpływu.

Schematyczna struktura poszczególnych sieci została zaprezentowana w odpowiednich podrozdziałach tj. 5.2, 5.3 ,5.4. Dokładna struktura w postaci kodu została przedstawiona w załączonych plikach:

- AutoSIO_Wariant1.py,
- AutoSIO_Wariant2.py,
- AutoSIO_Wariant3.py.

Poza różnicą w budowie sieci neuronowej, układ programu AutoSIO jest taki sam. w C.2 omówiono działanie programu przygotowującego dane wejściowe ResponceDataPreper, natomiast w C.3 przedstawiono dokumentację programu AutoSIO.

C.2 ResponseDataPreparer

C.2.1 Opis zadania

Zadaniem programu jest przygotowanie danych wejściowych do formatu przyjmowanego przez sieci neuronowe.

Wyniki symulacji uzyskiwane na podstawie symulatora dynamicznej odpowiedzi konstrukcji mostowej oraz informacje o pojazdach są zapisane w bazie danych w formacie „*pkl*". Odpowiedzi konstrukcji mostowej należy podczytać z bazy danych, sprowadzić do odpowiedniego formatu akceptowalnego przez sieć neuronową, podzielić na zbiory uczące, walidacyjne i testowe. Dodatkowo dla pojazdów należy przygotować informacje o konfiguracji osi oraz prędkości przejazdu, jak również stworzyć funkcje wpływu poszczególnych osi.

Sieć neuronowa przyjmuje następujące dane:

- Ścieżkę do folderu, gdzie znajdują się odpowiedzi konstrukcji mostowej od przejazdu pojazdów
- Ścieżkę do folderu, gdzie znajdują się informacje o pojazdach z populacji
- Rozmiar oczekiwanej paczki informacji. Sieć neuronowa nie jest w stanie przeanalizować wszystkich danych jednocześnie z uwagi na organiczną ilość pamięci operacyjnej. z tego względu przygotowuje się mniejsze paczki danych, na których uczy się sieć.
- Flaga losowości, czyli określenie czy wybór plików ma być losowy czy deterministyczny. Sprawdzenie postępu uczenia sieci neuronowej wykonuje się na stałym zbiorze danych, niebiorących udział w uczeniu i trenowaniu sieci. Ważne jest, aby populacja kontrolna była stała, aby uzyskać miarodajne wskaźniki dokładności działania sieci. Natomiast przy trenowaniu sieci ważne, aby trenować sieć na jak największej liczbie losowych danych.

Program zwraca następujące dane:

- Ścieżki przeanalizowanych plików. Daje nam to możliwość śledzenia, na których danych system się już uczył, a które nie były jeszcze sieci prezentowane.
- Sygnały odpowiedzi konstrukcji mostowej w postaci listy sygnałów o stałych długościach.
- Konfiguracje osi oraz prędkości przejazdu w postaci listy list parametrów.
- Rzeczywiste wartości nacisku w postaci listy list nacisków poszczególnych osi – wykorzystywane są do tylko do ewaluacji dokładności działania sieci
- Funkcje wpływu osi pojazdów w postaci listy list funkcji wpływu. Każda pozycja zawiera listę funkcji wpływu dla wszystkich osi pojazdów.

Klasa **ResponseDataPreper** zawiera następujące metody:

- konstruktor **`__init__()`**,
- **`okresl_LW()`**,
- **`podczytaj_pliki_pkl()`**,
- **`przygotuj_i_interpoluj()`**,

- **ekstr_war_z_słownikow()**,
- **przeskaluj_wartosci_czasu()**,
- **wygeneruj_funkcje()**,
- **podzial_danych()**,
- **przygotuj_dane()**,

Ogólną procedurę działania programu można opisać w następujących krokach:

1. Inicjalizacja poprzez zdefiniowanie ścieżek do folderów głównych zawierających pliki w formacie „pkl" oraz ustawienie parametrów, takich jak liczba plików do wczytania i flaga wyboru (losowy lub deterministyczny). Foldery dla danych uczących oraz danych testowych są definiowane osobno.
2. Wczytywanie plików z podanych folderów, wyodrębnienie sygnałów odpowiedzi konstrukcji oraz słowników pojazdów do osobnych list.
3. Ekstrakcja wartości z słowników przez iterację przez listę słowników pojazdów i wyodrębnienie wartości odpowiadających określonym symbolom (np. prędkość, odległości między osiami, czasy pojawienia się osi na moście, czasy zjazdu osi z obiektu mostowego).
4. Przycinanie i interpolacja sygnałów w celu sprowadzenia sygnału do przedziału czasowego, w którym pojazd faktycznie znajduje się na moście, oraz definiowanie stałej długości wektora sygnałów.
5. Generowanie funkcji wpływu poprzez przeskalowanie wartości czasów wjazdu i zjazdu osi pojazdów do przedziału <0,1>, gdzie 0 oznacza moment pojawienia się pierwszej osi na moście, a 1 – zjazd ostatniej osi. Algorytm generuje funkcje wpływu dla poszczególnych osi pojazdów na podstawie tych przeskalowanych czasów.
6. Podział danych na zestawy treningowe i testowe przy użyciu funkcji **train_test_split** z biblioteki **scikit-learn**.
7. Zwrócenie danych do programu AutoSIO w uporządkowanym formacie.

C.2.2 Struktura programu

C.2.2.1 Metoda inicjująca: ***__init__(*params)***

Metoda inicjalizująca jest konstruktorem klasy ResponseDataPreparer, który przygotowuje dane dla sieci neuronowej.

Parametry wejściowe:

- **`sciezka_solvera (str)`**: Ścieżka systemowa do programu wykonawczego PhysicsSolver z pakietu Symulatora Odpowiedzi Konstrukcji Mostowej.
- **`parametry_jednostkowe (dict)`**: Parametry jednostkowe symulacji dla wyznaczenia funkcji wpływu od obciążenia jednostkowego.

Poszczególne kroki:

1. Przypisuje **`sciezka_solvera`** do **`self.sciezka_solvera`**.
2. Wstępna inicjalizacja **`self.Linia_Wpływu`** na **`None`**.
3. Wywołuje **`okresl_LW(parametry_jednostkowe)`**, która na podstawie parametrów jednostkowych tworzy teoretyczną funkcję wpływu. Docelowo na istniejących obiekcie, konieczne jest zastosowanie Linii Wpływu Mierzonej Wielkości na podstawie próbnego obciążenia.

C.2.2.2 Metoda: ***okresl_LW(*params)***

Określa linię wpływu na podstawie podanych wartości parametrów symulacji.

Parametry wejściowe:

- **`slownik_wartosci (dict)`**: Słownik zawierający wartości parametrów symulacji.

Metoda nie zwraca bezpośrednio żadnych wartości, ale aktualizuje zmienną instancji.

Poszczególne kroki:

1. Dodanie do ścieżki systemowej modułu PhysicsSolver z pakietu Symulator Dynamicznej Odpowiedzi Konstrukcji Mostowej.
2. Dynamiczne ładowanie modułu solvera.
3. Tworzenie instancji klasy **`PhysicsSolver`**.
4. Wykonywanie symulacji odpowiedzi dynamicznej, czyli uzyskanie funkcji wpływu jako odpowiedzi konstrukcji mostowej na przejazd siły z quasi-statyczną prędkością.
5. Aktualizacja **`self.Linia_Wpływu`**.

C.2.2.3 Metoda: ***podczytaj_pliki_pkl(*params)***

Podczytuje pliki formatu „*pkl*” z danych folderów, łącząc ich zawartość w listy.

Parametry wejściowe:

- **`sciezka_folderu (str)`**: Ścieżka do folderu głównego zawierającego informacje o populacji pojazdów.
- **`sciezka_z_wynikami (str)`**: Ścieżka do folderu głównego zawierającego wyniki symulacji, które są uporządkowane i trzymane w podfolderach.
- **`podfoldery (list)`**: Jest to lista podfolderów. w celu systematyzacji danych stworzono osobne foldery dla różnych sylwetek pojazdów.
- **`ilosc (int)`**: Maksymalna liczba plików do wczytania z każdego podfolderu. Każdy plik zawiera określoną liczbę symulacji, np. 1000. w zależności od mocy obliczeniowej komputera możemy analizować większą liczbę symulacji w tym samym czasie.
- **`losowosc (bool)`**: Flaga określająca, czy wybór plików ma być losowy czy deterministyczny.

Produkty wyjściowe:

- **`foldery (list)`**: Lista ścieżek do wczytanych plików.
- **`slowniki_danych (list)`**: Lista słowników pojazdów wczytanych z plików w formacie „*pkl*”.
- **`wyniki_symulacji (list)`**: Lista wyników symulacji dynamicznej odpowiedzi konstrukcji wczytanych z plików w formacie „*pkl*”.
- **`zmienne_czasowe (list)`**: Lista zmiennych czasowych dla wczytanych symulacji w formacie „*pkl*”.

Poszczególne kroki:

1. Inicjalizacja list poprzez utworzenie pustych list wyjściowych: **wyniki_symulacji**, **foldery**, **zmienne_czasowe**, **slowniki_danych**.
2. Iteracja przez podfoldery w celu tworzenia dokładnych systemowych ścieżek dostępu do plików, sortowanie tych ścieżek oraz w zależności od flagi losowości wyboru, podczytanie określonej liczby plików.
3. Aktualizacja produktów wyjściowych poprzez wczytywanie zawartości plików w formacie **„pkl”** dla każdego wybranego pliku oraz dołączenie zawartości do odpowiedniej listy.

*C.2.2.4 Metoda: **przygotuj_i_interpoluj(*params)***

Przygotowuje i interpoluje dane konstrukcji obiektu mostowego.

Parametry wejściowe:

- **`Wyniki_symulacji (list)`**: Lista odpowiedzi konstrukcji obiektu mostowego.
- **`zmienna_czasu (list)`**: Lista zmiennych czasowych odpowiadających danym odpowiedziom.

- **`oczekiwana_dlugosc (int)`**: Oczekiwana długość interpolowanych danych.
- **`parametr_dociecia (int)`**: Liczba punktów do przycięcia z końca danych odpowiedzi i zmiennej czasowej (domyślnie 1000).

Produkty wyjściowe:

- **`list`**: Lista przeanalizowanych i interpolowanych danych.

Poszczególne kroki:

1. Inicjalizacja listy wyników poprzez utworzenie pustej listy wyjściowej.
2. Iteracja przez dane i zmienne czasowe, usuwanie z dynamicznej odpowiedzi konstrukcji mostowej wartości, gdy pojazd znajduje się poza obiektem.
3. Interpolacja danych poprzez zdefiniowanie funkcji interpolacyjnej. Ze względu na różną konfigurację osi i długości sygnałów odpowiedzi mostowej, na podstawie istniejącego sygnału interpolowany jest nowy sygnał o określonej długości oraz stałych odstępach czasowych między poszczególnymi wartościami.
4. Dodanie interpolowanych danych do listy wyników.

*C.2.2.5 Metoda: **ekstr_war_z_slownikow(*params)***

Metoda ekstrahuje wartości z listy słowników parametrów symulacji.

Parametry wejściowe:

- **`wartosci_slownikow (list)`**: Lista słowników zawierających wartości parametrów symulacji.

Produkty wyjściowe:

- **`v_list (list)`**: Lista wartości prędkości przejazdu **`v`**.
- **`A_list (list)`**: Lista list wartości odległości między osiami **`a_1, a_2, a_3, a_4`**.
- **`l_list (list)`**: Lista wartości odległości osi **`l_osi`**.
- **`P_list (list)`**: Lista list sum wartości mas **`M_i1`** i **`M_i2`** dla osi od 1 do 5.
- **`tp_list (list)`**: Lista list wartości czasów pojawienia się osi **`t_1`**, **`t_2`**, **`t_3`**, **`t_4`**, **`t_5`**.
- **`tk_list (list)`**: Lista list wartości czasów zjazdu osi **`tk_1`**, **`tk_2`**, **`tk_3`**, **`tk_4`**, **`tk_5`**.

Poszczególne kroki:

1. Inicjalizacja list poprzez utworzenie pustych list: **v_list**, **A_list**, **l_list**, **P_list**, **tp_list**, **tk_list**.
2. Iteracja przez słowniki wartości:
    a. Wyodrębnienie i dodanie wartości **v** do **v_list**.
    b. Wyodrębnienie i dodanie wartości **a_1**, **a_2**, **a_3**, **a_4** do **A_list**.
    c. Wyodrębnienie i dodanie wartości **l_osi** do **l_list**.
    d. Wyodrębnienie i sumowanie wartości **M_i1** i **M_i2** dla osi od 1 do 5, dodanie sumy do **P_list**.

e. Wyodrębnienie i dodanie wartości czasów pojawienia się osi **t_1**, **t_2**, **t_3**, **t_4**, **t_5** do **tp_list**.
f. Wyodrębnienie i dodanie wartości czasów zjazdu osi **tk_1**, **tk_2**, **tk_3**, **tk_4**, **tk_5** do **tk_list**.

3. Zwrócenie produktów wyjściowych.

*C.2.2.6 Metoda: **przeskaluj_wartosci_czasu(*params)***

Metoda przeskalowuje wartości czasów wjazdu i zjazdu osi pojazdów do określonego przedziału $< 0,1 >$, gdzie 0 oznacza moment, kiedy pierwsza oś pojawia się na obiekcie mostowym, a 1, gdy ostatnia oś zjeżdża z niego.

Parametry wejściowe:

- **`czasy_wjazdu (list)`**: Lista list wartości czasów wjazdu osi na belkę.
- **`czasy_zjazdu (list)`**: Lista list wartości czasów zjazdów osi z belki.
- **`Liczba_osi (list)`**: Lista wartości określających liczbę osi w każdym zestawie danych.

Produkty wyjściowe:

- **`przeskalowane_czasy_wjazdu (list)`**: Lista list przeskalowanych wartości czasów wjazdu.
- **`przeskalowane_czasy_zjazdu (list)`**: Lista list przeskalowanych wartości czasów zjazdu.

Poszczególne kroki:

1. Inicjalizacja list wynikowych poprzez utworzenie pustych list **przeskalowane_czasy_wjazdu** i **przeskalowane_czasy_zjazdu**.
2. Iteracja przez zestawy czasów:
   a. Określenie minimalnej i maksymalnej wartości czasów wjazdu i zjazdu dla każdego zestawu danych.
   b. Skalowanie czasów wjazdu oraz zjazdu do przedziału o określonej długości.
   c. Dodanie przeskalowanych czasów do listy wynikowej.
3. Zwrócenie produktów wyjściowych.

*C.2.2.7 Metoda: **wygeneruj_funkcje(*params)***

Generuje funkcje wpływu dla zadanych czasów wjazdu i zjazdu oraz liczby osi pojazdu.

Parametry wejściowe:

- **`LW (numpy.ndarray)`**: Funkcja wpływu, która ma zostać interpolowana.
- **`czasy_wjazdu (list)`**: Lista list wartości czasów wjazdu osi (tp) dla różnych zestawów danych.

- **`czasy_zjazdu (list)`**: Lista list wartości czasów zjazdu osi (tk) dla różnych zestawów danych.
- **`l_list (list)`**: Lista liczby osi pojazdów dla różnych zestawów danych.

Produkty wyjściowe:

- **`list`**: Lista list funkcji wpływu dla poszczególnych osi pojazdu.

Poszczególne kroki:

1. Interpolacja funkcji wpływu poprzez stworzenie funkcji interpolacyjnej z wykorzystaniem scipy.interpolate.interp1d i interpolowanie funkcji wpływu do funkcji dyskretnej o odpowiedniej liczbie punktów, tak aby jej długość odpowiadała oczekiwanej przez sieć neuronową długości sygnału.
2. Inicjalizacja pustej listy wyjściowej.
3. Generowanie funkcji wpływu dla każdej osi pojazdu:
    a. Przyjęcie, że funkcja wpływu w całej swojej dziedzinie ma wartość 0.
    b. Określenie czasu wjazdu oraz czasu zjazdu danej osi.
    c. Interpolacja funkcji wpływu pomiędzy określonymi momentami czasowymi.
    d. Dodanie wygenerowanej funkcji wpływu do listy wyjściowej.
4. Zwrócenie listy wyjściowej.

*C.2.2.8 Metoda: **podzial_danych(*params)***

Metoda dzieli dane na zestawy treningowe i testowe.

Parametry wejściowe:

- **`U (list or numpy.ndarray)`**: Dane wejściowe (sygnały) do podziału.
- **`parametry_AI (list or numpy.ndarray)`**: Parametry do podziału.
- **`Ps (list or numpy.ndarray)`**: Naciski do podziału.
- **`LWs (list or numpy.ndarray)`**: Linie wpływu do podziału.

Produkty wyjściowe:

- **`sygnal_trening (numpy.ndarray)`**: Zestaw treningowy danych wejściowych - sygnały odpowiedzi konstrukcji mostowej.
- **`sygnal_test (numpy.ndarray)`**: Zestaw testowy danych - sygnały odpowiedzi konstrukcji mostowej.
- **`av_trening (numpy.ndarray)`**: Zestaw treningowy parametrów.
- **`av_test (numpy.ndarray)`**: Zestaw testowy parametrów.
- **`naciski_trening (numpy.ndarray)`**: Zestaw treningowy nacisków.
- **`naciski_test (numpy.ndarray)`**: Zestaw testowy nacisków.
- **`FW_trening (numpy.ndarray)`**: Zestaw treningowy linii wpływu.
- **`FW_test (numpy.ndarray)`**: Zestaw testowy linii wpływu.

Poszczególne kroki:

1. Podział danych wejściowych poprzez użycie funkcji **train_test_split** z biblioteki **scikit-learn** do podziału danych na zestawy treningowe i testowe (**test_size=0.2**, **random_state=liczba_stała_w_funkcji**). Parametr **test_size** określa podział w proporcji 8:2 dla danych treningowych i walidacyjnych, a **random_state** zapewnia stałe ziarno losowości, co jest istotne, gdy metoda jest wywoływana niezależnie dla różnych list, takich jak sygnały odpowiedzi, parametry pojazdu, wartości nacisków i funkcje wpływu. **Liczba_stała_w_funkcji** może być losowym parametrem zdefiniowanym przed wywołaniem metody **train_test_split**.
2. Konwersja danych do typu **float64**, który jest formatem oczekiwanym przez sieć neuronową.
3. Zwrócenie wyników poprzez zwrócenie zestawów treningowych i testowych dla sygnałów, parametrów, nacisków i linii wpływu.

*C.2.2.9 Metoda: **przygotuj_dane(*params)***

Metoda przygotowuje dane do treningu i testowania sieci neuronowej. Jest to główna metoda wywoływana przez program AutoSIO. Zarządza całym procesem przygotowywania danych.

Parametry wejściowe:

- **`sciezka_folderu (str)`**: Ścieżka do folderu głównego zawierającego podfoldery z plikami w formacie „*pkl*”.
- **`sciezka_rozwiazan (str)`**: Ścieżka do folderu z rozwiązaniami odpowiadającymi plikom w podfolderach.
- **`podfolder (list)`**: Lista nazw podfolderów do przetworzenia.
- **`ilosc (int)`**: Maksymalna liczba plików do wczytania z każdego podfolderu.
- **`losowosc (bool)`**: Flaga określająca, czy wybór plików ma być losowy (True) czy deterministyczny (False).

Produkty wyjściowe:

- **`sciezki_plików (list)`**: Lista ścieżek do wczytanych plików.
- **`sygnaly_trening (numpy.ndarray)`**: Zestaw treningowy sygnałów.
- **`sygnaly_test (numpy.ndarray)`**: Zestaw testowy sygnałów.
- **`av_trening (numpy.ndarray)`**: Zestaw treningowy parametrów.
- **`av_test (numpy.ndarray)`**: Zestaw testowy parametrów.
- **`Naciski_trening (numpy.ndarray)`**: Zestaw treningowy nacisków.
- **`Naciski_test (numpy.ndarray)`**: Zestaw testowy nacisków.
- **`FW_trening (numpy.ndarray)`**: Zestaw treningowy funkcji wpływu.
- **`FW_test (numpy.ndarray)`**: Zestaw testowy funkcji wpływu.

Poszczególne kroki:

1. Wczytanie plików w formacie „**pkl**” poprzez wywołanie metody **podczytaj_pliki_pkl** w celu wczytania plików w formacie „**pkl**”.
2. Przycinanie i interpolacja sygnałów poprzez wywołanie metody **przygotuj_i_interpoluj** w celu przycięcia i interpolacji sygnałów.
3. Ekstrakcja wartości z słowników poprzez wywołanie metody **ekstr_war_z_slownikow** w celu wyodrębnienia wartości z listy słowników parametrów pojazdów.
4. Przeskalowanie wartości czasów wjazdu i zjazdu poprzez wywołanie metody **przeskaluj_wartosci_czasu** w celu przeskalowania wartości czasów wjazdu i zjazdu osi pojazdów.
5. Generowanie funkcji wpływu poprzez wywołanie metody **wygeneruj_funkcje** w celu wygenerowania funkcji wpływu dla zadanych czasów wjazdu i zjazdu oraz liczby osi pojazdów.
6. Podział danych na zestawy treningowe i testowe poprzez wywołanie metody **podzial_danych** w celu podziału danych na zestawy treningowe i testowe.
7. Zwrócenie wyników poprzez zwrócenie krotki zawierającej ścieżki plików, zestawy treningowe i testowe sygnałów, parametrów, nacisków i funkcji wpływu.

C.3 AutoSIO

C.3.1 Struktura programu

Zadaniem programu jest określenie nacisków na podstawie dynamicznej odpowiedzi obiektu mostowego. Procedury programu dla wszystkich wariantów systemu są takie same. Różnica między systemami polega jedynie na różnej zaimplementowanej architekturze sieci neuronowej.
System, wykorzystujący sieci neuronowe, przyjmuje następujące dane:

- Sygnał odpowiedzi konstrukcji mostowej jako wektor o stałej długości,
- Wektor informacji o przejeździe, który zawiera pięć informacji tj. prędkość w m/s oraz odległości między osiami pojazdu,
- Tablica funkcji wpływu zawierająca maksymalnie 5 wektorów. Każdy wektor opisuje lokalizację oraz teoretyczny wpływ wybranej osi na quasi-statyczną odpowiedź konstrukcji mostowej,

Produktem sieci neuronowej jest:

- Odtworzony sygnał wejściowy,
- Wartości nacisków na poszczególne osie (wartości wewnętrznych neuronów sieci),

Klasa **AutoSIO** zawiera następujące metody:

- konstruktor **`__init__()`**,
- **`zwroc_data_preparator()`**,
- **`zwroc_data_worker()`**,
- **`buduj_model()`**,
- **`podsumowanie_modelu()`**,
- **`trenuj_siec()`**,
- **`zaladuj_model()`**,
- **`weryfikuj()`**,
- **`wyznacz_roznice_oszacowan()`**,
- **`zapisz_postep()`**,

Ogólną procedurę działania programu można opisać w następujących krokach:

1. W pierwszej kolejności definiowany jest konstruktor klasy. Konstruktor przechowuje ścieżki systemowe dostępu do folderów z danymi treningowymi oraz danymi testowymi. Są to 2 osobne foldery, aby faktycznie ewaluować system na danych, na których nigdy się nie uczył. Dodatkowo konstruktor przechowuje klasę pomocniczą, **`ResponseDataPreparer`**. Konstruktor przechowuje również model sieci neuronowej.
2. Następnie tworzona jest sieć neuronowa poprzez wywołanie metody **`buduj_model()`**. w środku tej metody określona jest dokładna i indywidualna architektura sieci neuronowej dla każdego z wariantów systemu.

3. Przed rozpoczęciem treningu za pomocą klasy pomocniczej **`ResponseDataPreparer`** pobierane są dane testowe, czyli takie na których będzie ewaluowana dokładność działania systemu opartego o sieci neuronowe.
4. W odgórnie zdefiniowanej przez użytkownika pętli wykonywany jest proces uczenia oraz ewaluacji postępów uczenia sieci neuronowej. Proces ten jest wykonywany aż, do uzyskania wymaganej efektywności sieci:
    1. Losowane są różne pojazdy oraz odpowiedzi konstrukcji od przejazdów tych pojazdów.
    2. Z wykorzystaniem **`Symulatora Dynamicznej Odpowiedzi Konstrukcji Mostowej`** przygotowywany jest sygnał odpowiedzi konstrukcji oraz wektor informacji o odległościach między osiami oraz prędkości przejazdu. **`ResponseDataPreparer`** przygotowuje również funkcje wpływu osi pojazdu.
    3. Po przygotowaniu danych następuje trenowanie sieci neuronowej.
    4. Przeprowadzana jest ewaluacja dokładności działania sieci na danych testowych. Wyniki pośrednie są zapisywane w bazie danych w celach oceny procesu efektów uczenia.

C.3.2 Opis metod

C.3.2.1 Metoda inicjująca: ***__init__(*params)***

Metoda inicjalizująca jest konstruktorem klasy AutoSIO, który przygotowuje obiekt do pracy z danymi wejściowymi dotyczącymi kształtów sygnałów, parametrów wejściowych oraz funkcji wpływu.
Parametry wejściowe:

- **`ksztalt_sygnalu_wejsciowego`**: kształt sygnału wejściowego.
- **`ksztalt_param_wej`**: kształt parametru wejściowego, który zawiera informacje o prędkości przejazdu oraz konfiguracji między osiami
- **`ksztalt_funkcji_wplywu`**: kształt funkcji wpływu, tablica zawierająca wektory funkcji wpływu dla osi pojazdu.
- **`funkcje_wplywu`**: funkcje wpływu opisujące lokalizację oraz teoretyczny wpływ wybranej osi na quasi-statyczną odpowiedź konstrukcji mostowej.

Metoda nie zwraca bezpośrednio żadnych wartości, ale inicjalizuje atrybuty instancji klasy.
Poszczególne kroki:

1. Inicjalizacja atrybutów instancji i przypisanie wartości przekazanych parametrów **`ksztalt_sygnalu_wejsciowego`**, **`ksztalt_param_wej`**, **`ksztalt_funkcji_wplywu`** do odpowiednich atrybutów instancji.
2. Ustawienie ścieżki systemowej do Symulatora Odpowiedzi Konstrukcji Mostowej oraz programu pomocniczego **`ResponseDataPreper`**, danych pojazdów, wyników symulacji oraz danych testowych.
3. Inicjalizacja zmiennej na model sieci neuronowej. Wstępnie jest to wartość **`None`**, ale po wywołaniu funkcji buduj_model() zostanie zapisany w niej model sieci.

*C.3.2.2 Metoda: **zwroc_data_preparator (*params)***

Metoda importuje i zwraca klasę **`ResponseDataPreparator`** z modułu znajdującego się pod podaną ścieżką.

Parametry wejściowe:

- **`sciezka_modulu`**: Ścieżka systemowa do pliku modułu zawierającego klasę **`ResponseDataPreparator`**.

Produkty wyjściowe:

- **`ResponseDataPreparator`**: Zaimportowana z podanego modułu klasa.

Poszczególne kroki:

1. Import klasy **`ResponseDataPreparator`** z pliku znajdującego się pod podaną ścieżką systemową.

*C.3.2.3 Metoda: **zwroc_data_worker (*params)***

Metoda importuje i zwraca klasę **`Worker`** z programu Symulator Dynamicznej Odpowiedzi Konstrukcji Mostowej znajdującej się pod podaną ścieżką.

Parametry wejściowe:

- **`sciezka_modulu`**: Ścieżka systemowa do pliku zawierającego klasę **`Worker`**.

Produkty wyjściowe:

- **`Worker`**: Zaimportowana klasa **`Worker`**.

Poszczególne kroki:

1. Import klasy **`Worker`** z modułu znajdującego się pod podaną ścieżką i zwrócenie jej.

*C.3.2.4 Metoda: **buduj_model (*params)***

Metoda buduje model sieci neuronowej do przetwarzania sygnałów i parametrów wejściowych oraz generuje wyjściowe sygnały statyczne.

Metoda nie przyjmuje parametrów. Cała architektura sieci jest definiowana ręcznie przez autora programu. Dokładna struktura sieci jest przedstawiona w załącznikach, odpowiednio:

- AutoSIO1_PhD.py
- AutoSIO2_PhD.py
- AutoSIO3_PhD.py

Produkty wyjściowe:

- **`Model`**: Skompilowany model sieci neuronowej.

Poszczególne kroki:

1. Deklaracja kształtów wejść do sieci neuronowej. Deklarowane są oczekiwane przez sieć neuronową kształty wejścia takie jak, sygnał odpowiedzi konstrukcji mostowej, kształt wektora

parametrów pojazdów oraz kształt funkcji wpływu. Zastosowano technologię sieci w postaci Autodekodera, określono więc, że sieć neuronowa będzie analizować cały sygnał jednocześnie. Założono, że sygnał będzie skalowany do odpowiedniej długości tj. 2000 jednostek.

   a. Sygnał odpowiedzi konstrukcji mostowej jest więc wektorem o stałej długości. Nic nie stoi jednak na przeszkodzie, aby w przyszłości analizować większą liczbę sygnałów z obiektu mostowego.
   b. Kształt wektora opisującego parametry pojazdu jest stały. Wstępnie założono na obiekcie występowanie do 5 osi jednocześnie, ale nic nie stoi na przeszkodzie, aby w kolejnych wersjach analizować większą liczbę.
   c. Kształt funkcji wpływu jest zdefiniowany przez liczbę osi oraz przyjętą długość wektora sygnału. Jest to więc macierz o wymiarach 5x2000.

2. Definicja parametrów sieci takie jak funkcje aktywacji poszczególnych sztucznych neuronów, liczby filtrów dla warstw splotowych, regularyzatory oraz ograniczenia nakładane na poszczególne neurony.
3. Budowa właściwiej sieci, w zależności od wariantu sieci, implementowane są poszczególne bloki oraz warstwy sieci neuronowej.
4. Kompilacja modelu i przyjęcie optymalizatora oraz funkcji kosztu.
5. Aktualizacja modelu zapisanego w konstrukcji instancji klasy: Podczas inicjalizacji klasy, zadeklarowana jest zmienna na model oraz przypisano jej wartość `None`, czyli nic. Dopiero po wywołaniu metody pod zmienną `self.model` będzie się znajdować sieć neuronowa.

*C.3.2.5 Metoda: podsumowanie_modelu (*params)*

Metoda drukuje informacje szczegółowe o budowie sieci neuronowej. Wyszczególnia poszczególne połączenia oraz monitoruje parametry. Metoda nie przyjmuje żadnych parametrów wejściowych.
Produkty wyjściowe:

- **`str`**: Podsumowanie modelu w postaci tekstowej.

Poszczególne kroki:

1. **Wywołanie metody summary** zbudowanego modelu Keras, która drukuje szczegółowe informacje o warstwach modelu, kształtach wyjść oraz liczbie parametrów.

*C.3.2.6 Metoda: trenuj_siec (*params)*

Metoda trenuje model sieci neuronowej na danych treningowych oraz walidacyjnych.
Parametry wejściowe:

- **`epoki (int)`**: Liczba epok treningowych.
- **`sciezka_zapisu_modelu (str)`**: Ścieżka systemowa do zapisu modelu po każdej epoce.
- **`sciezki_danych (str)`**: Ścieżka systemowa do danych używanych do treningu i walidacji.
- **`sygnaly_trening (numpy.ndarray)`**: Treningowa tablica sygnałów odpowiedzi konstrukcji mostowej.
- **`sygnaly_walidacja (numpy.ndarray)`**: Walidacyjna tablica sygnałów odpowiedzi konstrukcji mostowej.
- **`param_poj_trening (numpy.ndarray)`**: Tablica parametrów pojazdów dla danych treningowych.
- **`param_poj_test (numpy.ndarray)`**: Tablica parametrów pojazdów dla danych walidacyjnych.
- **`funkcje_wplywu_trening (numpy.ndarray)`**: Tablica funkcji wpływu dla danych treningowych.
- **`funkcje_wplywu_walidacja (numpy.ndarray)`**: Tablica funkcji wpływu dla danych walidacyjnych.

Produkty wyjściowe:

- **`historia (History)`**: Obiekt historii treningu, zawierający informacje o przebiegu procesu uczenia. Zawiera w sobie funkcję kosztu dla danych treningowych jak i walidacyjnych.
- **`sciezki_danych (str)`**: Ścieżka do danych użytych do treningu i walidacji.

Poszczególne kroki:

1. Zdefiniowanie callback'ów:
    a. Redukcja prędkości uczenia poprzez redukcję **`ReduceLROnPlateau`** do monitorowania strat walidacyjnych (**`val_loss`**), redukcji prędkości uczenia o połowę po trzech epokach bez poprawy, z minimalną prędkością uczenia 0.000001.
    b. Ustawienie **`ModelCheckpoint`** do zapisywania modelu po każdej epoce, w tym wag i struktury, bez nadpisywania najlepszych wag.
2. Trening modelu poprzez wywołanie metody **`fit`** na modelu z danymi treningowymi i walidacyjnymi. Ustawienie liczby epok, wielkości partii (**`batch_size=100`**), **`shuffle=True`** oraz callbacków
3. Zwrócenie historii treningu i ścieżki do danych historycznych.

*C.3.2.7 Metoda: zaladuj_model (*params)*

Metoda ładuje model sieci neuronowej z określonej ścieżki systemowej.
Parametry wejściowe:

- **`sciezka_modelu (str)`**: Ścieżka systemowa do pliku zawierającego zapisany model.

Produkty wyjściowe:

- **`Model`**: Załadowany model Keras.

Poszczególne kroki:

1. Wywołanie funkcji **`load_model`** z podaną ścieżką do pliku modelu i zwrócenie załadowanego modelu.

*C.3.2.8 Metoda: weryfikuj (*params)*

Metoda weryfikuje model sieci neuronowej na podstawie danych treningowych.
Parametry wejściowe:

- **`sygnaly_test (numpy.ndarray)`**: Tablica sygnałów odpowiedzi konstrukcji mostowej.
- **`param_poj_test (numpy.ndarray)`**: Tablica parametrów pojazdów dla danych.
- **`naciski_test (numpy.ndarray)`**: Tablica rzeczywistych nacisków osi pojazdu.
- **`funkcje_wplywu_test (numpy.ndarray)`**: Tablica funkcji wpływu dla pojazdu.

Produkty wyjściowe:

- **`DataFrame`**: plik zawierający ewaluację modelu, w tym różnice między oszacowanymi a rzeczywistymi naciskami.

Poszczególne kroki:

1. Tworzenie jądra modelu zwracającej wyjścia **`f1`** do **`f5`**. Program odwołuje się do istniejącej, wytrenowanej sieci neuronowej, ale zamiast oczekiwanego odtworzenia sygnału wejściowego, sieć zwraca wartości neuronów szacujących naciski na osie.
2. Predykcja oszacowań nacisków na podstawie danych treningowych.
3. Wyznaczanie różnic między oszacowanymi a rzeczywistymi naciskami za pomocą metody **`wyznacz_roznice_oszacowan`**.
4. Zwrócenie **`DataFrame`** z wynikami ewaluacji.

*C.3.2.9 Metoda: wyznacz_roznice_oszacowan (*params)*

Metoda oblicza różnice między oszacowanymi a rzeczywistymi naciskami oraz tworzy **`DataFrame`** z wynikami.

Parametry wejściowe:

- **`oszacowania_naciskow (numpy.ndarray):`** Tablica oszacowanych nacisków dla każdej osi.
- **`rzeczywiste_naciski (numpy.ndarray)`**: Tablica rzeczywistych nacisków dla każdej osi.
- **`param_pojazdu (numpy.ndarray)`**: Tablica parametrów pojazdu.

Produkty wyjściowe:

- **`DataFrame`**: **`DataFrame`** zawierający obliczone różnice, rzeczywiste naciski, sumę nacisków oraz parametry pojazdu.

Poszczególne kroki:

1. Inicjalizacja bazy danych błędów poprzez stworzenie pustej listy do przechowywania wyników.
2. Obliczanie różnic dla każdego pojazdu poprzez iteracje przez rzeczywiste naciski. Dla każdej osi, obliczanie różnicy między oszacowanymi a rzeczywistymi naciskami (jeśli nacisk rzeczywisty jest niezerowy). Sumowanie całkowitych oszacowanych nacisków oraz liczba aktywnych osi.
3. Tworzenie wierszy bazy danych poprzez dodawanie różnic, rzeczywistych nacisków, sumy nacisków oraz parametrów pojazdu do listy wyników.
4. Tworzenie **DataFrame** dzięki konwersja listy wyników do **`DataFrame`** z odpowiednimi kolumnami.

*C.3.2.10 Metoda: zapisz_postep (*params)*

Metoda zapisuje postęp analizy i historii błędów do plików w formacie „*pkl*".

Parametry wejściowe:

- ***przeanalizowane_pliki (list)***: Lista przeanalizowanych plików.
- ***historia_bledu_treningowego (list)***: Historia błędów treningowych.
- ***historia_bledu_walidacyjnego (list)***: Historia błędów walidacyjnych.
- ***postep (list of DataFrame)***: Lista DataFrame zawierających błędy oszacowania nacisków dla cyklu treningowego.
- ***docelowa_sciezka_zapisu (str)***: Docelowa ścieżka systemowa do zapisu plików w formacie „*pkl*".

Poszczególne kroki:

1. Zapis ścieżek przeanalizowanych plików, historii funkcji kosztu dla danych treningowych i walidacyjnych oraz różnic oszacowania dla kolejnego cyklu treningowego