POLITECHNIKA GDAŃSKA
WYDZIAŁ MECHANICZNY

Katedra Mechaniki i Mechatroniki

CEZARY BUCHHOLZ

TECHNIKI PROJEKTOWANIA MECHATRONICZNEGO SYSTEMU **NADZOROWANIA RUCHU TRÓJKOŁOWEJ** PLATFORMY MOBILNEJ Z ZASTOSOWANIEM ENERGETYCZNEGO **WSKAŹNIKA JAKOŚCI**

ROZPRAWA DOKTORSKA

PROMOTOR:
dr hab. inż. Krzysztof J. Kaliński, prof. nadzw. PG

GDAŃSK 2012

Pracę dedykuje Babci, Mamie oraz Profesorowi,
bez których praca by nie powstała.

Spis treści

Wykaz skrótów. Definicje

Skrót	Wyjaśnienie
ABS	– Anti-lock Breaking System
ALU	– Arithmetic Logic Unit
CD&S	– Design Control and Simulation
cpr	– couts per turn
CPU	– Central Processing Unit
cRIO	– compact Reconfigurable Inputs Outputs
DC	– Direct Current
DP	– Dynamic Position
FPGA	– Field Programable Gate Array
GSM	– Global System for Mobile Communications
HILS	– Hardware in the Loop Simulations
HMI	– Human Machine Interface
HOST	– Komputer programisty (główny)
ms	– milisekunda
NI	– National Instruments
OSI	– Open Systems Interconnection
PWM	– Pulse Width Modulation
RT	– Real Time
TCP/IP	– Transmission Control Protocol/Internet Protocol
UMTS	– Universal Mobile Telecommunication Systems
WLAN	– Wireless Local Area Network
VHDL	– Very high speed integrated circuit Hardware Description Language

Optymalizacja energetyczna – inaczej optymalne warunki pracy układu (zastosowanego sterownika czasu rzeczywistego cRIO-9076 oraz zintegrowanego procesora 400 MHz), w którym poprzez mechatroniczny (w drodze badań doświadczalnych) dobór kroku całkowania równań różniczkowych (częstotliwość generowania sygnału sterującego) spełnia się warunek determinizmu czasowego sterownika (występuje znak równości pomiędzy odczytem stanu dwóch wewnętrznych zegarów sterownika – zegara symulacji/wykonania programu oraz zegara rzeczywistego), a występujące błędy numeryczne osiągają wartości satysfakcjonujące. W niniejszej pracy, dla danej jednostki CPU zwiększenie kroku całkowania z $\Delta t=0{,}001$ s do $\Delta t=0{,}005$ s spowodowało spełnienia warunku determinizmu czasowego (dla $\Delta t=0{,}001$ s, warunek ten nie jest dotrzymany) przy niezauważalnym wzroście błędów numerycznych. Należy zaznaczyć, że spełnienie warunku determinizmu czasowego dla zastosowanego algorytmu *on-line* jest krytyczne. Brak dotrzymania reżimu czasowego objawia się wydłużonym czasem obliczeń, a także – zwiększonym poborem prądu (większa liczba zasobów jednostki arytmetyczno-logicznej ALU tranzystorów bierze udział w obliczeniach) i tym samym – szybszym zużyciem energii elektrycznej, w tym przypadku – zmagazynowanej w bateriach.

Optymalizacja – w pracy rozumiana, jako szereg działań (usuwanie błędów, poprawa przepływu sygnałów, odciążenie jednostki CPU oraz pamięci operacyjnej sterownika, realizacja części kodu algorytmu w układzie FPGA, dobór współczynników macierzy **R** oraz **Q**) usprawniających wykonanie algorytmu, a w konsekwencji zapewnienie skutecznego nadzorowania integralnego systemu trójkołowej platformy mobilnej.

Deployment – proces zapisu (wraz z czynnościami poprzedzającymi ostateczną komendę *RUN*, rozpoczynającą rozpatrywany proces) powstałego oprogramowania do fizycznego sterownika (w tym przypadku - do systemu nadzorowania trójkołowej platformy mobilnej). W rozpatrywanym środowisku LabVIEW proces zapisu może być poprzedzony szeregiem czynności umożliwiających wybór opcji pozwalających na ustawienie specyficznych parametrów pracy sterownika czy (w trakcie zapisu a później wykonania kodu programu) ewentualną diagnostykę układu, usuwaniu błędów oraz monitorowaniu obciążenia jednostki CPU.

Operacyjność – możliwość wykonywania przez urządzenie (tj. platformę mobilną) zadanej czynności (lub grupy czynności) w ciągu danej jednostki czasu. Zazwyczaj, pożądana jest długa operacyjność tak, aby np. wykonanie jednej czynności (w ciągu 1 godziny) nie było związane z kilkukrotną potrzebą uzupełnienia zasobów energetycznych, np. baterii lub wymiany elementu, który na skutek użytkowania szybko ulega uszkodzeniu. Przez operacyjność (w przypadku platform mobilnych) należy również rozumieć pewną uniwersalność platformy w zakresie wykonywania różnych czynności w różnych warunkach terenowych, bądź atmosferycznych np. zdolność pokonywania stromych wzniesień.

Determinizm czasowy – cecha systemu sterowania gwarantująca wykonanie zadania w określonym, ustalonym przedziale czasowym.

Walidacja końcowa – integracja oraz uruchomienie na platformie docelowej systemu sterowania wraz z działaniami mającymi na celu potwierdzenie funkcjonowania zgodnie z przyjętymi założeniami.

1. Wstęp

W pracy omówiono techniki projektowania mechatronicznego (technika wirtualnego prototypowania, technika HILS oraz szybkie prototypowanie na platformie docelowej) wykorzystane w procesie kreowania koncepcji, projektowania, końcowego uruchomienia oraz przeprowadzenia badań eksperymentalnych systemu nadzorowania bazującego na energetycznym wskaźniku jakości. Proponowany algorytm nadzorowania zaimplementowano na zbudowanej dla potrzeb realizacji projektu, trójkołowej platformy kołowej własnej konstrukcji.

Zaprezentowany mechatroniczny proces projektowania uwzględniający interdyscyplinarność w procesie tworzenia modelu obliczeniowego, a także równoległość przeprowadzania badań i podejmowania strategicznych decyzji, umożliwił osiągnięcie optymalnych rozwiązań w skróconym cyklu realizowanego przedsięwzięcia naukowego, przy jednoczesnym uwzględnieniu stawianych wymagań konstrukcyjnych, rynkowych, a także ekonomicznych projektu. Ponadto, dzięki wykorzystaniu odpowiedniego środowiska informatycznego (tutaj LabVIEW wspomagane przez Maple), przeprowadzono szybką weryfikację założeń konstrukcyjnych oraz ich wstępną walidację. Należy jednak zaznaczyć, że fundamentem dla powstania pracy (realizowanego systemu nadzorowania) było środowisko LabVIEW, wraz z omówionymi w kolejnych rozdziałach modułami oprogramowania. Wybór środowiska, a także doświadczenia autora zdeterminowały fakt zastosowania określonego sterownika rozważanej platformy (cRIO), którego funkcjonalność stanowi jego integralną część oraz który jest w pełni zintegrowany z tym środowiskiem na wszystkich poziomach abstrakcji.

Zaprezentowany przebieg realizacji projektu bazujący na pryncypiach konstrukcji samojezdnych, które są szeroko stosowane w operacjach gdzie nie tylko czynnik ludzki wystawiony jest na wysokie ryzyko narażenia zdrowia i życia, ale także i tam, gdzie dokładność, wysoka efektywność oraz operacyjność brane są za czynnik decydujący. Długa operacyjność oraz niezawodność konstrukcji mobilnych wymaga realizacji wysoko wydajnych systemów zarządzania energią. Dzięki implementacji optymalnych algorytmów sterowania (tutaj energetyczny wskaźnik jakości) proces kontroli takich rozwiązań prowadzi do zmniejszenia zużycia energii, podniesienia wydajności energetycznej platformy, a także – do zwiększenia

dokładności realizowanych trajektorii ruchu oraz stabilności układu napędowego podczas przejazdu. Optymalizacja dotyczyła również poziomu wykonywania algorytmu (generowania optymalnych sygnałów sterujących). W trakcie jej realizacji autor, wykorzystując symulacje czasu rzeczywistego, dokonał podziału na część wykonywalną w procesorze Real Time oraz część wykonywaną w układzie FPGA.

Wykorzystane techniki oraz metodologia projektowania mechatronicznego umożliwiły rozwiązanie problemu optymalnego sterowania silnie nieliniowym obiektem badań (macierz bezwładności **M** oraz macierz efektów sił odśrodkowych bezwładności i efektów żyroskopowych **L** układu zależne od kąta i prędkości obrotu kierownicy). Istotnym w procesie sterowania platformą, a także i optymalizacji energetycznej rozwiązania stał się odpowiedni dobór kroku całkowania równań różniczkowych, a tym samym – wybór odpowiedniego sterownika platformy. Przeprowadzona analiza oraz proces wnioskowania ustalił wybór sterownika, który umożliwił kontrolowanie platformy w czasie rzeczywistym, a tym samym – realizację koncepcji sterowania przy wykorzystaniu algorytmu w trybie *on-line*.

Zademonstrowane symulacje badań oraz testy w czasie rzeczywistym, a także końcowa walidacja rozpatrywanego obiektu badań, potwierdziły słuszność zastosowanego podejścia w projektowaniu. Wykazano również, że proponowany energetyczny wskaźnik jakości jest efektywną metodą nadzorowania, która może być konkurencyjną alternatywą dla innych typów sterowników. Celowe wykorzystanie w prezentowanym projekcie technik projektowania mechatronicznego wpłynęło na znaczną redukcję czasu weryfikacji zadań cząstkowych oraz możliwość przebadania wielu wariantów, a także – pełną integrację przyjętego rozwiązania. Osiągnięto przy tym większe prawdopodobieństwo powodzenia realizacji zbudowanego obiektu przy równoczesnym uwzględnieniu jego optymalnej konstrukcji, niż przy zastosowaniu innych tradycyjnych (np. szeregowych) metod projektowania.

1.1. Wprowadzenie

Rosnąca złożoność maszyn oraz wymagania stawiane konstruktorom wpływają na ciągły proces poszukiwania optymalnych rozwiązań sprzyjających powstawaniu w krótkim okresie czasu prototypów i gotowych rozwiązań. Proces powstawania najpierw modelu obliczeniowego, poprzez symulację oraz prototypowanie do walidacji dzieła założeń konstrukcyjnych, musi uwzględniać podczas realizacji

poszczególnych tych etapów szeroko pojętą interdyscyplinarność nauk oraz złożoność pojęciową. Współczesną rolą inżyniera jest nie tylko tworzenie finalnych rozwiązań, ale także poszukiwanie oraz dobór optymalnych technik (metod, narzędzi) projektowania oraz składowych komponentów systemu [93].

Wysoce wyspecjalizowanymi urządzeniami są platformy mobilne, należące do klasy robotów autonomicznych [27], których konstrukcja oraz realizowane zadania wymagają często złożonego procesu projektowania tak, aby opracowana koncepcja odzwierciedlała założenia konstrukcyjne oraz była możliwa do zrealizowania przy równoczesnym zbilansowaniu zasobów ekonomicznych w trakcie projektowania, oraz energetycznych w trakcie realizacji przewidzianego zadania (realizacji algorytmu). Konstruuje się roboty proste (masowa produkcja np. robot typu kosiarka do trawy) spełniające określone zadania życia codziennego człowieka (np. wyręczające jego pracę), jak i bardzo zaawansowane (najczęściej specjalistyczne konstrukcje, tworzone pod kątem jednostkowych zamówień). Zarówno pod względem konstrukcyjnym (wyspecjalizowana konstrukcja dedykowana konkretnemu zadaniu oraz odporna na surowe warunki klimatyczne), jak i systemu sterowania (ze względu na częstą niemożliwość komunikacji w trybie *on-line* z obiektem), pożądana jest pełna inteligencja/autonomiczność robota. Niezawodność, jak też – jego długa operacyjność wiąże się z bezpośrednio z optymalizacją energetyczną systemu. Niewątpliwie, do specjalistycznych robotów należy zaliczyć te, które są przeznaczone do realizacji zadań podczas eksploracji kosmosu, bądź też zadań podmorskich (typu spawanie rur, łączenie elementów). Jest też dość powszechnym stosowanie specjalistycznych robotów w służbach rządowych oraz innych organizacjach, w których często jednostkowy czynnik ekonomiczny konstrukcji takiego robota jest sprawą drugorzędną. Podczas realizacji robotów specjalistycznych [20], ze względu na charakter zadań oraz liczbę zaangażowanych w projekt inżynierów różnych specjalności, rozważa się szereg wariantów, pomysłów oraz idei. Często na skutek błędnej interpretacji wyników, pomyłek, bądź nowych wymagań dochodzi do zmian opracowanej koncepcji, powrotu do pierwotnych ustaleń, bądź rezygnacji z proponowanego rozwiązania. Realizacja takiego projektu, wymaga stosowania odpornych oraz elastycznych reguł projektowania tak, aby pomimo istnienia licznych sprzężeń zwrotnych, możliwe było zachowanie ciągłości realizacji zadania (budowy mechatronicznego urządzania).

Stawiane wymagania dotyczące pracy w interdyscyplinarnej hierarchii przedsięwzięcia inżynierskiego, a także złożoność systemów i opisu modelu matematycznego, kreują potrzebę podejścia mechatronicznego w procesie projektowania tak, aby parametry oraz zmienne modelowanych zjawisk można było, na bieżąco w trakcie przeprowadzonych badań, weryfikować oraz dobierać według naukowych reguł i zasad [29, 41 45].

Obecnie, w wiodących biurach projektowych można zaobserwować tendencję do coraz częstszego sięgania (w trakcie realizacji projektu) po techniki projektowania mechatronicznego (m.in. wirtualnego prototypowania oraz ostatnio bardzo popularną technikę HILS), jako alternatywy dla projektowania szeregowego, pozbawionego tak pożądanej elastyczności.

Niezwykle istotnym zagadnieniem, wynikającym wprost z zastosowanej metodyki projektowania jest dobór odpowiedniego narzędzia informatycznego, które w pełni integruje w sobie możliwości realizacji stanowisk badawczych, a także tworzenia wirtualnych prototypów i symulacji w czasie rzeczywistym. Możliwości takie oferuje dość popularne środowisko LabVIEW [7, 66], które wraz z dostępnymi modułami oprogramowania dodatkowego, doskonale wpisuje się w tendencję mechatronicznego podejścia do realizacji projektu. Rozpatrywane środowisko, wraz z modułami Real Time, Control Design & Simulations oraz FPGA stało się fundamentem realizacji niniejszej pracy.

Pomimo konieczności budowy wyspecjalizowanych obiektów, obserwuje się trend do coraz częstszego przejmowania realizacji funkcji danego urządzenia przez wyspecjalizowanie oprogramowanie (algorytm) [19, 42]. Tendencja ta ściśle odzwierciedla fakt powstawania coraz doskonalszego oprogramowania, a także – jednostek elektronicznych, gdzie dany algorytm może być wykonywany. Niewątpliwie, istotnym tutaj elementem jest fakt istnienia ogromnych, potencjalnych możliwości testowania wielu wariantów oraz koncepcji, bez konieczności ponoszenia większych kosztów na budowę urządzeń fizycznych.

Zauważalnym trendem w mechatronice jest stosowanie sterowników czasu rzeczywistego oraz układów FPGA [82, 83]. Determinizm czasowy związany jest ściśle z koniecznością realizowania algorytmów w trybie *on-line* oraz podejmowania decyzji w ściśle określonej czasoprzestrzeni (tj. szeroko pojętego sterowania). Systemy podejmowania szybkich decyzji (a tym samym, optymalizacji koncepcji systemu sterowania) wymagają przeniesienia realizacji algorytmów bądź jego części

do układów FPGA, gdzie czas zadania może być realizowany z częstotliwością rzędu gigaherców (GHz).

Algorytmy realizowane w robotyce mobilnej zależą w głównej mierze od przyjętej koncepcji sterowania oraz od typu zadania realizowanego przez obiekt. Wzrost mocy obliczeniowej produkowanych obecnie jednostek operacyjnych CPU pozwala na tworzenie bardzo wyrafinowanych algorytmów czasu rzeczywistego, sterujących obiektem o często bardzo silnie nieliniowej i złożonej strukturze dynamicznych równań ruchu. Ze względu na chęć tworzenia w pełni inteligentnych obiektów oraz możliwości aproksymacji dowolnych odwzorowań nieliniowych, a także możliwości uczenia się w zmieniającym się otoczeniu, dużo uwagi poświęca się sztucznej inteligencji, tj. sieciom neuronowym oraz logice rozmytej [5, 19, 45]. Należy zauważyć, że pomimo ustalonej jednej tendencji w tworzeniu danego algorytmu sterowania, mobilne roboty, z uwagi na szerokie spektrum realizowanych zadań oraz zespołów wykonawczych biorących w nich udział, realizują algorytmy będące najczęściej ich hybrydami. System sterowania takich układów realizuje w jednej chwili kilka zadań sterowania jednocześnie tak, aby np. w sposób optymalny dotrzeć do celu omijając przeszkodę, przy równoczesnym procesie budowania bazy wiedzy eksperckiej dla przyszłych procesów wnioskowania. Na uwagę zwraca algorytm optymalnego wykorzystania energii (minimalizacji strat wynikających z powstałego błędu na skutek poruszania się obiektu po niewłaściwej trajektorii), będącego czynnikiem decydującym o możliwościach operacyjnych danego typu układu (tutaj platformy mobilnej) oraz jakościowych wykonania danego zadania.

Tworzenie oprogramowania, a także i realizacja całego przedsięwzięcia naukowego (w tym przypadku, budowy platformy mobilnej) odbywa się w środowiskach programistycznych, gdzie istnieją potencjalne możliwości realizacji techniki projektowania mechatronicznego. Zastosowane środowisko LabVIEW wspiera nie tylko proces tworzenia konstrukcji stosując techniki wirtualnego prototypowania oraz HILS [39], ale dzięki ogromnym możliwością sprzętowym, zintegrowanym ze środowiskiem, pozwala na zastosowanie w pełni jednolitej, równoległej metody projektowania mechatronicznego. Współpraca zastosowanego oprogramowania ze sprzętem (tj. sterowaniem czasu rzeczywistego cRIO z układem FPGA) umożliwia wydatne przyspieszenie procesu projektowania, w którym, w czasie rzeczywistym, przeprowadza się eksperymenty odwzorowujące projektowane elementy, bez konieczności ich fizycznego wytwarzania.

1.2. Cele i zakres pracy

Rosnąca popularność stosowania technik projektowania mechatronicznego, a także obserwacje związane ze zwiększającym się zapotrzebowaniem na układy czasu rzeczywistego oraz FPGA, przy jednoczesnym poszukiwaniu optymalnych technik sterowania wytwarzanym obiektem, skłoniły autora do zastosowania w praktyce (tj. realizacji systemu nadzorowania trójkołową platformą mobilną) powyższych aspektów oraz wykazania praktycznych korzyści z ich stosowania.

Realizacja zamierzenia, wymagała określenia przez autora następujących celów.

1. Wykorzystanie technik projektowania mechatronicznego (wirtualnego prototypowania, HILS oraz szybkiego prototypowania na sprzęcie docelowym) do realizacji systemu nadzorowania przy energetycznym wskaźniku jakości, a także – do jego implementacji w strukturach projektowanej równolegle trójkołowej platformy mobilnej.
2. Potwierdzenie doświadczalnie, na przykładzie zbudowanego obiektu badań własnej konstrukcji (trójkołowej platformy mobilnej), że nadzorowanie ruchu obiektu silnie nieliniowego (macierz bezwładności **M** oraz macierz efektów sił odśrodkowych bezwładności i efektów żyroskopowych **L,** zależne od kąta skrętu kierownicy) jest skuteczne przy wykorzystaniu zaimplementowanego do sterownika czasu rzeczywistego cRIO algorytmu sterowania optymalnego przy energetycznym wskaźniku jakości.
3. Wykazanie, że optymalizacja energetyczna systemu nadzorowania ruchu trójkołowej platformy mobilnej jest osiągalna poprzez mechatroniczny dobór kroku całkowania, wraz z wykorzystaniem prędkości korygujących.

Treść pracy została podzielona na 8 rozdziałów odzwierciedlających charakter projektowania mechatronicznego.

W rozdziale 1 autor dokonał wprowadzenia w zagadnienia będące tematem rozprawy. Omówił główne problemy wynikające ze stosowania robotyki mobilnej, a także zaprezentował przyjętą koncepcję realizacji badań. Autor przedstawił cele, zakres oraz tezy pracy.

Wprowadzenia w metodykę realizowanego przedsięwzięcia naukowego dokonano w rozdziale 2, gdzie autor omawia zastosowane techniki projektowania

mechatronicznego wykorzystane podczas realizacji systemu nadzorowania ruchu trójkołowej platform mobilnej.

Analiza zagadnień mechaniki układu, na bazie zdefiniowanego modelu trójkołowej platformy mobilnej, jest treścią 3 rozdziału. Korzystając ze zdefiniowanych równań różniczkowych, autor za pomocą metod numerycznych rozwiązuje odwrotne zadanie kinematyki oraz odwrotne zadanie dynamiki układu. Dla trzech wybranych trajektorii ruchu platformy mobilnej wykreślane są (drogą symulacji komputerowej) odpowiedzi układu (parametry platformy), będące podstawą weryfikacji przyjętego modelu obliczeniowego. Dodatkowo weryfikacja przyjętego rozwiązania platformy mobilnej (konstrukcji) dokonywana jest na bazie analizy mnożników Lagrange'a, w której autor sprawdza warunek dopuszczalnych sił tarcia w punktach styczności kół z podłożem.

Rozdział 4 poddaje dyskusji (drogą techniki wirtualnego prototypowania) zastosowany algorytm nadzorowania ruchu platformy mobilnej (energetyczny wskaźnik jakości). Badane są odpowiedzi układu (platformy) na pobudzenia sygnałem optymalnym, pochodzącym z realizowanego systemu nadzorowania. Dodatkowo autor porusza kwestię optymalizacji energetycznej systemu nadzorowania, a także - określa jego architekturę. Analizie podlega długość kroku całkowania równań różniczkowych, a także przeprowadzana jest dyskusja nad zdefiniowanymi prędkościami korygującymi będącymi częścią realizowanej optymalizacji, oraz ostatecznego sygnału sterującego. Autor uzupełnia treść rozdziału o opracowaną metodologię optymalizacji oraz określa jej ramy.

Metoda HILS, omówiona w rozdziale 5, stanowi nowy etap w tworzeniu systemu sterowania czasu rzeczywistego. Autor, dokonując emulacji platformy mobilnej, generuje dla każdej z rozpatrywanych trajektorii ruchu odpowiedzi układu na sygnały z rzeczywistego sterownika. Rozdział szczegółowo opisuje opracowaną architekturę systemu sterowania, a także implementację do jego struktur odpowiedniego algorytmu.

Przeprowadzone badania (walidacja opracowanego rozwiązania) oraz ich analiza są treścią rozdziału 6. Autor w pierwszej części realizuje prototypowanie sterownika oraz przeprowadza implementację systemu nadzorowania w strukturze platformy mobilnej. Ostatecznie, dokonując rzeczywistych przejazdów platformy po ustalonych trasach przejazdu, weryfikowane są przyjęte tezy pracy oraz wyznaczone cele. Rozdział wzbogaca opis realizacji przedsięwzięcia naukowego (przyjęty proces

projektowania mechatronicznego), a także dane techniczne przyjętego rozwiązania platformy mobilnej.

Podsumowanie otrzymanych rezultatów wypełnia rozdział 7 pracy.

W rozdziale 8 autor proponuje wskazówki (powstałe na skutek swoich przemyśleń w trakcie realizacji pracy) do dalszych badań nad robotyką mobilną, na bazie energetycznego wskaźnika jakości (tj. będącego przedmiotem rozważań niniejszej pracy).

Pracę uzupełniają 2 załączniki opisujące wykorzystane w trakcie realizacji pracy oprogramowanie LabVIEW oraz Maple.

1.3. Tezy pracy

Tezy pracy sformułowano następująco:

1. Techniki projektowania mechatronicznego, a w szczególności HILS wraz ze środowiskiem LabVIEW Real Time, przyczyniają się do znacznego zwiększenia prawdopodobieństwa poprawnej decyzji w procesie weryfikacji projektowanego systemu nadzorowania ruchu z wykorzystaniem trójkołowej platformy mobilnej oraz prowadzą do jego optymalizacji energetycznej poprzez mechatroniczny dobór kroku całkowania wraz z zastosowanymi prędkościami korygującymi.

2. Nadzorowanie ruchu trójkołowej platformy mobilnej, jako obiektu silnie nieliniowego (macierz bezwładności **M** oraz macierz efektów sił odśrodkowych bezwładności i efektów żyroskopowych **L** zależne od kąta skrętu kierownicy) jest skuteczne przy wykorzystaniu zaimplementowanego do sterownika czasu rzeczywistego cRIO algorytmu sterowania optymalnego przy energetycznym wskaźniku jakości.

Prawdziwość powyżej sformułowanych tez pracy została udowodniona poprzez:

– realizację testów czasu rzeczywistego, w których sprawdzano odpowiedź emulowanej platformy mobilnej na zadane optymalne sygnały sterujące, generowane przez rzeczywisty sterownik. Kryterium oceny była weryfikacja

otrzymanych wyników z rezultatami uzyskanymi w trakcie realizacji wirtualnego prototypowania. W trakcie realizacji badań z użyciem metody HILS, niezwykle istotnym stało się ustalenie optymalnej częstotliwości generowania sygnału sterującego, w którym osiągnięto kompromis pomiędzy występującymi błędami, a energetyczną sprawnością systemu sterowania;

- eksperymentalne potwierdzenie skuteczności sterowania obiektem silnie nieliniowym przy energetycznym wskaźniku jakości, poprzez analizę porównawczą otrzymanych rezultatów (zrealizowanych trajektorii oraz parametrów platformy) przejazdu platformy mobilnej dla zadanych trajektorii i wartości wynikających z przeprowadzonych w pracy symulacji komputerowych;
- eksperymentalne opracowanie metody doboru kroku całkowania, w którym, zapewnia się optymalną pracę sterownika (czas zegara systemu czasu rzeczywistego równy czasowi zegara symulacji), przy równoczesnym stosowaniu prędkości korygującej dla danego kroku całkowania, neutralizującej powstały błąd wyznaczania bieżących parametrów trajektorii ruchu.

1.4. Przegląd literatury

Przegląd literatury został przeprowadzony zgodnie z konwencją pracy oraz realizowanych badań. W pierwszej fazie autor kieruje swoją uwagę na publikacje poruszające temat mechatroniki, a ściślej – budowy robotów, stosowanych technik, w których ukazuje się naturę (złożoność) prowadzenia multidyscyplinarnego projektu. W dalszej kolejności autor zapoznaje się z konstrukcjami robotów mobilnych, a także – z metodyką tworzenia ich modelu obliczeniowego, bazującego na opisie kinematyki oraz dynamiki układu). Sporo uwagi autor poświęca procesowi sterowania platformy mobilnej. Rozpoznaje ogólne koncepcje, algorytmy oraz strategie sterowania takimi strukturami. Doświadczenia autora implikują stosowanie środowiska LabVIEW, jako platformy realizacji projektu mechatronicznego. Jednakże, dość nowoczesne podejście do rozwiązania problemu sterowania platformą powoduje konieczność zgłębienia zagadnień o systemach czasu rzeczywistego oraz rozwiązaniach bazujących na układach FPGA. Kolejnym zagadnieniem wziętym pod uwagę jest technika HILS projektowania mechatronicznego, której efekty ostatecznie weryfikują przyjętą

koncepcję systemu sterowania oraz rozwiązań sprzętowych. Rozważania na temat systemu sterowania kierują autora w stronę analizy metod optymalizacji układów, a także nowoczesnych systemów energetycznych.

W pracy [24] autorzy, na przykładzie projektu maszyny synchronicznej, w syntetyczny sposób omawiają zagadnienia związane z mechatroniką, jako elastycznym podejściem do projektowania. Użytecznym jest prezentacja procedury tworzenia hierarchicznego systemu wzajemnych interakcji oraz sprzężeń zwrotnych, pozwalających na rozwiązanie dowolnego problemu inżynierskiego na różnym poziomie szczegółowości. Kolejną, bardzo ciekawą pozycją definiującą mechatronikę i ukazująca interpretację jej twórców (Japonia), a także – szeroką pojętą integrację z edukacją oraz przemysłem jest publikacja [29]. Autor, bazując na licznych przykładach, definiuje zakres mechatroniki oraz podkreśla dużą złożoności takich systemów. Na uwagę zwraca pozycja [82], w której autor analizuje charakterystyczne techniki stosowane w projektowaniu mechatronicznym, a także na przykładzie robota równoległego omawia procedurę implementacji algorytmu oraz budowę systemu sterowania bazującego na układzie FPGA. Publikacje [20, 45] w precyzyjny sposób omawiają proces mechatronicznego projektowania robota inspekcyjnego. Autorzy poprzez głęboką dyskusję założeń i wymagań konstrukcyjnych, przeprowadzają proces kompleksowego modelowania, wnikliwej analizy oraz symulacji realizowanego przedsięwzięcia mechatronicznego.

Problematyka dotycząca zagadnień kinematyki oraz dynamiki rozważanego układu jest przedmiotem wielu publikacji. Wyczerpujące podejście do modelowania, identyfikacji oraz sterowania robotyką mobilną zostało zaprezentowane przez autorów publikacji [12 – 20, 27, 88, 99, 100]. Autorzy prac przedstawili w publikacjach wyniki wieloletnich prac naukowo-badawczych. W szczegółowy sposób pokazano metodykę tworzenia modeli dwu-, trój- i czterokołowych platform mobilnych. Zaprezentowano różne strategie sterowania, bazujące na logice rozmytej, sieciach neuronowych czy sterowaniu adaptacyjnym. Wartym odnotowania jest fakt prezentacji środowiska Maple wspomagającego proces projektowania poprzez symboliczne generowanie równań kinematyki oraz dynamiki modelu ruchu robota, a także jego symulację. Obszerniejsze omówienie problematyki modelowania symbolicznego omawiane jest w pracach [1, 60, 81, 94].

Szersze ujęcie nowoczesnej teorii sterowania, obejmujące sterowanie w warunkach niepewności, sterowanie kompleksami operacji oraz zastosowanie

sztucznej inteligencji, zostało zaprezentowane w pozycji [5]. Publikację uzupełnia literatura poświęcona algorytmom zastosowanym przy realizacji konkretnych strategii sterowania. Jedną z nich jest strategia bazująca na zadawaniu obiektowi badań takich momentów, aby robot poruszał się bez poślizgów. Autor publikacji [46, 47] rozprawia na konkretnym przykładzie robota SOLERO (czterokołowego robota terenowego) zalety tej metody z dość pozytywnym skutkiem. W kolejnej publikacji [95] autorzy omówili natomiast strategię kontroli momentu napędowego. Publikacja [32] prezentuje zalety algorytmu czasu rzeczywistego estymującego krytyczne parametry terenu, w którym porusza się robot mobilny, a tym samym – przyczyniającego się do wzrostu jego mobilności.

Robotyka mobilna dotyczy w dużej mierze eksploracji kosmosu, gdzie oprócz potrzeby zapewnienia optymalnego systemu sterowania (algorytmu) istnieje również wymóg umożliwienia poruszania się takiego obiektu w trudnym terenie. Warunek ten determinuje konieczność poszukiwania oraz analizy różnych koncepcji konstrukcji obiektów mogących spełnić takie wymagania. Najczęściej są to konstrukcje wielokołowe, o różnych koncepcjach zawieszenia. Ciekawe konstrukcje robotów są tematem wielu publikacji, np. CRAB [43, 91], EXOMARS [2, 44, 62], czy RCL [11].

Przyjęta koncepcja pracy, w której zaimplementowano system czasu rzeczywistego, nawiązała do obecnych trendów w realizacji metod sterowania. Postęp, jaki dokonał się w dziedzinie narzędzi informatycznych oraz możliwości sprzętowych, umożliwia stosowanie rozwiązań czasu rzeczywistego, w których analiza sygnałów wejściowych oraz proces decyzyjny systemu sterowania wykonywany jest w trybie *on-line*. Publikacje [66, 75, 82] szeroko dyskutują realizację takich systemów. Na uwagę zasługuje pozycja [61], która definiuje główne warunki oraz założenia systemu czasu rzeczywistego, a także – jego ogólną architekturę.

Autor pracy, mając do dyspozycji jednostkę czasu rzeczywistego cRIO zawierającą również programowalny układ FPGA, wykorzystał jej możliwości. Systemy czasu rzeczywistego, w których realizowane są wymagające algorytmy (ze względu na proces obliczeniowy), wymuszają stosowanie bardzo wydajnych jednostek obliczeniowych. Obserwuje się tendencje (z której autor również skorzystał), w której fragment kodu algorytmu, istotny dla zapewnienia wymaganej wydajności systemu wykonywany jest w układzie FPGA (pozostała część algorytmu wykonywana jest w części głównej sterownika, tj. procesorze Real Time). Zastosowany podział algorytmu znacząco wpływa na wydajność realizowanego procesu sterowania oraz optymalizuje

układ pod względem energetycznym. Szeroko na temat stosowania tych układów rozprawia wspomniana wcześniej pozycja [82]. Przykładami realizacji systemu sterowania robotami mobilnymi bazującymi na układach FPGA są prace [23, 83, 96].

Przyjęta metodyka projektowania oraz koncepcja systemu sterowania platformy mobilnej kierują w następnym kroku autora do poszukiwania optymalnych rozwiązań w dalszych badaniach przyczyniających się do skrócenia czasu projektowania oraz podniesienia prawdopodobieństwa podjęcia prawidłowej decyzji kończącej fazę tworzenia projektu platformy oraz architektury systemu sterowania. Obranym kierunkiem jest technika HILS [28, 39, 82] wiążąca weryfikację emulowanego modelu obiektu badań z zagadnieniami czasu rzeczywistego. Jest to technika bardzo popularna wykorzystywana podczas prowadzenia różnych projektów zarówno komercyjnych (np. testowanie sterownika Zintegrowanego Okrętowego Systemu Elektrycznego [85]) oraz czysto akademickich (np. testowanie systemu ABS [6, 53]). Inne przykłady zastosowań to testowanie sterownika bezzałogowego samolotu [10], czy testowanie robota podwodnego [63].

Jednym z końcowych etapów realizowanego przedsięwzięcia naukowego stały się (z uwagi na ograniczenia sprzętowe, a także – zamiar autora realizacji projektu optymalnego) zagadnienia optymalizacji systemu. Z uwagi na mechatroniczny charakter projektu, optymalizacja dotyczyła szeregu obszarów technicznych realizowanego przedsięwzięcia naukowego. Inspiracji autor poszukiwał w szeregu publikacjach. Oryginalnym pomysłem, rozważanym w artykule [87] jest minimalizacja zużycia energii robota mobilnego za pomocą generowania optymalnej drogi w terenie. Idea autorów polega w pierwszej kolejności na utworzeniu modelu obliczeniowego dla mapy terenu (jako wydatek energetyczny), po którym porusza się robot mobilny. Model ten bazuje na współczynniku tarcia (minimalizacja poślizgów) oraz polu grawitacyjnym (pokonywanie wzniesień). W kolejnym etapie, proponowany algorytm wyszukuje dla utworzonego modelu optymalną drogę pomiędzy dwoma punktami przy z góry zdefiniowanym przez użytkownika błędzie. Kolejna, ciekawa propozycja rozwiązująca problem optymalizacji została przedstawiona w artykule [98], w którym autorzy rozważają różne prędkości przejazdu robota, ściśle korespondujące z czasem podejmowania decyzji przez CPU. Badania przeprowadzono dla różnych rozkładów tych wartości (rozkład Gaussa, rozkład wykładniczy oraz ich warianty). Zaskakujące są rezultaty, które wskazują na optymalne wykorzystanie energii, gdy wartości te są stałe i niezmienne w całym cyklu przejazdu platformy.

Powyższy artykuł ściśle omawia problem optymalizacji, którego źródła należy upatrywać w systemie sterowania bądź algorytmie sterującym robotem. Z uwagi, o czym autor wspominał, na typowo mechatroniczny charakter projektu, część optymalizacji będzie w przyszłości dotyczyła (nie było to tematem pracy) nowej koncepcji systemu napędowego robota mobilnego, w którym energia kinetyczna będzie mogła być w dość łatwy sposób odzyskiwana i ponownie wykorzystana. W trakcie, gdy robot zwalnia (hamuje) energia mechaniczna ruchu robota zostaje zamieniana w układzie inwertera na energię elektryczną. Magazynem nowo wytworzonej energii są superkondensatory [54], które oprócz funkcji szybkiego obiorcy energii (funkcja magazynowania) pełnią również rolę odwrotną, tzn. funkcję szybkiego dostawcy tej energii. Istnieje szereg badań oraz projektów, w których nowoczesne systemy napędowe są testowane oraz implementowane [64, 89, 90].

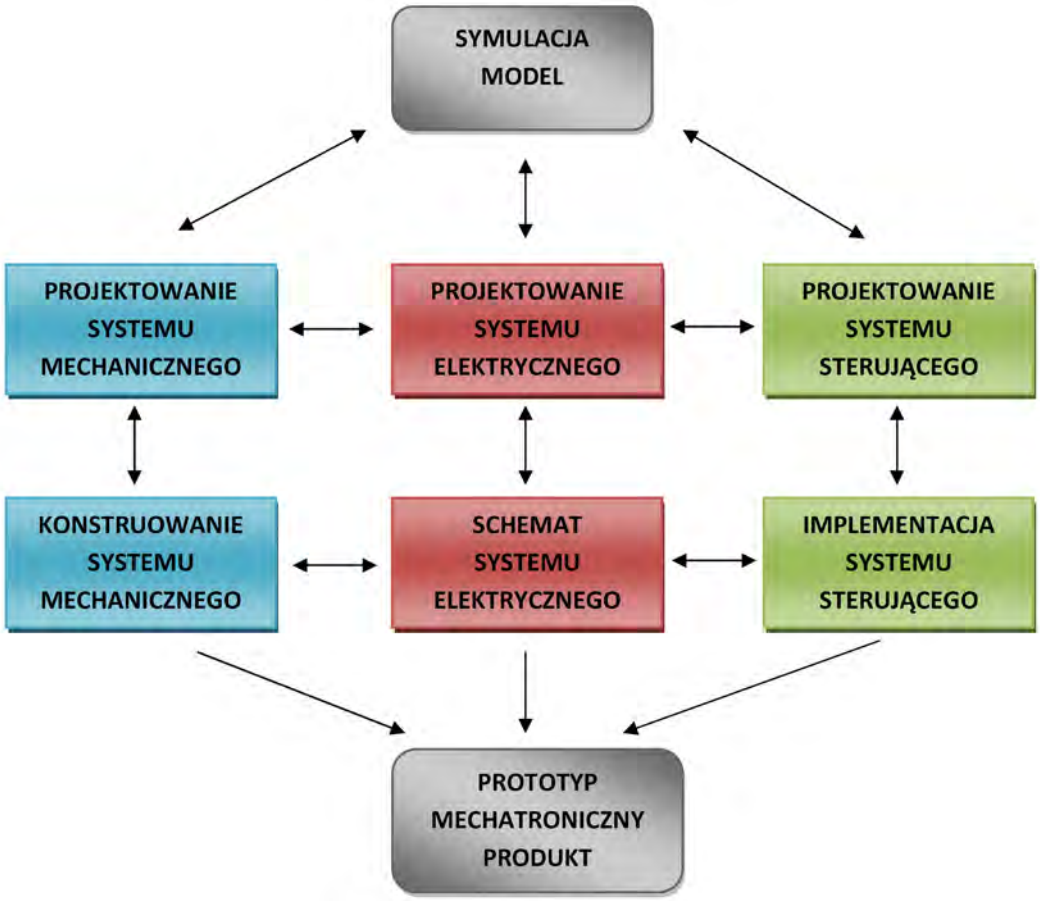
Cytowana literatura niniejszej pracy umożliwiła autorowi zapoznanie się z trendami oraz koncepcją projektowania mechatronicznego urządzeń oraz złożonych systemów, co w późniejszej fazie okazało się zagadnieniem fundamentalnym dla końcowego powodzenia realizowanego przedsięwzięcia. Opracowano oryginalny, dotychczas niespotykany system nadzorowania ruchu, powstały na bazie zastosowanego procesu projektowania mechatronicznego, spełniającego zakładane wymagania, w skróconym czasie jego trwania. Szczególnie kluczowym aspektem pracy stały się monografie poświęcone problematyce energetycznego wskaźnika jakości [9, 35], jako podstawy wyznaczania sygnału sterującego platformą mobilną, które w wydatny sposób umożliwiły poprawną realizację systemu nadzorowania oraz implementację algorytmu w struktury sterownika cRIO. Pozycje poświęcone technikom projektowania mechatronicznego znacznie przybliżyły sposób przeprowadzenia eksperymentów oraz analizy ich wyników, co w efekcie przyczyniło sią do wyboru środowiska LabVIEW, jako platformy programistycznej realizacji systemu nadzorowania. Cenną literaturą stały się pozycje poświęcone systemom czasu rzeczywistego w kontekście zastosowania techniki HILS oraz projektowania architektury systemu nadzorowania ruchu trójkołowej platformy mobilnej.

2. Techniki projektowania mechatronicznego

Wzrastająca złożoność realizowanych urządzeń oraz systemów powoduje, że powstający produkt nie jest już tylko dziełem bazującym na jednej dyscyplinie wiedzy np. mechanice. Funkcje realizowane przez projektowany obiekt, wymagają nierzadko szeregu innych urządzeń pomocniczych, których realizacja wymusza stosowanie dziedzin zupełnie odrębnych tak, aby tworzony produkt spełniał wymagania klienta oraz założenia konstrukcyjne. Powstaje produkt mechatroniczny, którego filozofia działania oraz konstrukcja jest syntetycznym dziełem realizowanego projektu. Podejście mechatroniczne przeobraża model projektowania sekwencyjnego w równoległy (rys. 2.1), gdzie w danej jednostce czasu realizowanych jest równocześnie kilka zadań cząstkowych, będących nierozerwalną częścią większego systemu oraz powiązanych za sobą na różnych poziomach funkcji i abstrakcji [29, 93]. Najczęściej, w projektowanej maszynie bądź urządzeniu, można wyróżnić część typowo mechaniczną, wykonawczą (w tym przypadku elektryczną; należy zaznaczyć, że, aktorami niektórych maszyn są układy hydrauliczne bądź pneumatyczne) oraz sterowania (przez którą należy rozumieć wyspecjalizowany sterownik wraz z częścią informatyczną w której wykonywany jest algorytm realizowanych funkcji danego obiektu). Dzięki istnieniu wzajemnych połączeń oraz zależności, każde z realizowalnych zadań cząstkowych może wymusić zmianę w innej części projektowanego systemu (zadaniu cząstkowym). Przykładowo, zmiana przełożenia w systemie układu różnicowego wymusza stosowanie (dla zachowania tej samej prędkości poruszania się obiektu) szybszy (często też większy) silnik elektryczny. Większy silnik elektryczny wymusza stosowanie innych sterowników silnika (z uwagi na pobór prądu). Podwyższona masa obiektu (często silnie nieliniowego) może powodować, w następstwie zmiany w realizacji algorytmu sterowania (np. inne wzmocnienia układu sterownika). Z powyższego przykładu rysuje się silna zależność projektowanych elementów składowych danego systemu (urządzenia). Silne powiązanie wymaga nie tylko stosowania procedur umożliwiających równolegle realizowanie zadań, ale także odpowiedniego systemu (architektury) projektu uwzględniającego częste zmiany, niekonwencjonalne metody badań oraz przeprowadzania testów, a także (co jest zdaniem autora najważniejsze) zespołu ludzi rozumiejących i stosujących pryncypia pracy zespołowej, wraz z nierozerwalną

wysoką wiedzą inżynierską. Zasadnym jest stosowanie podejścia wysoce elastycznego, w którym spodziewany proces zmian będzie niedostrzegalny (lub dostrzegalny w małym zakresie) w realizacji celu nadrzędnego.

Uelastycznienie struktury projektu oraz metodologii jego realizacji przyczynia się znacznie do podniesienia wydajności tworzonego dzieła oraz uwzględnienia pewnej wielowariantowości w proponowanym rozwiązaniu. Skrócenie czasu realizacji projektu (fazy powstawania prototypu oraz końcowej walidacji) czyni produkt bardziej konkurencyjnym na wymagającym i szybko zmieniającym się rynku.



Rys. 2.1. Schemat procesu projektowania mechatronicznego

Realizacja projektu, oprócz określenia niezbędnej architektury oraz stworzenia niezbędnych ram dla funkcjonowania wyselekcjonowanych specjalistów (składającą się w głównej mierze z mechatroników, pomijając obsługę administracyjną), wymaga przyjęcia właściwej metody/metod (w tym przypadku techniki) projektowania tak, aby możliwe było spełnienie postulatów projektowania mechatronicznego. Dotyczą one

spełnienia zasady równoległości, elastyczności, umożliwienia realizacji sprzężeń zwrotnych, skrócenia czasu projektowania oraz nierzadko możliwości wzbogacenia funkcjonalności układu lub systemu.

Do współczesnych technik projektowania mechatronicznego zastosowanych w pracy można zaliczyć:

- wirtualne prototypowanie,
- technikę HILS,
- szybkie prototypowanie na obiekcie docelowym [82].

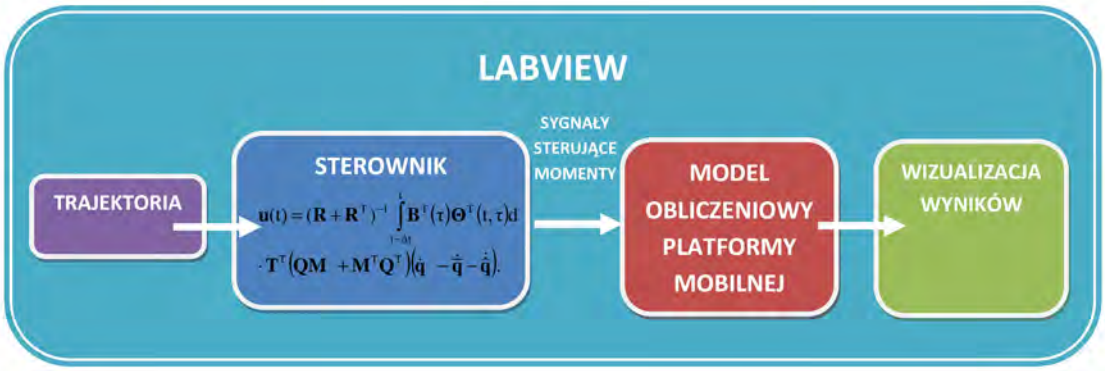
2.1. Wirtualne prototypowanie

Fundamentem założeń projektowania mechatronicznego jest symulacja komputerowa, będąca częścią platformy projektowania (narzędzia informatycznego), a służącą jako weryfikator przyjętych koncepcji (w tym przypadku, systemu nadzorowania oraz obiektu badań).

Wirtualne prototypowanie poprzez komputerowe emulowanie projektowanych obiektów umożliwia w bardzo krótkim czasie przeprowadzenie szerokiego spektrum ich symulacji, z uwzględnieniem pożądanej przez projektantów wielowariantowości.

Zamysłem wirtualnego prototypowania jest utworzenie modelu obliczeniowego projektowanego obiektu (odtwarzającego z pewną przyjętą dokładnością obiekt rzeczywisty), który w wirtualnej przestrzeni (środowisku komputerowym) zostaje poddany próbie działania, również wirtualnych (zamodelowanych, jak w tym przypadku) czynników zewnętrznych (może być to sterownik wraz z algorytmem, bądź inny sygnał wymuszający). Analizie podlega odpowiedź (uzyskana również w wirtualnej przestrzeni) takiego układu na zadane sygnały wejściowe.

W pracy zastosowano technikę wirtualnego prototypowania, której schemat został zaprezentowany na rys. 2.2. Koncepcją przeprowadzonego wirtualnego projektowania systemu nadzorowania była analiza odpowiedzi (w bloku „wizualizacja wyników”) utworzonego modelu obliczeniowego platformy mobilnej na zadane optymalne sygnały sterujące (tj. moment napędowy oraz kierujący, generowane przez projektowany system nadzorowania) dla przyjętych trajektorii ruchu. Szczegóły tej analizy zostały omówione w rozdziale czwartym.



Rys. 2.2. Ogólna koncepcja wirtualnego prototypowania

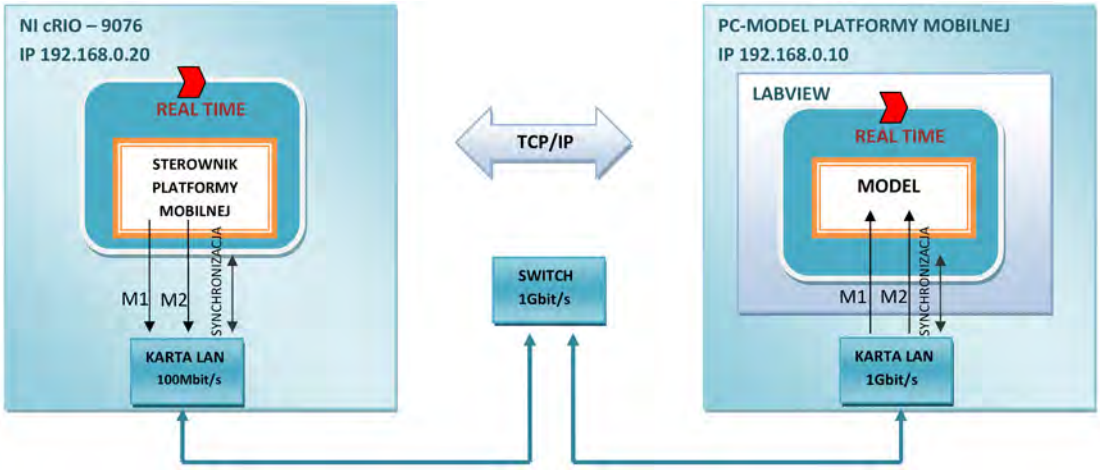
Kluczowym elementem wirtualnego prototypowania jest właściwy dobór narzędzi informatycznych (w tym przypadku autor wykorzystał środowisko LabVIEW [7, 92]), które będą miały niebagatelny wpływ na jakość projektowania, a także, o czym autor wspomniał – na zachowanie niezwykle ważnej elastyczności. Model systemu nadzorowania został utworzony na bazie matematycznego zapisu algorytmu (definicji sygnału sterującego, zależność (4.9)). Do modelowania platformy mobilnej autor wykorzystał równania różniczkowe kinematyki, oraz równania Lagrange'a II rodzaju wynikające bezpośrednio z nałożonych na platformę więzów nieholonomicznych. Rozwiązania równań różniczkowych (w środowisku Maple) zostały zapisane do środowiska LabVIEW tworząc schematy blokowe (podsystemy) umożliwiające w dalszej kolejności przeprowadzenie obliczeń numerycznych (w całym cyklu badań zastosowano stałokrokową metodę całkowania Eulera). Szczegóły implementacji zostały opisane w rozdziale 3.

Warto dodać, co również, było częścią realizowanej pracy, że istotnym elementem środowiska wirtualnego prototypowania jest istnienie mechatronicznego interfejsu użytkownika, umożliwiającego bezpośrednią korektę parametrów przeprowadzonych badań oraz modelowanych obiektów.

2.2. Symulacja czasu rzeczywistego (HILS)

Ze względu na wzrastającą złożoność projektowanych systemów mechatronicznych oraz na towarzyszący im często brak możliwości budowania prototypów (ograniczony dostęp do środków budżetowych i zasobów czasowych przedsięwzięcia), a także – z uwagi na zwiększającą się moc obliczeniową komputerów oraz odpowiednich środowisk testowych (tutaj LabVIEW), rośnie popularność technik projektowych, w których odzwierciedlany (emulowany) jest obiekt rzeczywisty (poprzez implementację modeli obliczeniowych) oraz badana jest odpowiedź takiego układu (modelu) na rzeczywiste sygnały sterujące, generowane przez rzeczywiste sterowniki układu bądź procesu. Wykorzystana technika znacznie zwiększa prawdopodobieństwo podjęcia właściwej decyzji w fazie realizacji projektu (architektury systemu sterowania, zastosowanych rozwiązań i wydajności sprzętowych, parametrów fizycznych oraz mechanicznych obiektu). Niekiedy, ważnym również czynnikiem jest uniknięcie strat w kosztownych urządzeniach, spowodowanych testowaniem różnych strategii sterowania oraz stabilność rozwiązania. Dodatkowo, realizacja testów HILS umożliwia badanie algorytmów, w których liczba zmiennych sterowanych (np. liczba jednostek napędowych (silników) lub wejściowych (np. liczba sensorów) może być dowolnie konfigurowana i optymalizowana (poprzez implementację do kodu sterownika właściwych pobudzeń oraz emulację aktorów danego procesu w modelu obiektu). W pracy wykorzystano metodę HILS, której architekturę zaprezentowano na rys. 2.3.

Utworzony model obliczeniowy trójkołowej platformy mobilnej (ten sam, który został wykorzystany w technice wirtualnego prototypowania) emuluje obiekt rzeczywisty. W tym samym czasie, rzeczywisty sterownik cRIO w czasie RT generuje optymalne sygnały sterujące. W rzeczywistym modelu sygnały (optymalne momenty) sterują układem napędowym i kierowniczym układu – silnikami DC. W tym przypadku, dla emulowanej platformy przy danych wartościach momentów (sygnałów z cRIO) rozwiązywane jest proste zadanie dynamiki (wynikające z równań Lagrange'a; odpowiednie przekształcenia dokonano w środowisku Maple, które następnie zamodelowano w środowisku LabVIEW) oraz proste zadanie kinematyki.



Rys. 2.3. Ogólna koncepcja konfiguracji testu HILS

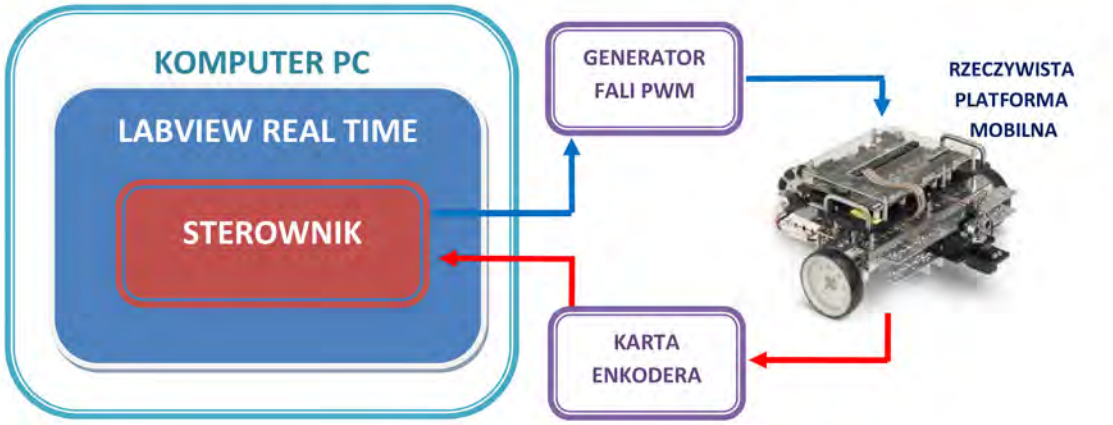
Emulowany model obliczeniowy platformy jest niezależnym programem czasu rzeczywistego, zrealizowanym na bazie zainstalowanego na komputerze PC środowiska LabVIEW z modułem Real Time (zaimplementowany blok emulowanego modelu platformy umieszczono w pętli modułu CS&D, którą następnie zsynchronizowano z pętlą modułu Real Time, szczegóły w rozdziale 5). Spełnienie warunku determinizm czasowego podczas przepływu sygnałów pomiędzy obiema pętlami RT, tzn. pętlą sterownika cRIO generującego optymalne sygnały sterujące, a pętlą modelu obliczeniowego platformy, odbywa się przy udziale zmiennych globalnych czasu rzeczywistego o charakterze kolejek FIFO, które są transportowane poprzez protokół TCP/IP [67, 72, 73]. Sterownik cRIO oraz komputer PC, na którym emulowana jest platforma, został połączony w sieć LAN (standard OSI) poprzez oddzielny *switch* (przełącznik).

Zastosowana technika uzupełnia technikę wirtualnego prototypowania o możliwość optymalizacji systemu nadzorowania (poprzez optymalizację kodu oraz odpowiedni dobór macierzy **R**, **Q**, szczegóły w rozdziale piątym), a także – weryfikacji wydajności przyjętej platformy sprzętowej – sterownika (dobranego drogą analizy wszystkich parametrów wpływających na poprawność sterowania przy jednoczesnym zapewnieniu odpowiedniej wydajności energetycznej systemu). Dodatkowym atutem jest możliwość przeprowadzenia w czasie rzeczywistym walidacji parametrów geometrycznych oraz mechanicznych projektowanej platformy mobilnej.

2.3. Szybkie prototypowanie na obiekcie docelowym

W momencie, gdy dostępny jest obiekt badań, a zachodzi potrzeba realizacji sterownika (np. po przeprowadzeniu modyfikacji systemu mechanicznego, gdzie część funkcji została zastąpiona przez oprogramowanie lub jest realizowana automatycznie) stosuje się technikę szybkiego prototypowania, umożlwiającą znalezienie optymalnego rozwiązania (metodą eksperymentalną) zarówno od strony sprzętowej, jak i programistycznej (optymalizacja algorytmu).

Zamysłem omawianej techniki jest prototypowanie sterownika w środowisku oraz na sprzęcie umożliwiającym w czasie rzeczywistym generowanie sygnałów sterujących, a także – tworzenie kodu algorytmu oraz jego kompilację. Technika, z uwagi na wymagania sprzętowe oraz informatyczne, jest bardzo droga w zastosowaniu. Na rys. 2.4. zamieszczono klasyczną konfigurację rozpatrywanej techniki dla niniejszego projektu.

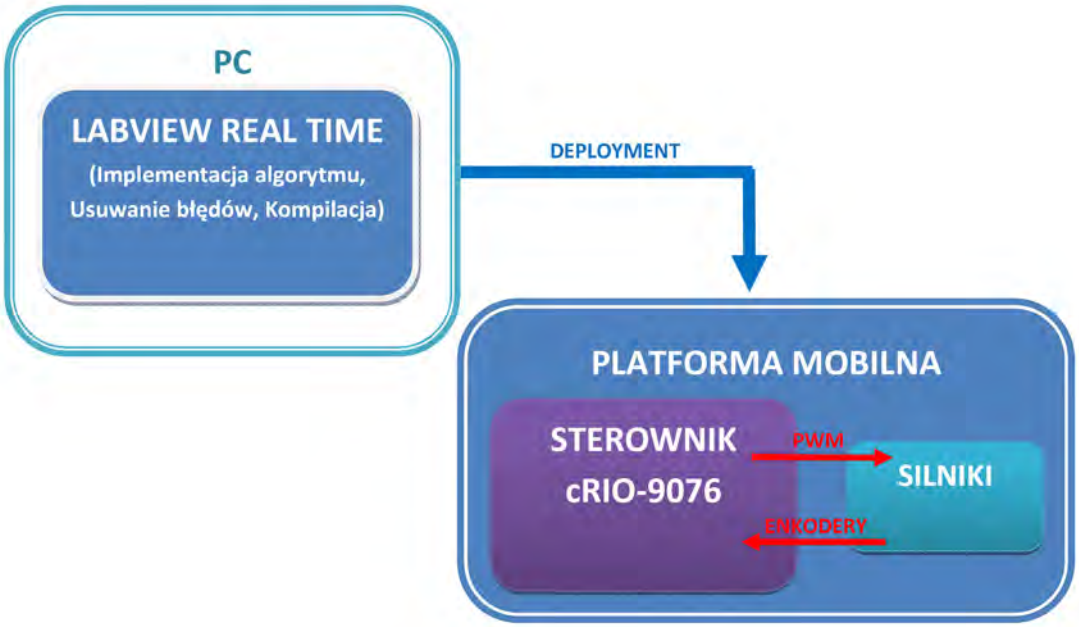


Rys. 2.4. Ogólna koncepcja techniki szybkiego prototypowania na obiekcie docelowym

Podczas prototypowania sterownika, obiekt (platforma mobilna) jest połączony za pomocą urządzeń wejścia/wyjścia (w tym przypadku, generatora fali PWM oraz karty enkodera) z komputerem PC. Komputer, wraz ze środowiskiem LabVIEW, umożliwia przeprowadzenie symulacji zachowania sterownika oraz sprawdzania poprawność realizowanego algorytmu (energetyczny wskaźnik jakości).

Z uwagi na fakt, iż prace rozwojowe nad realizacją projektu były prowadzone równolegle (w tym samym czasie powstawała koncepcja systemu nadzorowania oraz

obiektu badań), a także – dość wczesnej decyzji o wyborze sterownika (co w późniejszej fazie umożliwiło przeprowadzenie testów w technice HILS) autor zastosował koncepcję szybkiego prototypowania na sprzęcie docelowym (jednostce sterownika docelowego cRIO – 9076), wyróżniającą się jednoczesnym przeprowadzaniem procesu integracji oraz prototypowania. Szersze omówienie techniki szybkiego prototypowania na sprzęcie docelowym przedstawiono w rozdziale 6. Konfiguracja tej metody została zaprezentowana na rys. 2.5.



Rys. 2.5. Przyjęta konfiguracja techniki szybkiego prototypowania na obiekcie docelowym

2.4. Podsumowanie

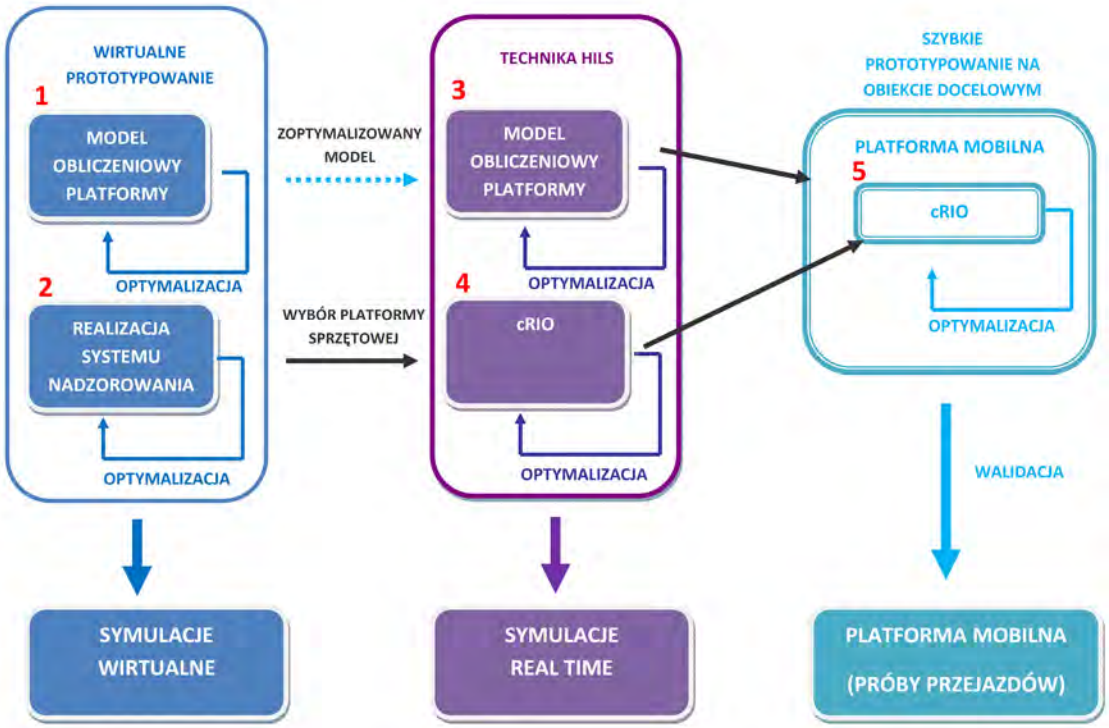
W niniejszej pracy zastosowano wszystkie wymienione powyżej techniki projektowania mechatronicznego. Powiązanie pomiędzy nimi zademonstrowano na rys. 2.6. Zaprezentowana metodologia uwzględnia wszystkie fazy projektowania mechatronicznego systemu nadzorowania trójkołowej platformy mobilnej, a także – relacje zachodzące pomiędzy nimi. Zastosowane techniki (wirtualnego prototypowania oraz HILS) umożliwiają w dość krótkim czasie utworzenie modelu projektowanego sterownika, wraz z tworzonym równocześnie obiektem badań, prześledzenie zachowania się ich w trakcie wirtualnej symulacji (analiza odpowiedzi na optymalne sygnały sterownika), bądź w symulacji czasu rzeczywistego, przy

równoczesnym uwzględnieniu w fazie projektowania części mechanicznej, elektrycznej oraz sterowania (stworzeniu finalnej i szczegółowej koncepcji projektu).

Niezwykle cenną jest technika HILS, która w tym przypadku weryfikuje przyjętą architekturę sprzętową systemu sterowania, umożliwia zoptymalizowanie zastosowanego algorytmu, a także – ustala ostateczną konstrukcję (w tym przypadku) platformy mobilnej.

Warto zwrócić uwagę na technikę szybkiego prototypowania na docelowym obiekcie, w którym występuje, po pierwsze proces integracji rzeczywistego sterownika (powstałego na skutek zastosowania dwóch poprzednich technik projektowania) z obiektem badań, aby poprzez proces optymalizacji dokonać uruchomienia systemu oraz jego walidacji (eksperymentów).

Każda zastosowana technika optymalizuje realizowane funkcje układu (szczegółowy zakres każdej z zademonstrowanej optymalizacji został zamieszczony w tab. 2.1), co w rezultacie skutkuje powstaniem obiektu spełniającego założenia projektowe przy wymogu realizacji postawionych funkcji – funkcji celu – realizacji skutecznego systemu nadzorowania trójkołową platformą mobilną (przy jednoczesnym zminimalizowaniu kosztów przedsięwzięcia naukowego oraz zredukowaniu prawdopodobieństwa podjęcia się realizacji błędnej koncepcji projektu).



Rys. 2.6. Wykorzystane techniki projektowania mechatronicznego

Tab. 2.1. Zakres optymalizacji dla poszczególnych technik projektowania mechatronicznego

Nr	Zakres optymalizacji
1	Parametry geometryczne: wymiary platformy. Parametry mechaniczne: określenie parametrów dla układu napędowego oraz kierowniczego, określenie trajektorii ruchu oraz prędkości maksymalnej przejazdu poprzez przeprowadzenie analizy mnożników Lagrange'a.
2	Poprawność zapisu algorytmu. Dobór współczynników macierzy **R** i **Q**, wprowadzenie prędkości korygujących, analiza możliwości, wydajności sprzętowej (badania sprawności energetycznej układu) poprzez badanie wpływu długości kroku całkowania na długość przeprowadzonych symulacji oraz błędu w przebiegach trajektorii, określenie architektury sprzętowej. Sprawdzanie warunku determinizmu czasowego (realizowane również w ramach HILS).
3	Parametry mechaniczne: Dostosowanie parametrów mechanicznych do zastosowanego typu sterownika, określenie typu silników, enkoderów, typu i wielkości przekładni. Ustalenie kształtu i wymiarów platformy uwzględniającej mocowania silników, układu różnicowego, mechanicznych połączeń. Umiejscowienie układu sterującego oraz zasilającego. Zbilansowanie ciężaru i rozkładu mas.
4	Zapis i czytelność algorytmu (zastosowanie pętli, wykorzystanie zmiennych globalnych czasu rzeczywistego o charakterze kolejek FIFO, grupowanie elementów w podsystemy). Dobór współczynników macierzy **R** i **Q**, optymalizacja wydajności poprzez określenie optymalnego kroku całkowania, podział wykonywania algorytmu na części cRIO RT oraz układu FPGA.
5	Dobór współczynników macierzy **R** i **Q**, dobór wzmocnienia i nastaw sterowników NI – 9505 (dla fali PWM), regulacja odczytu enkoderów. Przeprowadzenie końcowej weryfikacji zapisu algorytmu. Usuwanie błędów i zbędnych bloków (minimalizacja algorytmu).

3. Model trójkołowej platformy mobilnej

Praktyka inżynierska nakazuje, aby nowa idea, produkt bądź myśl techniczna była poprzedzona przeprowadzeniem symulacji komputerowych potwierdzających trafność podjętych decyzji, weryfikację założeń konstruktorskich, bądź modyfikację pewnej założonej wizji projektowanego systemu. Naprzeciw tym wymaganiom wychodzi projektowanie mechatroniczne, mogące, często dzięki zastosowanym narzędziom informatycznym powiązać w jedną wspólną całość, szereg często bardzo odległych od siebie dziedzin wiedzy, a także uwzględnić równocześnie wiele aspektów technicznych oraz detali.

3.1. Charakterystyka ogólna

Podstawą niezbędną do przeprowadzenia badań oraz rozwojem przyjętej koncepcji (myśli) jest budowa modelu obliczeniowego realizowanego pomysłu. Istotnym zagadnieniem w procesie budowy jest ustalenie akceptowalnego poziomu szczegółowości przyjętego modelu, odzwierciedlającego rzeczywisty obiekt, system bądź proces. Ograniczenia sprzętowe, brak wystarczającej wiedzy, a także brak istotnego wpływu na funkcjonowanie układu powoduje, że nie modeluje się niektórych zjawisk fizycznych. Pomija się elementy, budując model prostszy, mniej złożony, jednocześnie pozwalający z dużym prawdopodobieństwem odwzorować rzeczywisty charakter projektowanego obiektu.

Tworząc model danego obiektu, istotnym zagadnieniem jest opis kinematyki i dynamiki układu. Opis kinematyki robotów mobilnych jest ściśle uzależniony od ograniczeń, jakimi podlegają te roboty. Robot mobilny ze względu na swoją budowę, ma narzucone na swoją konstrukcję więzy. Narzucenie więzów, powoduje, że wartości, jakie mogą przyjmować zmienne konfiguracyjne i ich pochodne, podlegają ograniczeniom. W dalszej części, analiza więzów układu prowadzi do powstania opisu kinematyki robota, w którym istnieje możliwość określenia jego parametrów liniowych ruchu (droga, prędkość, przyspieszenie) oraz kątowych (kąt obrotu, prędkość kątowa, przyspieszenie kątowe) [19, 99, 100].

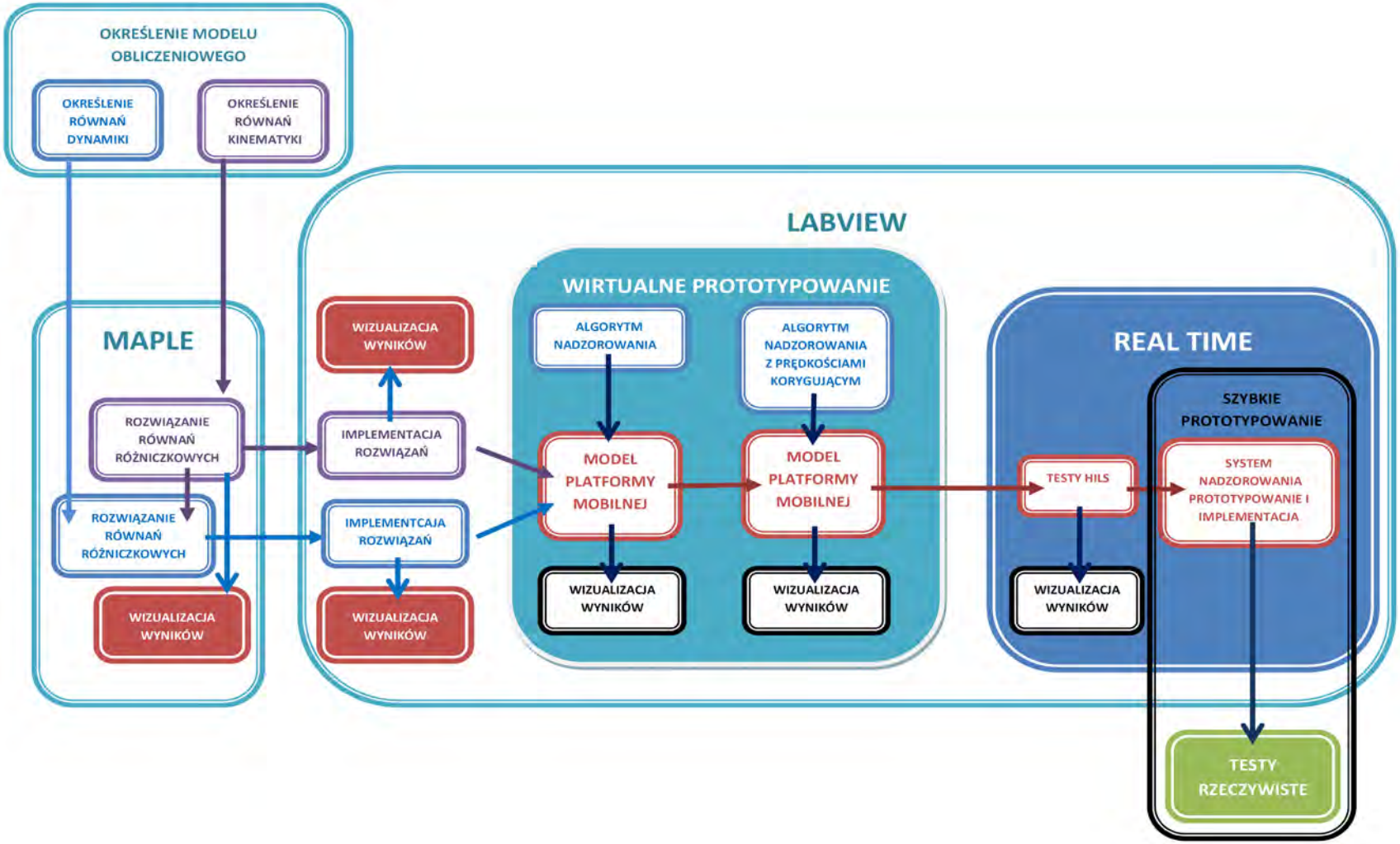
W przypadku opisu dynamiki układu, definiuje się związek zachodzący pomiędzy sygnałem podawanym na wejścia układu, a sygnałami wyjściowymi stanowiącymi odpowiedź układu. Modele obliczeniowe, zwykle przedstawiane są za pomocą układu równań różniczkowych lub transmitancji operatorowych (Laplace'a lub Fouriera) [4, 8, 33]. Wybór formy modelu obliczeniowego jest ściśle uzależniony od przyjętej metody konstrukcji obiektu (tj. funkcji, jaką spełnia). W przypadku trójkołowej platformy mobilnej przyjęto do opisu jej ruchu równania Lagrange'a II rodzaju z mnożnikami. W trakcie tworzenia modelu, niezwykle ważnym aspektem jest zdefiniowanie metod jego analizy (zastosowanych metod projektowania, uruchomienia oraz przeprowadzania badań eksperymentalnych), a także ścisłe powiązanie z koncepcję układu sterowania. Zastosowane w pracy równania, pozwoliły na określenie momentów napędzającego oraz kierującego (które sterują silnikami DC), a także mnożników Lagrange'a, które wykorzystano do zbadania całkowitych sił tarcia suchego, których wartości dla zapewnienia jazdy bez poślizgu kół, nie powinny przekraczać wartości granicznych.

Tworzenie modelu obliczeniowego platformy, a w późniejszej perspektywie realizacja całego przedsięwzięcia naukowego, będzie bazować na współczesnym projektowaniu mechatronicznym, które nie tylko uwzględnia możliwość projektowania równoległego, ale także (co jest niezmiernie istotne i pożądane) jest na tyle elastyczne w formie, aby na każdym etapie badań istniała możliwość przeanalizowania różnych wariantów oraz koncepcji. Niezwykle pożądaną cechą takiego podejścia (mechatronicznego, równoległego) jest możliwość realizacji (w procesie projektowania) szeregu sprzężeń zwrotnych, szybkiej zmiany ustawień oraz parametrów tak, aby końcowy efekt był jak najbardziej zbieżny z przyjętymi założeniami konstrukcyjnymi. Na rys. 3.1 została zademonstrowana koncepcja przeprowadzonych w pracy badań.

Autor w niniejszej pracy postanowił, aby platformą projektowania mechanicznego było środowisko LabVIEW. Zastosowane środowisko pozwoliło autorowi na przeprowadzenie badań nad zachowaniem się układu w przestrzeni (analiza kinematyki), analizę dynamiki układu, dobór jego parametrów geometrycznych oraz mechanicznych. Realizacja techniki wirtualnego prototypowania umożliwiła autorowi weryfikację opracowanego wcześniej modelu platformy, a także weryfikację jego odpowiedzi na sygnały generowane z prototypowanego sterownika.

W późniejszej fazie, środowisko LabVIEW zostało wykorzystane podczas realizacji (w czasie rzeczywistym) testów HILS. Ostatecznie, poprzez realizację techniki szybkiego prototypowania w systemie docelowym przeprowadzono integrację oraz walidację (eksperymentalną ocenę zgodności) systemu nadzorowania z platformą mobilną.

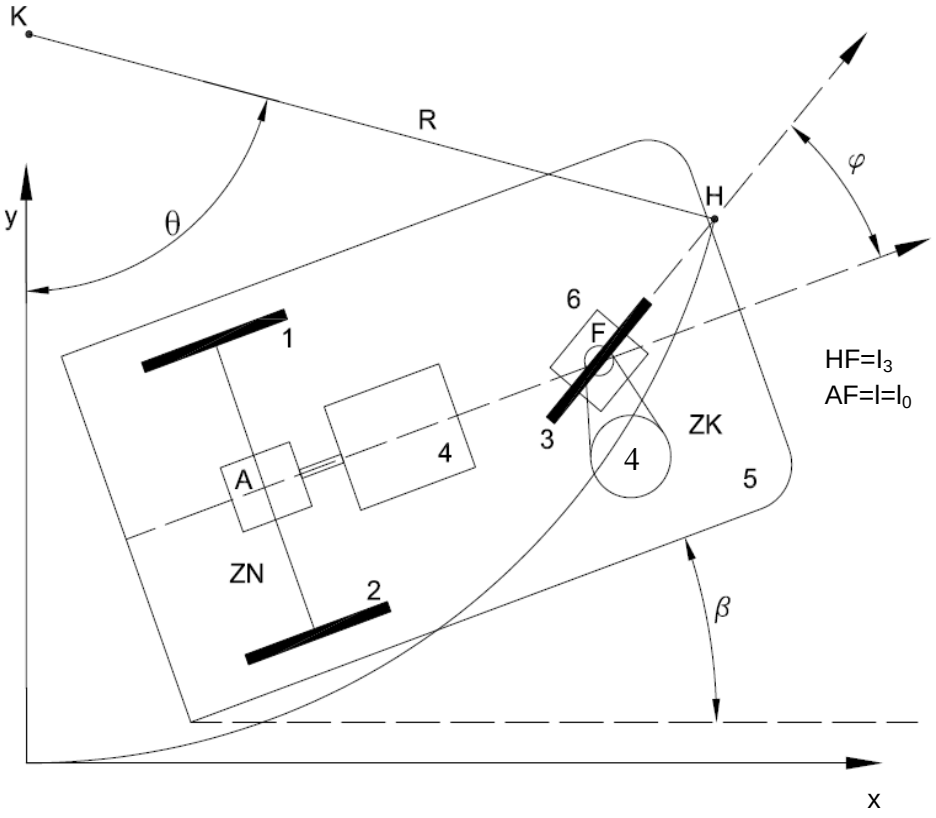
W celu zachowania elastyczności projektu autor zadecydował, aby równania różniczkowe (kinematyki, dynamiki) zostały rozwiązane w środowisku Maple. Zastosowanie takiego zabiegu pozwoliło na otrzymanie rozwiązań w postaci parametrycznej. Następnie, parametryczna postać równań została zaimplementowana do środowiska LabVIEW [41]. Autor, w pierwszym etapie realizacji przedsięwzięcia naukowego, postanowił sprawdzić zaimplementowane oraz rozwiązane numerycznie (w LabVIEW) parametryczne równania kinematyki i dokonał procesu ich weryfikacji z rozwiązaniami (tych samych równań) otrzymanych w środowisku Maple.



Rys. 3.1. Koncepcja badań mechatronicznych platformy mobilnej

3.2. Model obliczeniowy

Mając na uwadze przedstawioną powyżej koncepcję przeprowadzanych badań, a także przyjęte wymagania konstrukcyjne, autor określił w pierwszej fazie model obiektu badań. Rys. 3.2 przedstawia te rozważania.



Rys. 3.2. Model trójkołowej platformy mobilnej

W prezentowanym modelu można wyróżnić główne części: podwozie (rama) 5, zespół napędowy ZN oraz zespół kierujący ZK. Założono, że system napędowy będzie bazował na dwóch kołach 1 i 2, które poprzez mechanizm różnicowy będą napędzane silnikiem elektrycznym prądu stałego 4. W zespole kierującym znajduje się kierownica 6, która wraz z kołem 3 obraca za pomocą zainstalowanego na ramie platformy silnika prądu stałego 4. Prędkości kątowe poszczególnych kół określono za pomocą pochodnych współrzędnych α_1, α_2 oraz α_3. Kątem obrotu kierownicy względem ramy oznaczono, jako φ, natomiast przez β oznaczono chwilowy kąt obrotu platformy. Symbolem θ określono kąt orientacji, należącego do kierownicy punktu H na torze kołowym.

3.3. Kinematyka platformy mobilnej

Przedstawiony model trójkołowej platformy mobilnej pozwolił, bazując na więzach nieholonomicznych nałożonych na punkt A platformy, na zdefiniowanie prędkości uogólnionych platformy mobilnej, a także na przeprowadzanie analizy zadania odwrotnego kinematyki, którego efektem było wyznaczenie parametrów ruchu platformy.

Założenie o braku występowania poślizgów kół platformy mobilnej, powoduje, że wektor prędkości punktu A będzie zgodny z kierunkiem wzdłużnej osi symetrii ramy. Zależność pomiędzy rzutami wektora prędkości tego punktu na osie 0xy, na bazie więzów nieholonomicznych przyjmuje postać [19]:

$$\dot{x}_A \, tg\beta = \dot{y}_A, \tag{3.1}$$

Zakładając, ze punkt H porusza się po okręgu o promieniu R, współrzędne tego punktu będą zmieniać się według zależności:

$$\begin{aligned} x_H &= R\sin\theta, \\ y_H &= R(1-\cos\theta). \end{aligned} \tag{3.2}$$

Wówczas, rzuty wektora prędkości punktu H na osie x i y będą opisane zależnością:

$$\begin{aligned} R\dot{\theta}\cos\theta &= \dot{x}_A - l\dot{\beta}\sin\beta - l_3(\dot{\beta}+\dot{\varphi})\sin(\beta+\varphi), \\ R\dot{\theta}\sin\theta &= \dot{y}_A - l\dot{\beta}\cos\beta - l_3(\dot{\beta}+\dot{\varphi})\cos(\beta+\varphi). \end{aligned} \tag{3.3}$$

Jeżeli znana jest prędkość v_A poruszania się punktu A, to

$$\dot{x}_A = v_A\cos\beta, \qquad \dot{y}_A = v_A\sin\beta. \tag{3.4}$$

Układy równań (3.1), (3.3) oraz (3.4) pozwalają na zdefiniowanie układu równań:

$$\begin{cases} v_A\cos\beta = \dot{\beta}\left[l\sin\beta + l_3\sin(\beta+\varphi)\right] + \dot{\varphi}\, l_3\sin(\beta+\varphi) + \dot{\theta}\, R\cos\theta \\ v_A\sin\beta = -\dot{\beta}\left[l\cos\beta + l_3\cos(\beta+\varphi)\right] - \dot{\varphi}\, l_3\cos(\beta+\varphi) + \dot{\theta}\, R\cos\theta. \\ v_A\, tg\varphi = \dot{\beta}\, l \end{cases} \tag{3.5}$$

Układ równań (3.5) można zapisać w postaci:

$$\mathbf{J}_1(\mathbf{q})\dot{\mathbf{q}} = \begin{bmatrix} v_A\cdot\cos\beta \\ v_A\cdot\sin\beta \\ v_A\cdot tg\varphi \end{bmatrix}, \tag{3.6}$$

gdzie poprzez $\mathbf{J}_1(\mathbf{q})$ oznaczono jakobian, który jest określony jako:

$$\mathbf{J}_1(\mathbf{q}) = \begin{bmatrix} l\sin\beta + l_3\sin(\beta+\varphi) & l_3\sin(\beta+\varphi) & R\cos\theta \\ -l\cos\beta - l_3\cos(\beta+\varphi) & -l_3\cos(\beta+\varphi) & R\sin\theta \\ 1 & 0 & 0 \end{bmatrix}, \tag{3.7}$$

a $\mathbf{q}$ jest wektorem prędkości kątowych:

$$\dot{\mathbf{q}} = \left[\dot{\beta}, \dot{\varphi}, \dot{\theta}\right]^{\mathrm{T}}. \tag{3.8}$$

Z zależności (3.6) wynika, że poszukiwany wektor prędkości kątowych opisany jest zależnością:

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{\beta} \\ \dot{\varphi} \\ \dot{\theta} \end{bmatrix} = \mathbf{J}_1^{-1}(\mathbf{q}) \cdot \begin{bmatrix} v_A \cdot \cos\beta \\ v_A \cdot \sin\beta \\ v_A \cdot tg\beta \end{bmatrix}. \tag{3.9}$$

Zakładając, że nie występują poślizgi boczne ani wzdłużne, przedstawiamy równania styczności punktu każdego z kół platformy z płaszczyzną, po której się poruszają.

I tak, dla koła pierwszego:

$$\dot{x}_A + \dot{\beta} l_1 \cos\beta - \dot{\alpha}_1 r\cos\beta = 0, \qquad \dot{y}_A + \dot{\beta} l_1 \sin\beta - \dot{\alpha}_1 r\sin\beta = 0. \tag{3.10}$$

Dla koła drugiego, zależności te przyjmują postać:

$$\dot{x}_A - \dot{\beta} l_1 \cos\beta - \dot{\alpha}_2 r\cos\beta = 0, \qquad \dot{y}_A - \dot{\beta} l_1 \sin\beta - \dot{\alpha}_2 r\sin\beta = 0. \tag{3.11}$$

W przypadku koła trzeciego (kierującego), równanie styczność punktu z płaszczyzną można zapisać jako:

$$\dot{x}_A - \dot{\beta} l_1 \sin\beta - \dot{\alpha}_3 r\cos(\beta+\varphi) = 0, \qquad \dot{y}_A + \dot{\beta} l_1 \sin\beta - \dot{\alpha}_3 r\sin(\beta+\varphi) = 0. \tag{3.12}$$

Równania (3.10), (3.11) oraz (3.12), określają szukane parametry kątów obrotu poszczególnych kół. Równania te można zapisać w postaci układu:

$$\begin{cases} v_A\cos\beta + \dot{\beta} l_1\cos\beta = \dot{\alpha}_1 r\cos\beta \\ v_A\cos\beta - \dot{\beta} l_1\cos\beta = \dot{\alpha}_2 r\cos\beta \\ v_A\cos\beta - \dot{\beta} l_1\sin\beta = \dot{\alpha}_3 r\cos\left(\beta+\varphi\right) \end{cases}. \tag{3.13}$$

Rozwiązanie równań (3.5) oraz (3.13) pozwala na (dla zdefiniowanej trajektorii ruchu) określenie parametrów ruchu platformy (φ, β, θ, α_1, α_2, α_3). Autor założył, że do celów przeprowadzenia doświadczeń zbada zachowanie się platformy przy przejeździe wzdłuż trzech trajektorii tj. typu „sinus", „okrąg" oraz „parabola" [22, 30, 36, 42, 84]. Poniżej zostały zademonstrowane rozważania dla trajektorii typu „okrąg". Do analizy pozostałych torów przejazdu (trajektorii) platformy mobilnej,

autor zdefiniował odpowiednią funkcję opisującą trajektorię ruchu punktu H tj.(x_H = f(trajektoria), y_H = f(trajektoria). Zgodnie z zamieszczonym rys. 3.1 do rozwiązania parametrycznego równań różniczkowych zastosowano środowisko Maple. Szersze omówienie, zastosowanego w pracy środowiska Maple zostało umieszczone w załączniku do niniejszej pracy (p. 10.2).

3.3.1. Realizacja trajektorii typu „sinus”

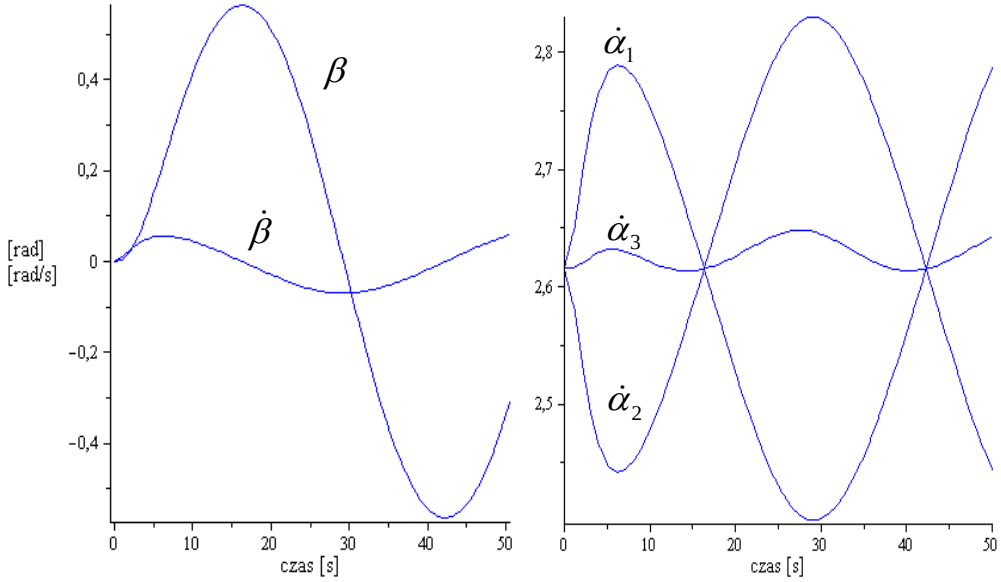
Poszukujemy parametrów opisu ruchu trójkołowej platformy mobilnej po trajektorii typu „sinus”. Podczas symulacji platforma wykona tor przejazdu pokazany na rys. 3.4. (prawy) oraz rys. 3.12 (prawy). Wykresy otrzymano drogą graficznej ilustracji rozwiązania równań różniczkowych, dobierając odpowiedni czas trwania symulacji. Należało określić równania opisujące w czasie ruch punktu charakterystycznego obiektu. Tor ruchu opisany jest równaniem (3.14):

$$y_H = A\sin(\frac{2\pi}{T}x_H). \tag{3.14}$$

Równania (3.5) oraz (3.13) dla danej trajektorii zostały rozwiązane metodą symboliczną w środowisku Maple. Obliczenia w środowisku Maple zrealizowano w trybie konwersacyjnym. Otrzymane na podstawie przekształceń symbolicznych rozwiązanie stanowi układ sześciu równań różniczkowych:

$$\begin{cases} \dot{\varphi} = -[\ v_A l_0 \sin\beta - A_0 l_0 \sin(\frac{2\pi x_H}{T_C}) + v_A l_0 \cos\beta\, tg\varphi + \\ + v_A l_3 \cos(\beta+\varphi) tg\varphi\ ]/[\ l_3 l_0 \cos(\beta+\varphi)\ ], \\ \dot{\beta} = \dfrac{v_A tg\varphi}{l_0}, \\ \dot{x}_H = [\ v_A \sin(\beta+\varphi)\sin\beta - A_0 \sin(\beta+\varphi)\sin(\frac{2\pi x_H}{T_C}) + v_A \sin(\beta+\varphi)\cos\beta\, tg\varphi + \\ + v_A \cos(\beta+\varphi)\cos\beta - v_A \cos(\beta+\varphi)\sin\beta\, tg\varphi\ ]/\cos(\beta+\varphi), \\ \dot{\alpha}_1 = \dfrac{v_A(l_1 tg\varphi + l_0)}{r_1 l_0}, \\ \dot{\alpha}_2 = -\dfrac{v_A(l_1 tg\varphi - l_0)}{r_1 l_0}, \\ \dot{\alpha}_3 = -\dfrac{v_A(tg\varphi\sin\beta - \cos\beta)}{r_1\cos(\beta+\varphi)}. \end{cases} \tag{3.15}$$

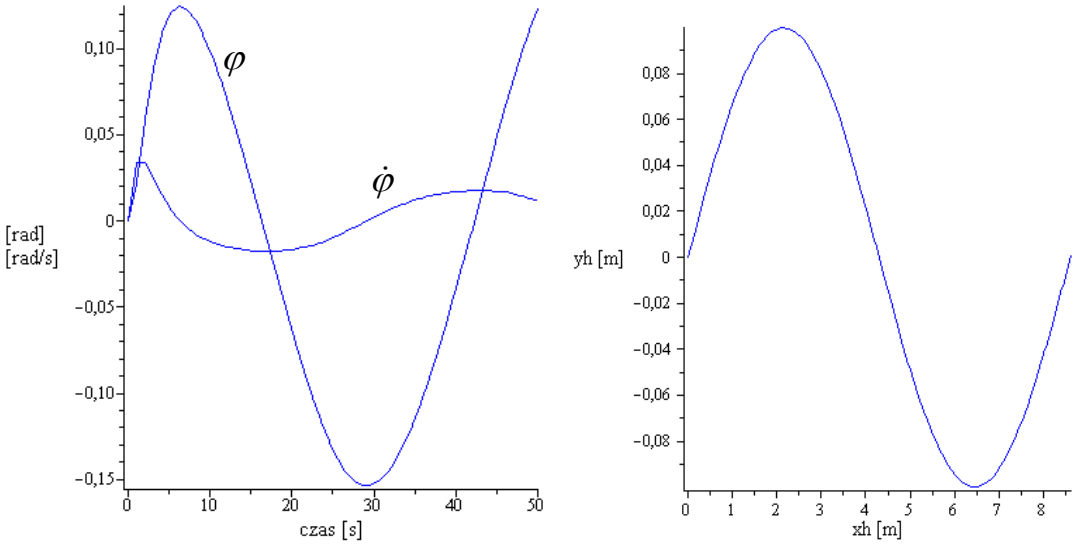
Otrzymany układ równań różniczkowych jest układem nieliniowym. Dalsza jego analiza możliwa jest przy zastosowaniu metod numerycznych. W celu weryfikacji poprawności modelowania (implementacji) układu równań do środowiska LabVIEW, autor postanowił przeprowadzić dodatkowe rozwiązanie numeryczne dla każdej trajektorii w środowisku Maple (zgodnie z rys. 3.1). Rozwiązanie tych równań wygenerowano dla kroku całkowania 0,005 s stosując metodę Eulera (rozważania na temat przyjętego kroku całkowania i metody jego doboru podjęte zostały w dalszej części pracy).



Rys. 3.3. Kąt obrotu platformy (lewy). Prędkości kątowe kół platformy (prawy)

Na rys. 3.3. (lewy) zamieszczono, wykreślone na podstawie zależności (3.15) rozwiązanie numeryczne chwilowego kąta obrotu ramy β oraz jego pochodnej. Z zamieszczonych wykresów wynika, że platforma podczas całego okresu przejazdu zachowuje płynny i łagodny tor jazdy. Świadczy o tym brak jakichkolwiek „ostrych" przejść w chwilowej prędkości kąta obrotu, co mogłoby świadczyć o wystąpieniu lokalnych przyspieszeń oraz powodowałoby utratę stabilności. Dodatkowo, na uwagę zasługuje również mała prędkość kątowa, co z pewnością predestynuje obiekt do zapewnienia mu sterowalności w trakcie poruszania się po wyznaczonej trajektorii. Rys. 3.3. (prawy) obrazuje przebiegi poszczególnych prędkości kół platformy. Podobnie, jak to miało miejsce dla przebiegów kąta obrotu platformy i jego pochodnej, tak i w tym przypadku, zmiany prędkości są bardzo łagodne. Nie występują gwałtowne zmiany, co świadczy o stabilnym oraz płynnym ruchu

platformy. Ze względu na fakt, iż moment napędowy platformy przekazywany jest przez silnik DC oraz wspólny dla obu kół układ różnicowy, to prędkości kątowe obu kół 1 i 2 są bezpośrednio uzależnione od prędkości wału silnika. Doskonale ta sytuacja zobrazowana jest na rozpatrywanym rysunku, na którym w trakcie pokonywania łuku jedno koło przyspiesza, drugie zwalnia. Na uwagę zasługuje, niemalże stała prędkość kątowa koła 3 (brak przyspieszenia). Jest to o tyle komfortowa sytuacja, iż jest to prędkość koła kierownicy, które w znacznej mierze decyduje o realizacji zamierzonej trajektorii przejazdu. Brak przyspieszeń, oznacza brak szarpnięć mogących być przyczyną ewentualnych błędów w położeniu platformy (realizacji ustalonej trasy przejazdu).



Rys. 3.4. Kąt obrotu kierownicy (lewy). Tor ruchu punktu charakterystycznego (prawy)

Na rys. 3.4 (lewy) został przedstawiony chwilowy kąt obrotu oraz chwilowa prędkość kątowa obrotu kierownicy platformy. Krzywa, swoim charakterem, nawiązuje do krzywej z rys. 3.3 (lewy). W tym jednak przypadku, chwilowy kąt obrotu kierownicy rośnie szybciej, aby swoje maksimum osiągnąć w 8 s przejazdu. Odzwierciedleniem tej zależności jest wykres prędkości kąta obrotu kierownicy, który już w 3 s osiąga prędkość maksymalną (szybciej, niż prędkość obrotu platformy). Przyczynę tej różnicy należy upatrywać w geometrii (konstrukcji) platformy – najpierw skrętu doznaje kierownica, później nadąża za tą zmianą kąt obrotu platformy. Z zestawiania tych dwóch wykresów wynika dodatkowy wniosek, że o ile zmiany kąta obrotu kierownicy są szybsze, to osiągane wartości kątowe są mniejsze, niż prędkość

obrotu kąta platformy. Również i w tym przypadku wytłumaczenia należy doszukiwać się w przyjętej konstrukcji obiektu. Fakt jest o tyle istotny dla systemu nadzorowania platformą, że niewielka zmiana kąta obrotu kierownicy w dość znaczący sposób (ale z pewnym opóźnieniem) będzie powodowała zmiany w położeniu platformy, a tym samym – wpływała na poprawność realizacji toru przejazdu.

Rys. 3.4 (prawy) obrazuje rozwiązanie numeryczne równania różniczkowego położenia punktu charakterystycznego we współrzędnych kartezjańskich 0xy. Autor poprzez dobór czasu trwania przejazdu zapewnił, aby punkt charakterystyczny projektowanej platformy przebył trasę równą pełnemu okresowi krzywej sinus. Platforma mobilna w ciągu 50 s przejeżdża wzdłuż osi x około 8,5 m.

Rozwiązań równań różniczkowych docelowo autor poszukiwał w środowisku LabVIEW (jak wcześniej wspomniano graficzna reprezentacja rozwiązań dokonana w środowisku Maple posłużyła jedynie w celu weryfikacji rozwiązań otrzymanych w środowisku LabVIEW). Rozwiązania wygenerowano metodą numeryczną Eulera (podobnie, jak to miało miejsce w środowisku Maple) korzystając z modułu CD&S omawianego programu oraz przyjmując identyczną długość kroku całkowania tj. 0,005 s. Szersze omówienie zastosowanego środowiska LabVIEW zamieszczono w załączniku niniejszej pracy (p. 10.1).

Otrzymane rozwiązania w postaci parametrycznej zaimplementowano do środowiska LabVIEW korzystając z bogatej biblioteki dostępnych w module CD&S bloków (znanych z teorii modelowania czy identyfikacji).

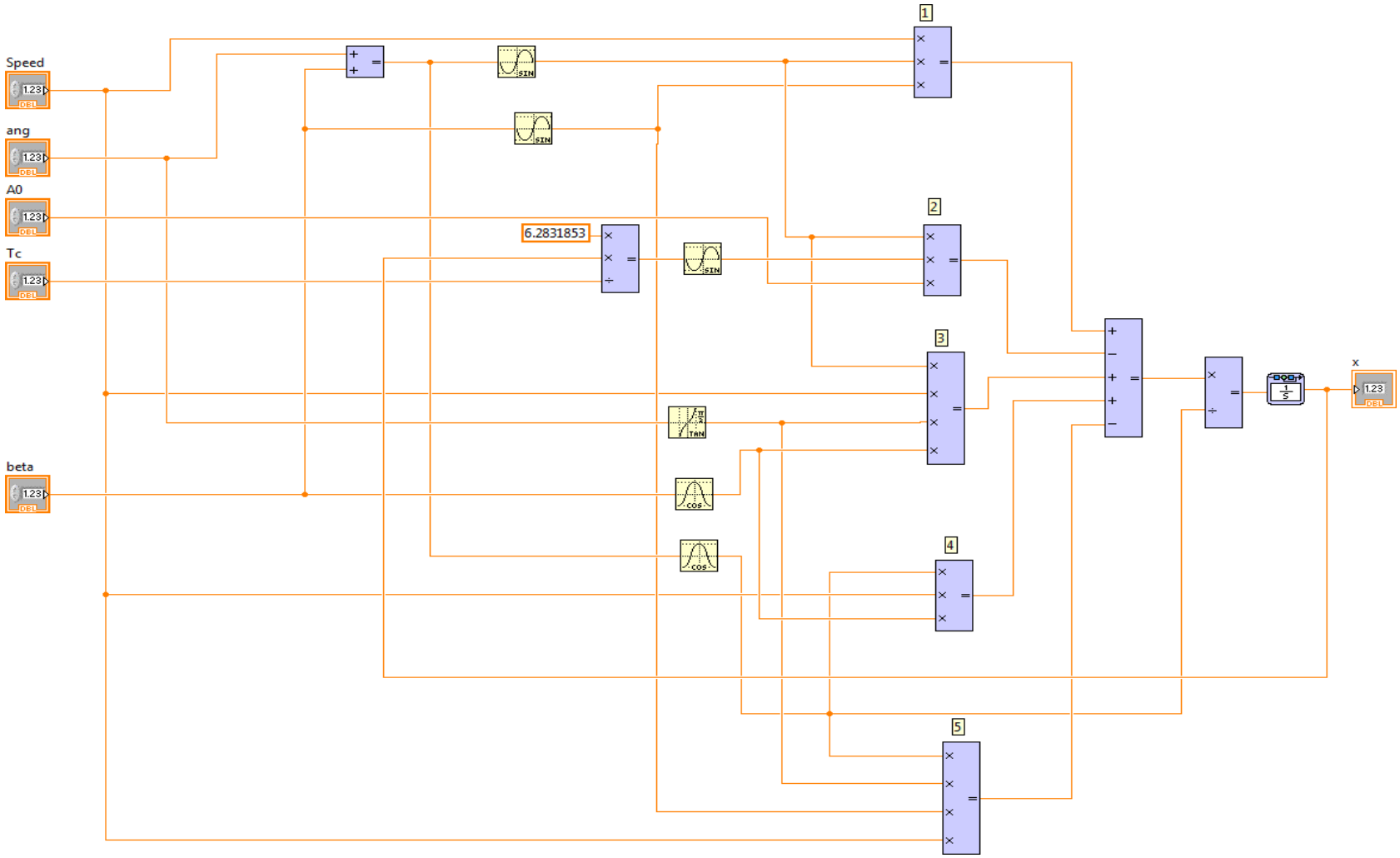
Koncepcja symulacji w tym module opiera się na stworzeniu tzw. *control simulation loop*. W jej wnętrzu autor umieścił i połączył w bloki system niezbędny do rozwiązania ustaloną metodą numeryczną danego równania różniczkowego. Do realizacji systemu sterowania wykorzystano przede wszystkim bloki umożliwiające całkowanie i różniczkowanie odpowiednich wartości sygnałów. W celu eliminacji błędów oraz uzyskania lepszej przejrzystości kodu (przy niezwykle złożonym – ze względu na złożone równania różniczkowe – systemie nadzorowania; do budowy systemu użyto ponad 1300 różnych bloków i tym samym połączeń) bloki stanowiące rozwiązanie danego równania różniczkowego grupowano w tzw. podsystemy. Należy nadmienić, że dla każdej trajektorii powtórzono procedurę modelowania (rozwiązania przebiegów prędkości kątowych oraz kąta obrotu

platformy dla wszystkich rozpatrywanych trajektorii są identyczne i zapisano je tylko raz).

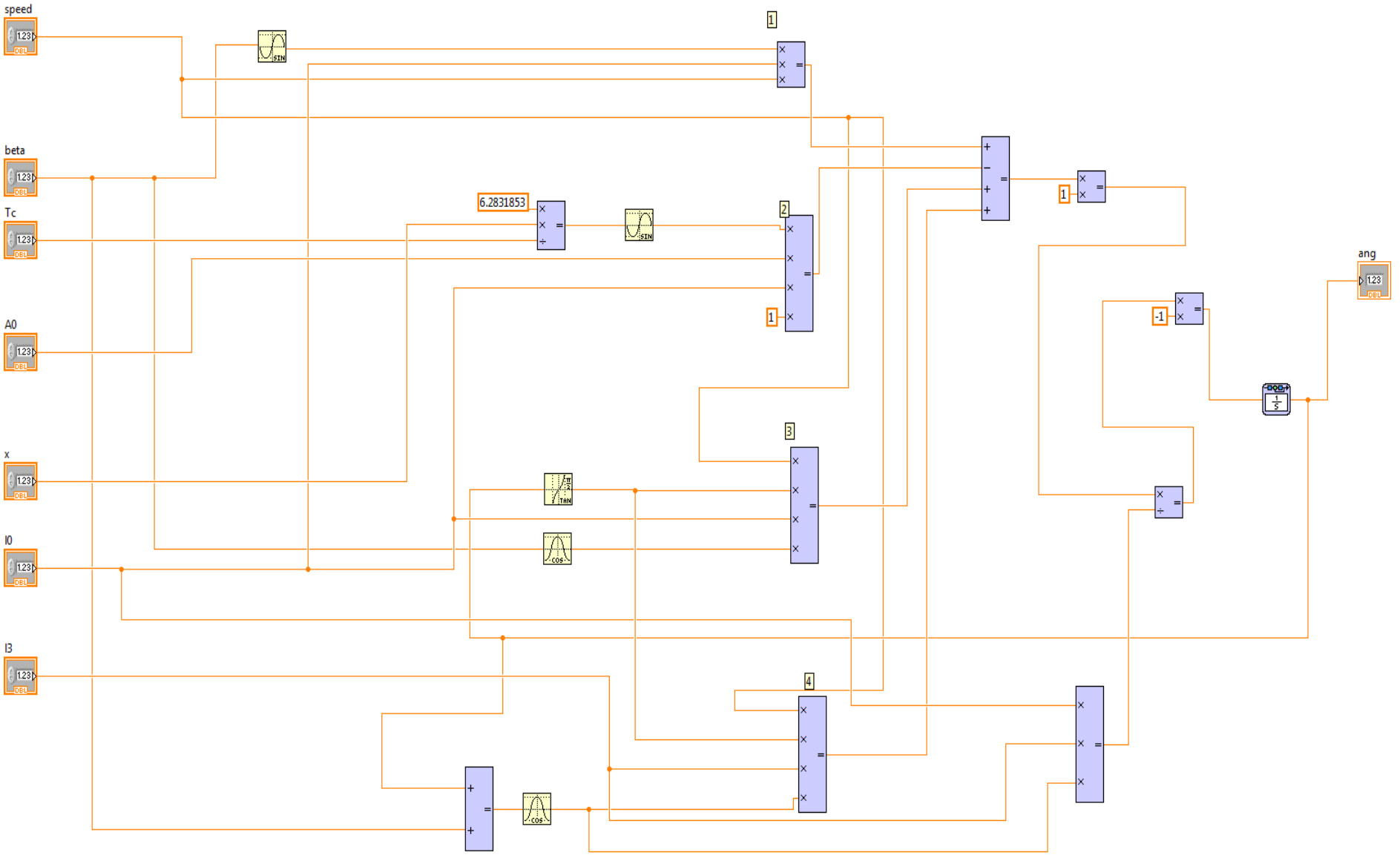
Schematy zaimplementowanych podsystemów dla danego parametru rozwiązania zaprezentowano na rys. 3.5 – 3.10. Przytoczone podsystemy pokazano w pracy dla trajektorii typu „sinus"; dla pozostałych trajektorii tj. trajektorii typu „parabola" oraz „okrąg" mechanizm tworzenia jest identyczny.

Sygnałem wyjściowym dla każdego z widocznych na rys. 3.5, 3.6 oraz 3.7 podsystemów jest rozwiązanie (chwilowa wartość) równania różniczkowego. Powyższy sygnał otrzymywany jest drogą działań arytmetycznych. Końcowym etapem tych operacji jest wyliczenie przez blok 1/s w każdym ustalonym kroku Δt całki sygnału, która jednocześnie stanowi sprzężenie zwrotne (sygnał) dla następnego kroku całkowania [41]. Sygnałami wejściowymi są rozwiązania pochodzące z innych podsystemów otrzymanych dla danej chwili czasu t (wszystkie sześć równań rozwiązywane jest w tym samym czasie).

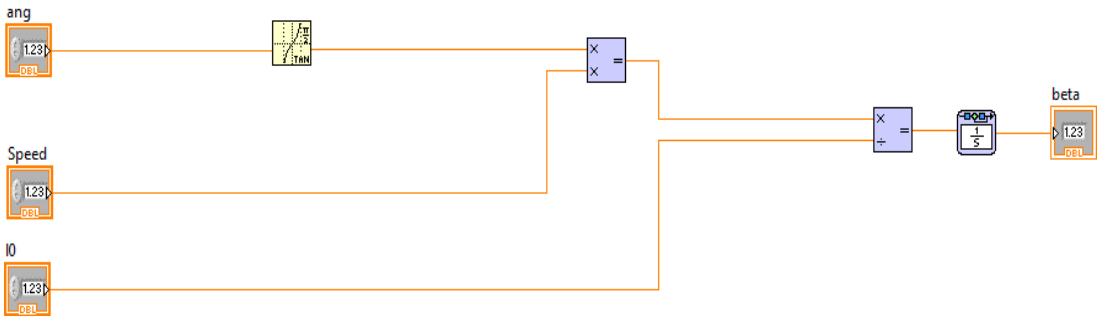
Na rys. 3.5 zademonstrowano zapisane do środowiska LabVIEW rozwiązanie dla punktu charakterystycznego trajektorii typu „sinus". Jest to równanie silnie nieliniowe, którego rozwiązania należy poszukiwać posługując się metodami numerycznymi. Na rys. 3.6 zaprezentowano implementację dla kąta kierownicy platformy, a na rys. 3.7 – dla kąta obrotu ramy. Należy zauważyć, że prędkości kątowe kół 1 oraz 2 są symetryczne również i w konwencji utworzonych schematów podsystemów (rys. 3.8 oraz rys. 3.9). Jedyną różnicą jest fakt mnożenia prędkości kątowej koła 2 przez wartości –1. Wynika to bezpośrednio z równań otrzymanych drogą symboliczną (3.15). Warto zwrócić uwagę, że o ile implementacja do LabVIEW rozwiązania równań „unikatowych" dla danej trajektorii tj. dla kąta obrotu kierownicy oraz trajektorii punktu charakterystycznego, wymaga istnienie sprzężenia zwrotnego, to dla pozyskanych parametrów taka konieczność nie istnieje. W tym przypadku rozwiązania otrzymuje się jedynie drogą działań arytmetycznych.



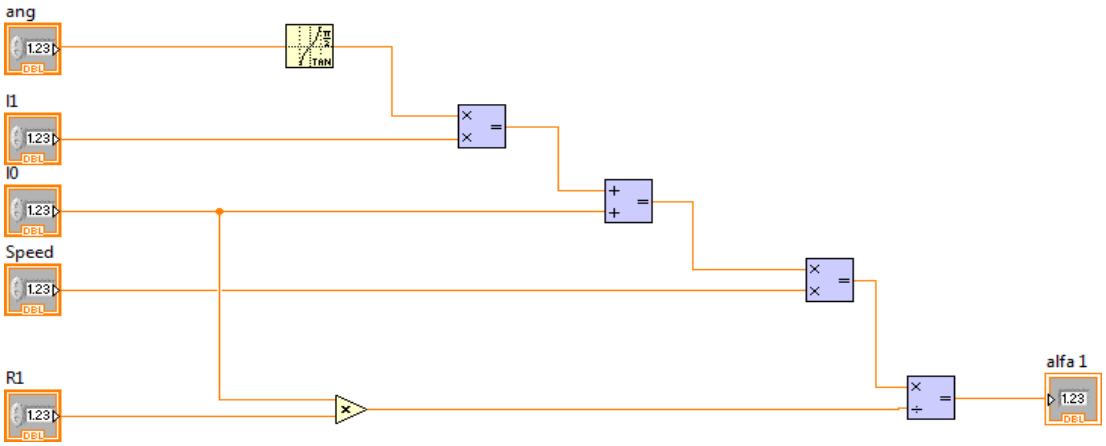
Rys. 3.5. Zapisane rozwiązanie dla punktu charakterystycznego H platformy mobilnej



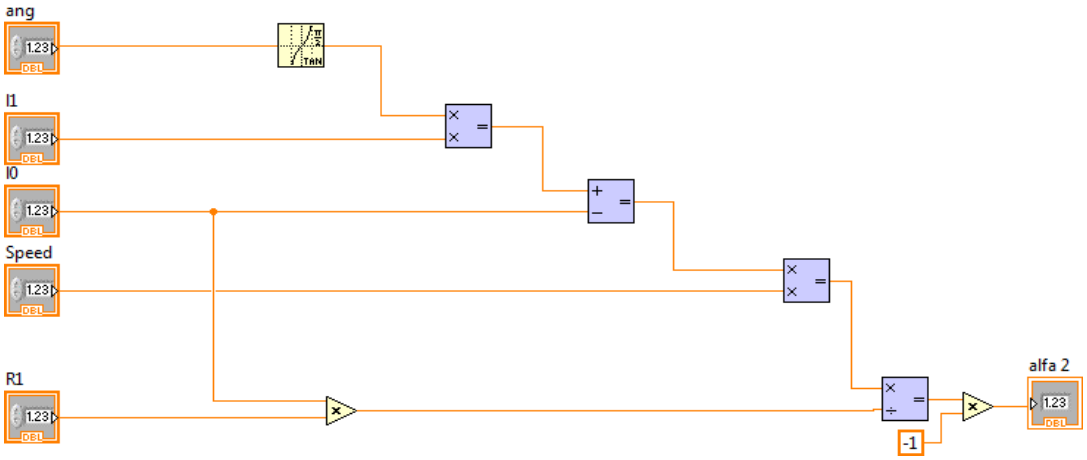
Rys. 3.6. Zapisany kąt obrotu kierownicy φ platformy mobilnej



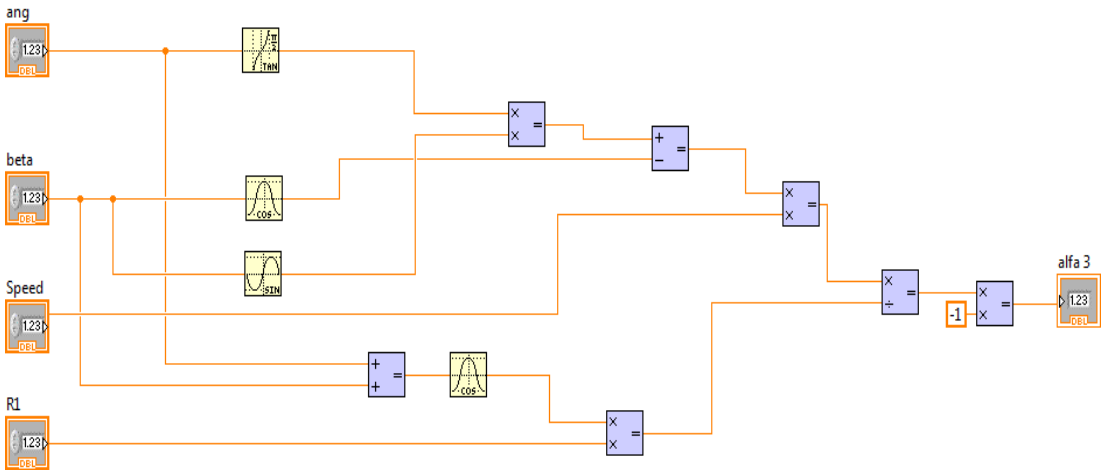
Rys. 3.7. Zapisany kąt obrotu ramy β platformy mobilnej



Rys. 3.8. Zapisana prędkość kątowa koła 1 α_1 platformy mobilnej



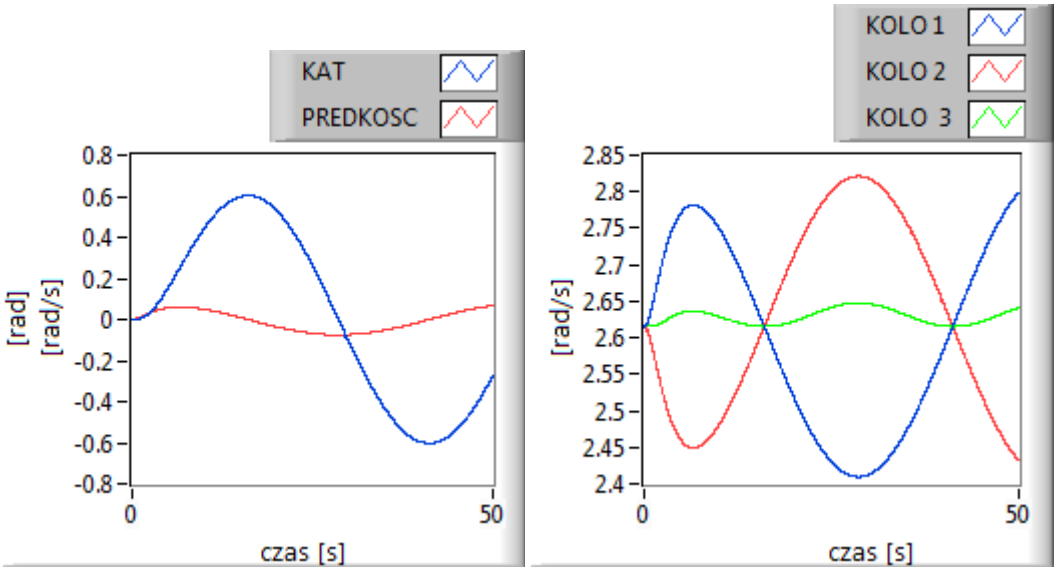
Rys. 3.9. Zapisana prędkość kątowa koła 2 α_2 platformy mobilnej



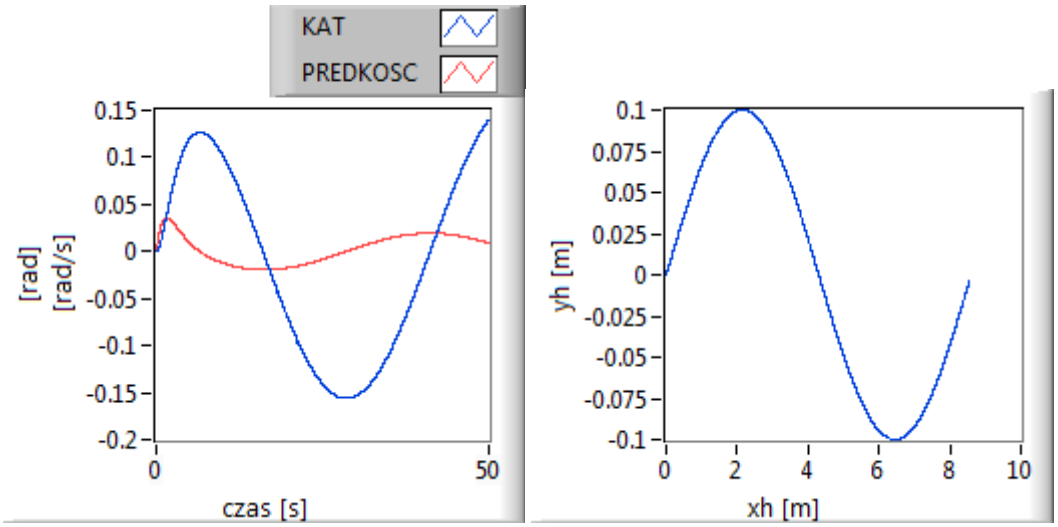
Rys. 3.10. Zapisana prędkość kątowa koła 3 α_3 platformy mobilnej

Podobnie jak to miało miejsce w środowisku Maple, wygenerowano zapisane w środowisku LabVIEW rozwiązania graficzne dla otrzymanego układu równań (3.15).

Wykreślony w środowisku LabVIEW chwilowy kąt obrotu platformy, jak i jego pochodna (rys. 3.11, lewy) jest całkowicie zbieżny z tym otrzymanym za pomocą środowiska Maple. Warto zauważyć, że warunkiem koniecznym realizacji kolejnych faz projektowania systemu nadzorowania trójkołowej platformy mobilnej, było poprawne rozwiązanie równań różniczkowych modelowania ruchu platformy. Otrzymanie identycznych rezultatów w obu zastosowanych platformach programistycznych utwierdziło autora w przekonaniu o poprawnym rozwiązaniu oraz zapisaniu rozwiązań do środowiska docelowego LabVIEW. Niezwykle cennym dla porównania obu rozwiązań było ustalenie tej samej metody numerycznej tj. metody Eulera oraz przyjęcie identycznego kroku całkowania (0,005 s). Rys. 3.11 (prawy) oraz rys. 3.12 (lewy i prawy) potwierdzają powyższe rozważania o poprawności otrzymanych rezultatów oraz właściwej implementacji rozwiązań otrzymanych drogą symboliczną do środowiska LabVIEW.



Rys. 3.11. Kąt obrotu platformy (lewy). Prędkości kątowe kół platformy (prawy)



Rys. 3.12. Kąt obrotu kierownicy (lewy). Tor ruchu punktu charakterystycznego (prawy)

3.3.2. Realizacja trajektorii typu „parabola”

W przypadku, gdy platforma porusza się po trajektorii typu „parabola”, funkcja opisująca ruch punktu charakterystycznego przyjmuje zależność:

$$y_H = x_H^4 . \tag{3.16}$$

Dla tak określonej trajektorii ruchu zapisano równania różniczkowe, które następnie rozwiązano w środowisku Maple. Należy zaznaczyć, że równania prędkości kątowych poszczególnych kół platformy, a także funkcja opisująca kąt położenia platformy nie są jawnie uzależnione od charakteru trajektorii ruchu. Dlatego

otrzymane zależności są tożsame z zależnościami otrzymanymi dla trajektorii typu „sinus” (3.15). Otrzymany układ rozwiązań równań różniczkowych można wyrazić następującymi zależnościami:

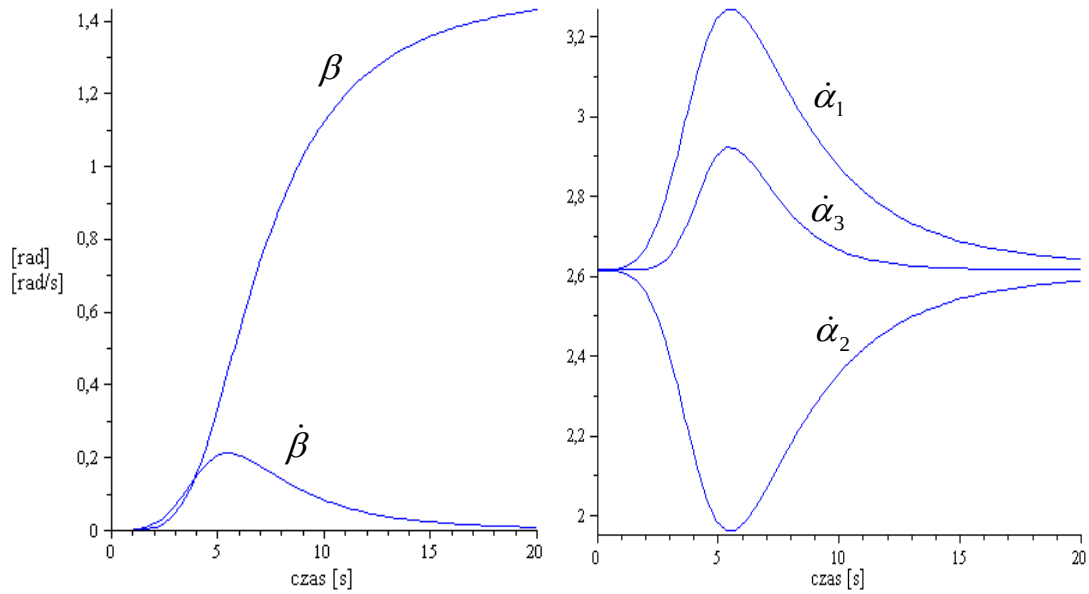
$$\begin{cases} \dot{\varphi} = -\left[\; v_A(l_0 \sin\beta - l_3\, tg\varphi \cos(\beta+\varphi) - 4x_H^3 l_0 \cos\beta + 4x_H^3 l_0 \sin\beta \; tg\varphi + \right. \\ \left. + 4x_H^3 l_3 \sin(\beta+\varphi) tg\varphi + l_0 \cos\beta \; tg\varphi \; \right] / \left[\; l_3 l_0 (4x_H^3 \sin(\beta+\varphi) + \cos(\beta+\varphi)) \; \right], \\ \dot{\beta} = \dfrac{v_A\, tg\varphi}{l_0}, \\ \dot{x}_H = \left[\; v_A(\sin(\beta+\varphi)\sin\beta + \sin(\beta+\varphi)\cos\beta\, tg\varphi + \cos(\beta+\varphi)\cos\beta - \right. \\ \left. - \cos(\beta+\varphi)\cos\beta\, tg\varphi) \; \right] / \left[\; 4x_H^3 \sin(\beta+\varphi) + \cos(\beta+\varphi) \; \right], \\ \dot{\alpha}_1 = \dfrac{v_A(l_1\, tg\varphi + l_0)}{r_1 l_0}, \\ \dot{\alpha}_2 = -\dfrac{v_A(l_1\, tg\varphi - l_0)}{r_1 l_0}, \\ \dot{\alpha}_3 = -\dfrac{v_A(tg\varphi \sin\beta - \cos\beta)}{r_1 \cos(\beta+\varphi)}. \end{cases} \tag{3.17}$$

Podobnie, jak to miało miejsce w przypadku rozważań dla trajektorii typu „sinus”, do dalszej analizy zastosowano metody numeryczne. W obu przypadkach (dla Maple i LabVIEW) autor przyjął krok całkowania równy 0,005 s. Autor ustalił, iż platforma dla trajektorii typu „parabola” będzie poruszała się po torze opisanym zależnością (3.16). Jest to funkcja czwartego rzędu, której wartości y rosną szybciej, co bezpośrednio determinuje profil pokonywanego łuku.

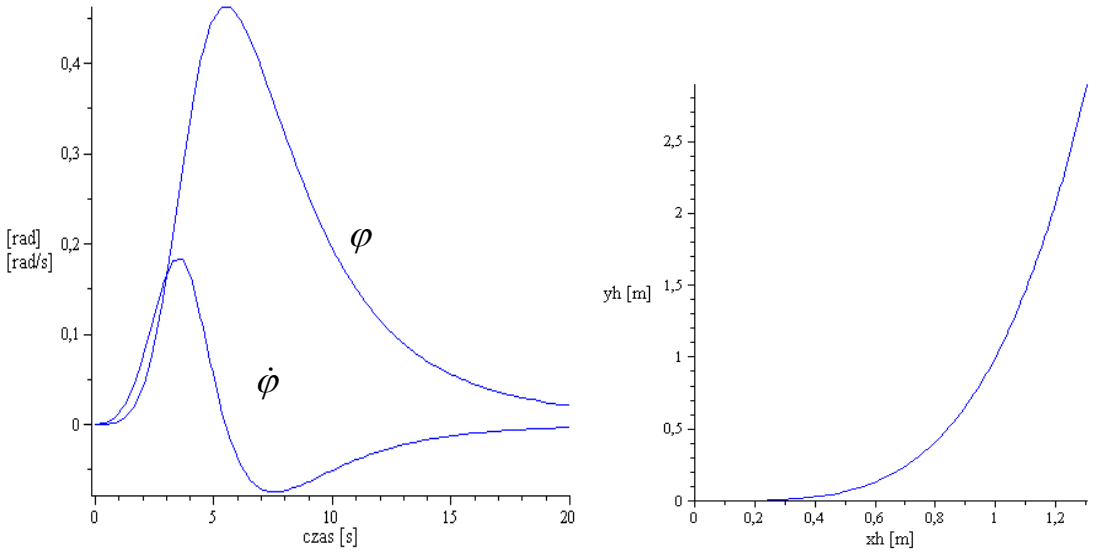
W trakcie 20 s przejazdu, platforma mobilna pokonuje trasę wykreśloną na (rys. 3.14 prawy). W tym czasie chwilowy kąt obrotu ramy (rys. 3.13 lewy) doznaje przemieszczenia zgodnie w wykreślonym rozwiązaniem równania różniczkowego. Otrzymana krzywa ma charakter bardzo łagodny, co bezpośrednio wpływa na komfort przejazdu (w przyszłości podobna platforma może zostać użyta do transportu ludzi bądź materiałów niebezpiecznych, czy kruchych). Dodatkowo, co jest niezwykle istotne z punktu widzenia użytkowania robotów mobilnych, przebyta w łagodny sposób ustalona trajektoria wpływa decydująco na optymalne wykorzystanie zasobów energetycznych platformy (nagłe szarpnięcia są przyczyną powstania odchyłek od ustalonego toru jazdy, co skutkuje dodatkowym poborem mocy niezbędnej do powrotu na dany tor przejazdu, a także, co jest domeną coraz częściej stosowanych

napędów elektrycznych, że w trakcie gwałtownych zmian prędkości bądź rozruchu, pobór prądu (wyjątkiem są napędy bazujące na przemiennikach częstotliwości, tzw. *softstarty*) wzrasta kilkakrotnie. Szersza dyskusja poświęcona energetycznym zagadnieniom projektowanego systemu nadzorowania platformy mobilnej została przeprowadzona w kolejnych rozdziałach pracy.

Na (rys. 3.13 prawy) przestawiono chwilowe zmiany prędkości kątowych platformy. Otrzymane krzywe są efektem charakteru przejazdu platformy po rozpatrywanej trajektorii. Występujące w 5,5 s wartości ekstremalne osiąganych prędkości kątowych kół platformy świadczą o fakcie pokonywania w tym czasie największego promienia krzywizny ustalonej trasy przejazdu. Również chwilowa wartość prędkości kąta obrotu platformy osiąga swoje maksimum w 5,5 s (rys. 3.13 lewy). W przypadku chwilowych zmian prędkości kątowej kierownicy (rys. 3.14 lewy) wartości ekstremalne osiągane są w 3,5 s. Tak, jak to autor zaznaczył poprzednio, wartość kąta skrętu platformy, na skutek jej geometrii działa z pewnym opóźnieniem w stosunku do skrętu kierownicy, ale jednak skutki jej działania są bardziej zauważalne (większe wartości kąta β niż kąta φ, por. rys. 3.13 lewy). Warto zwrócić uwagę, że wszystkie (rys. 3.13 oraz rys. 3.14) otrzymane rozwiązania, mają charakter krzywych o łagodnych i równomiernych zboczach narastania oraz opadania. Platforma pokonuje założoną trasę w optymalny i precyzyjny sposób (brak gwałtownych przyspieszeń mogących być przyczyną utraty stabilności).



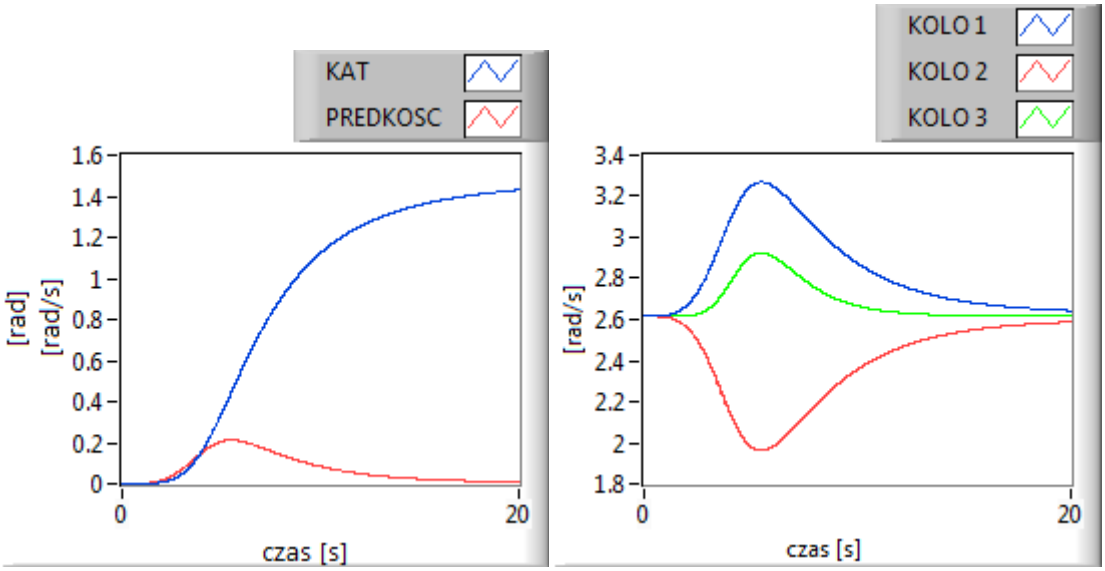
Rys. 3.13. Kąt obrotu platformy (lewy). Prędkości kątowe kół platformy (prawy)



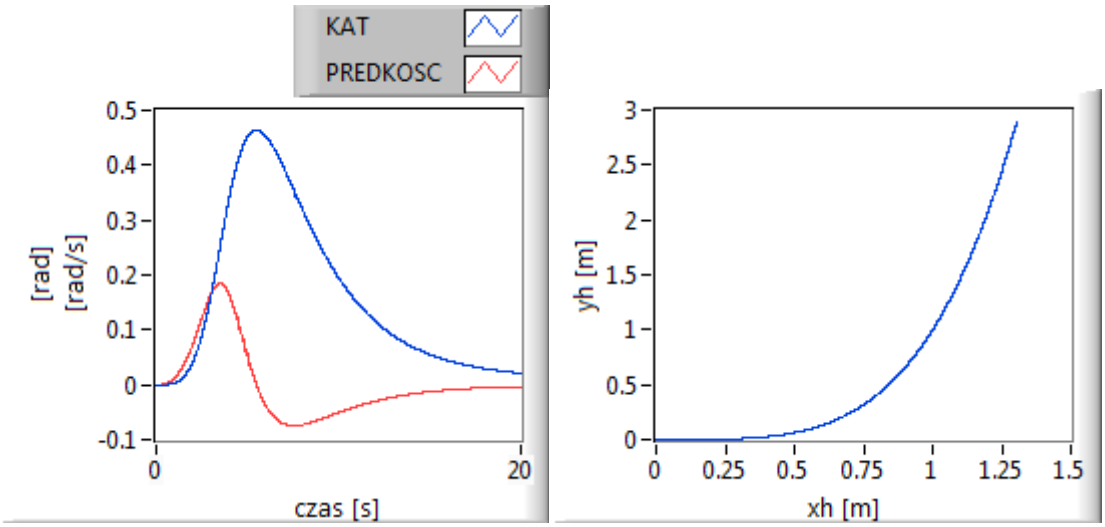
Rys. 3.14. Kąt obrotu kierownicy (lewy). Tor ruchu punktu charakterystycznego (prawy)

Podobnie, jak to miało miejsce dla trajektorii typu „sinus", otrzymane rozwiązania drogą przekształceń symbolicznych w Maple zapisano do środowiska LabVIEW. Kolejno przeprowadzając symulacje w module CD&S otrzymano rozwiązania graficzne modelowanych równań (rys. 3.15. oraz rys. 3.16).

Wykreślony w środowisku LabVIEW chwilowy kąt obrotu platformy, jak i jego pochodna (rys. 3.15 lewy) są całkowicie zbieżne, z tymi otrzymanymi za pomocą środowiska Maple. Otrzymanie identycznych rezultatów w obu zastosowanych platformach programistycznych utwierdziło autora w przekonaniu (jak to miało miejsce dla trajektorii typu „sinus") o poprawnym rozwiązaniu oraz zaimplementowaniu rozwiązań do środowiska docelowego LabVIEW. Niezwykle cennym dla porównania obu rozwiązań, było ustalenie tej samej metody numerycznej (tj. metody Eulera) oraz przyjęcie identycznego kroku całkowania (0,005 s). Rys. 3.15 (lewy i prawy) oraz rys. 3.16 (lewy i prawy), potwierdzają powyższe rozważania o poprawności otrzymanych rezultatów oraz właściwej implementacji rozwiązań otrzymanych drogą symboliczną w środowisku LabVIEW.



Rys. 3.15. Kąt obrotu platformy (lewy). Prędkości kątowe kół platformy (prawy)



Rys. 3.16. Kąt obrotu kierownicy (lewy). Tor ruchu punktu charakterystycznego (prawy)

3.3.3. Realizacja trajektorii typu „okrąg”

Schemat postępowania przy tworzeniu równań kinematyki został szczegółowo opisany w 3.3. W przypadku, gdy platforma porusza się po trajektorii typu „okrąg”, funkcja opisująca ruch punktu charakterystycznego opisana jest zależnością (3.2) tj. $x_h = R\sin\theta$, $y_h = R(1 - \cos\theta)$.

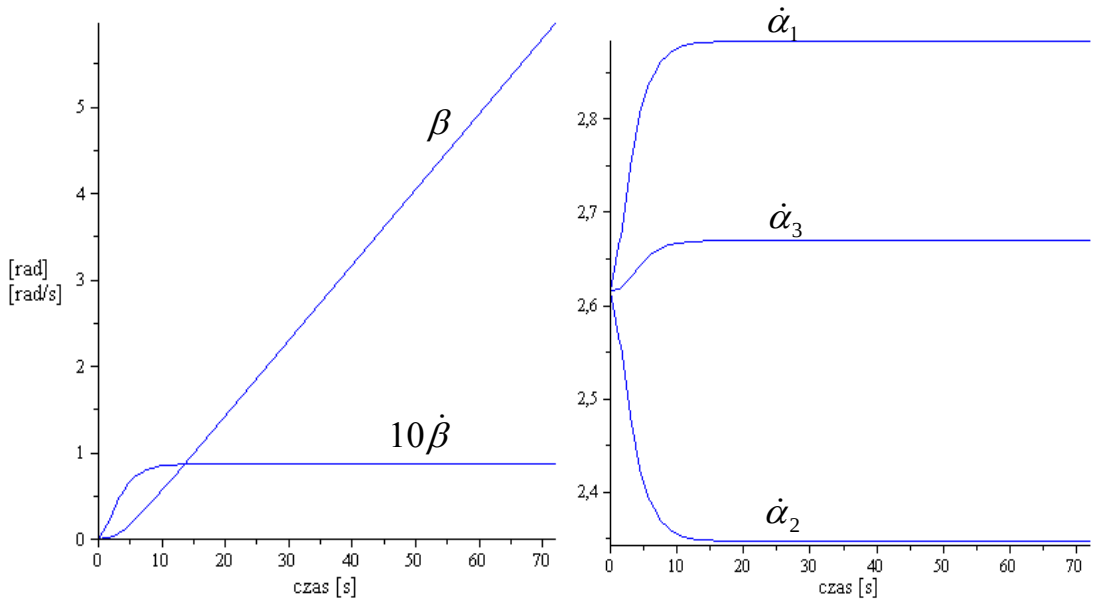
Podobnie, jak to miało miejsce dla trajektorii typu „sinus” oraz „parabola”, równania różniczkowe (3.5) oraz (3.13) rozwiązano w środowisku Maple. Otrzymamy wówczas zależności:

$$\begin{cases} \dot{\varphi} = -[\ v_A(-l_0\cos\beta\sin\theta - l_3\,tg\varphi\cos(\beta+\varphi) + l_0\,tg\varphi\,\sin\beta\,\sin\theta + \\ + l_3\,tg\varphi\,\sin(\beta+\varphi)\sin\theta + l_0\sin\beta\cos\theta + l_0\,tg\varphi\cos\beta\cos\theta + \\ + l_3\cos\theta\,tg\varphi\cos(\beta+\varphi))\ ]/[\ l_3 l_0(\sin(\beta+\varphi)\sin\theta + \cos(\beta+\varphi)\cos\theta)\ ], \\ \dot{\beta} = \dfrac{v_A\,tg\varphi}{l_0}, \\ \dot{\theta}\ = [\ v_A(\cos(\beta+\varphi)\cos\beta - \cos(\beta+\varphi)tg\varphi\,\sin\beta + \sin(\beta+\varphi)\sin\beta + \\ + \sin(\beta+\varphi)tg\varphi\cos\beta)\ ]/[\ R(\sin(\beta+\varphi)\sin\theta + \cos(\beta+\varphi)\cos\theta)\ ], \\ \dot{\alpha}_1 = \dfrac{v_A(l_1\,tg\varphi + l_0)}{r_1 l_0}, \\ \dot{\alpha}_2 = -\dfrac{v_A(l_1\,tg\varphi - l_0)}{r_1 l_0}, \\ \dot{\alpha}_3 = -\dfrac{v_A(tg\varphi\sin\beta - \cos\beta)}{r_1\cos(\beta+\varphi)}. \end{cases} \tag{3.18}$$

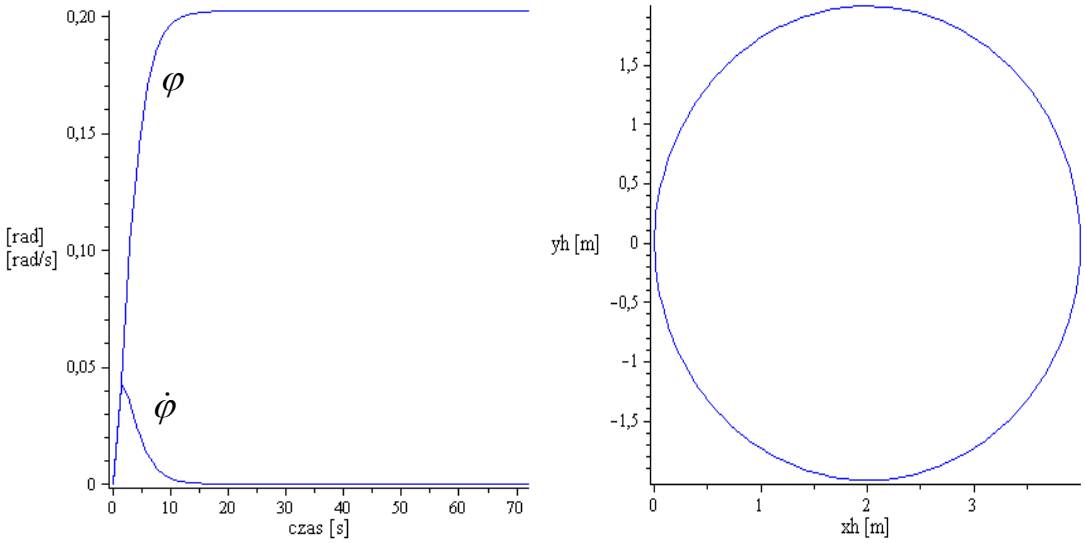
Poszukiwanie rozwiązań zależności opisanych układem równań (3.18) wiązało się z zastosowaniem metod numerycznych. Podobnie, jak to miało miejsce poprzednio, autor skorzystał z metody stałokrokowej Eulera oraz przyjął krok całkowania 0,005 s. Otrzymane rezultaty przeprowadzonych obliczeń przedstawiono na rys. 3.17 oraz rys. 3.18. Czas trwania przejazdu (72 s) został tak dobrany, aby w tym okresie platforma zatoczyła jedno pełne koło o promieniu 2 m (rys. 3.18 prawy). Ze względu na charakterystyczny tor przejazdu, kąt obrotu platformy od 5 s przejazdu wzrasta liniowo (rys. 3.17 lewy), natomiast prędkość od tej samej chwili czasu przyjmuje wartość stałą (platforma porusza się w jednej ustalonej pozycji). Ze

względu na bardzo małą prędkość obrotu ramy, autor chcąc zademonstrować rezultaty obu przebiegów (kąt i jego pochodną) dokonał pomnożenia każdej otrzymanej wartości prędkości dziesięciokrotnie. Aktorem otrzymanych wyników, dla parametrów obrotu ramy, jest wartość chwilowego kąta obrotu kierownicy, zadającą tor jazdy platformy. Również w tym przypadku (rys. 3.18 lewy), ze względu na charakter trajektorii, po której porusza się platforma, wartość kąta obrotu kierownicy jest od około 8 s trwania przejazdu stała (kierownica nie zmienia swojego położenia). Prędkość obrotu kierownicy po tym czasie maleje niemalże do zera i utrzymuje wartość stałą na pozostałym odcinku trasy. Wyznaczone wartości kąta kierownicy, jak i ramy platformy, mają bezpośredni wpływ na przebiegi chwilowych prędkości obrotu kół (rys. 3.17 prawy), które od około 10 s czasu przejazdu mają wartość stałą (brak przyspieszenia). Ze względu na wspomniany wcześniej układ różnicowy przenoszący napęd na koła 1 i 2, wartości prędkości tych kół są symetryczne względem obrotu wału silnika napędowego DC.

Z otrzymanych rozwiązań graficznych parametrów platformy wynika, że platforma pokonuje trasę optymalnie, nie zużywając zasobów energetycznych na ewentualną korektę trasy przejazdu (które mogą być nieodzowne w momencie powstania szarpnięć, gwałtownych zmian kierunku, gwałtownych przyspieszeń lub utraty stabilności).

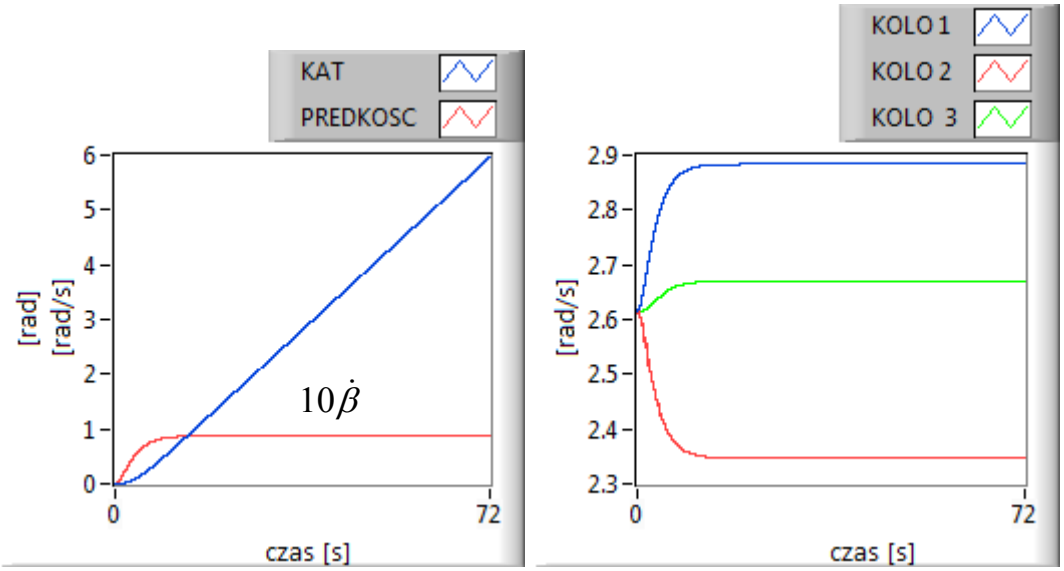


Rys. 3.17. Kąt obrotu platformy (lewy). Prędkości kątowe kół platformy (prawy)

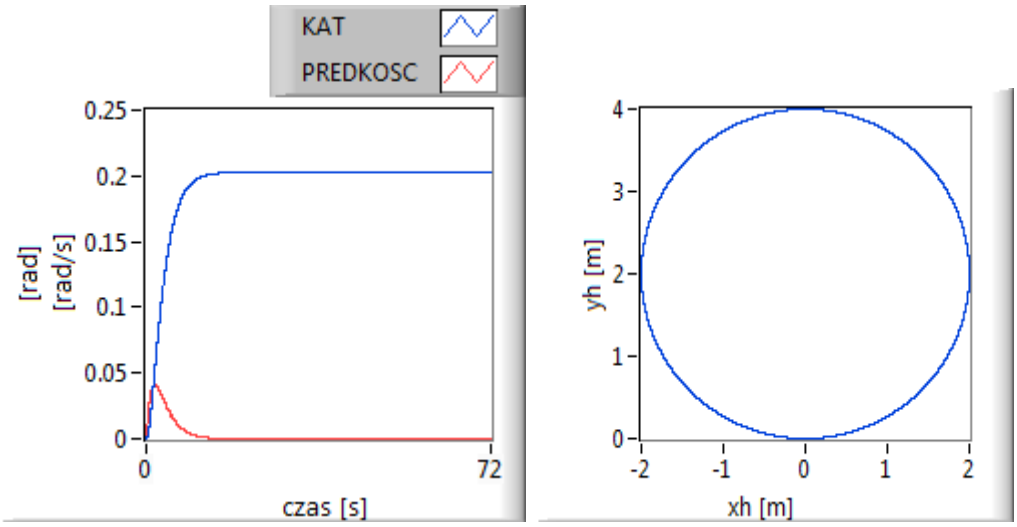


Rys. 3.18. Kąt obrotu kierownicy (lewy). Tor ruchu punktu charakterystycznego (prawy)

W rzeczywistości, o czym autor wspominał, system sterowania platformą zostanie zbudowany na bazie środowiska LabVIEW, skąd zostaną wygenerowane optymalne sygnały sterujące. Fakt ten warunkuje poprawne zapisanie równań kinematyki tak, aby platforma poruszała się po właściwych trajektoriach ruchu. Wykreślone przebiegi (rys. 3.19 i 3.20) świadczą o poprawnym zapisaniu równań, na bazie których otrzymano rozwiązanie odwrotnego zadania kinematyki. Wartości chwilowych parametrów platformy są tożsame z wartościami dla przebiegów otrzymanych w środowisku Maple. Autor, podobnie jak to miało miejsce dla trajektorii typu „sinus" oraz „parabola", do rozwiązania równań różniczkowych wykorzystał metodę numeryczną Eulera ustawiając krok całkowania 0,005 s.



Rys. 3.19. Kąt obrotu platformy (lewy). Prędkości kątowe kół platformy (prawy)



Rys. 3.20. Kąt obrotu kierownicy (lewy). Tor ruchu punktu charakterystycznego (prawy)

3.3.4. Podsumowanie wyników symulacji komputerowych

Przeprowadzone symulacje komputerowe kinematyki platformy mobilnej umożliwiły autorowi, w dość krótkim okresie czasu, uzyskanie w docelowym środowisku LabVIEW odpowiedzi w postaci przebiegów chwilowych parametrów platformy na zadane trajektorie ruchu. Wykorzystane dodatkowo środowisko Maple, przy swojej wielkiej zalecie operowania na zmiennych symbolicznych (parametrycznych rozwiązań równań różniczkowych) zostało również wykorzystane, do przeprowadzenia obliczeń numerycznych, których rezultaty w późniejszym etapie

zastosowano podczas weryfikacji poprawności implementacji rozwiązań odwrotnego zadania kinematyki do środowiska LabVIEW.

Symulacje komputerowe kinematyki układu znacząco skróciły czas weryfikacji tras przyjazdu platformy, a także wartości – jej chwilowych parametrów. Znaczącej optymalizacji uległa również przyjęta koncepcja powstającego równolegle obiektu badań. Niezwykle istotnym elementem docelowej platformy programistycznej LabVIEW, dla realizowanego przedsięwzięcia naukowego, jest fakt powstania na każdym etapie rozwoju oprogramowania mechatronicznego interfejsu użytkownika, umożliwiającego szybką korektę, bądź ewentualną zmianę ustawiań. W tym przypadku, swoboda doboru nastaw umożliwiła rozważeniu różnych prędkości przejazdu platformy, a także konfiguracji trasy jej przejazdu. Dodatkowo, autor optymalizując konstrukcje platformy, dobierał różne jej parametry (długość, szerokość ramy, rozstaw kół oraz osi itp.) tak, aby zapewnić platformie na całej długości realizowanej trasy stabilną oraz płynna jazdę.

Dokonując analizy otrzymanych wyników przeprowadzonych symulacji, a także mając na uwadze silną nieliniowość układu, należy wnioskować, że w trakcie przejazdów wirtualnych, projektowana platforma porusza po założonych trajektoriach. Przyjmowane wartości parametrów platformy są wynikiem oczekiwanych zjawisk fizycznych. Niedostrzegalne są sytuacje (np. ewentualne zmiany chwilowej wartości parametru) mogące być przyczyną utraty stabilności, kontaktu kół z podłożem lub ewentualnych poślizgów.

We wszystkich zrealizowanych przejazdach otrzymane drogą środowiska Maple oraz środowiska LabVIEW rozwiązania cechuje całkowita zbieżność.

3.4. Dynamika platformy mobilnej

W drugim etapie symulacji komputerowych, autor przeprowadził analizę dynamiki układu, niezbędną w procesie zapewnienia strategii sterowania trójkołową platformą mobilną i polegającą na ustaleniu właściwych sygnałów sterujących.

Rozwiązanie odwrotnego zadania dynamiki pozwala na określenie pożądanych sygnałów sterujących (w tym przypadku momentów napędowych silników DC, które będą w trakcie stosowania technik projektowania mechatronicznego wartościami referencyjnymi, dla realizowanego systemu nadzorowania platformą mobilną), a także – jest podstawą określenia niezbędnych parametrów (macierzy) dla zdefiniowanego w następnym rozdziale algorytmu sterowania platformą mobilną, bazującego na energetycznym wskaźniku jakości. Utworzenie właściwego modelu jest o tyle istotne, że otrzymane w procesie analizy momenty napędowe będą bezpośrednio korespondowały z jakością sterowania systemu, a także wpływały na realizację przez platformę zadanej trajektorii oraz na wydajność energetyczną systemu. Autor przeprowadził badania nad dynamiką układu dla identycznych trajektorii oraz czasów przejazdu, co w przypadku rozwiązania odwrotnego zadania kinematyki.

Autor w trakcie przeprowadzonych badań postanowił do opisu ruchu platformy posługiwać się równaniami Lagrange`a II rodzaju. W przypadku obiektu nieholonomicznego, jakim jest badana platforma mobilna, ruch jej może być wyrażony równaniem [34]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathrm{T}}{\partial \dot{\mathbf{q}}}\right)^{\mathrm{T}} - \left(\frac{\partial \mathrm{T}}{\partial \mathbf{q}}\right)^{\mathrm{T}} = \mathbf{f} + \mathbf{B}(\mathbf{q})\mathbf{u} + \mathbf{J}^{\mathrm{T}}(\mathbf{q})\boldsymbol{\lambda}, \tag{3.19}$$

gdzie: $\mathbf{q} = [x_A, y_A, \beta, \alpha, \psi, \varphi]^T$ – wektor współrzędnych uogólnionych , $T = T(\mathbf{q}, \dot{\mathbf{q}})$ – energia kinetyczna, **B –** macierz wejść, **f** – wektor zakłóceń, **u –** wektor sygnałów sterujących **J**(**q**) – macierz Jakobianu, $\boldsymbol{\lambda}$ – wektor mnożników Lagrange`a, ψ **–** kąt obrotu koła zastępczego kół 1 i 2.

Równanie (3.19) umożliwia rozwiązanie prostego i odwrotnego zadania dynamiki oraz wyznaczenie mnożników Lagrange`a, będących w analizowanym przypadku siłami tarcia działającymi w płaszczyźnie styczności kół z jezdnią.

Mając na uwadze fakt nieholoniczności rozpatrywanego układu (więzy zależne od prędkości) oraz rozważając rys. 3.2, na którym przedstawiano model zastępczy platformy, równania opisujące więzy nałożone na prędkości współrzędnych (x_A, y_A, β, α, ψ, φ) mogą zostać opisane układem równań:

$$\begin{cases} \dot{x}_A - r\dot{\psi}\cos\beta = 0, \\ \dot{y}_A - r\dot{\psi}\sin\beta = 0, \\ \dot{x}_A - l\dot{\beta}\sin\beta - r\dot{\alpha}\cos(\beta+\alpha) = 0, \\ \dot{y}_A + l\dot{\beta}\cos\beta - r\dot{\alpha}\sin(\beta+\alpha) = 0. \end{cases} \tag{3.20}$$

Równania więzów nieholonomicznych dla rozważanej platformy można zapisać w postaci:

$$\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0}. \tag{3.21}$$

Poddając analizie równanie (3.20) oraz (3.21) można określić Jakobian, który może być opisany, jako:

$$\mathbf{J}(\mathbf{q}) = \begin{bmatrix} 1 & 0 & 0 & 0 & -r\cos\beta & 0 \\ 0 & 1 & 0 & 0 & -r\sin\beta & 0 \\ 1 & 0 & -l\sin\beta & -r\cos(\beta+\alpha) & 0 & 0 \\ 0 & 1 & l\cos\beta & -r\sin(\beta+\alpha) & 0 & 0 \end{bmatrix}. \tag{3.22}$$

Równania różniczkowe opisujące dynamikę rozważanego układu mogą być zapisane w zwięzłej formie poprzez równanie macierzowe, wyrażone zależnością:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{L}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} = \mathbf{f} + \mathbf{B}(\mathbf{q})\mathbf{u} + \mathbf{J}^T(\mathbf{q})\boldsymbol{\lambda}, \tag{3.23}$$

gdzie: $\mathbf{M}(\mathbf{q})$ – macierze bezwładności układu, $\mathbf{L}(\mathbf{q},\dot{\mathbf{q}})$ – macierz sił żyroskopowych i Coriolisa.

Operacja odsprzęgania mnożników Lagrange`a pozwala na zapisanie macierzowego równania dynamiki projektowanej platformy w formie bardziej dogodnej [38], niż to było prezentowane w pracach [20, 59], tj.:

$$\begin{cases} \mathbf{M}^*\ddot{\mathbf{q}}_\mathrm{i} + \mathbf{L}^*\dot{\mathbf{q}}_\mathrm{i} = \mathbf{f}^* + \mathbf{B}^*\mathbf{u} \\ \mathbf{M}_{\mathrm{di}}\ddot{\mathbf{q}}_\mathrm{i} + \mathbf{M}_{\mathrm{dd}}\ddot{\mathbf{q}}_\mathrm{d} + \mathbf{L}_{\mathrm{di}}\dot{\mathbf{q}}_\mathrm{i} + \mathbf{L}_{\mathrm{dd}}\dot{\mathbf{q}}_\mathrm{d} = \mathbf{f}_\mathrm{d} + \mathbf{B}_{\mathrm{ud}}\mathbf{u} + \mathbf{J}_\mathrm{d}^\mathrm{T}\boldsymbol{\lambda}, \end{cases} \tag{3.24}$$

gdzie:

$$\mathbf{M}^* = \mathbf{M}_{\mathrm{ii}} + \mathbf{M}_{\mathrm{id}}\mathbf{J} - \mathbf{J}_\mathrm{i}^\mathrm{T}\mathbf{J}_\mathrm{d}^{-\mathrm{T}}\mathbf{M}_{\mathrm{di}} - \mathbf{J}_\mathrm{i}^\mathrm{T}\mathbf{J}_\mathrm{d}^{-\mathrm{T}}\mathbf{M}_{\mathrm{dd}}\mathbf{J}, \tag{3.25}$$

$$\mathbf{L}^* = \mathbf{L}_{\mathrm{ii}} + \mathbf{L}_{\mathrm{id}}\mathbf{J} - \mathbf{J}_\mathrm{i}^\mathrm{T}\mathbf{J}_\mathrm{d}^{-\mathrm{T}}\mathbf{L}_{\mathrm{di}} - \mathbf{J}_\mathrm{i}^\mathrm{T}\mathbf{J}_\mathrm{d}^{-\mathrm{T}}\mathbf{L}_{\mathrm{dd}}\mathbf{J} + \mathbf{M}_{\mathrm{id}}\dot{\mathbf{J}} - \mathbf{J}_\mathrm{i}^\mathrm{T}\mathbf{J}_\mathrm{d}^{-\mathrm{T}}\mathbf{M}_{\mathrm{dd}}\dot{\mathbf{J}}, \tag{3.26}$$

$$\mathbf{B}^{*} = \mathbf{B}_{\mathrm{ui}} - \mathbf{J}_{\mathrm{i}}^{\mathrm{T}} \mathbf{J}_{\mathrm{d}}^{-\mathrm{T}} \mathbf{B}_{\mathrm{ud}}, \tag{3.27}$$

$$\mathbf{f}^{*} = \mathbf{f}_{\mathrm{i}} - \mathbf{J}_{\mathrm{i}}^{\mathrm{T}} \mathbf{J}_{\mathrm{d}}^{-\mathrm{T}} \mathbf{f}_{\mathrm{d}}. \tag{3.28}$$

Wektor współrzędnych uogólnionych **q** może zostać przedstawiony w postaci,

$$\mathbf{q} = [\ \mathbf{q}_{\mathrm{i}}^{\ T}, \mathbf{q}_{\mathrm{d}}^{\ T}\ ]^{T}, \tag{3.29}$$

gdzie:

$\mathbf{q_i}$ – wektor współrzędnych uogólnionych niezależnych,

$\mathbf{q_d}$ – wektor współrzędnych uogólnionych zależnych.

Dalej, podstawiając równanie (3.29) do równania (3.21), równanie więzów nieholonomicznych może być wyrażone jako:

$$\begin{bmatrix} \mathbf{J}_{\mathrm{i}} & \mathbf{J}_{\mathrm{d}} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_{\mathrm{i}} \\ \dot{\mathbf{q}}_{\mathrm{d}} \end{bmatrix} = \mathbf{0}, \tag{3.30}$$

gdzie:

$$\dot{\mathbf{q}}_{\mathrm{d}} = \underbrace{-\mathbf{J}_{\mathrm{d}}^{-1} \mathbf{J}_{\mathrm{i}}}_{\mathbf{J}} \dot{\mathbf{q}}_{\mathrm{i}} = \mathbf{J}\dot{\mathbf{q}}_{\mathrm{i}}, \tag{3.31}$$

$$\ddot{\mathbf{q}}_{\mathrm{d}} = \dot{\mathbf{J}}\,\dot{\mathbf{q}}_{\mathrm{i}} + \dot{\mathbf{J}}\,\ddot{\mathbf{q}}_{\mathrm{i}}. \tag{3.32}$$

Pierwsze równanie układu (3.24) umożliwia przeprowadzenie analizy prostego i odwrotnego zadania dynamiki bez konieczności stosowania mnożników Lagrange`a. Dla tak zdefiniowanego układu, pamiętając że $\mathbf{q_d} = [x_A, y_A, \beta, \alpha]^T$ oraz $\mathbf{q_i} = [\psi, \varphi]$ rozpatrywane równanie macierzowe można zapisać w postaci układu dwóch równań różniczkowych:

$$\begin{aligned}
&\left[\left(m_1 + m_2 + m_3 + m_5\right) r^2 + I_{z1} + I_{z2} + I_{z3} \frac{1}{\cos^2 \varphi}\right] \ddot{\psi} + I_{x3} \frac{r}{l_0} \ddot{\varphi} tg\varphi + \\
&+ \left[I_{x1} + I_{x2} + I_{x3} + 2I_{z1} h_1^2 + I_{z5} + \left(m_1 + m_2\right) l_1^2 + m_3 {l_0}^2 + m_5 l_2^2\right] \frac{r^2}{{l_0}^2} \ddot{\psi}\, tg^2\varphi + \\
&+ \left\{\left[I_{x1} + I_{x2} + I_{x3} + 2I_{z1} h_1^2 + I_{z5} + \left(m_1 + m_2\right) l_1^2 + m_3 {l_0}^2 + m_5 l_2^2\right] \frac{r^2}{{l_0}^2} + I_{z3}\right\} \dot{\psi}\, \dot{\varphi} \frac{tg\varphi}{\cos^2 \varphi} = \\
&= M_N - N_1 f_1 - N_2 f_2 - N_3 f_3 \frac{1}{\cos\varphi} + \left(M_S - M_0 \operatorname{sgn} \dot{\varphi}\right) \frac{r}{l_0} tg\varphi,
\end{aligned} \tag{3.33}$$

$$I_{x3} \frac{r}{l_0} \ddot{\psi}\, tg\varphi + I_{x3} \ddot{\varphi} + I_{x3} \frac{r}{l_0} \frac{1}{\cos^2 \varphi} \dot{\psi}\, \dot{\varphi} = M_S - M_0 \operatorname{sgn} \dot{\varphi}, \tag{3.34}$$

gdzie : m_1, m_2, m_3, m_5 – masy poszczególnych części układu tj. kolejno koła 1, koła 2, koła 3 oraz ramy; I_{x1}, I_{x2}, I_{x3}, I_{z1}, I_{z2}, I_{z3}, I_{z5} – masowe momenty bezwładności części (odpowiednio koła 1, koła 2, koła 3, ramy platformy) określone względem odpowiednich osi (x oraz z); f_1, f_2, f_3, – współczynniki toczenia odpowiednich kół 1, 2 oraz 3; N_1, N_2, N_3 – siły nacisku odpowiednich kół 1,2 oraz 3; M_0 – moment oporu w parze ciernej koło-jezdnia, występujący przy skręcie koła 3; M_S – moment kierujący; M_N – moment napędzający; l_0, l_1, r – parametry geometryczne platformy.

Dla tak przyjętego modelu dynamiki układu (zależności (3.33) oraz (3.34)) została określona macierz bezwładności:

$$\mathbf{M} = \begin{bmatrix} \left\{ \left[\left(m_1 + m_2 + m_3 + m_5\right) r^2 + I_{z1} + I_{z2} + I_{z3} \dfrac{1}{\cos^2 \varphi} \right] + \right. & \\ \left[I_{x1} + I_{x2} + I_{x3} + 2I_{z1} \left(\dfrac{l_1}{r} \right)^2 + I_{z5} + \right. & I_{x3} \dfrac{r}{l_0} tg\varphi \\ \left. + \left(m_1 + m_2\right) l_1^2 + m_3 {l_0}^2 + m_5 l_2^2 \right] \left(\dfrac{r}{l_0} \right)^2 tg^2 \varphi \; \right\} & \\ I_{x3} \dfrac{r}{l_0} tg\varphi & I_{x3} \end{bmatrix}, \tag{3.35}$$

macierz efektów sił odśrodkowych bezwładności i efektów żyroskopowych:

$$\mathbf{L} = \begin{bmatrix} \left\{ \left\{ \left[I_{x1} + I_{x2} + I_{x3} + 2I_{z1} \left(\dfrac{l_1}{r} \right)^2 + I_{z5} + \right. \right. \right. & \\ \left. \left. + \left(m_1 + m_2\right) l_1^2 + m_3 l^2 + m_5 l_2^2 \right] \left(\dfrac{r}{l_0} \right)^2 + I_{z3} \; \right\} & 0 \\ \left. \cdot \dot{\varphi} \dfrac{tg\varphi}{\cos^2 \varphi} \; \right\} & \\ 0 & I_{x3} \dfrac{r}{l_0} \dfrac{1}{\cos^2 \varphi} \dot{\psi} \end{bmatrix}, \tag{3.36}$$

macierz sterowań układu:

$$\mathbf{B}_u = \begin{bmatrix} \dfrac{r}{l_0} tg\varphi & 1 \\ 1 & 0 \end{bmatrix}, \tag{3.37}$$

macierz zakłóceń:

$$\mathbf{f} = \begin{bmatrix} -N_1 f_1 - N_2 f_2 - N_3 f_3 \dfrac{1}{\cos\varphi} - M_0 \operatorname{sgn}\dot{\varphi}\dfrac{r}{l_0} tg\varphi \\ -M_0 \operatorname{sgn}\dot{\varphi} \end{bmatrix}, \tag{3.38}$$

oraz macierz sygnałów sterujących:

$$\mathbf{u} = \begin{Bmatrix} M_S \\ M_N \end{Bmatrix}. \tag{3.39}$$

Otrzymany model obliczeniowy dynamiki układu (równania (3.33) oraz (3.34)) jest bardzo złożony. Analiza jego jest bardzo niewygodna i nastręcza wiele trudności. Warto zauważyć, że wszystkie określone powyżej macierze platformy są silnie nieliniowe, uzależnione od kąta skrętu kierownicy, a także od kąta obrotu koła zastępczego (macierz **L**). Fakt ten determinuje potrzebę poszukiwania optymalnych rozwiązań systemu sterowania, a także stosowania metod i narzędzi (w tym przypadku Maple, LabVIEW) projektowania mechatronicznego mogącego optymalizować określoną koncepcję oraz przyjęte założenia.

Ze względu na skomplikowany proces zapisu równań symbolicznych, a także – w późniejszym etapie konieczność stosowania metod numerycznych dla pierwszej i drugiej pochodnej danego parametru, autor postanowił użyć środowiska Maple wyłącznie w celu rozwiązania układu równań dynamiki, tj. wyznaczenia momentów napędowego oraz kierującego, a od tej chwili – kontynuować pracę wyłącznie w środowisku LabVIEW.

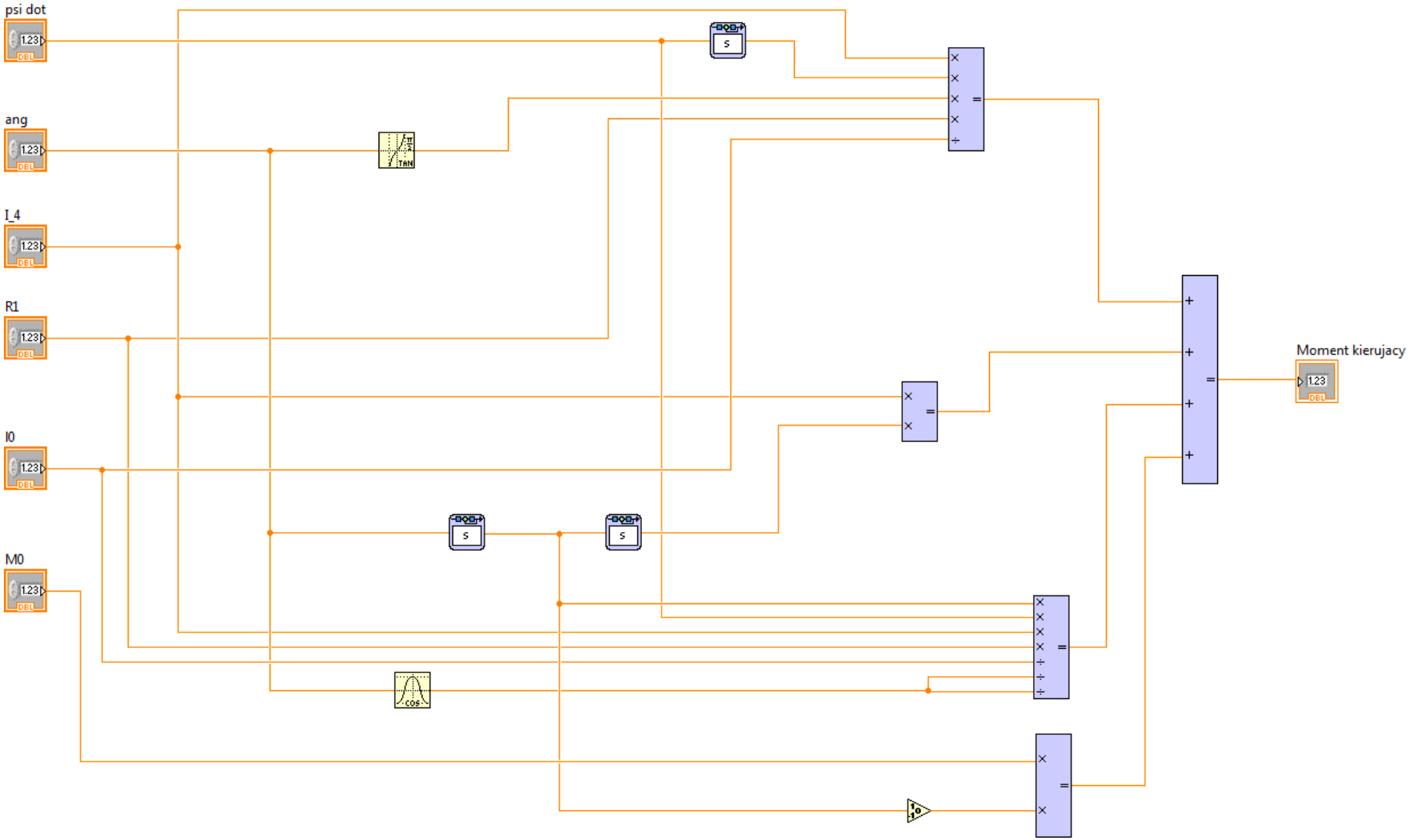
Dokonując zapisu równań (3.33) oraz (3.34) metodą symboliczną w środowisku Maple oraz przeprowadzając proces obliczeń, autor otrzymał równania parametryczne dla danych momentów sterujących platformą. Równania wyrażają się następującymi zależnościami:

$$M_S = \frac{\left[\ I_{x3} r\,\ddot{\psi} tg\varphi \cos^2\varphi + I_{x3}\ddot{\varphi}\, l_0 \cos^2\varphi + I_{x3}\, r\,\dot{\psi}\,\dot{\varphi} + M_0 \operatorname{sgn}\dot{\varphi}\, l_0 \cos^2\varphi\ \right]}{l_0 \cos^2\varphi}, \tag{3.40}$$

$$M_N = \big[\ \ddot{\psi} l_0^2 I_{z1} \cos^2\varphi + \ddot{\psi} l_0^2\, I_{z2} \cos^2\varphi + N_1 f_1 l_0^2 \cos^2\varphi + N_2\, f_2\, l_0^2 \cos^2\varphi +$$
$$+ N_3\, f_3\, l_0^2 \cos^2\varphi + \ddot{\psi}\, l_0^2\, I_{z3} + \ddot{\psi}\, l_0^2\, r^2\, m_1 \cos^2\varphi + \ddot{\psi} l_0^2\, r^2\, m_2 \cos^2\varphi +$$
$$+ \ddot{\psi} l_0^2\, r^2\, m_3 \cos^2\varphi + \ddot{\psi} l_0^2\, r^2\, m_5 \cos^2\varphi + \ddot{\psi}\, I_{x1}\, r^2 \cos^2\varphi\, tg^2\varphi + \ddot{\psi}\, I_{x2}\, r^2 \cos^2\varphi tg^2\varphi +$$
$$+ 2\ddot{\psi}\, I_{z1} l_1^2 \cos^2\varphi tg^2\varphi + \ddot{\psi}\, I_{z5} r^2 \cos^2\varphi tg^2\varphi + 16\ddot{\psi}\, l_1^2 r^2 \cos^2\varphi tg^2\varphi + \dot{\varphi}\dot{\psi}\, I_{x1}\, r^2 tg\varphi +$$
$$+ \dot{\varphi}\dot{\psi}\, I_{x2} r^2 tg\varphi + 2\dot{\varphi}\dot{\psi}\, I_{z1} l_1^2\, tg\varphi + \dot{\varphi}\dot{\psi}\, I_{z5}\, r^2\, tg\varphi + 16\, \dot{\varphi}\dot{\psi}\, l_1^2\, r^2\, tg\varphi + \dot{\varphi}\, I_{z3}\, l_0^2\, tg\varphi +$$
$$+ \ddot{\psi}\, l_1^2\, r^2\, m_2 \cos^2\varphi tg^2\varphi + \ddot{\psi}\, l_0^2\, r^2\, m_3 \cos^2\varphi tg^2\varphi + \ddot{\psi}\, l_2^2\, r^2 m_5 \cos^2\varphi tg^2\varphi +$$
$$+ \dot{\varphi}\dot{\psi}\, l_1^2\, r^2\, m_2\, tg\varphi + \dot{\varphi}\dot{\psi}\, l_0^2\, r^2 m_3\, tg\varphi + \dot{\varphi}\dot{\psi}\, l_2^2\, r^2 m_5\, tg\varphi\ \big] / \big[\ l_0 \cos^2\varphi\ \big]. \tag{3.41}$$

Otrzymane równania przedstawione w formie parametrycznej autor zapisał do oprogramowania LabVIEW (moduł CD&S). Ze względu na dużą złożoność utworzonego modelu dla momentu napędzającego, autor ograniczył się do pokazania (rys. 3.21) zapisanego momentu kierującego (zależności (3.34) oraz (3.40) są identyczne), którego równanie pozwala na numeryczne obliczenie sygnału sterującego rozważaną platforma mobilną. Proces generowania momentu (sygnału sterującego) odbywa się z częstotliwością zdefiniowaną przez ustaloną długość kroku całkowania. Autor w niniejszym podpunkcie zamieścił wyniki otrzymane dla kroku całkowania Δt=0,005 s oraz metody stałokrokowej Eulera. Kolejne rozdziały (4 oraz 5) przedstawiają dyskusję związaną z analizą wpływu długości kroku całkowania na proces sterowania platformą, a także motywem jego przyjęcia. Ze względu na fakt, iż wyliczane momenty nie są pochodnymi, nie jest konieczne obliczanie całek tych sygnałów (całki sygnałów generowane są jedynie dla α, β oraz punktu charakterystycznego). Wartości generowane są przy pomocy zapisanych dla potrzeb realizacji obliczeń arytmetycznych przekształceń na sygnałach (parametrach) platformy pochodzących z rozwiązania odwrotnego zadania kinematyki. Parametry odwrotnego zadania kinematyki stanowią wejścia do podsystemu dla danego momentu. Sygnałem wyjściowym z tego bloku jest chwilowa wartość pożądanego momentu (kierującego lub napędowego). Warto zauważyć, że otrzymane równanie momentu napędzającego jest uzależnione również od momentu kierującego. Natomiast wartość momentu kierującego może być wyliczona tylko na podstawie znajomości wartości parametrów otrzymanych drogą rozwiązania odwrotnego zadania kinematyki platformy.

Zapisane do środowiska LabVIEW momenty sterujące, zostały wykorzystane do przeprowadzenia symulacji ruchu trójkołowej platformy mobilnej, na trzech, zdefiniowanych we wcześniejszych podrozdziałach pracy, trajektoriach ruchu. Warto zaznaczyć, że obliczenia numeryczne dla zadania odwrotnego kinematyki oraz zadania odwrotnego dynamiki generowane są w tej samej chwili czasu t (np. obliczona wartość parametru kąta skrętu kierownicy jest w tym samym kroku całkowana Δt sygnałem wejściowym dla generacji momentu kierującego, również generowanego w chwili czasu t). W dalszej części pracy, oba momenty posłużyły do numerycznego obliczenia wartości pożądanych sygnałów (wartości chwilowych), które powinny być generowane przez system sterujący tak, aby platforma poruszała się po ustalonych trajektoriach. Sygnały te będą stanowiły punkt referencyjny dla docelowych sygnałów generowanych przez zaimplementowany algorytm.



Rys. 3.21. Schemat rozwiązania dla momentu kierującego Ms platformy mobilnej w środowisku LabVIEW

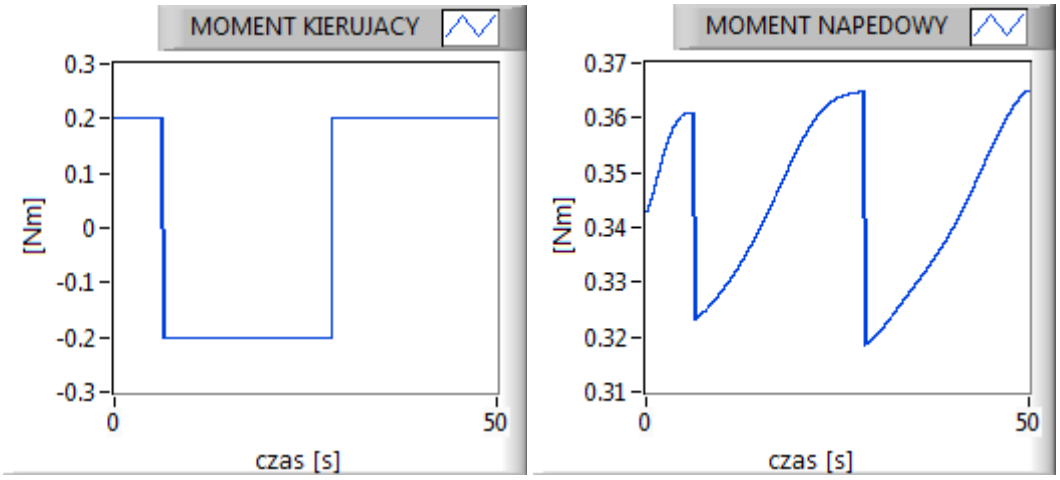
3.4.1. Realizacja trajektorii typu „sinus”

Na podstawie otrzymanego rozwiązania odwrotnego zadania dynamiki, autor przeprowadził symulacje komputerowe dla trajektorii typu „sinus”. Zapisane momenty w połączeniu z dedykowanymi dla danej trajektorii parametrami platformy, otrzymanymi podczas rozwiązania zadania odwrotnego kinematyki, posłużyły do przeprowadzenia obliczeń numerycznych, których rezultaty przedstawiono w postaci graficznej. Obliczenia numeryczne zostały przeprowadzone dla długości kroku całkowania równego 0,005 s oraz dla stałokrokowej metody całkowania Eulera. Na rys. 3.22 (lewy) przedstawiono wykres chwilowego momentu kierującego platformą. Moment kierujący przyjmuje przez większość część przejazdu platformy (nie wliczając okresów przejściowych, tzn. zmiany kierunku obrotu kierownicy) dwie wartości o tej samej amplitudzie, lecz o przeciwnym znaku. Skręt koła kierownicy w jedną stronę wymaga momentu siły 0,2 Nm, który dostarczany jest przez silnik DC. Wartość ta jest utrzymywana do czasu, w którym punkt charakterystyczny platformy osiąga swoje maksimum (w przebiegu funkcji ruchu tego punktu po trajektorii typu „sinus”) i kierownica ulega skrętowi w stronę przeciwną. W tym czasie (około 8 s) moment kierujący zmienia swój zwrot również na przeciwny. Wartość momentu jest identyczna tj. 0,2 Nm, lecz z przeciwnym znakiem (silnik obraca się w drugą stronę). Wartość momentu kierującego utrzymywana jest do czasu kolejnego skrętu, który występuje w 28 s, gdy punkt charakterystyczny osiąga swoje drugie ekstremum. Po tym czasie moment kierujący zmienia znak na przeciwny i przyjęta wartość 0,2 Nm utrzymywana jest do końca czasu trwania przejazdu.

Ze względu na charakter przebiegu momentu napędowego (rys. 3.22, prawy, niewielkie zmiany momentu w otoczeniu 0,34 Nm), autor postanowił zamieścić wykres na osobnym rysunku (a nie na wspólnym z momentem kierującym), ilustrującym charakter zmian momentu w trakcie trwania przejazdu. Otrzymany charakter przebiegu chwilowego momentu napędowego jest tożsamy z wartością momentu kierującego. Wartość momentu napędowego rośnie do 8 s, w której następuje skręt kierownicy. Skręt kierownicy platformy mobilnej, powoduje spadek wartości momentu do 0,325 Nm, która następnie wzrasta, aby w 28 s osiągnąć poziom 0,364 Nm. Po tym czasie następuje ponowny skręt kierownicy, spadek momentu i ponowny jego wzrost, aż do końca trasy przejazdu. Warto zwrócić uwagę, że chociaż

zmiany w chwilowej wartości momentu napędowego sięgają 12%, ze względu na bezwładność silnika zmiany te (w trakcie przeprowadzania eksperymentów) będą słabo odczuwalne.

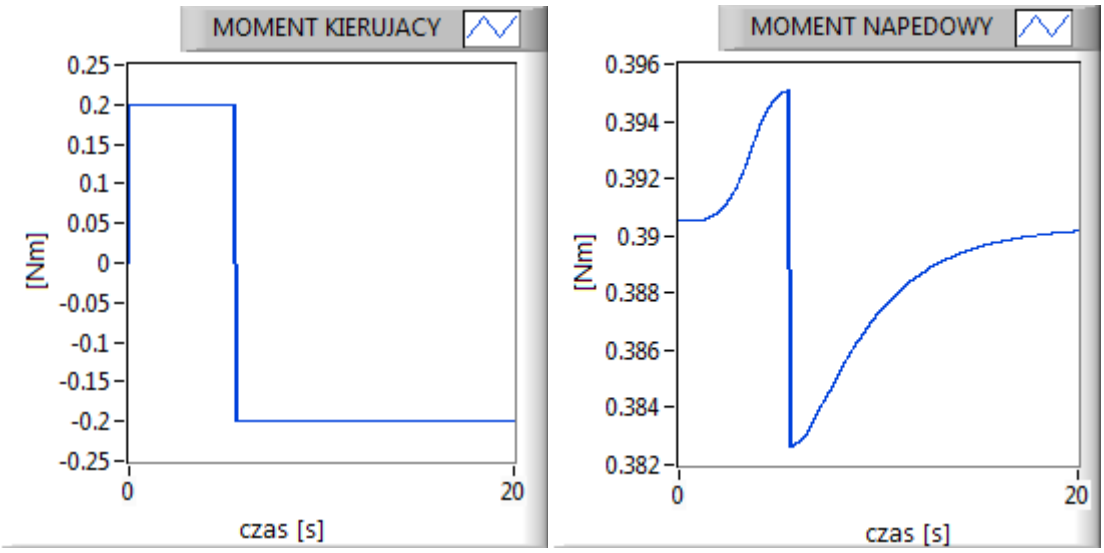
Należy zaznaczyć, że otrzymane przebiegi dla obu momentów (tj. kierującego oraz napędowego) odznaczają się charakterem odzwierciedlającym typowo fizyczny (mechaniczny) aspekt poruszania się platformy. Moment wzrasta, zmienia kierunek bądź utrzymuje stałą wartość w tych miejscach, które wynikają bezpośrednio z wyznaczonej trasy. W prezentowanych przebiegach brak jest jakichkolwiek zauważalnych oznak szarpnięć, czy lokalnych nieciągłości. Na uwagę zasługuje jedynie moment zmiany kierunku jazdy platformy, objawiający się gwałtownymi zmianami wartości momentu. Warto jednak zauważyć, że wykreślone momenty są momentami idealnymi (pożądanymi). Dodatkowo, jest to parametr pośredni (zadawany zespołowi napędowemu oraz kierującemu). W rzeczywistości szybka zmiana momentu będzie wywoływała na skutek bezwładności układu wolniejszy (z pewnym opóźnieniem) skutek (prędkości silników będą podążały za przyjętą wartością momentu).



Rys. 3.22. Chwilowa wartość momentu kierującego (lewy) oraz momentu napędowego (prawy) dla trajektorii typu „sinus”

3.4.2. Realizacja trajektorii typu „parabola”

Kolejną trajektorią ruchu, dla której autor przeprowadził symulacje komputerowe (numeryczne rozwiązanie równań różniczkowych w środowisku LabVIEW) była parabola. W tym przypadku, jak i dla trajektorii typu sinus, krok całkowania został ustawiony na 0,005 s, przy stałokrokowej metodzie całkowania. Sygnałami wejściowymi dla podsystemów momentu kierującego i napędowego były parametry z zadania odwrotnego kinematyki generowane dla danej trajektorii ruchu (paraboli). Rozwiązanie dla momentu kierującego zostało zaprezentowane na rys. 3.23 (lewy). Jego kształt po części nawiązuje do charakteru momentu uzyskanego dla trajektorii typu „sinus”. Również i w tym przypadku, chwilowe wartości zmian analizowanego momentu są bezpośrednim odzwierciedleniem zmian wartości chwilowej kąta skrętu kierownicy. Wyraźnie dostrzegalna jest sytuacja, w której zmiana momentu kierującego na przeciwny (wartość momentu utrzymująca się do 5,5 s przejazdu na stałym poziomie 0,2 Nm, ulega zmianie na tą samą wartość, lecz z przeciwnym znakiem) wiąże się bezpośrednio ze zmianą kierunku skrętu kierownicy (por. rys. 3.16 lewy). W dalszej cześć trasy, platforma po wyjściu z łuku kontynuuje powrót kierownicy do pozycji neutralnej (jazda na wprost), co wiąże się z utrzymaniem przyjętej wartości momentu do końca realizowanego przejazdu.



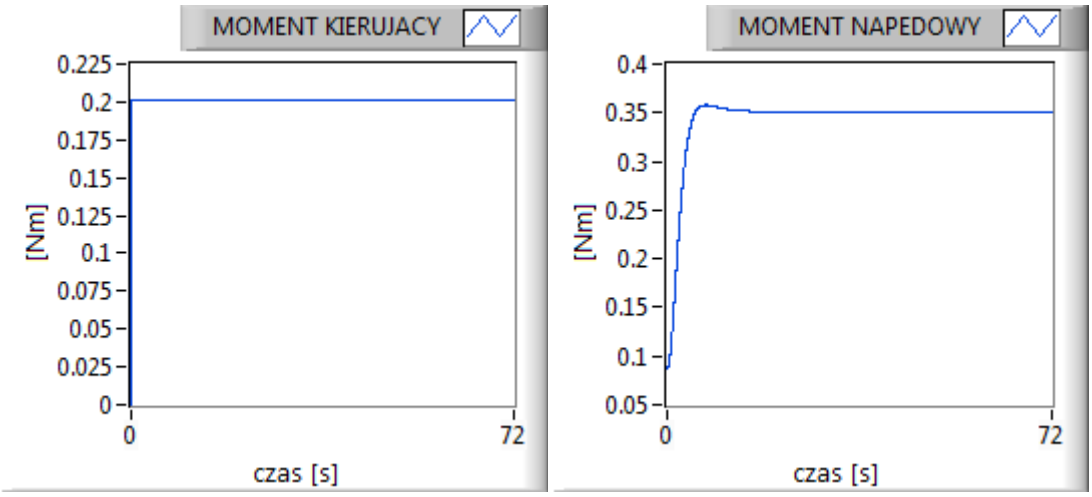
Rys. 3.23. Chwilowa wartość momentu kierującego (lewy) oraz momentu napędowego (prawy) dla trajektorii typu „parabola”

Zmiany chwilowej wartości momentu napędowego (rys 3.23 prawy) wiernie odzwierciedlają charakter realizowanej trajektorii oraz reakcji układu na skręt kierownicy. Z przebiegu trajektorii wynika, że platforma nie doznaje gwałtownych skrętów (pokonuje zakręt opisany krzywą paraboliczną). Pokonywany łuk trasy jest bardzo łagodny. Przejawia się to w przebiegu chwilej wartości rozpatrywanego momentu, gdzie zmiany sięgają rzędu tysięcznych części Nm. Do 5,5 s platforma pokonuje łuk, by po tym czasie, na skutek zmiany kierunku skrętu kierownicy, wartość uległa gwałtownej zmianie do 0,3828 Nm. Po tym czasie, aż do zakończenia przejazdu, wartość momentu ulega nieznacznemu wzrostowi (do wartości 0,3905 Nm).

Charakter przebiegów wartości chwilowych momentów platformy przy jeździe po trajektorii typu „parabola" jest tożsamy z wartościami otrzymanymi dla trajektorii typu „sinus". Również w tym przypadku, brak jest jakichkolwiek lokalnych nieciągłości. Pojawiające się nagłe zmiany wartości obu momentów w 5,5 s przejazdu platformy są wynikiem zmiany kierunku obrotu kierownicy na przeciwny. Jednak, o czym autor wspominał powyżej, moment dla silnika napędowego oraz dla sterującego układem kierowniczym jest parametrem zadawanym. Odpowiedzią układu, uzyskiwaną z pewnym opóźnieniem (na skutek bezwładności układu), są prędkości silników.

3.4.3. Realizacja trajektorii typu „okrąg"

Numeryczne rozwiązanie równań dynamiki zostało przeprowadzona również dla trajektorii typu „okrąg". Przyjęto wartości parametrów symulacji komputerowej identyczne jak dla poprzednich trajektorii, tzn. autor ustawił krok całkowania 0,005 s, przy stałokrokowej metodzie całkowania Eulera. Chwilowa wartość momentu kierującego została przedstawiona na rys. 3.24 (lewy). Z uwagi na fakt, iż platforma porusza się po okręgu, kierownica platformy po osiągnięciu odpowiedniego dla ustalonego promienia okręgu 2 m kąta skrętu (trwa to 5 s, rys. 3.20 lewy) utrzymuje tą pozycję do końca trwania przejazdu, tj. jednokrotnego przejazdu po rozpatrywanym okręgu. Sytuacja ta wiernie odzwierciedla fakt przebiegu chwilowej wartości momentu kierującego, która w 5 s osiąga wartość 0,2 Nm i jest utrzymywana do końca trasy przejazdu.



Rys. 3.24. Chwilowa wartość momentu kierującego (lewy) oraz momentu napędowego (prawy) dla trajektorii typu „koło"

Przebieg chwilowego moment napędowego ma zbliżony charakter do momentu kierującego, jednak wartość utrzymująca się od 5 s (tzn. po osiągnięciu przez koło kierownicy właściwego kąta skrętu) na stałym poziomie jest o około 75% większa (0,36 Nm). Skutkiem zadanej wartość rozpatrywanego momentu napędowego są wartości prędkości kół napędowych i kierującego, które również od 5 s utrzymują stałą wartość, aż do końca czasu trwania przejazdu (por. rys. 3.19 prawy).

3.4.4. Analiza mnożników Lagrange'a

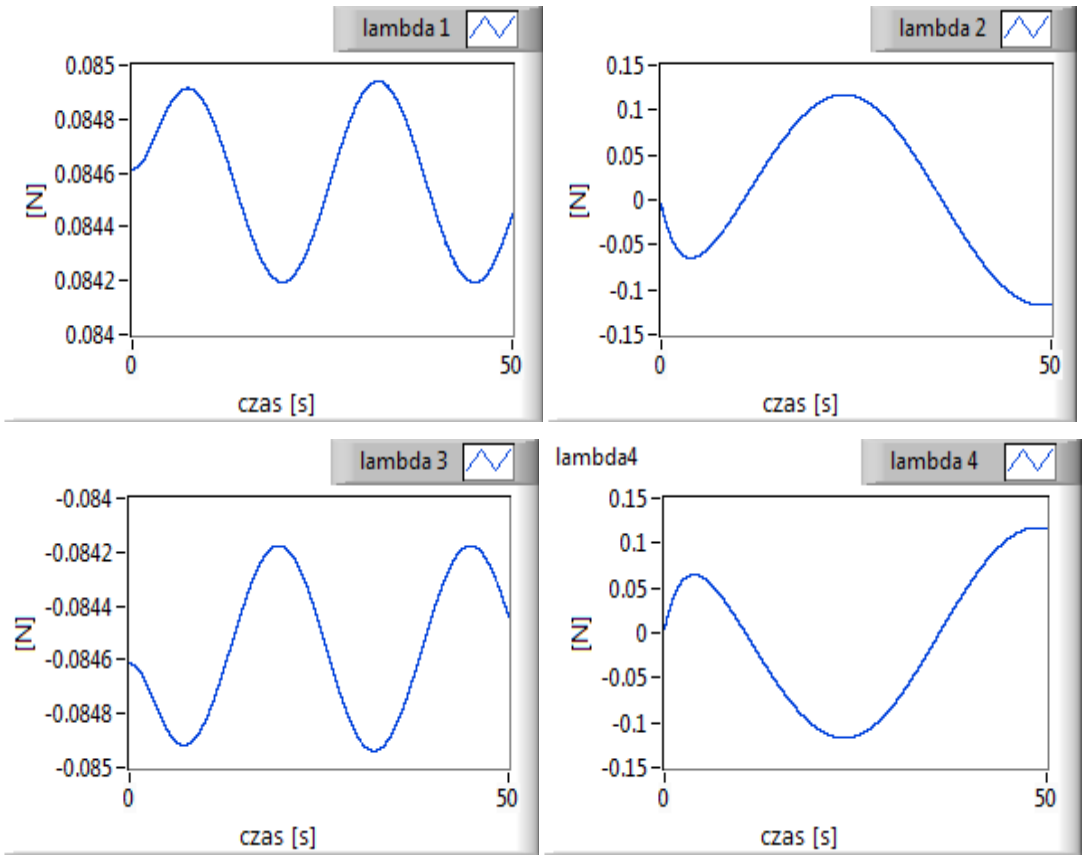
Drugie równanie macierzowe układu (3.24) można przestawić w postaci umożliwiającej przeprowadzenie analizy mnożników Lagrange'a (λ_1, λ_2, λ_3 oraz λ_4) [20, 21]. Otrzymamy wówczas układ równań:

$$\begin{aligned}&(m_1+m_2+m_3+m_5)\,r\,(\ddot{\psi}\cos\beta-\frac{r}{l}\dot{\psi}^2\,tg\varphi\sin\beta)-\\&-(m_3l-m_5l_2)\frac{r}{l}(\ddot{\psi}\,tg\varphi\sin\beta+\frac{r}{l}\dot{\psi}^2\,tg^2\varphi\,\cos\beta+\dot{\psi}\,\dot{\varphi}\frac{\sin\beta}{\cos^2\varphi})=\lambda_1+\lambda_3,\end{aligned}\tag{3.42}$$

$$\begin{aligned}&(m_1+m_2+m_3+m_5)\,r\,(\ddot{\psi}\sin\beta-\frac{r}{l}\,\dot{\psi}^2\,tg\varphi\cos\beta)-\\&-(m_3\,l-m_5\,l_2)\frac{r}{l}(\ddot{\psi}\,tg\varphi\,\cos\beta+\frac{r}{l}\dot{\psi}^2\,tg^2\varphi\,\sin\beta+\dot{\psi}\,\dot{\varphi}\frac{\cos\beta}{\cos^2\varphi})=\lambda_2+\lambda_4.\end{aligned}\tag{3.43}$$

Tak określony układ równań pozwolił autorowi na weryfikację określonej koncepcji platformy mobilnej. Podczas tej analizy autor zbadał zmiany sił tarcia suchego w punktach styczności kół z jezdnią i dokonał weryfikacji czy określone chwilowe zmiany tych sił tarcia nie przekraczają wartości granicznych. Równania (3.42) oraz (3.43) zostały zapisane w postaci symbolicznej do środowiska Maple, gdzie dokonano ich rozwiązania. Autor wykorzystał środowisko LabVIEW w dalszej ich analizie. W procesie modelowania otrzymanych rozwiązań do LabVIEW zastosowano identyczną procedurę postępowania, jaka została wykorzystana w przypadku modelowania rozwiązań równań kinematyki i dynamiki platformy. Do rozwiązania rozpatrywanego układu równań wykorzystano metody numeryczne. Przyjęto stałokrokową metodę całkowania Eulera oraz krok całkowania 0,005 s.

Na rys. 3.25 zademonstrowano (autor przeprowadził analizę mnożników Lagrange'a dla trajektorii typu „sinus", jako najbardziej wymagającej krzywej, po której porusza się platforma w realizowanych badaniach) graficzne rozwiązania mnożników Lagrange'a.



Rys. 3.25. Chwilowe wartości mnożników Lagrange'a dla trajektorii typu „sinus"

Całkowitą wartość siły tarcia działającą na odpowiednie koła (kierujące T_N oraz zastępcze T_M), można określić korzystając z zależności:

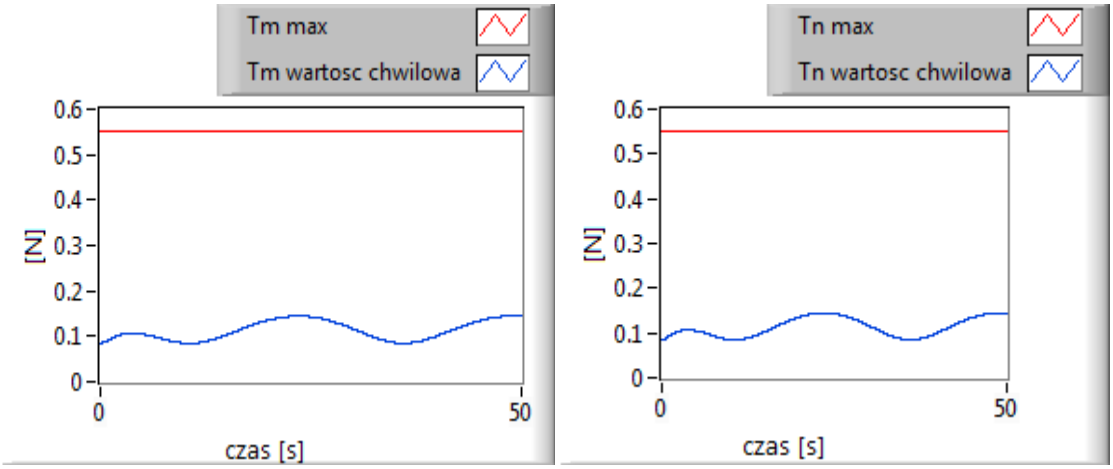
$$T_N = \sqrt{\lambda_1^2 + \lambda_2^2},\ T_M = \sqrt{\lambda_3^2 + \lambda_4^2}. \tag{3.44}$$

Natomiast, wartości graniczne dla poszczególnych kół określone są zależnościami:

$$T_{Nmax} = \mu_N \cdot (N_1 + N_2),\ T_{M\,max} = \mu_N \cdot N_3. \tag{3.45}$$

Gdzie : μ_N, μ_M – współczynniki tarcia suchego poszczególnych kół (kierującego i zastępczego).

Na podstawie otrzymanych wartości chwilowych, autor korzystając z zależności (3.44), określił całkowite siły tarcia działające na odpowiednie koła, które porównał z ich wartościami granicznymi (3.45). Rys. 3.26 przedstawia, otrzymane rezultaty tej analizy.



Rys. 3.26. Chwilowa wartość siły tarcia (niebieska) oraz wartość graniczna (czerwony) dla koła kierującego 1(lewy) oraz koła zastępczego (prawy) dla trajektorii typu „sinus”

Otrzymane rezultaty przeprowadzonej analizy mnożników Lagrange'a potwierdzają, że określone (chwilowe wartości całkowitej siły tarcia dla poszczególnych kół platformy są około 5,5 razy mniejsze od wartości granicznych) parametry platformy zapewniają jej przejazd bez poślizgów, a tym samym - przy zbilansowanym wydatku energetycznym (ewentualne poślizgi mogłyby doprowadzić do utraty sterowalności, a tym samym - do poruszania się platformy po niewłaściwym torze).

3.4.5. Podsumowanie wyników symulacji komputerowych

Przeprowadzone symulacje komputerowe dynamiki układu umożliwiły określenie pożądanych (referencyjnych) sygnałów sterujących (momentów napędowych silników DC). Autor dokonując szeregu symulacji wirtualnych na wyznaczonych trasach dobierał tak parametry fizyczne platformy (w ramach przyjętych założeń konstrukcyjnych oraz możliwości realizacji projektu), m.in. masy poszczególnych części, ich rozkład itp., aby wypracować finalną koncepcję, w myśl której zapewniona jest płynność przejazdu platformy bez poślizgów (analiza mnożników Lagrange'a). Moment kierujący oraz napędowy, którego numeryczne rozwiązania wygenerowano w środowisku LabVIEW dla wszystkich zrealizowanych tras przejazdu, jest wynikiem przyjętej trajektorii ruchu oraz przyjmuje tylko takie wartości, które wynikają bezpośrednio z zasad mechaniki. Gwałtowne zmiany chwilowych wartości rozpatrywanych momentów (jako wartości zadanych), ze względu na fakt istnienia bezwładności układu, w rzeczywistych warunkach nie będą zauważalne. Zmiana będzie dokonywała się w sposób płynny. Na uwagę zasługują również małe zmiany wartości chwilowych momentu napędowego na wszystkich trasach przejazdu. Podczas realizacji rzeczywistych przejazdów platformy chwilowe zmiany tego momentu będą miały pomijalny wpływ. Dużo rolę w zapewnieniu precyzji ruchu platformy będzie odgrywał moment kierujący, którego zmiany w wartościach chwilowych są znaczące (dla trajektorii typu „sinus" oraz „parabola", zmiana na wartość przeciwną). Ustalona koncepcja platformy będzie stanowiła fundament do dalszych badań oraz dalszej analizy. Pozwoli również na przeprowadzenie wirtualnego prototypowania oraz testów techniką HILS.

4. Wirtualne prototypowanie systemu nadzorowania ruchu platformy mobilnej

Współczesne energetyczne systemy napędowe pojazdów kołowych stają się coraz bardziej wydajne i niezawodne. Wysoka sprawność energetyczna poszczególnych składników tych systemów pozwala na budowę układów przewidzianych do realizacji przedsięwzięć oraz operacji, których często główną oceną jest ich dostępność (operacyjność). Dodatkowo, na źródła energii nakładają się wymagania związane z szybkością jej odbioru bądź magazynowania (np. superkondensator). Jednakże, to proces sterowania obiektem oraz przepływem energii, a także algorytm odzwierciedlający mechatroniczny model układu przyczynia się do lepszego jej wykorzystania i wyeliminowania zjawisk negatywnie wpływających na własności trakcyjne układu [31,46, 97].

Autor, kierując się założeniami konstrukcyjnymi realizowanego projektu podjął się próby implementacji energetycznego wskaźnika jakości (rozwijanego w Katedrze Mechaniki i Mechatroniki Politechniki Gdańskiej) pozwalającego na skuteczne nadzorowanie obiektu badań.

Proces realizacji systemu nadzorowania ruchu platformy rozpoczęto od realizacji techniki wirtualnego prototypowania, której koncepcja została zobrazowana na rys. 2.2. Ideą przeprowadzonego projektowania było zbadanie odpowiedzi układu (modelu platformy mobilnej) na sygnały sterujące pochodzące od prototypowanego sterownika. Badania przeprowadzono dla trzech wybranych trajektorii ruchu platformy. Wartościami referencyjnymi dla przeprowadzonych doświadczeń były wyniki symulacji uzyskane w rozdziale 3 tj. zadane trajektorie ruchu platformy.

4.1. Sterowanie optymalne przy energetycznym wskaźniku jakości

Optymalny sygnał sterujący otrzymany w wyniku minimalizacji zmiennego w czasie energetycznego wskaźnika jakości jest wyrażony zależnością [9, 35]:

$$\mathbf{u}(t) = (\mathbf{R} + \mathbf{R}^T)^{-1} \int_{t-\Delta t}^{t} \mathbf{B}^T(\tau)\boldsymbol{\Theta}^T(t,\tau)\mathrm{d}\tau \cdot \mathbf{T}^T\left(\mathbf{QM} + \mathbf{M}^T\mathbf{Q}^T\right)\left(\dot{\mathbf{q}} - \dot{\bar{\mathbf{q}}}\right), \tag{4.1}$$

gdzie:

$\mathbf{Q}$ – macierz bezwymiarowych współczynników wpływu,

$\mathbf{R}$ – macierz efektu sygnału sterującego,

$\dot{\bar{\mathbf{q}}}$ – wektor prędkości uogólnionych układu w ruchu zadanym,

$\dot{\mathbf{q}}$ – wektor prędkości uogólnionych układu sterowanego,

$\mathbf{M}$ – macierz bezwładności,

$\mathbf{B}$ – macierz wejść,

$\mathbf{\Theta}(t,\tau)$ – rozwiązanie macierzowego równania różniczkowego jednorodnego [35].

Zdefiniowany zależnością (4.1) wskaźnik jakości wyraża różnicę pomiędzy rzeczywistą energią platformy mobilnej w chwili *t* a energią, jaką powinna mieć platforma w tej samej chwili *t*, jednakże gdyby poruszała się po dokładnie wyznaczonej trajektorii wynikającej z założeń procesu sterowania.

Omawiany wskaźnik jakości należy do grupy wskaźników energetycznych, gdzie cechą wiodącą (zaletą) wyróżniającą go spośród innych (np. wskaźnikiem całkowym) jest możliwości generowania optymalnych sygnałów sterujących (w tym przypadku momentu kierującego i sterującego) w trybie *on-line*. Cecha ta umożliwia podjęcie w każdej chwili czasu *t* (co krok Δt) optymalnej decyzji sterującej, co objawia się faktem, iż dany obiekt (tu realizowana platforma mobilna) porusza się w każdej chwili czasu po optymalnej trasie, a wartość chwilowa energii kinetycznej oraz potencjalnej (tutaj tylko kinetyczna) wynika wyłącznie z przyjętej trajektorii ruchu (plus ewentualna odchyłka powstała na skutek skończonej dokładności jednostek generujących sygnał sterujący, a także skończonej dokładności zastosowanych komponentów użytych do budowy obiektu).

Bardzo silna nieliniowość równań dynamiki (wszystkie macierze są silnie nieliniowe) oraz złożoność opisu modelu obliczeniowego platformy, a także dodatkowo charakter zastosowanego algorytmu (generowanie sygnałów *on-line*) narzucił wymóg realizacji wysokowydajnego sterownika czasu rzeczywistego. Do realizacji rzeczywistego systemu nadzorowania autor postanowił zastosować sterownik firmy National Instruments cRIO-9076 (szczegóły techniczne oraz koncepcja zastosowanego rozwiązania zostaną przedstawione w rozdziale 5 pracy). Dokonany wybór sterownika jest nieprzypadkowy, bowiem pochodzi on od tego samego producenta, z którego wywodzi się zastosowane w pracy środowisko LabVIEW. Oba te elementy są ze sobą ściśle zintegrowane, zarówno pod kątem czysto abstrakcyjnym (filozofia działania, architektura) jak i funkcjonalnym. LabVIEW

stanowi środowisko programistyczne, w którym interfejsem zewnętrznym (układami wejścia/wyjścia) jest dostarczany sprzęt (tutaj sterownik cRIO).

W rozważanych badaniach wektor sygnałów sterujących został określony, jako:

$$\mathbf{u} = \begin{bmatrix} M_1 & M_2 \end{bmatrix}^T, \tag{4.2}$$

gdzie:

M_1 – moment napędowy,

M_2 – moment kierujący.

Algorytm sterowania opisany zależnością (4.1) oraz niezbędne przekształcenia (metoda rozwiązania całek oznaczonych) umożliwiające wyznaczenie optymalnego sygnału sterującego (moment kierujący oraz napędowy) zostały zaimplementowane do środowiska LabVIEW, tworząc wirtualny model sterownika platformy mobilnej. Do procesu implementacji autor wykorzystał podobny zestaw bloków, co w przypadku implementacji równań różniczkowych, jednak dodatkowo posłużył się blokami umożliwiającymi realizację działań na macierzach.

W pracy [35] opisano procedurę obliczania całek oznaczonych występujących w zależności (4.1). W tym celu dzielimy czas sterowania na *m* przedziałów o długościach:

$$\Delta t_i = t_i - t_{i-1}. \tag{4.3}$$

W rezultacie, otrzymujemy m+1 chwil czasu t_i, i=0,1,…*m*. Tranzycję stanu dla chwili czasu t_i oraz t_{i-1} można określić w następujący sposób:

$$\mathbf{\Theta}(t_i, t_{i-1}) \cong \sum_{k=0}^{\infty} \mathbf{A}^k(t_i) \frac{\Delta t_i^k}{k!}. \tag{4.4}$$

Wówczas przy obliczaniu całek oznaczony występujących w zależności (4.1) korzystamy ze wzoru:

$$\int_{t_{i-1}}^{t_i} \mathbf{B}^T(\tau)\mathbf{\Theta}^T(t,\tau)\mathrm{d}\tau \cong \mathbf{B}^T(t_i) \cdot \sum_{k=0}^{\infty} \frac{\left[\mathbf{A}(t_i)\Delta t_i\right]^k}{k!} \Delta t_i \tag{4.5}$$

W zależności (4.1), macierz bezwładności **M** (zależność 3.35) oraz efektów sił odśrodkowych bezwładności i efektów żyroskopowych **L** (zależność 3.36) nie ulega zmianom podczas kroku całkowania Δt. Dodatkowo, macierze **M** oraz **L** wyznaczane są w chwili początkowej t_{i-1}, zamiast w chwili docelowej t_i.

Przyjęta konwencja realizacji techniki wirtualnego prototypowania pozwoliła autorowi, wykorzystując sterownik wirtualny (zaimplementowane do LabVIEW

równanie (4.1), a dokładniej (4.9)) wraz z zamodelowaną (w rozdziale 3) wirtualną (emulowaną) platformą mobilną na wygenerowanie rozwiązań (parametrów kinematyki oraz dynamiki) w postaci graficznej dla wszystkich trajektorii ruchu platformy.

Charakter realizowanych badań, a także sytuacja, w której od dość wczesnej fazy projektu autor posiadał docelowy sterownik (cRIO-9076), narzuciła konieczność jednoczesnej realizacji wirtualnego prototypowania oraz przeprowadzenia (w ramach techniki HILS) procesu weryfikacji warunku determinizmu czasowego w rzeczywistym sterowniku (tak, aby ustalony krok całkowania był możliwy do realizacji w ramach przyjętego rozwiązania sprzętowego).

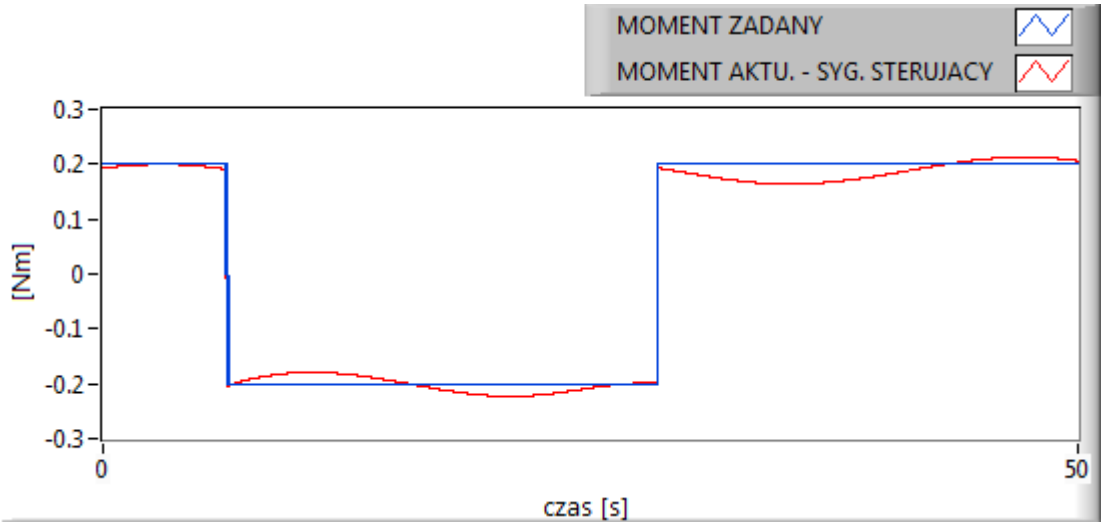
Należy zauważyć, że w tym przypadku elementem łączącym obie techniki jest przeprowadzony proces optymalizacji układu, w którym ustala się architekturę wykonania algorytmu oraz podejmuje decyzję o długości kroku całkowania.

Ostatecznie, w podpunkcie 4.1 prezentowane są rezultaty badań (wirtualnego prototypowania) dla przyjętego (drogą przeprowadzonej optymalizacji) kroku całkowania Δt=0,005 s (szczegółowy opis doboru kroku całkowania jak i analiza zostały przedstawione w p. 4.3, 4.4 oraz rozdziale 5) oraz zastosowanych (w sygnale sterującym) prędkości korygujących (szczegóły w p. 4.2).

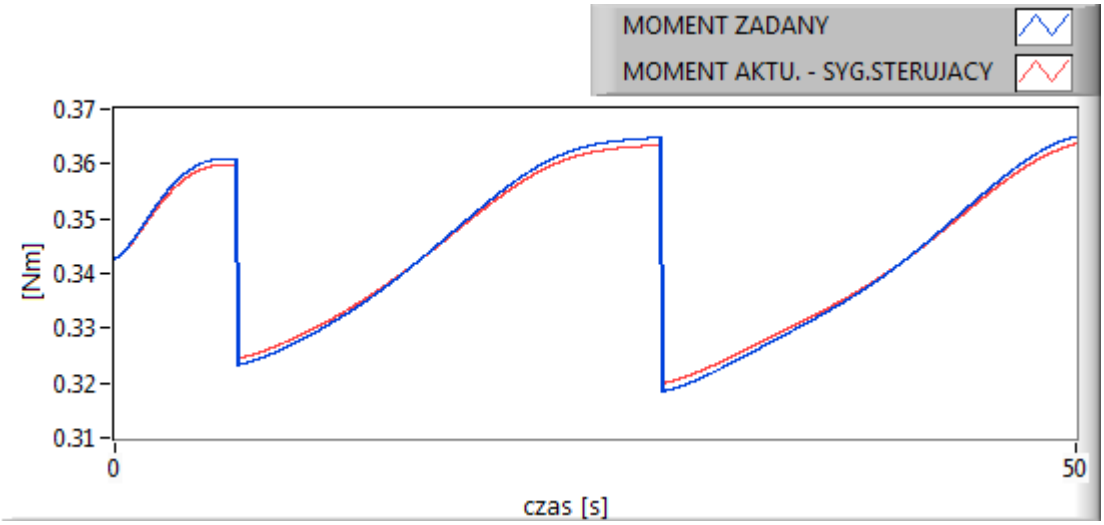
Mając na uwadze iteracyjny proces prototypowania systemu nadzorowania tzn. powtarzany proces: 1. Symulacja wirtualna; 2. Sprawdzenie warunku determinizmu czasowego w rzeczywistym sterowniku; 3. Korekta ustawień, (należy zaznaczyć, że klasyczna metoda wirtualnego prototypowania nie uwzględnia sprawdzania warunku determinizmu), autor dla każdej wartości chwilowej parametru platformy (wynikającego z kinematyki oraz dynamiki układu) generował w celach referencyjnych (możliwości wprowadzania ewentualnych korekt do prototypowanego sterownika) trajektorie idealne (zadane) dla tych parametrów. Należy podkreślić, że proces prototypowania odbywa się w wirtualnej przestrzeni, w której emulowany jest sterownik, jak i obiekt. Fakt ten determinuje sytuację, w której algorytm sterujący oraz równania różniczkowe opisujące zamodelowany obiekt rozwiązywane są drogą numeryczną. Przyjęcie cyklu maszynowego na poziomie Δt=0,005 s (a także innych, rozważanych podczas przeprowadzanych badań) powoduje powstanie błędów w generowanych rozwiązaniach. Dodatkowo, czynnikiem negatywnie wpływającym na uzyskane rezultaty jest fakt nadzorowania obiektu silnie nieliniowego.

4.1.1. Wirtualne prototypowanie systemu nadzorowania dla trajektorii typu „sinus”

Proces wirtualnego prototypowania został rozpoczęty od sytuacji, w której zbadano odpowiedź samego sterownika układu dla zadanej trajektorii ruchu platformy. Pierwszą rozpatrzoną trajektorią ruchu była trajektoria typu „sinus”.



Rys. 4.1. Chwilowa wartość sygnału sterującego dla momentu kierującego (niebieski) oraz momentu kierującego zadanego (czerwony) dla trajektorii typu „sinus”

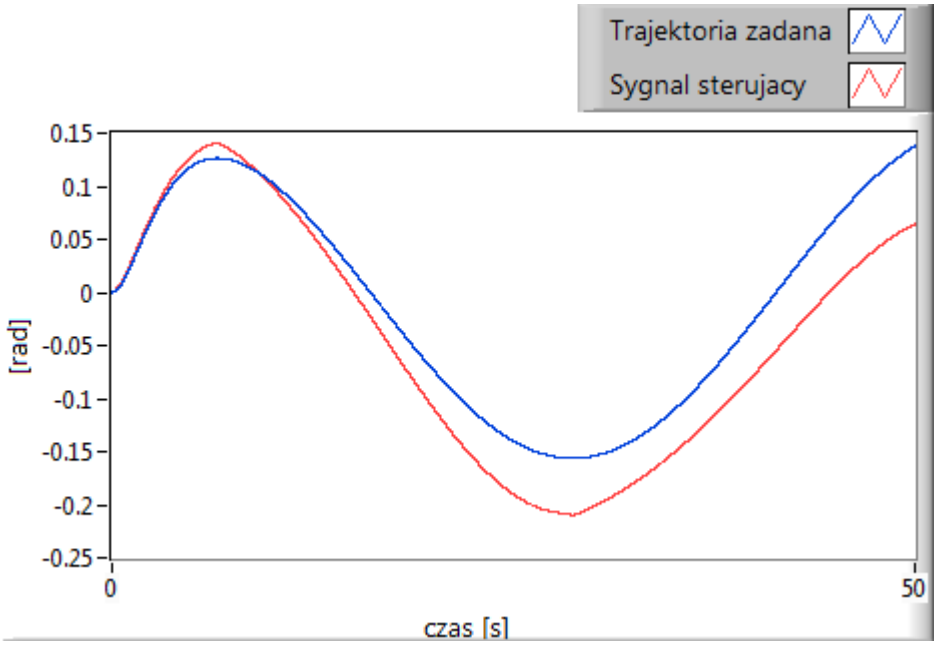


Rys. 4.2. Chwilowa wartość sygnału sterującego dla momentu napędowego (niebieski) oraz momentu napędowego zadanego (czerwony) dla trajektorii typu „sinus”

Otrzymane graficzne (rys. 4.1 oraz rys. 4.2) rozwiązania równania (4.9) potwierdzają poprawność implementacji algorytmu sterowania do środowiska LabVIEW. Zarówno chwilowa wartość momentu napędowego jak i kierującego są zbieżne z wartościami zadanymi (rozwiązania równań Lagrange’a, pokazane w

rozdziale 3, a stanowiące przypadek referencyjny). Występujące błędy (odchyłki) są następstwem skończonej dokładności zastosowanej metody numerycznej dla danego kroku całkowania, a także – wynikającej ze sterowania bardzo silnie nieliniowym obiektem. Należy zauważyć, że każda odchyłka (szczególnie widoczna dla momentu kierującego w drugiej części przejazdu) może być zminimalizowana poprzez odpowiedni dobór współczynników macierzy **R** i **Q** oraz wzmocnień rzeczywistego sterownika silnika DC.

Następstwem dobranych współczynników macierzy **R** oraz **Q** (wirtualnego sterownika) była realizacja prototypowania, w którym wirtualna platforma porusza się (po trajektorii typu „sinus") na podstawie optymalnych sygnałów generowanych przez projektowany sterownik. Na rys. 4.3 – 4.5 zamieszczono graficzne rezultaty tych badań.



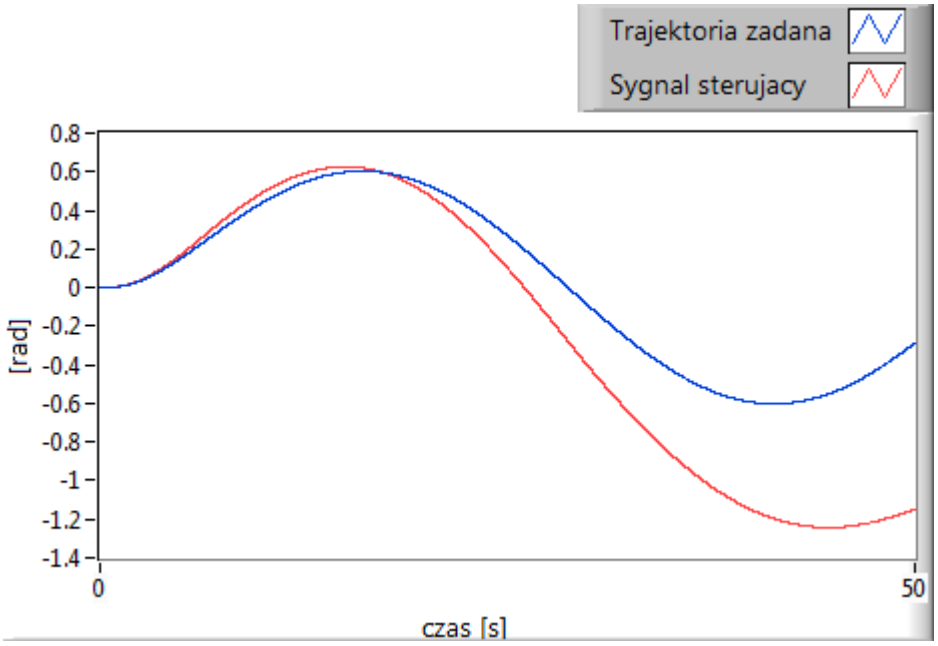
Rys. 4.3. Chwilowa wartość kąta obrotu kierownicy (przy sygnale sterującym – kolor czerwony, wartość zadana – kolor niebieski) dla trajektorii typu „sinus"

Rys. 4.3 obrazuje chwilową wartość kąta obrotu kierownicy. Czerwonym kolorem oznaczono krzywą uzyskaną w przypadku, kiedy ruch zamodelowanej (wirtualnej) platformy jest wynikiem optymalnego sygnału sterującego. Kolorem niebieskim oznaczono trajektorię zadaną. W dalszych badaniach konwencja w przyjętych oznaczeniach jest identyczna.

Występujące różnice w przebiegach chwilowych wartości rozważanego parametru kąta tłumaczy się (o czym autor wspomniał powyżej) skończoną dokładnością zastosowanej metody numerycznej, silną nieliniowością układu, a także

szeregiem przekształceń całkowo-różniczkowych. Pomimo, istnienia licznej grupy czynników zniekształcających prowadzone badania, wyniki odzwierciedlają zachowanie się rzeczywistego systemu. Trajektorie są do siebie bardzo zbieżne, choć wirtualnej platformie nadawane są zbyt duże sygnały sterujące (w 30 s oraz 50 s błąd sięga około 35%). Należy podkreślić, że sygnałami sterującymi są momenty, a prezentowane wartości chwilowe parametrów kinematyki platformy mobilnej są wynikiem ich działania. Podczas procesu integracji rzeczywistego sterownika (algorytmu) sterownik zostanie poddany strojeniu, umożliwiającemu wyeliminowanie występujących w tych badaniach przeregulowań.

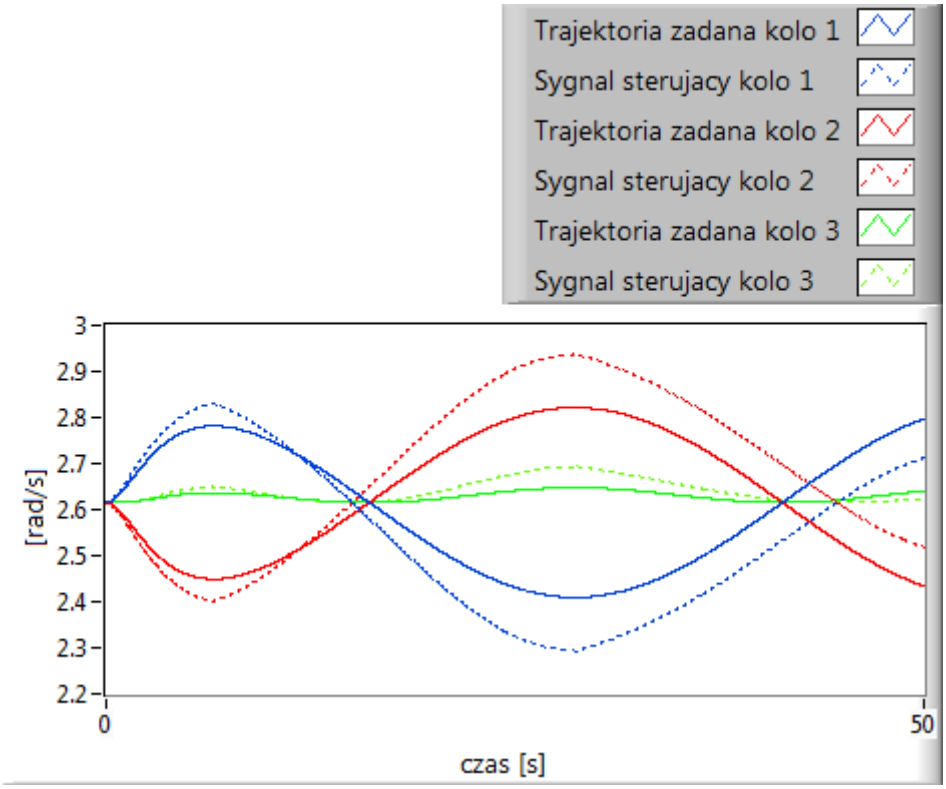
Na rys. 4.4 wykreślono wartość chwilową kąta obrotu platformy, który bezpośrednio odzwierciedla położenie kierownicy platformy mobilnej. Fakt, ten determinuje sytuację, w której zbyt duży obrót koła kierownicy (przy optymalnym sygnale sterującym) sprawia, iż rozpatrywany kąt położenia platformy również doznaje znacznej dewiacji od wartości zadanej (największy błąd notuje się na końcu trwania symulacji).



Rys. 4.4. Chwilowa wartość kąta obrotu platformy (przy sygnale sterującym – kolor czerwony, wartość zadana – kolor niebieski) dla trajektorii typu „sinus”

Zbyt duże wartości sygnałów sterujących (momentów) przekładają się również na zwiększone wartości prędkości kątowych poszczególnych kół platformy, których wartości są ściśle uzależnione od chwilowych wartości kąta skrętu kierownicy. Rys. 4.5 obrazuje te przebiegi. Koła poruszają się znacznie szybciej, niż wynika to z przyjętych wartości zadanych. Pomimo wyższych wartości chwilowych w

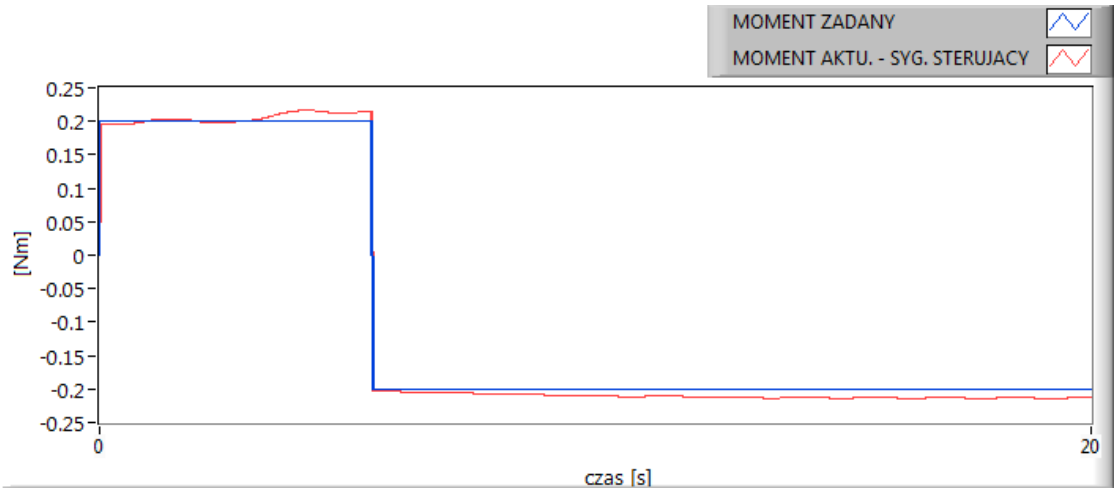
otrzymanych przebiegach, brak jest dostrzegalnych lokalnych nieciągłości czy też sytuacji mogących być przyczyną lokalnych przyspieszeń lub utraty stabilności.



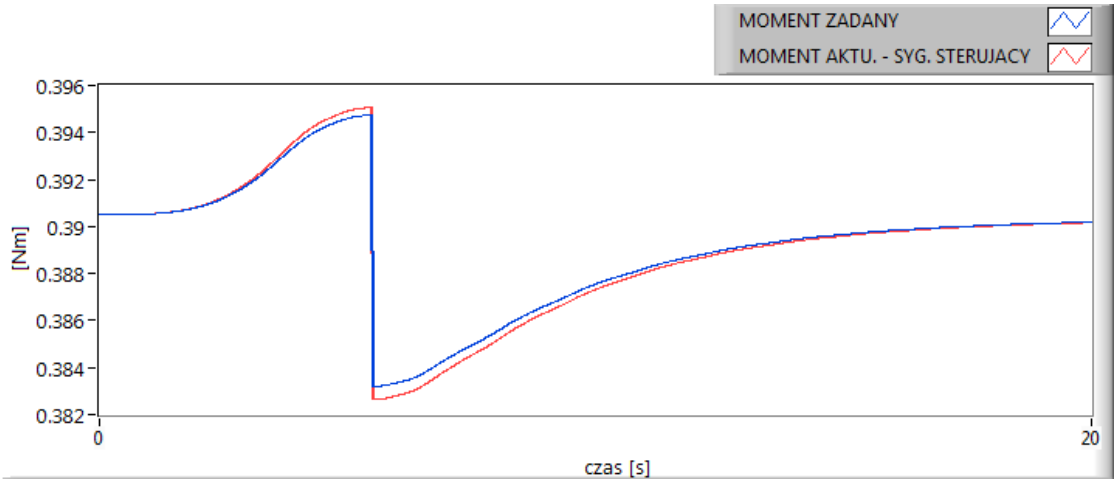
Rys. 4.5. Chwilowe wartość prędkości kątowych kół (przy sygnale sterującym – linia przerywana, wartość zadana – linia ciągła) dla trajektorii typu „sinus”

4.1.2. Wirtualne prototypowanie systemu nadzorowania dla trajektorii typu „parabola”

Kolejną trajektorią ruchu platformy, dla której przeprowadzono proces wirtualnego prototypowania, była trajektoria typu „parabola”. Również w tym przypadku badania rozpoczęto od analizy odpowiedzi sterownika.



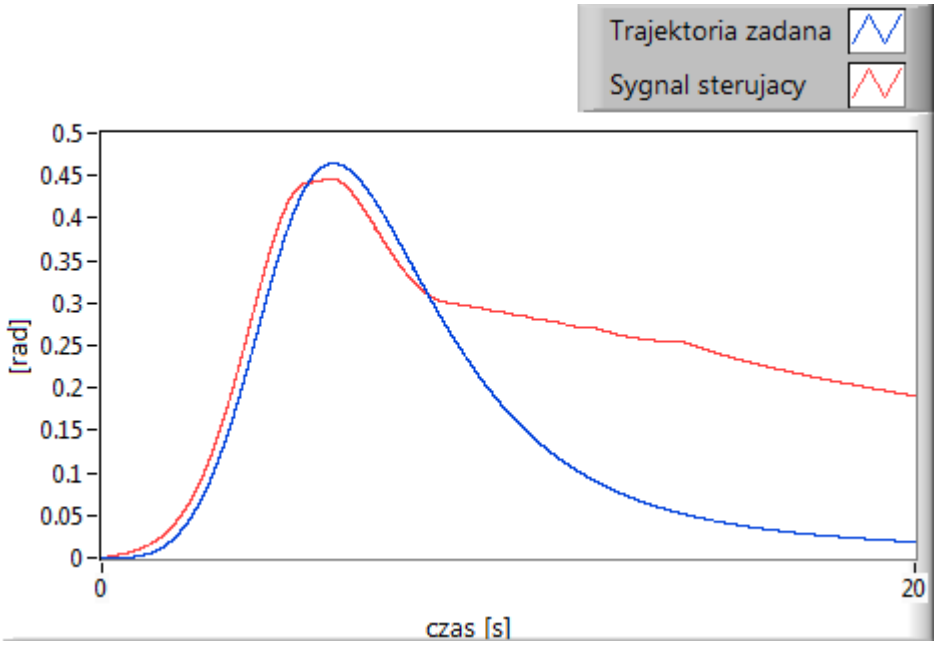
Rys. 4.6. Chwilowa wartość sygnału sterującego dla momentu kierującego (niebieski) oraz momentu kierującego zadanego (czerwony) dla trajektorii typu „parabola”



Rys. 4.7. Chwilowa wartość sygnału sterującego dla momentu napędowego (niebieski) oraz momentu napędowego zadanego (czerwony) dla trajektorii typu „parabola”

Podczas przejazdu wirtualnej platformy po trajektorii typu „parabola” (rys. 4.6 oraz rys. 4.7), zaimplementowany algorytm, podobnie jak dla trajektorii „sinus”, generuje sygnały bardzo zbieżne z zadanymi. Występujące odchyłki, podobnie jak poprzednio, tłumaczy się skończoną dokładnością zastosowanych metod numerycznych, a także silną nieliniowością sterowanego układu. Na uwagę zasługuje mała różnica (około 1%) w przebiegach wartości chwilowej momentu zadanego oraz aktualnego. Niewielkie przeregulowania występujące w otrzymanych przebiegach, zostaną wyeliminowane poprzez dobór odpowiednich współczynników wzmocnienia sterownika silników DC.

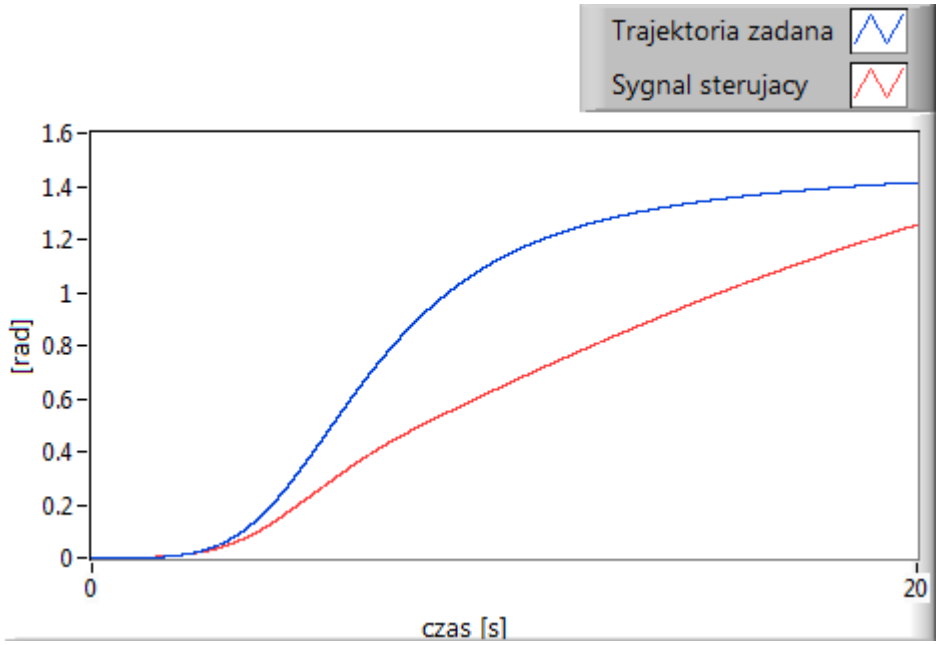
Proces iteracyjnego strojenia sterownika (doboru współczynników macierzy **R** oraz **Q**) umożliwił w kolejnej fazie przeprowadzenie badań nad zachowaniem się modelu wirtualnego. Na rys. 4.8 – 4.10 wykreślono chwilowe wartości parametrów platformy, uzyskanych przy pobudzeniu sygnałem optymalnym (z prototypowanego sterownika).



Rys. 4.8. Chwilowa wartość kąta obrotu kierownicy (przy sygnale sterującym – kolor czerwony, wartość zadana – kolor niebieski) dla trajektorii typu „parabola”

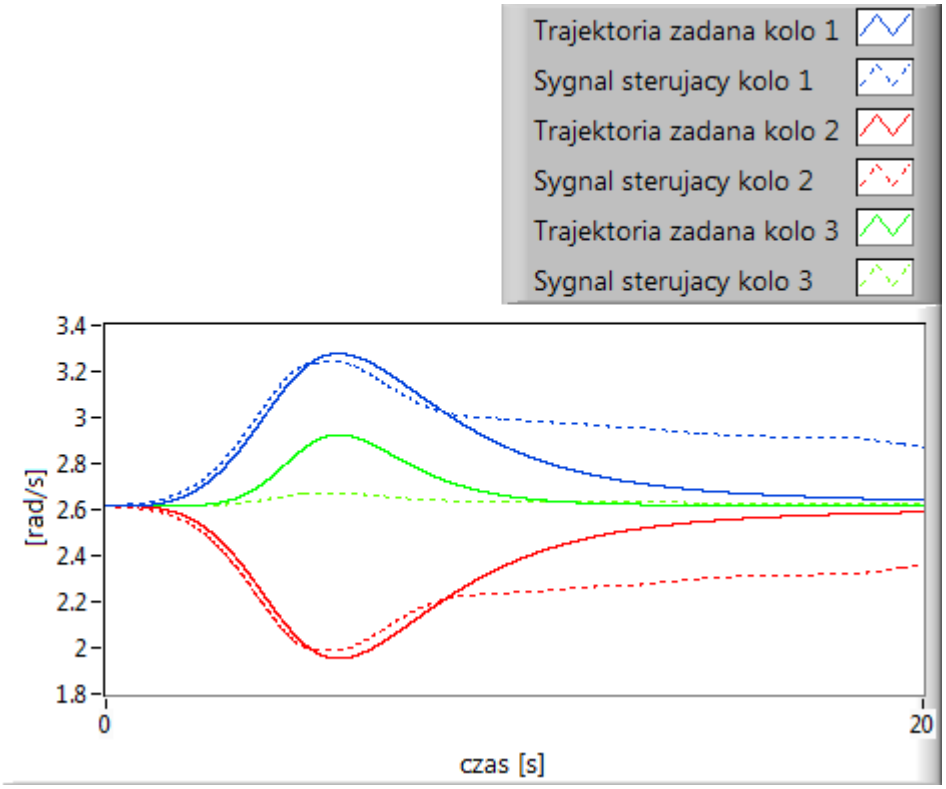
Rys 4.8 przedstawia sytuację, w której platformie od 7 s są nadawane zbyt małe wartości sygnału sterującego, co skutkuje powstaniem błędów w trajektorii przebiegu wartości chwilowej obrotu koła kierownicy. Pomimo, że optymalne wartości momentów napędowego jak i kierującego generowane przez sterownik (rys. 4.6 oraz rys. 4.7) są zbieżne z wartościami zadanymi (uzyskanymi w drodze rozwiązania równań Lagrange’a), to silna nieliniowość układu, a także występujące błędy numeryczne, decydują o niedokładnym odwzorowaniu trajektorii zadanej.

Podobnie, jak to miało miejsce w przypadku trajektorii typu „sinus”, wartość kąta obrotu kierownicy determinuje wartość kąta obrotu platformy. Na rys. 4.9 zobrazowano wartość chwilową tego przebiegu. Błąd w położeniu kierownicy ma istotny wpływ na wartości błędu kąta obrotu platformy, które w 10 s sięgają 80%. Warto zauważyć, że w drugim okresie przejazdu platformy wartość powstałego błędu maleje.



Rys. 4.9. Chwilowa wartość kąta obrotu platformy (przy sygnale sterującym – kolor czerwony, wartość zadana – kolor niebieski) dla trajektorii typu „parabola”

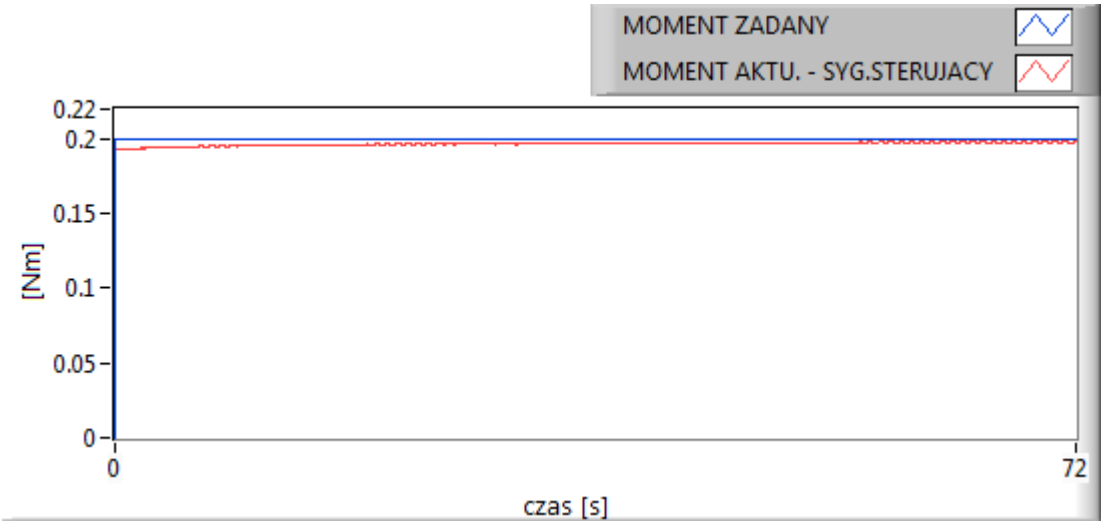
Z uwagi na fakt, iż wartość kąta obrotu kierownicy wpływa bezpośrednio na prędkości poszczególnych kół, również od 7 s zarysowuje się różnica w uzyskanych przebiegach (rys. 4.10). Platforma porusza się szybciej, niż wynika to z trajektorii zadanych. Podobnie, jak to miało miejsce dla kąta obrotu ramy platformy, błąd w szybkości poruszania się platformy ulega zmniejszeniu, aby w końcowej fazie wirtualnego przejazdu osiągnąć wartość około 12%.



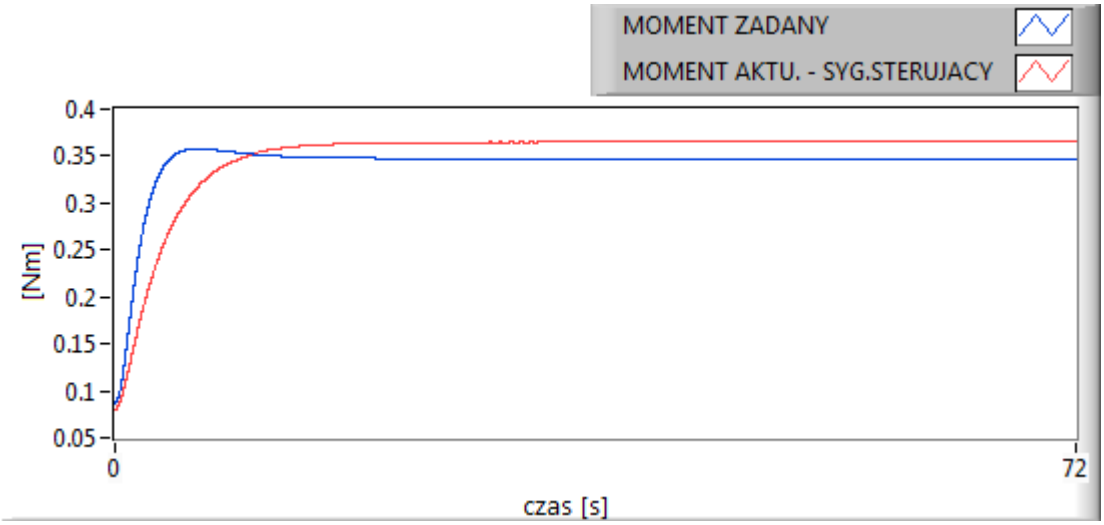
Rys. 4.10. Chwilowe wartość prędkości kątowych kół (przy sygnale sterującym – linia przerywana, wartość zadana – linia ciągła) dla trajektorii typu „parabola”

4.1.3. Wirtualne prototypowanie systemu nadzorowania dla trajektorii typu „okrąg”

Realizacja procesu wirtualnego prototypowania dla przejazdu platformy po krzywej typu „okrąg” (rys. 4.11 oraz rys. 4.12), potwierdziła wnioski uzyskane podczas analizy wyników dla trajektorii typu „sinus” oraz „parabola” (przyczyna występujących błędów tkwiąca w zastosowanej metodzie numerycznej). W tym jednak przypadku, koło kierujące, po osiągnięciu właściwego kąta względem platformy (w 8 s), nie zmienia swego położenia, aż do końca trwania przejazdu. Fakt ten skutkuje stałym (zmieniającym się w niewielkim zakresie) błędem występującym dla wartości chwilowej momentu kierującego i napędowego. Powyższe przeregulowanie zostanie zminimalizowane, tak jak to miało miejsce dla innych trajektorii, przez odpowiedni dobór współczynników wzmocnień sterowników.

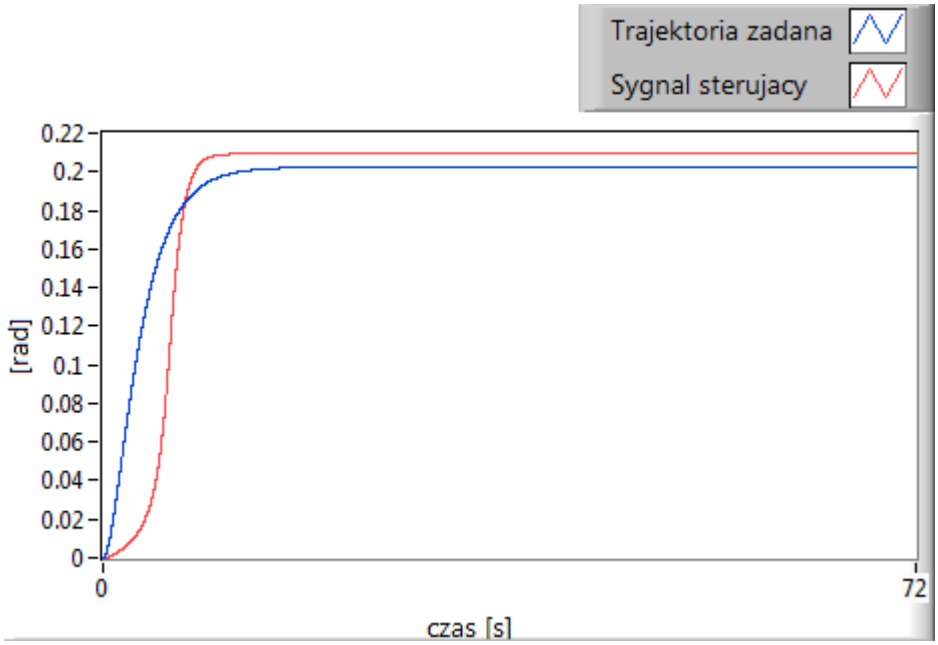


Rys. 4.11. Chwilowa wartość sygnału sterującego dla momentu kierującego (niebieski) oraz momentu kierującego zadanego (czerwony) dla trajektorii typu „okrąg”



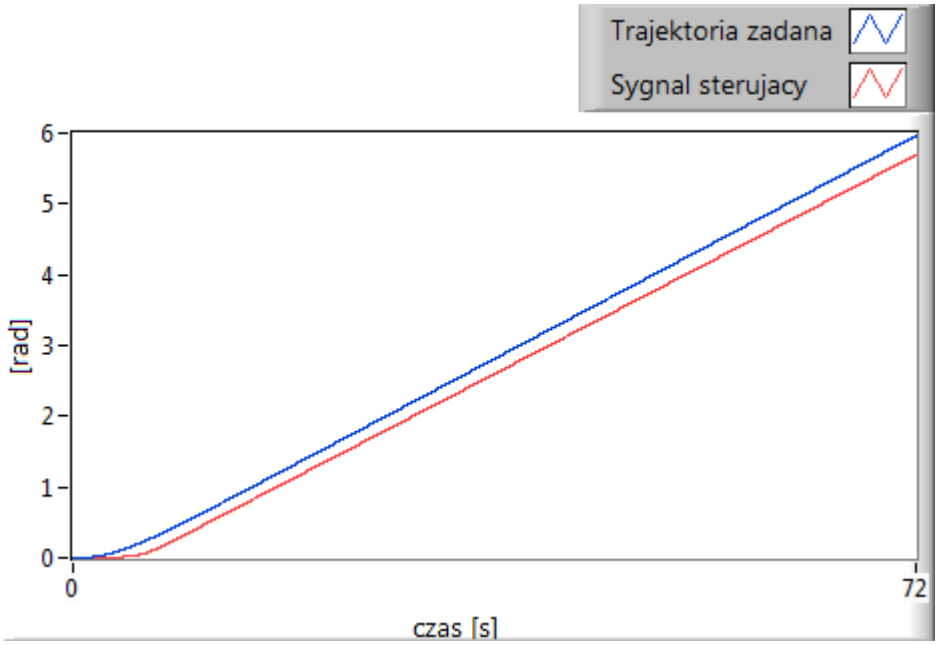
Rys. 4.12. Chwilowa wartość sygnału sterującego dla momentu napędowego (niebieski) oraz momentu napędowego zadanego (czerwony) dla trajektorii typu „okrąg”

Na rys. 4.13 zamieszczono wyniki wirtualnego prototypowania dla kąta obrotu kierownicy. Z przebiegu wartości chwilowej tego kąta wynika, że tylko w pierwszej części przejazdu (do 10 s) wirtualna platforma doznaje dewiacji (błędu od założonego ruchu). W tym czasie koło kierownicy obraca się, aby osiągnąć pozycję niezbędną do przejazdu po wyznaczonej trajektorii. Osiągnięty kąt odbiega od wartości zadanej, co skutkuje utrzymaniem się niewielkiej wartości błędu (2,5%) aż do końca trwania przejazdu.



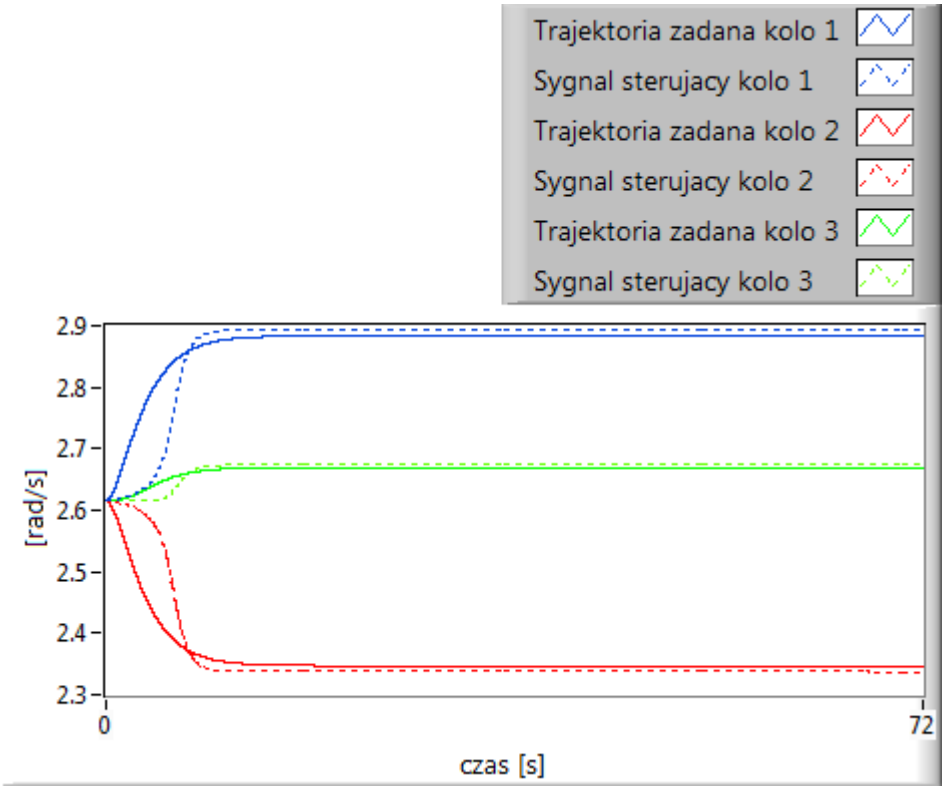
Rys. 4.13. Chwilowa wartość kąta obrotu kierownicy (przy sygnale sterującym – kolor czerwony, wartość zadana – kolor niebieski) dla trajektorii typu „okrąg"

Niewielka wartość błędu w chwilowej wartości kąta obrotu koła kierownicy determinuje fakt, iż chwilowa wartość kąta obrotu platformy (rys. 4.14) również niewiele różni się od wartości zadanej. Od około 5 sekundy aż do końca trwania przejazdu utrzymuje się stały błąd na poziomie 5%.



Rys. 4.14. Chwilowa wartość kąta obrotu platformy (przy sygnale sterującym – kolor czerwony, wartość zadana – kolor niebieski) dla trajektorii typu „okrąg"

Błąd wartości chwilowych prędkości kątowych poszczególnych kół platformy (rys. 4.15) ze względu na niewielki błąd kąta obrotu kierownicy, dostrzegalny jest tylko do 8 s. W pozostałej części pokonywanej trasy, kołom nadawane są niemal identyczne wartości, co wynika z wartości zadanych.



Rys. 4.15. Chwilowe wartość prędkości kątowych kół (przy sygnale sterującym – linia przerywana, wartość zadana – linia ciągła) dla trajektorii typu „okrąg”

4.1.4. Podsumowanie wirtualnego prototypowania systemu nadzorowania

Przedstawione rozwiązania graficzne (rys. 4.1 – 4.15) optymalnych sygnałów sterujących, a także będących ich konsekwencją odpowiedzi obiektu (platformy), potwierdzają poprawność implementacji algorytmu będącego przedmiotem badań. Wirtualnej platformie nadawane są momenty o wartościach różniących się od wartości zadanej maksymalnie o 5%. Pomimo silnej nieliniowości układu, na wszystkich trasach przejazdu brak jest jakichkolwiek, lokalnych odchyłek mogących świadczyć o utracie stabilności, czy wejścia kół w poślizg.

Na uwagę zasługują wartości chwilowe uzyskane dla trajektorii typu „sinus”, która jest krzywą niezwykle wymagającą dla realizowanej platformy. Pomimo trudnego technicznie przejazdu, projektowany system sterowania generuje momenty zbieżne z wartościami zadanymi. Dla pozostałych trajektorii system zachowuje

podobną tendencję (szczególnie dla trajektorii typu „okrąg”), w przypadku której optymalne sygnały sterujące tylko w nieznaczny sposób różnią się od wartości zadanych.

Na skutek uciążliwego procesu obliczeń, wypływających negatywnie na wierność odtworzenia trajektorii zadanych, na uwagę zasługują jedynie dość duże błędy dla trajektorii typu „parabola”, dla której błędy w trakcie przejazdu sięgają 80% wartości zadanej.

Zaimplementowany do środowiska LabVIEW algorytm, z racji swojej natury (matematycznego zapisu) jest wymagający, biorąc pod uwagę wykorzystanie pamięci operacyjnej oraz zasobów procesora. Proces mnożenia macierzy o nieliniowych współczynnikach, czy też obliczanie macierzy odwrotnej, absorbuje w sposób znaczący zasoby komputera (w trakcie realizacji obliczeń procesor obciążony jest w około 100%). Dlatego, niezwykle istotnym okazał się proces optymalizacji, tj.:

- zapisu algorytmu w środowisku LabVIEW oraz miejsca jego wykonania (postanowiono wydzielić część wykonywaną w procesorze Real Time oraz w jednostce FPGA; szczegóły omówiono w rozdziale 5);
- energetycznej (w sposób najbardziej zauważalny) systemu sterowania (tj. dobór kroku całkowania).

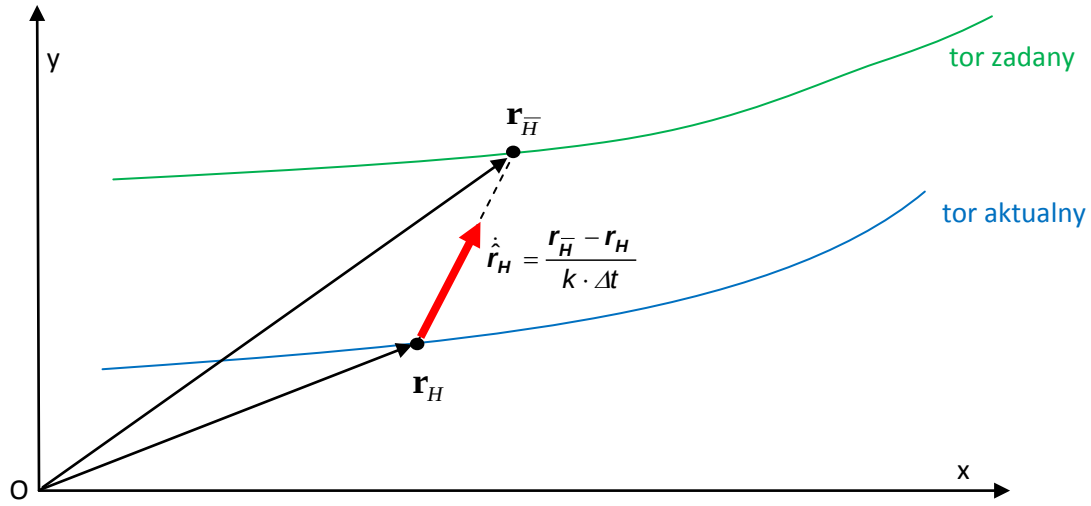
Wykorzystana technika wirtualnego prototypowania znacznie skróciła czas weryfikacji zastosowanych rozwiązań oraz przyjętych koncepcji. W trakcie realizacji badań, optymalizacji podległ proces zapisu algorytmu w środowisku LabVIEW (poprawie uległa czytelność algorytmu; wyeliminowanie części bloków pozwoliło zmniejszyć wykorzystanie zasobów procesora, a także pamięci operacyjnej), a także – zawężono zakres poszukiwań wartości elementów macierzy **R** oraz **Q**.

4.2. Prędkości korygujące

Główną swoją uwagę autor, w realizowanych badaniach skierował na wykorzystaniu technik projektowania mechatronicznego do realizacji systemu sterowania, który dzięki efektywnej implementacji energetycznego wskaźnika jakości, pozwala na skuteczne nadzorowanie obiektu badań.

Swoje pierwsze rozważania autor podjął rozpatrując zagadnienie poruszania się platformy mobilnej po wyznaczonej trajektorii. W trakcie tego ruchu system sterowania (algorytm) w każdej zdefiniowanej chwili czasu, za pomocą metod numerycznych (realizowanych w CPU) oblicza chwilowe wartości sygnału sterującego (w tym przypadku, moment napędowy oraz kierujący otrzymany drogą rozwiązania równań różniczkowych). Obliczona całka sygnału sterującego generowana jest w równych odstępach czasu Δt (autor wykorzystał stałokrokową metodę Eulera), definiowanych jako długość kroku całkowania. W tym okresie czasu system sterowania platformy mobilnej jest nieaktywny (nie jest generowany sygnał sterujący). Podczas braku aktywności systemu sterowania platforma przebywa określoną drogę, z której wynika różnica położenia platformy względem zadanego toru ruchu [38]. W celu skorygowania zaistniałej różnicy położeń punktu H na torze ruchu, polegającej na „naprowadzeniu" punktu H z położenia aktualnego w chwili czasu t (opisanego wektorem $\mathbf{r}_H$) do położenia zadanego w tej samej chwili czasu t (opisanego wektorem $\mathbf{r}_{\overline{H}}$), zdefiniowano wektor prędkości korygujących $\dot{\hat{\mathbf{r}}}_H$ (rys. 4.16). W wyniku rzutowania wektora prędkości korygujących na osie płaskiego kartezjańskiego układu współrzędnych Oxy otrzymujemy:

$$\dot{\hat{x}}_H = \frac{\overline{x}_H - x_H}{k \cdot \Delta t}, \; \dot{\hat{y}}_H = \frac{\overline{y}_H - y_H}{k \cdot \Delta t} \tag{4.6}$$



Rys. 4.16. Błąd położenia punktu H platformy mobilnej wynikający z przyjętego kroku całkowania Δt

Przyjmując, że prędkość korygująca punktu charakterystycznego A wynosi:

$$\hat{v}_A = \dot{\hat{\psi}} r, \tag{4.7}$$

gdzie: $\dot{\hat{\psi}}$ – korygująca prędkość kątowa zastępczego koła napędowego, oraz wykorzystując zależności (4.6) można określić (stosując podstawianie w równaniach różniczkowych kinematyki układu, tj. zależność (3.5) oraz (3.13)) prędkości korygujące dla wszystkich parametrów platformy. Autor do rozwiązania układu równań (3.5), (3.13) oraz (4.6) i (4.7) wykorzystał pakiet Maple. Przykładowo, dla trajektorii typu „okrąg" otrzymano układ równań (4.8), który w późniejszej fazie zaimplementowano do środowiska LabVIEW.

$$\begin{cases}
\dot{\hat{\varphi}} = -\big[\ l_0\,\dot{\hat{x}}_H \sin\beta + l_0\,\dot{\hat{x}}_H \cos\beta\, tg\varphi + l_3\,\dot{\hat{x}}_H\, tg\varphi \cos(\beta+\alpha) + \\
+ l_3\,\dot{\hat{y}}_H\, tg\varphi\, \sin(\beta+\alpha) - l_0\,\dot{\hat{y}}_H \cos\beta + l_0\,\dot{\hat{y}}_H \sin\beta\ tg\varphi\ \big] / \big[\ l_3\, l_0 (\sin(\beta+\varphi)\sin\beta + \\
+ \sin(\beta+\varphi)\ \cos\beta\, tg\varphi + \cos(\beta+\varphi)\cos\beta - \cos(\beta+\varphi)\, tg\varphi \sin\beta)\ \big], \\
\dot{\hat{\beta}} = \big[\ (\dot{\hat{x}}_H \cos(\beta+\alpha) + \dot{\hat{y}}_H \sin(\beta+\alpha))\, tg\varphi\ \big] / \big[\ l_0\,(\sin(\beta+\varphi)\sin\beta + \\
+ \sin(\beta+\varphi)\cos\beta\, tg\varphi + \cos(\beta+\varphi)\cos\beta - \cos(\beta+\varphi)\, tg\varphi \sin\beta)\ \big], \\
\dot{\hat{\psi}} = \big[\ \dot{\hat{x}}_H (\cos(\beta+\alpha) + \dot{\hat{y}}_H \sin(\beta+\alpha))\ \big] / \big[\ r\,(\sin(\beta+\varphi)\sin\beta + \sin(\beta+\varphi)\cos\beta\, tg\varphi + \\
+ \cos(\beta+\varphi)\ \cos\beta - \cos(\beta+\varphi)\, tg\varphi \sin\beta)\ \big], \\
\dot{\hat{\alpha}}_1 = \big[\ \dot{\hat{x}}_H (\cos(\beta+\alpha) + \dot{\hat{y}}_H \sin(\beta+\alpha))(l_0 + l_1\, tg\varphi)\ \big] / \big[\ r\, l_0 (\sin(\beta+\varphi)\sin\beta + \\
+ \sin(\beta+\varphi)\ \cos\beta\, tg\varphi + \cos(\beta+\varphi)\ \cos\beta - \cos(\beta+\varphi)\, tg\varphi\ \sin\beta)\ \big], \\
\dot{\hat{\alpha}}_2 = -\big[\ \dot{\hat{x}}_H (\cos(\beta+\alpha) + \dot{\hat{y}}_H \sin(\beta+\alpha))\ (-l_0 + l_1\, tg\varphi)\ \big] / \big[\ r\, l_0 (\sin(\beta+\varphi)\sin\beta + \\
+ \sin(\beta+\varphi)\ \cos\beta\, tg\varphi + \cos(\beta+\varphi)\ \cos\beta - \cos(\beta+\varphi)\, tg\varphi\ \sin\beta)\ \big], \\
\dot{\hat{\alpha}}_3 = -\big[\ \dot{\hat{x}}_H (\cos(\beta+\alpha) + \dot{\hat{y}}_H \sin(\beta+\alpha))(-\cos\beta + tg\varphi\ \sin\beta)\ \big] / \big[\ (\sin(\beta+\varphi)\sin\beta + \\
+ \sin(\beta+\varphi)\ \cos\beta\, tg\varphi + \cos(\beta+\varphi)\cos\beta - r\cos(\beta+\varphi) tg\varphi\ \sin\beta)\ \cos(\beta+\varphi)\ \big].
\end{cases} \tag{4.8}$$

W przypadku sterowania ruchem układu silnie nieliniowego, jakim jest trójkołowa platforma mobilna, energetyczny wskaźnik jakości (zależność 4.1) należy zmodyfikować o wartości prędkości korygujących. W rezultacie, otrzymujemy optymalny sygnał sterujący wyrażony wzorem:

$$\mathbf{u}(t) = (\mathbf{R} + \mathbf{R}^T)^{-1} \int_{t-\Delta t}^{t} \mathbf{B}^{\mathrm{T}}(\tau)\boldsymbol{\Theta}^T(t,\tau)d\tau \cdot \mathbf{T}^T\left(\mathbf{QM}\ + \mathbf{M}^T\mathbf{Q}^T\right)\left(\dot{\mathbf{q}} - \dot{\bar{\mathbf{q}}} - \dot{\hat{\mathbf{q}}}\right), \tag{4.9}$$

gdzie: $\dot{\hat{\mathbf{q}}}$ – wektor uogólnionych prędkości korygujących.

Kierunek wektora $\dot{\hat{\mathbf{r}}}_H$ jest znany. Mechatronicznemu doborowi (poprzez współczynnik skalujący *k* długość kroku całkowania Δt) podlega jego długość tak, aby operacja dodawania otrzymanego wektora kompensowała otrzymaną odchyłkę, czyli występujący błąd powstały na skutek przejazdu w czasie Δt platformy po drodze, bez sygnału sterującego. Przyjmując nieskończenie mały krok całkowania, błąd (długość wektora) będzie zmierzał do zera. Niestety, co później wykaże autor, będzie to obarczone znacznym wzrostem czasu obliczeń, w przypadku wirtualnego prototypowania [38]. Należy zaznaczyć, że realizowany przez autora system nadzorowania platformą mobilną jest systemem czasu rzeczywistego Real Time (nie jest dopuszczalne, aby ustawianie czasu przejazdu platformy na 50 s zajmowało w rzeczywistości 58 s). W tym przypadku czas obliczeń jest ściśle zdeterminowany (czas symulacji równy jest czasowi rzeczywistemu, wynikającemu z pracy zegara systemowego). Autor mając do dyspozycji jedną z najbardziej wydajnych jednostek sterujących National Instruments, nie mógł dowolnie dobierać prędkości pracy procesora. Ten parametr w całym cyklu badań pozostał stały. Powyższa sytuacja determinowała fakt, iż musiał wystąpić kompromis pomiędzy szybkością obliczeń (długością kroku całkowania) tak, aby zapewnić determinizm czasowy obliczeń, a wartością występującego błędu. Należy zauważyć (co jest kluczowe w tych rozważaniach), że dla danego kroku całkowania występujący błąd może zostać skompensowany przez dodanie w równaniu sygnału sterującego wartości prędkości korygujących. Dzięki takiemu zabiegowi, dla danego kroku całkowania osiąga się zamierzoną optymalizację energetyczną, tzn. zastosowany procesor pracuje w czasie rzeczywistym (ponieważ jest spełniony przyjęty przez autora warunek optymalności dla danej jednostki CPU), a błędy wynikające z zastosowanego kroku całkowania mogą być skutecznie kompensowane przez prędkości korygujące (dla zadanego kroku całkowania).

4.3. Optymalizacja energetyczna układu

Problem, rozważanego powyżej doboru długości kroku całkowania, podczas generowania sygnału sterującego (tj. momentów napędowego i kierującego) jest ścisłe uzależniony od zastosowanego procesora projektowanego systemu nadzorowania oraz bezpośrednio wpływa na wydajność energetyczną realizowanego układu platformy.

Autor rozważając zagadnienie wpływu zastosowanej jednostki CPU, ograniczył swoje rozważania do generalnych wytycznych (reguł wynikających z zachodzących zjawisk fizycznych) zakładając wzrost zużycia energii wraz ze wzrostem prędkości pracy CPU (liczby wykonywanych operacji w danej jednostce czasu). Objawia się to wydłużonym czasem obliczeń (jeżeli rozwiązania nie są generowane w czasie Real Time), większym poborem prądu (większa liczba zasobów (tranzystorów) jednostki arytmetyczno-logicznej ALU bierze udział w obliczeniach), a także zwiększoną intensywnością procesu chłodzenia rozpatrywanej jednostki. Należy zaznaczyć, że wydajność energetyczna systemu autonomicznego, jakim jest platforma mobilna, jest krytycznym parametrem, który powinien być minimalizowany. W przypadku realizacji projektów komercyjnych, optymalizacja systemu energetycznego powinna być przeprowadzona, mając w zamyśle zadanie realizowane przez projektowaną platformę oraz czas realizacji, np. 120 min. Konstruktorzy takich układów mają zazwyczaj zdefiniowane źródło energii (np. baterie bądź ogniwa paliwowe, w przypadku których pojemność jest determinowana przyjętą konstrukcją obiektu). Dostępna pojemność (ilość możliwej do dostarczenia energii) będzie determinowała architekturę sprzętową systemu sterowania oraz, co jest przedmiotem badań autora, długość kroku całkowania. Jest oczywistym, że w przypadku, gdy dostęp do energii jest sprawą drugorzędną (np. platforma porusza się w dzień po obszarze pustynnym, a źródłem energii są wydajne ogniwa słoneczne), architektura systemu sterowania może bazować na najbardziej wydajnych jednostkach CPU, a krok całkowania może przyjąć dowolnie małe rozmiary (jeżeli dodatkowo ogranicznikiem nie są źródła finansowania projektu i zapewni się odpowiednie układy chłodzenia). W niniejszej pracy pojemność źródła (baterii) nie była przedmiotem rozważań. Autor dobrał typ baterii tak, aby zapewnić przejazd po wytyczonej trasie, dostarczając energii do systemu sterowania oraz silników. Główną wytyczną doboru baterii był jej ciężar tak, aby całkowita masa platformy nie przekroczyła 10 kg. Masa platformy została ustalona po przeprowadzonej analizie wydajności systemu sterowania, w zestawieniu z założeniami konstrukcyjnymi stawianymi projektowanemu obiektowi, a także możliwościami (w tym i dostępności) sprzętowymi niezbędnymi w procesie realizacji koncepcji m.in. silników DC oraz mechanizmu różnicowego (w platformie zastosowano najbardziej trwałą jednostkę dostępną na rynku).

Brak rozważań związanych z pojemnością źródła energii determinuje koncepcję przeprowadzonych badań oraz analizy, w której autor dla danego procesora

poszukiwał kompromisu pomiędzy występującymi błędami wynikającymi z przyjętej długości kroku całkowania, a wydajnością energetyczną (w przypadku techniki wirtualnego prototypowania był to czas przeprowadzenia symulacji, w przypadku testów HILS [48, 68-70] wymagania dotyczyły zapewnienia równości pomiędzy czasem przeprowadzenia symulacji, a czasem rzeczywistym). Dla podkreślenia warto zauważyć, że w trakcie tych poszukiwań autor nie brał pod uwagę wpływu pojemności źródła energii na parametry operacyjne platformy.

4.4. Metodologia optymalizacji systemu nadzorowania

Obszar badań został ograniczony do zbadania jednej jednostki CPU (Intel Core i5, M560 @2,76 GHz) w przypadku realizacji techniki wirtualnej, gdzie wykorzystano komputer klasy PC, oraz jednego procesora wynikającego z zastosowanego sterownika cRIO (400MHz) – dla techniki HILS (szczegółowo opisanej w rozdziale 5).

Autor projektując system nadzorowania trójkołowej platformy mobilnej, rozważał kilka wartości kroku całkowania, tj. 0,01 s, 0,005 s, 0,001 s oraz 0,0005 s. Powstały system powinien być zoptymalizowany energetycznie tzn. pod względem wydajności (jakości) sterowania, w sensie minimalizacji wartości błędów występujących podczas przejazdu platformy oraz sprawności energetycznej (optymalne zużycie energii), przy jednoczesnym zapewnieniu czasu rzeczywistego (determinizmu) podczas generowania sygnału sterującego [37]. Badania zostały przeprowadzone dla rozważanych trajektorii „sinus", „parabola" oraz „okrąg".

Dla podanych kroków całkowania oraz przyjętego modelu obliczeniowego projektowanej platformy, autor wygenerował (stosując metody numeryczne) w zrealizowanym systemie sterowania (w środowisku LabVIEW) optymalne sygnały sterujące (moment napędowy i kierujący). W następnym kroku zbadano wpływ wartości występujących błędów wynikających z różnicy pomiędzy trajektorią zadaną i trajektorią zrealizowaną, a czasem przeprowadzonych symulacji wirtualnych (wirtualnego prototypowania). Należy zauważyć, że w technice wirtualnego prototypowania, w której operujemy modelem obliczeniowym sterownika i platformy, czas przeprowadzenia symulacji nie jest zdeterminowany. Zależy on ściśle od przyjętej metody numerycznej (w tym, długości kroku całkowania), szybkości pracy

procesora, wielkości pamięci operacyjnej oraz przyjętej architektury systemu, bądź algorytmu. W tym przypadku, o czym autor wspominał powyżej, dane definiujące typ procesora nie są parametrami. Badaniu podlegała zdolność procesora do generowania sygnału sterującego przy zadanym kroku całkowania (oraz występujących błędów). Ze względu na konieczność sprawdzenia powtarzalności otrzymanych rezultatów, autor postanowił przeprowadzić analizę dla momentu kierującego oraz napędowego. Fakt, iż obie analizy przeprowadzano równocześnie (w tym samym czasie generowano rozwiązania dla momentu kierującego i napędowego) zdeterminował czas obliczeń, który w obu przypadkach jest identyczny. Należy zauważyć, że autor tak dokonał przekształceń algorytmu (proces implementacji do środowiska LabVIEW), że nie istniała fizyczna możliwość odseparowania obu rodzajów analizy (sygnały generowane są przez ten sam blok podsystemu rys. 5.5).

Wyniki przeprowadzonej analizy zestawiono w tab. 4.1 oraz tab. 4.2, z których wynika, że zmniejszeniu kroku całkowania towarzyszy mniejszy błąd, jednak obarczone jest to zwiększonym czasem przeprowadzenia symulacji, a co za tym idzie – wzrostem zapotrzebowania energetycznego. Należy zaznaczyć, że w celu zrównania otrzymanych czasów przeprowadzonych symulacji, wraz ze zmniejszeniem kroku całkowania należałoby zwiększyć szybkość pracy procesora, a tym samym – poddać rekonfiguracji system chłodzenia układu sterowania. Efektem byłoby zwiększenie zapotrzebowania energetycznego i co za tym idzie – spadek tak pożądanej operacyjności platformy mobilnej.

Dodatkowo, na uwagę zasługuje fakt (szersza dyskusja poniżej), który został przez autora wykorzystany podczas optymalizacji energetycznej systemu nadzorowania (dokładniej przy wyborze kroku całkowania). Zmniejszenie kroku całkowania z Δt=0,005 s do Δt=0,001s pociągnęło za sobą kilkukrotny wzrost czasu obliczeń (a tym samym wystąpiła sytuacja, w której, dla jednostki cRIO nie mógł być spełniony warunek determinizmu czasowego). Jednakże różnica błędu pomiędzy czasem Δt=0,001 s a Δt=0,005 s jest na tyle mała, że stało się bezcelowym poszukiwaniem innych rozwiązań sprzętowych, umożliwiających realizację systemu nadzorowania dla kroku całkowania Δt=0,001 s.

Tab. 4.1. Wyniki przeprowadzonych badań wpływu kroku całkowania na występujące błędy oraz czas obliczeń dla rozpatrywanych trajektorii. Moment kierujący. Przez „–” oznaczono fakt otrzymania wyników obarczonych błędami wykraczającymi poza fizyczną interpretacje. Dalsza ich analiza została pominięta

Krok całkowania [s]	**Czas obliczeń [ms]**	**Średnia wartość strat energetycznych= $\frac{\sum_i \lvert \Delta \mathbf{r}_i \rvert}{i}$ [Nm]**
Trajektoria „sinus” – przejazd 50 s		
0,01	7848	0,026436
0,005	18985	0,000686
0,001	91056	0,0001018
0,0005	145843	0,0000834096
Trajektoria „parabola” – przejazd 72 s		
0,01	2350	-
0,005	5883	0,001652
0,001	32057	0,00012
0,0005	63386	0,0000540326
Trajektoria „okrąg” – przejazd 20 s		
0,01	9129	-
0,005	20227	0,000234256
0,001	167075	0,0000675311
0,0005	540690	0,0000102256

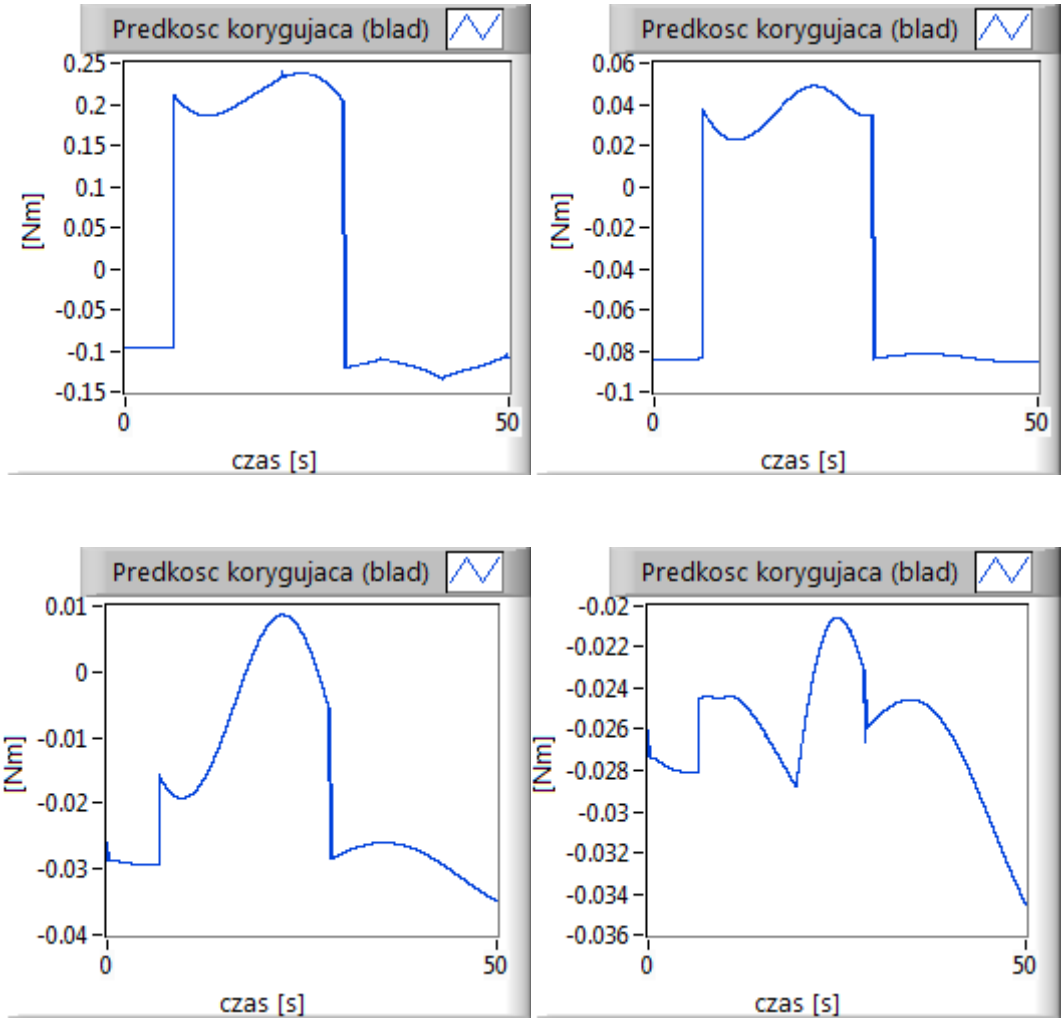
Tab. 4.2. Wyniki przeprowadzonych badań wpływu kroku całkowania na występujące błędy oraz czas obliczeń dla rozpatrywanych trajektorii. Moment napędowy. Przez „–" oznaczono fakt otrzymania wyników obarczonych błędami wykraczającymi poza fizyczną interpretacje. Dalsza ich analiza została pominięta

Krok całkowania [s]	**Czas obliczeń [ms]**	**Średnia wartość strat energetycznych= $\frac{\sum_i \lvert \Delta \mathrm{r_i} \rvert}{i}$ [Nm]**
Trajektoria „sinus" – przejazd 50 s		
0,01	7848	0,0000366063
0,005	18985	0,00000919
0,001	91056	0,00000132878
0,0005	145843	0,000000583862
Trajektoria „parabola" – przejazd 72 s		
0,01	2350	-
0,005	5883	0,00000363
0,001	32057	0,000000287
0,0005	63386	0,000000126
Trajektoria „okrąg" – przejazd 20 s		
0,01	9129	-
0,005	20227	0,00013329
0,001	167075	0,0000151013
0,0005	540690	0,00000712851

Zastosowana w tab. 4.1 oraz tab. 4.2 wartość średnia strat energetycznych (definiowana, jako suma wartości bezwzględnych różnicy położeń na torze zadanym i torze aktualnym dla wszystkich punktów wynikających z przyjętego kroku całkowania, w których generowany jest optymalny sygnał sterujący) została wprowadzona jedynie w celu pokazania sumarycznego wpływu zastosowanego kroku całkowania na przebieg realizowanych symulacji (wartość błędu, czas obliczeń).

Autor pracy w przyjętej koncepcji algorytmu sterowania platformą (która różni się od powszechni stosowanego całkowego wskaźnika jakości [33]) posługuje się wartością chwilową tej prędkości, która pozwala na bieżące (*on-line*) śledzenie jej zmiany.

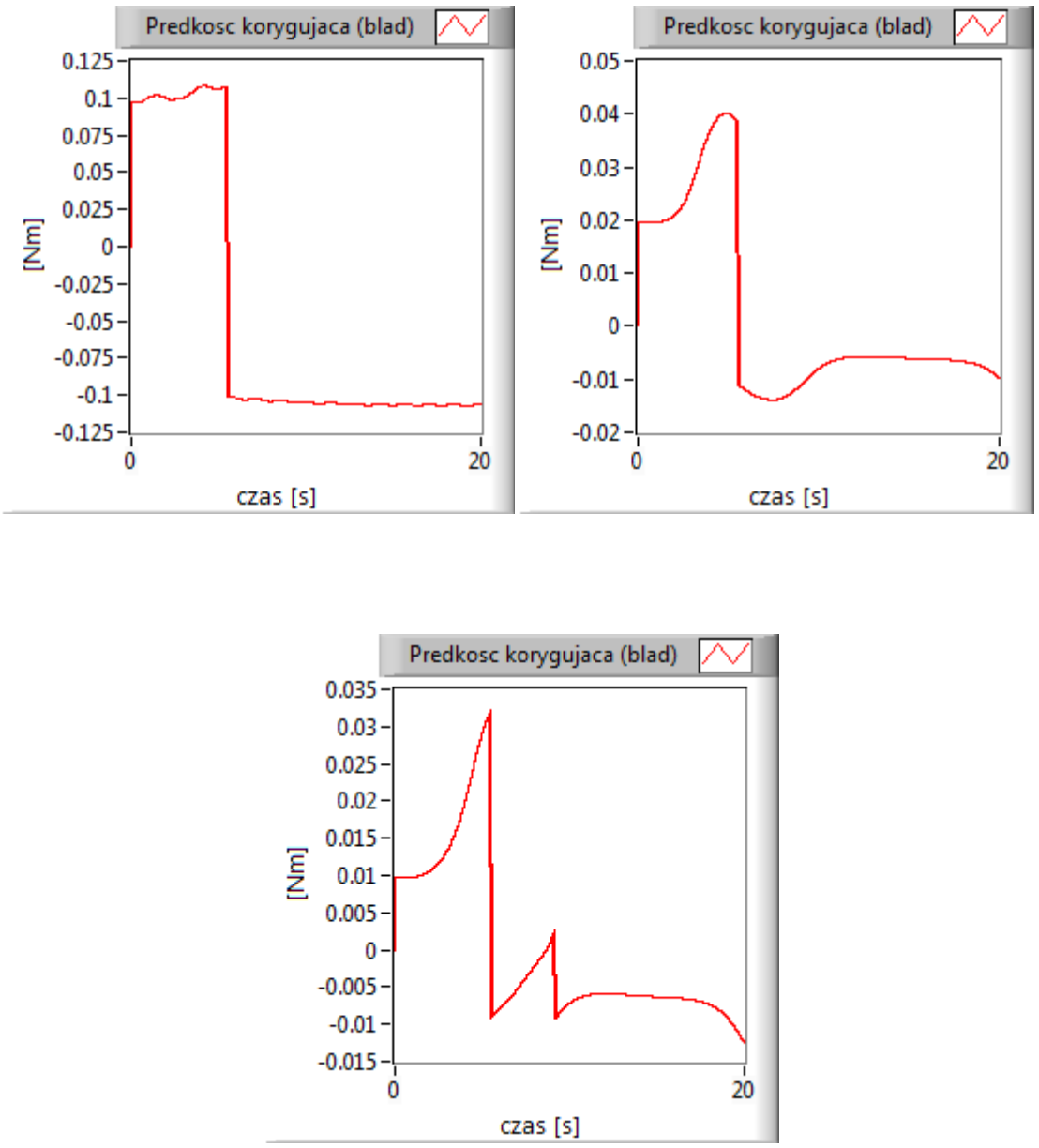
Wykresy chwilowych wartości prędkości korygującej, dla rozpatrywanych kroków całkowania oraz dla przyjętych trajektorii poruszania się platformy, przedstawiono na rys. 4.17, 4.18 oraz 4.19 (dla momentu kierującego).



Rys. 4.17. Chwilowe wartości prędkości korygującej(moment kierujący; trajektoria typu „sinus") dla kroków całkowania: górny lewy Δt=0,01 s, górny prawy Δt=0,005 s, dolny lewy Δt=0,001 s, dolny prawy Δt=0,0005 s

Przebiegi wartości chwilowych prędkości korygujących wykreślone dla trajektorii typu „sinus" (rys. 4.17) potwierdzają przeprowadzoną powyżej analizę. Zmniejszenie kroku całkowania skutkuje większą dokładnością (częstszym generowaniem sygnału) przeprowadzonych obliczeń (rozwiązania równań różniczkowych dla parametrów platformy oraz wygenerowania optymalnego sygnału sterującego przez zaimplementowany algorytm przy pomocy metod numerycznych). Następstwem tego jest sytuacja, dla której platforma mobilna (punkt charakterystyczny H) pozostaje przez krótszy czas bez sterowania, co w efekcie

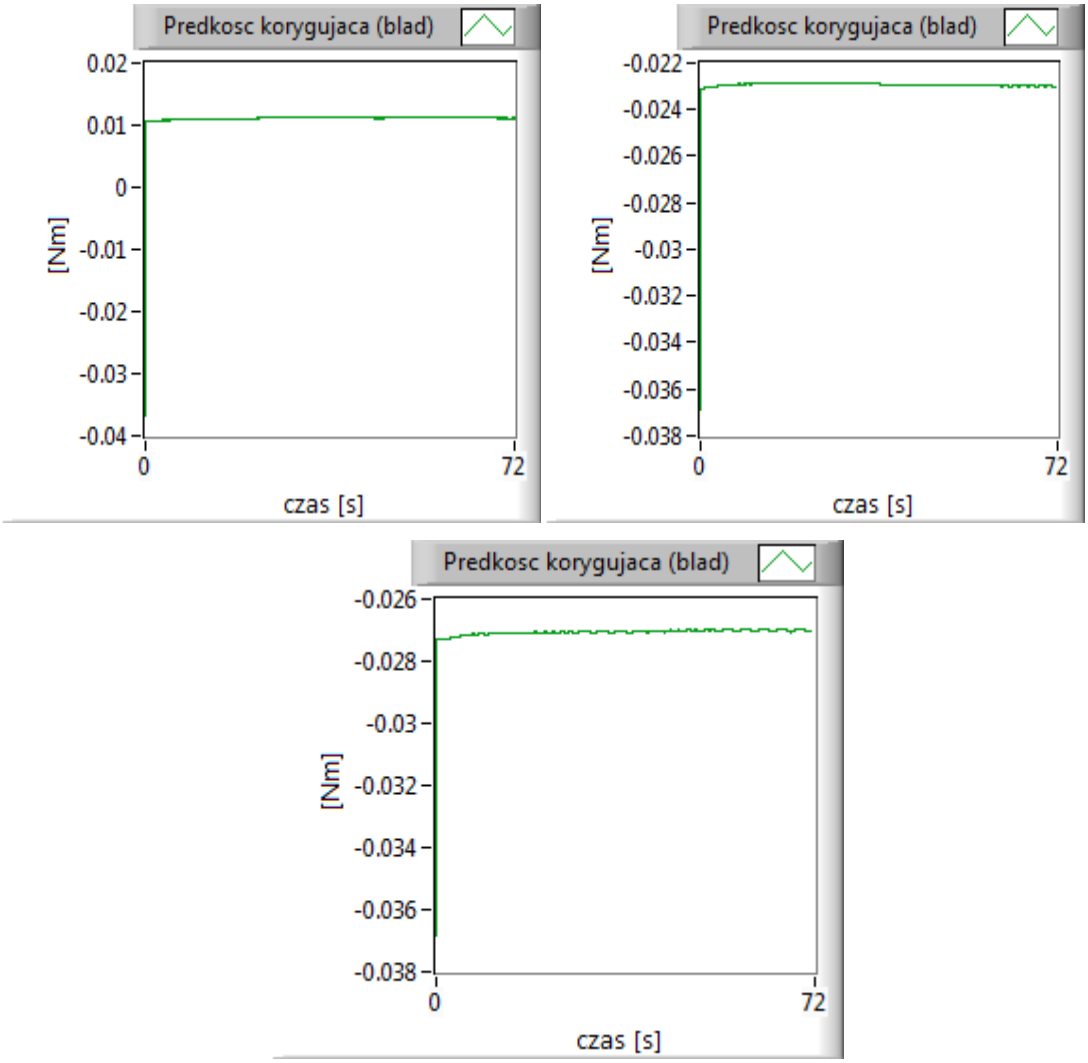
determinuje powstaniem mniejszego błędu (krótszej długość wektora korygującego $\dot{\hat{\mathbf{r}}}_H$). Powyższa sytuacja, jest dobrze zilustrowana na rozpatrywanych rysunkach. Amplituda błędu dla kroku całkowania Δt=0,01 s jest o rząd wielkości większa niż dla kroku całkowania Δt=0,001 s. Należy zauważyć, że o ile dla kroku całkowania Δt=0,01 s oraz Δt=0,005 s wartość chwilowej prędkości korygującej odzwierciedla swoją charakterystyką (kształtem) odpowiednie przebiegi dla momentu kierującego (rys. 4.1) tak, w przypadku kroków krótszych tj. Δt=0,001 s oraz Δt=0,0005 s trudno doszukać się jakichkolwiek zależności. Przebiegi mają charakter bardzo nieregularny, zaś błędy nie przekraczają setnych części Nm. Niezwykle istotnym faktem, o czym autor wspominał powyżej, był wzrost czasu obliczeń (wraz ze zmniejszeniem kroku całkowania), co w rezultacie (podczas weryfikacji determinizmu czasowego w rzeczywistym sterowniku cRIO) powodowało niespełnianie warunku czasu rzeczywistego. Na uwagę zasługuje fakt, że dla kroku całkowania Δt=0,0005 s chwilowe prędkości korygujące przyjmują tylko wartości ujemne. Inaczej wygląda sytuacja dla kroków całkowania o mniejszych wartościach, tj. Δt=0,01 s, Δt=0,005 s oraz Δt=0,001 s, dla których wartości prędkości korygującej są raz dodatnie, innym razem – ujemne (podążają za wartościami chwilowymi momentu kierującego).



Rys. 4.18. Chwilowe wartości prędkości korygującej(moment kierujący; trajektoria typu „parabola") dla kroków całkowania: górny lewy Δt=0,005 s, górny prawy Δt=0,001 s, dolny Δt=0,0005 s

Na rys. 4.18 pokazano wartości chwilowe prędkości korygującej dla trajektorii typu „parabola". Z uwagi na otrzymanie dla kroku całkowania Δt=0,01 s błędów prędkości korygującej wykraczających poza fizyczną interpretację, autor nie zamieścił otrzymanego rezultatu tego przebiegu. Interpretacja otrzymanych rezultatów dla trajektorii typu „parabola", jest tożsama z wynikami analizy dla trajektorii typu „sinus". Zmniejszenie kroku całkowania w trakcie generowania sygnału sterującego determinuje zmniejszenie wartości występujących błędów, przy jednoczesnym, wcześniej omawianym wzroście czasu obliczeń. Dla kroków całkowania Δt=0,005 s oraz Δt=0,001 s, podobnie jak dla trajektorii typu „sinus", wartość prędkości

korygującej nie odzwierciedla kształtem krzywej wartości momentu kierującego. Należy zwrócić uwagę, że dla kroku całkowania wynoszącego Δt=0,0005 s otrzymano (w przeciwieństwie do trajektorii typu „sinus") wartość zarówno dodatnią jak i ujemną.

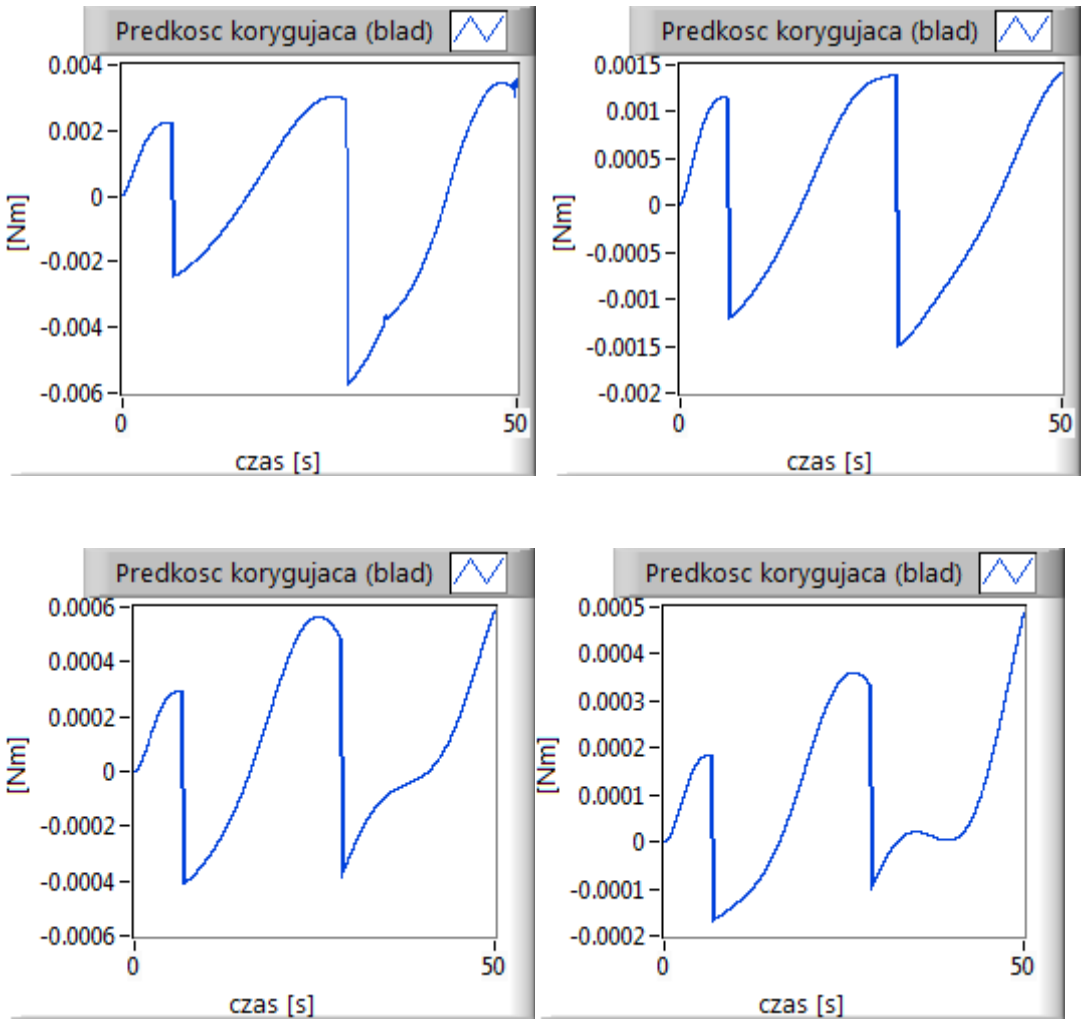


Rys. 4.19. Chwilowe wartości prędkości korygującej(moment kierujący; trajektoria typu „okrąg") dla kroków całkowania: górny lewy: Δt=0,005 s, górny prawy Δt=0,001 s, dolny Δt=0,0005 s

Wyniki otrzymane dla trasy przejazdu typu „okrąg" (rys. 4.19) potwierdzają wnioski otrzymane z wcześniejszych badań dla trajektorii typu „sinus" oraz „parabola" i wykazują ścisłą zależność pomiędzy długością kroku całkowania, wartością błędów i czasem trwania obliczeń. W tym jednak przypadku, dla wszystkich kroków całkowania (otrzymanie dla kroku całkowania Δt=0,01 s błędów wykraczających poza fizyczną interpretację spowodowało zaniechanie analizy tego przypadku) wartości chwilowe prędkości korygujących, odzwierciedlają kształt

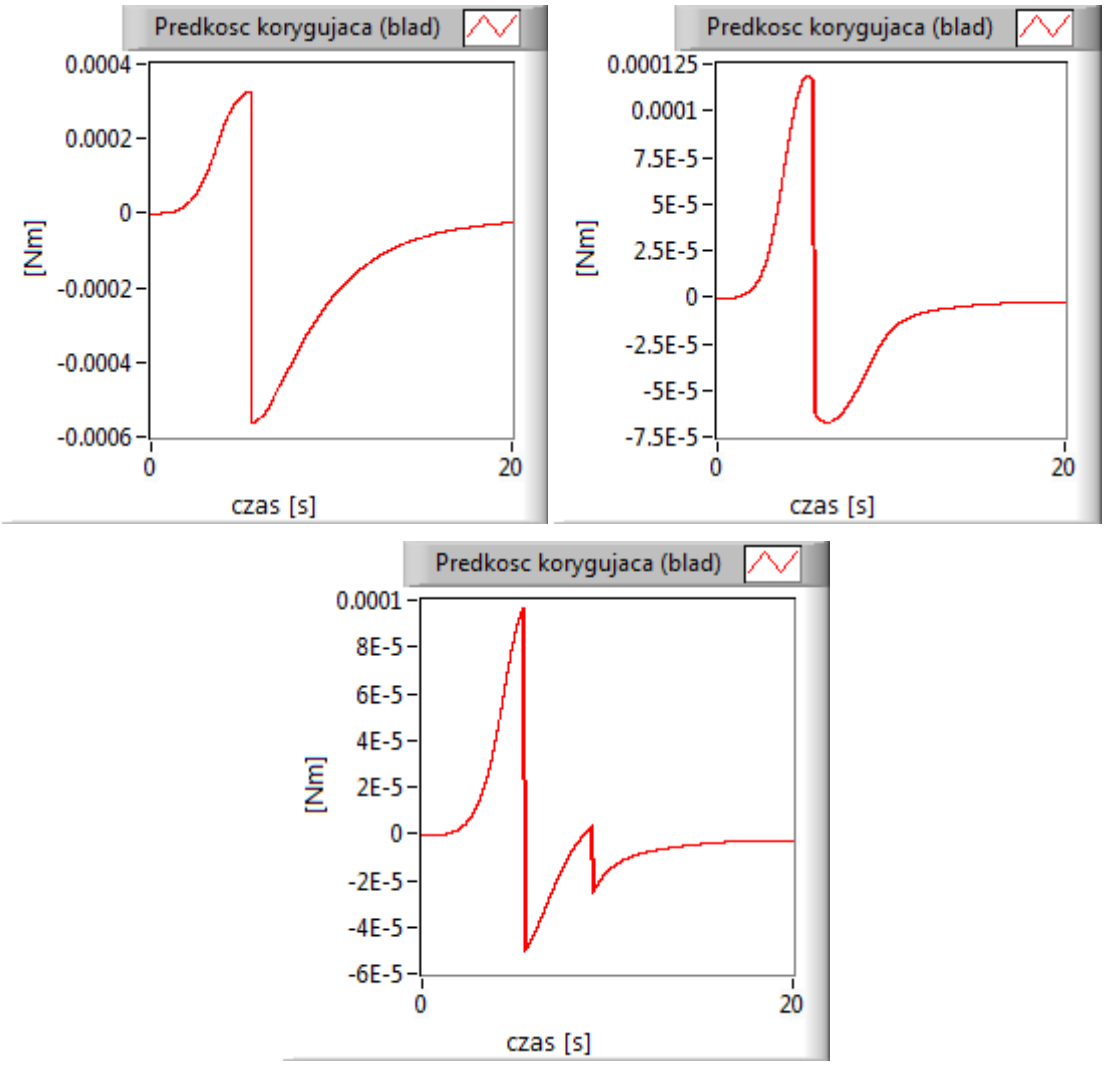
przebiegu momentu kierującego (brak wspomnianej powyżej nieliniowości w otrzymanych charakterystykach). Fakt ten jest ściśle związany z charakterem momentu kierującego, który przez większą część trasy (od 8 s) jest stały.

Identyczna analiza została dokonana dla momentu napędowego, której wyniki całkowicie potwierdzają przeprowadzoną analizę dla momentu kierującego. Na rys. 4.20. wykreślono ponownie dla trajektorii typu „sinus" wartość chwilową prędkości korygującej. Bliższa analiza otrzymanych wyników potwierdza poprzedni wniosek, w którym uzgodniono, że stosowanie mniejszego kroku całkowania skutkuje większą dokładnością przeprowadzonych obliczeń, co w efekcie determinuje powstanie mniejszego błędu. Dla wszystkich analizowanych kroków całkowania charakter przebiegu wartości prędkości korygującej jest zbieżny z przebiegiem momentu napędowego dla tego typu trajektorii. W chwili czasu t, w czasie którego koło kierujące dokonuje obrotu, następuje gwałtowana zmiana momentu napędowego. Objawia się to wyraźną zmianą wartości prędkości korygującej, która zmienia swoją wartość na przeciwną (zgodnie z wartością chwilową analizowanego momentu). Chwila ta wyznacza również początek kolejnego okresu narastania wartości błędu (prędkości korygującej), który rośnie do czasu następnej zmiany kąta obrotu kierownicy. Wyjątek stanowi wykres dla kroku Δt=0,0005 s, gdzie od 40 s do 45 s błąd nieznacznie maleje.



Rys. 4.20. Chwilowe wartości prędkości korygującej (moment napędowy; trajektoria typu „sinus") dla kroków całkowania: górny lewy Δt=0,01 s, górny prawy Δt=0,005 s, dolny lewy Δt=0,001 s, dolny prawy: Δt=0,0005 s

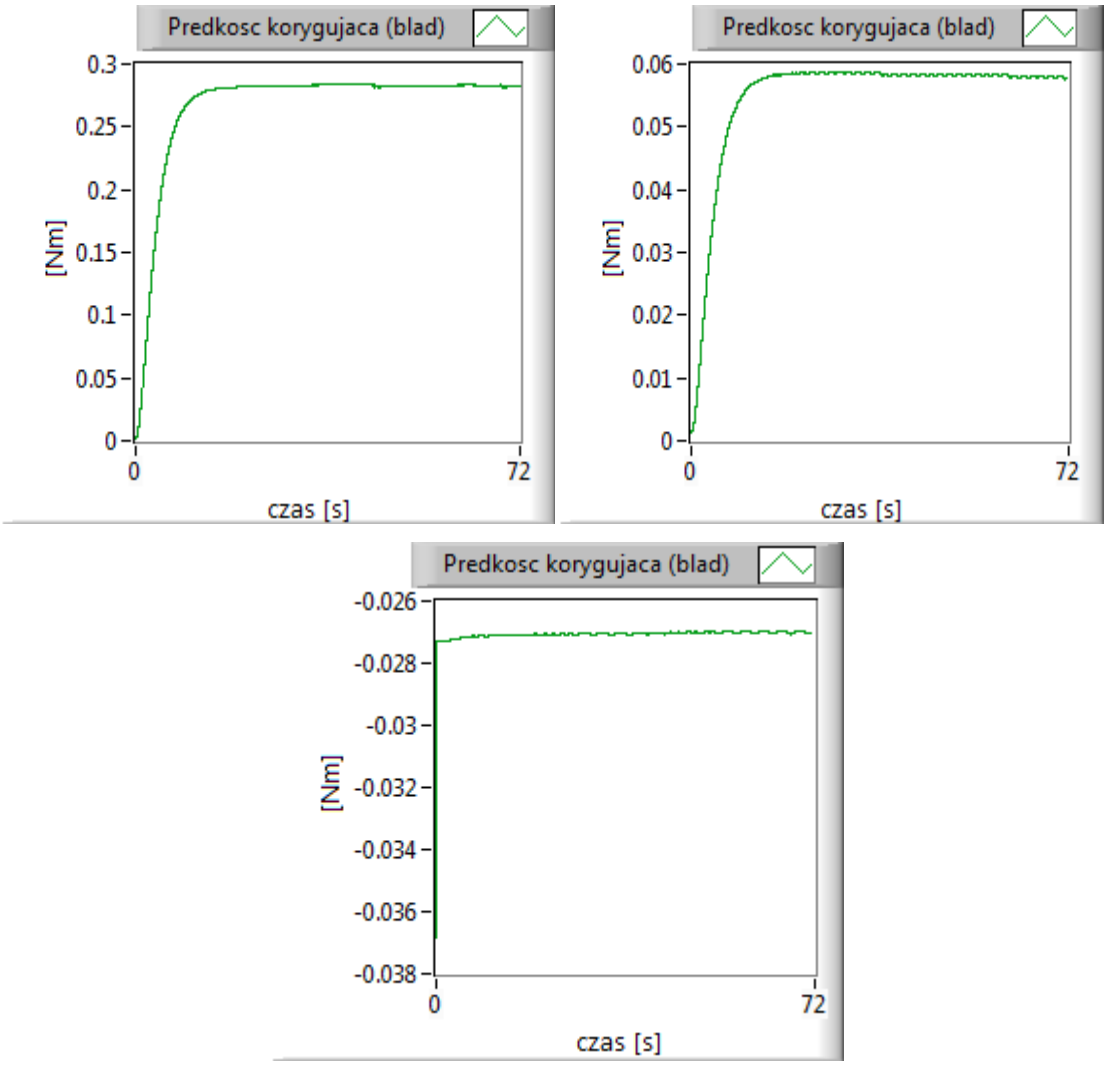
Rezultaty dla przypadku, gdy platforma mobilna porusza się po trajektorii typu „parabola", ściśle nawiązują do przypadku wcześniejszego tzn., gdy platforma poruszała się po torze typu „sinus". Zaprezentowane na rys. 4.21 wartości chwilowych prędkości korygujących potwierdzają opisaną powyżej tendencję do zmniejszania się wartości błędu, wraz ze zwiększeniem się częstotliwości generowania optymalnego sygnału sterującego (mniejszy krok całkowania). Należy zauważyć, że powstałe średnie błędy dla wszystkich kroków całkowania (dla rozpatrywanej trajektorii ruchu platformy) stanowią znikomą wartość w stosunku do wartości danego momentu (rzędu tysięcznych części wartości maksymalnej momentu napędowego). Również w tym przypadku (jak to miało miejsce dla momentu kierującego) otrzymane rezultaty dla kroku całkowania Δt=0,01 s całkowicie eliminowały je z dalszej analizy (brak fizycznej interpretacji).



Rys. 4.21. Chwilowe wartości prędkości korygującej (moment napędowy; trajektoria typu „parabola") dla kroków całkowania: górny lewy Δt=0,005 s, górny prawy Δt=0,001 s, dolny Δt=0,0005 s

Dla przypadku, w którym platforma porusza się po trajektorii typu „okrąg", otrzymana wartość chwilowa prędkości korygującej dla rozpatrywanych kroków całkowania została uwidoczniona na rys. 4.22. Z tego samego względu, co dla momentu kierującego, nie został pokazany przebieg dla kroku całkowania Δt=0,01 s. Występujące błędy dla tego przypadku nie mają swojej reprezentacji fizycznej, dlatego zostały wykluczone z dalszej analizy. Ze względu na charakter przebiegu wartości chwilowej momentu napędowego, nie dostrzega się w przebiegach chwilowych wartości prędkości korygujących większych fluktuacji, czy też jak to miało miejsce dla trajektorii typu „sinus" gwałtownych zmian wartości sygnału. Podczas poruszania się platformy po trajektorii typu „okrąg", raz osiągnięta wartość otrzymywana jest do końca trasy. Z otrzymanych przebiegów wartości prędkości

korygujących rysuje się jasna zależność wartości powstającego błędu przy stosowaniu większego kroku całkowania. Należy jednak jeszcze raz podkreślić, że mniejszy błąd obarczony jest zwiększonym czasem przeprowadzania symulacji. Z uwagi na fakt, iż wartość prędkości korygującej jest dodawana w równaniu optymalnego sygnału sterującego, będzie mogła docelowo neutralizować powstały błąd w tym lokalne oscylacje, które mogłyby doprowadzić do utraty stabilności poruszającej się platformy.



Rys. 4.22. Chwilowe wartości prędkości korygującej(moment napędowy; trajektoria typu „okrąg") dla kroków całkowania: górny lewy Δt=0,005 s, górny prawy Δt=0,001 s, dolny Δt=0,0005 s

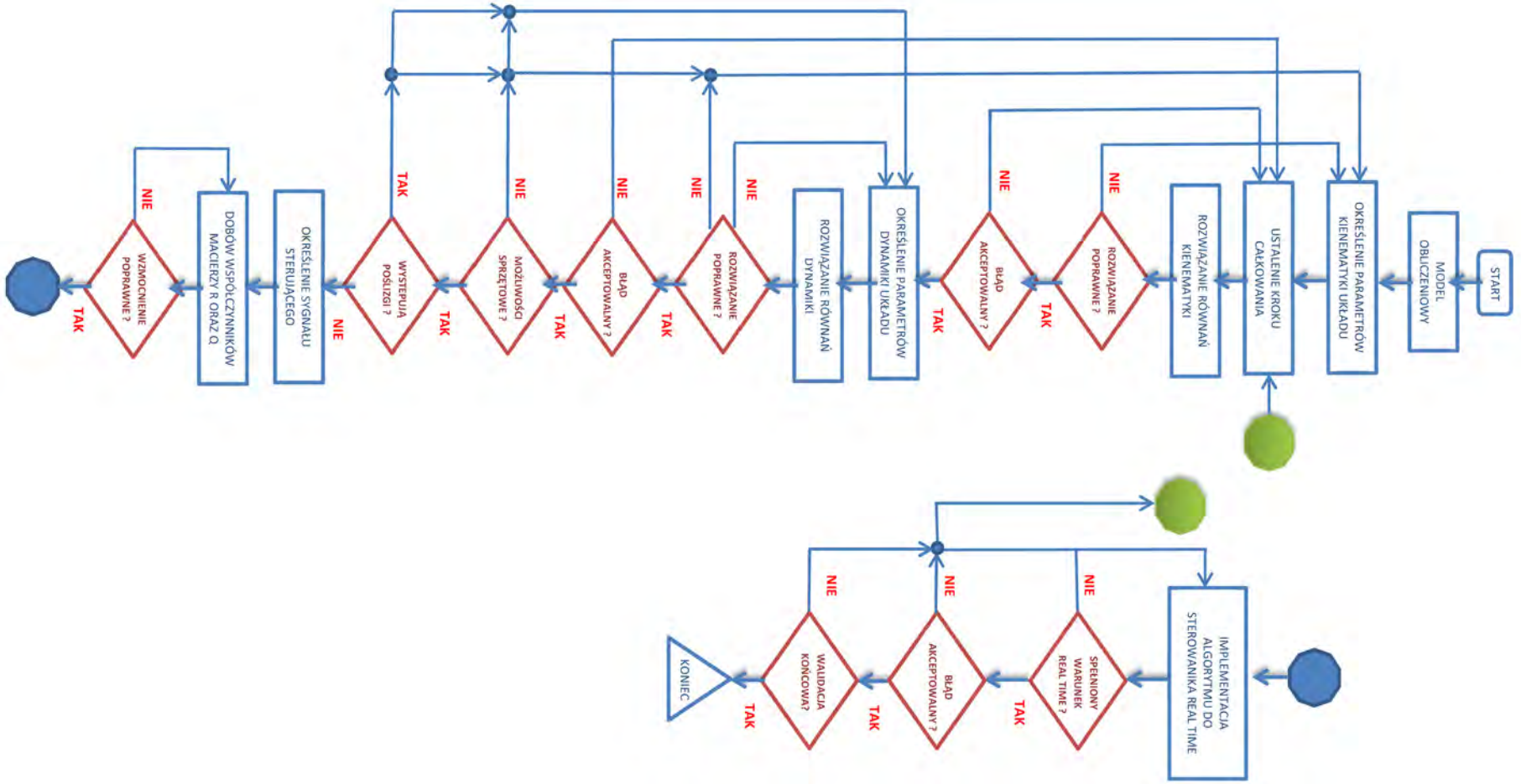
Analiza otrzymanych wyników z wykorzystaniem techniki wirtualnego prototypowania, a także przyjęte wstępnie założenia konstrukcyjne systemu nadzorowania, sugerowały zastosowanie kroku całkowania wynoszącego 0,001 s.

Niestety, w trakcie implementacji algorytmu do sterownika cRIO (weryfikacja warunku determinizmu czasowego – drugi etap w iteracyjnym, wirtualnym prototypowaniu sterownika) oraz późniejszych prób testów HILS, autor zauważył, że wspomniany powyżej warunek czasu rzeczywistego nie jest spełniony dla Δt=0,001 s, tzn. wskazania zegarów czasu symulacji oraz czasu rzeczywistego nie są zbieżne. Zastosowany w sterowniku procesor Real Time o częstotliwości pracy 400 MHz nie był w stanie generować (tj. obliczać w czasie RT) całki sygnału z tak narzuconą szybkością. Obciążenie procesora, jak i pamięci operacyjnej w całym cyklu przeprowadzania testu (dla Δt=0,001) wynosiło 100%. Dodatkowo, sterownik w trakcie tego okresu nie reagował na inne zewnętrzne przerwania (np. stop symulacji). Drogą kolejnej analizy, tj. wydajności sprzętowej oraz optymalizacji rozwiązania, autor dokonał pozytywnej próby zastosowania kroku 5 razy dłuższego. Różnica w występujących błędach dla kroków 0,001 s a 0,005 s jest niedostrzegalna i pomijalna.

Ostatecznie, na podstawie symulacji analizy obciążenia procesora, wykorzystania pamięci operacyjnej oraz spełnienia warunku determinizmu czasowego zadecydowano, że optymalnym krokiem całkowania będzie wartość 0,005 s (tj. sygnał sterujący będzie generowany z częstotliwością 200 Hz). Fakt ten utwierdził w przekonaniu, że osiągnięto sytuację optymalizacji energetycznej (optymalnych warunków układu) projektowanego rozwiązania poprzez zapewnienie kompromisu pomiędzy długością kroku całkowania, obciążalnością procesora (bilansem energetycznym), a wymaganiami realizacji systemu nadzorowania w czasie rzeczywistym.

Na rys. 4.23 zilustrowano w sposób schematyczny zastosowaną metodologię optymalizacji systemu nadzorowania platformy mobilnej. Demonstrowany schemat optymalizacji zawiera wszystkie etapy wykorzystane przez autora podczas mechatronicznego projektowania systemu nadzorowania. Należy zauważyć, że część występujących sprzężeń zwrotnych (np. po rozwiązaniu zadania dynamiki) może zostać skierowana do dwóch różnych bloków wykreślonego schematycznie algorytmu postępowania. W przypadku otrzymania rozwiązania niepoprawnego (np. moment sterujący lub napędowy nie ma swojej interpretacji fizycznej) następuje powrót do etapu, w którym zostają określone ponownie parametry dynamiki układu np. masa poszczególnych części. W tym samym czasie istnieje również możliwość powrotu do etapu, w którym następuje ponowna korekta parametrów kinematyki czy geometrii układu (np. promienia koła platformy). Niezwykle istotnym zagadnieniem w

przeprowadzanej optymalizacji (projektowaniu) jest określenie możliwości sprzętowych platformy, np. możliwości zakupu motoreduktora o wybranych parametrach. W tym przypadku niepomyślna weryfikacja (brak na rynku takich rozwiązań lub zakup wyspecyfikowanej jednostki przekracza budżet projektu) powoduje ponowny powrót do etapów (opcjonalnie, w zależności od potrzeb), w których następuje ponowny dobór parametrów dynamiki lub kinematyki układu. Ze względu na dość szybką ocenę możliwości oraz przydatności jednostki sterującej (sterownik cRIO) do realizowanego projektu badań, proces weryfikacji wydajności sprzętowych systemu sterowania (dobór właściwego CPU lub wielkość pamięci operacyjnej) został wyłączony z omawianej metodologii optymalizacji.



Rys.4.23. Zastosowana metoda optymalizacji projektowanej platformy mobilnej

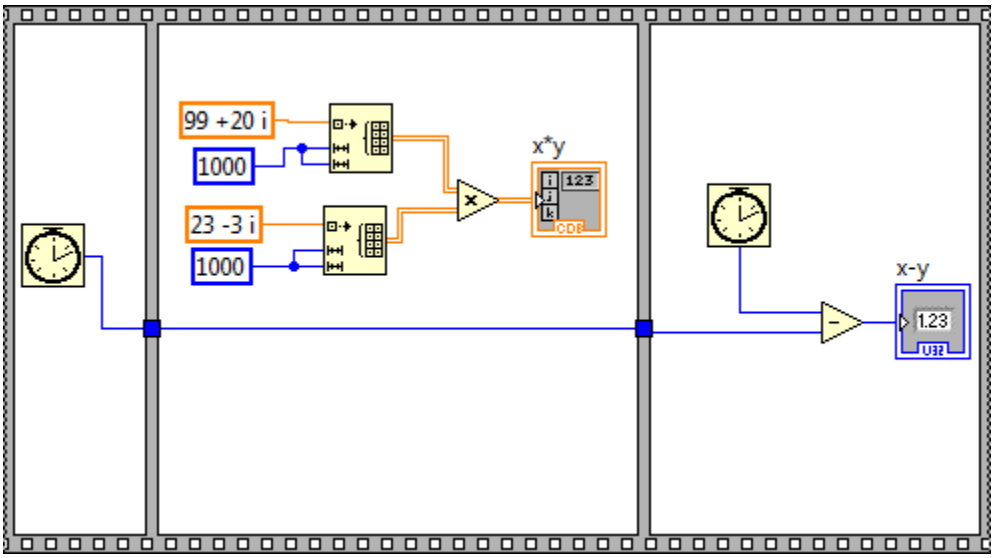
5. System nadzorowania ruchu platformy mobilnej w czasie rzeczywistym

Obecna w projektowaniu rozwiązań mechatronicznych tendencja do korzystania z najnowszych oraz kompleksowych środowisk projektowych (dających możliwość prototypowania, projektowania, symulacji oraz integracji z dedykowanym sprzętem (tutaj LabVIEW), a także weryfikacji w czasie rzeczywistym (RT) [75, 76] projektowanego obiektu (tutaj technika HILS) [56], została wykorzystana do przeprowadzania badań nad odpowiedzią emulowanej na zewnętrznym systemie LabVIEW Real Time trójkołowej platformy mobilnej na pobudzenie sygnałem generowanym przez rzeczywisty sterownik czasu rzeczywistego cRIO firmy National Instruments [66]. Wykorzystana w pracy technika HILS [78] została zastosowana w wielu przedsięwzięciach inżynierskich. Należy tu wymienić m.in. przemysł samochodowy [6, 49], lotniczy [26, 50], morski militarny [85], elektryczno-elektroniczny [3, 25, 51], systemów automatyki [55] i mechatroniki [52, 57, 58], a także – przemysł maszynowy [86]. Technika HILS jest drugą techniką, po wirtualnym prototypowaniu, zastosowaną w prezentowanym przedsięwzięciu naukowym (patrz rys. 3.1 oraz rys. 2.6) dotyczącym projektowania systemu nadzorowania ruchu trójkołową platformą mobilną.

5.1. Koncepcja systemu sterowania Real Time

Realizacja sytemu nadzorowania trójkołowej platformy mobilnej, bazującą na algorytmie *on-line*, pociągnęła za sobą zastosowanie systemu Real Time, tj. w pełni deterministycznego systemu, w którym odpowiedź (generowanie sygnału), a także czas wykonania zadania (operacji) muszą być ściśle określone. Systemy sterowania bazujące na środowisku Microsoft Windows mogą być bardzo wydajne, jeśli wziąć pod uwagę szybkość wykonywania obliczeń, a także – ich dokładność. Jednak takie aplikacje, pomimo swoich niepodważalnych zalet, nie gwarantują nieraz krytycznego w systemach sterowania, czasu rzeczywistego. Głównym powodem jest fakt bazowania takich aplikacji na niedeterministycznym systemie operacyjnym, jakim jest Windows. Dla potwierdzenia niedeterministyczności systemu, przeprowadzono test (rys. 5.1), w którym proste obliczenie wyniku mnożenia dwóch liczb zespolonych

„zajmuje” za każdym razem procesorowi (pracującemu pod nadzorem systemu Windows) różny okres czasu (od 21 do 36 ms).



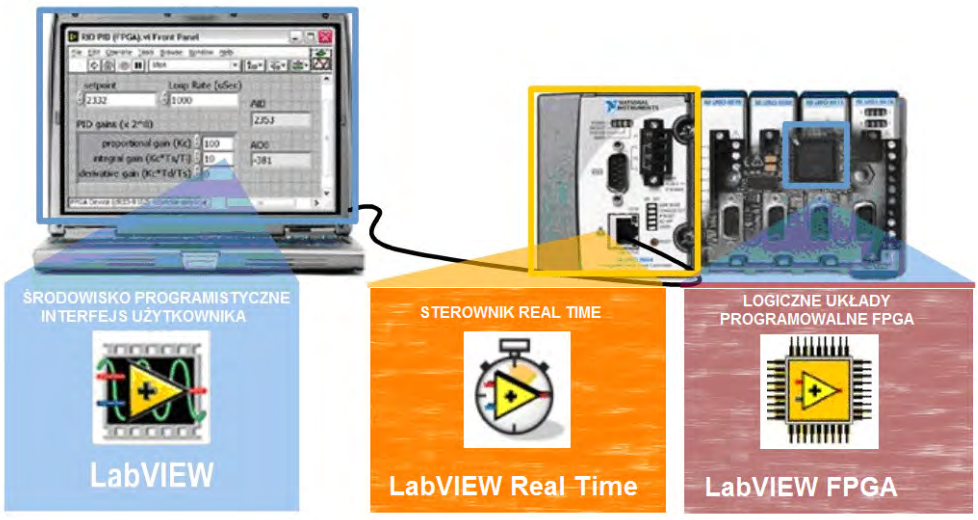
Rys. 5.1. Schemat testu niedeterministyczności systemu operacyjnego Windows, przeprowadzonego w środowisku LabVIEW

Test przeprowadzono w środowisku LabVIEW bez modułu Real Time. Powodem takiej sytuacji jest fakt, iż system Windows nie jest systemem dedykowanym do jednego zadania. Obsługuje on inne programy pracujące w tle oraz reaguje na różne przerwania z takich urządzeń, jak klawiatura, czy też sprzęt dołączany do portów USB. Zastosowanie modułu LabVIEW Real Time pozwala na uruchamianie aplikacji w oddzielnym systemie operacyjnym (pętli Real Time), w którym zadania są ściśle zdeterminowane i nadane mogą im być niezbędne priorytety wykonywalności. Dla powyższego przykładu z mnożeniem dwóch liczb zespolonych, czas przeprowadzenia obliczeń w takim systemie (RT) będzie zawsze taki sam (niezmienny).

W procesie projektowania systemu sterowania trójkołowej platformy mobilnej niezwykle istotnym i ważnym elementem okazał się determinizm czasowy, tzn. generowanie optymalnego sygnału sterującego musi odbywać się w ściśle określonym czasie. Dodatkowo, czas przeprowadzenia symulacji ruchu platformy musi być równy czasowi rzeczywistemu, jaki upłynął (dalej czas Real Time).

W ogólności, niedeterministyczne systemy sterowania bazujące na systemie operacyjnym Windows muszą koegzystować z innymi procesami obsługiwanymi

przez system, a także – dzielić wspólne, często ograniczone zasoby. Taka konfiguracja, uniemożliwia zapewnienie ścisłej czasoprzestrzeni wykonania określonego zadania przez proces. System Real Time firmy NI (szczegóły zostaną omówione w kolejnych podrozdziałach) poprzez swoją konfigurację (rys. 5.2), w skład której wchodzą kontroler (procesor) Real Time, programowalne układy cyfrowe FPGA, moduły wejścia/wyjścia, a także dedykowane środowisko programistyczne, zapewnia dedykowanemu procesowi (jednej aplikacji) ściśle określony czas wykonania zadania (pętli), a także – możliwość nadania podzadaniom aplikacji ściśle określonych priorytetów wykonalności.



Rys. 5.2. Koncepcja sterownika cRIO

5.2. Metodologia implementacji systemu sterowania w czasie rzeczywistym

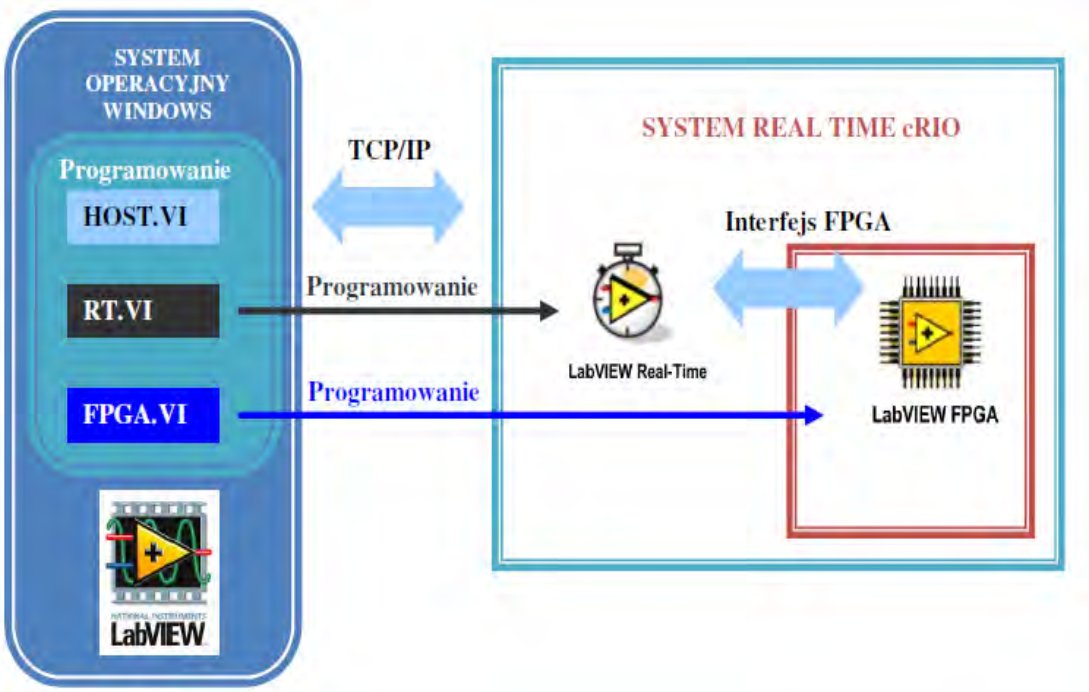
Mając na uwadze specyfikę systemu Windows, a także wymagania stawiane realizowanemu algorytmowi w trybie *on-line*, autor postanowił, że całość projektu mechatronicznego obejmująca zastosowane techniki projektowania, zostanie zrealizowana przy wykorzystaniu jednej wspólnej platformy obliczeniowej, dostarczonej przez firmę National Instruments.

W skład wykorzystanej platformy wchodzą:

- LabVIEW Professional Development System 2010;
- moduł Control Design & Simulations 2010;
- moduł Real Time 2010;
- moduł FPGA 2010;
- kontroler Real Time cRIO-9074;
- sterowniki (dwa) silników – DC NI-9505;
- końcówki mocy (dwie) NI-9931.

Nie bez znaczenia z punktu widzenia wyboru platformy sterowania (oprócz spełnienia koniecznego warunku deterministyczności w procesie sterowania) okazał się fakt, że zastosowane moduły sterownika silnika DC NI – 9505 umożliwiają zmianę stanów wyjść („odświeżanie") z częstotliwością do 20 MHz. Moduł może wykonywać powierzone zadanie szybciej niż ustalona (drogą optymalizacji, podczas realizacji techniki wirtualnego prototypowania oraz weryfikacji warunku determinizmu czasowego) częstotliwość generowania optymalnych sygnałów sterujących. Z tego powodu, proces kontroli układu napędowego oraz kierowniczego przebiega zgodnie z przyjętą koncepcją.

Strukturę programowania, jaka została przyjęta do realizacji systemu sterowania platformą, przedstawiono na rys. 5.3.



Rys. 5.3. Struktura programowania w środowisku LabVIEW sterownika cRIO

Aplikacja LabVIEW (wraz z wyżej wymienionymi modułami) została zainstalowana na wydajnym komputerze klasy PC (Dell, Intel Core i5, M560 @2,76 GHz, 8GB RAM). Dostępne środowisko, poprzez liczne interfejsy użytkownika (HMI), kompilatory kodu (do cRIO oraz FPGA) oraz managerów sprzętu (analizującym m.in. obciążenie procesora oraz wykorzystanie pamięci RAM) umożliwiło zastosowanie metody projektowania równoległego, a także – stworzyło możliwość pełnej integracji realizacji projektu (realizowanego w tym szczególnym przypadku, co w ogólności nie musi być regułą) z jednego miejsca roboczego.

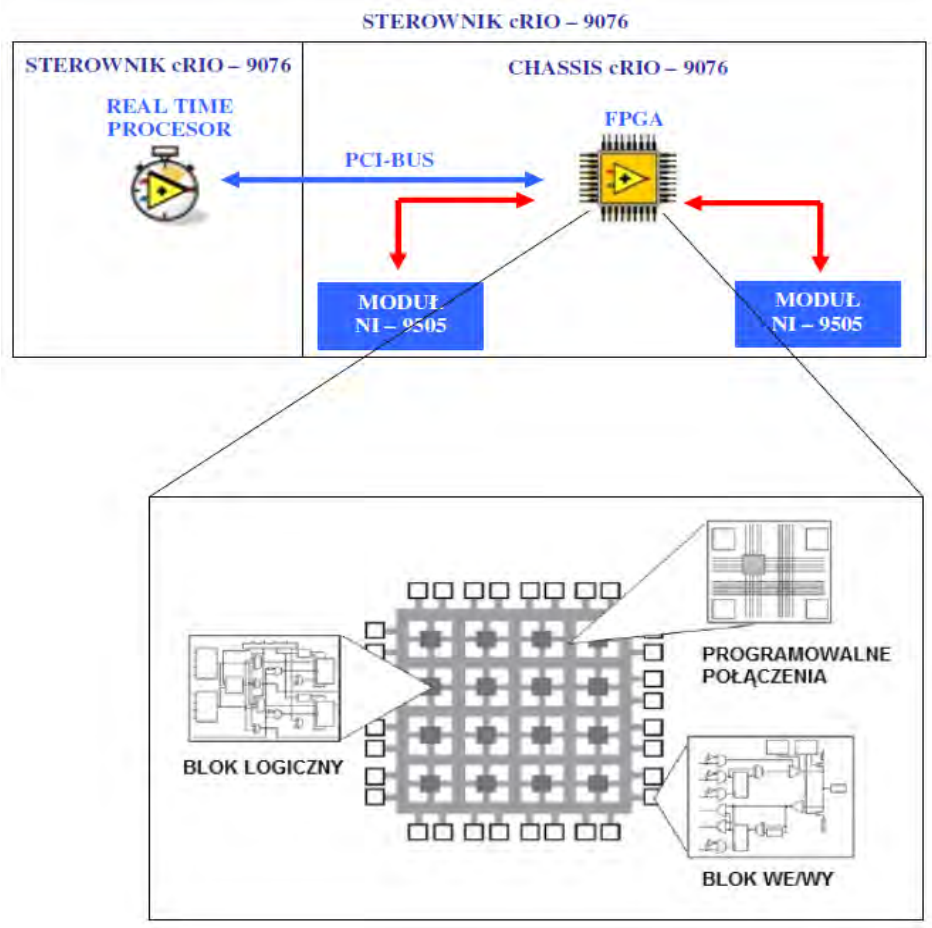
Wygenerowane w środowisku Maple rozwiązania parametryczne (tj. bez określania wartości poszczególnych wielkości fizycznych) równań różniczkowych kinematyki oraz dynamiki układu zostały zaimplementowane dzięki modułowi CD&S w systemie LabVIEW. Zastosowanie takiego zabiegu (parametryzacja) pozwoliło autorowi, w trakcie projektowania platformy oraz przeprowadzonym w późniejszym okresie licznych symulacji, na natychmiastową korektę parametrów czy też niezbędnych ustawień. Ułatwiło to również, właściwy dobór parametrów platformy (rozmiar kół, rozstaw osi, maksymalna masa itp.), a także – wykorzystywanych w procesie symulacji komputerowych wartości elementów macierzy **R** oraz **Q** w zależności opisującej optymalny sygnał sterujący.

Dla wybranych trajektorii ruchu (typu „sinus”) została przeprowadzona analiza mnożników Lagrange’a (patrz rozdział 3) niezbędnych do określenia wartości całkowitych sił tarcia ślizgowego, które to wartości dla zapewnienia jazdy bez poślizgu kół, nie powinny przekraczać wartości tarcia granicznego.

Na rys. 5.4 została przedstawiona architektura użytego do budowy systemu sterowania sterownika (szerszy opis zamieszczono w podrozdziałach 5.2.1 oraz 5.2.2). Jest to najnowsza jednostka firmy NI (premiera w USA w kwietniu 2011). Zastosowany sterownik jednomodułowy jest zintegrowany mechanicznie, tzn. część RT CPU została połączona w jedną obudowę z *Chassis*, w której znajduje się FPGA (w ogólności, obie te jednostki dobiera się osobno i łączy mechanicznie, rozważając wydajność CPU RT oraz FPGA, a także liczbę slotów na moduły wejść/wyjść w kontekście zastosowania do konkretnej aplikacji). Niezależne układy cyfrowe CPU RT oraz FPGA programuje się z poziomu oddzielnych modułów LabVIEW; pierwszy z poziomu LabVIEW RT, natomiast drugi z poziomu LabVIEW FPGA.

W trakcie prac implementacyjnych algorytmu do systemu LabVIEW, autor postanowił podzielić program realizujący system sterowania platformą na dwie części. Część główna – generowanie sygnału sterującego (realizacja algorytmu) została osadzona bezpośrednio na kontrolerze Real Time cRIO. Kontroler na podstawie założonej trajektorii, analizy danych z enkoderów, oblicza w czasie rzeczywistym, wykorzystując algorytm sterowania optymalnego przy energetycznym wskaźniku jakości, optymalne sygnały sterujące. Obliczone wartości sygnałów sterujących, drogą interfejsów magistrali sprzętowej przesyłane są do układów logicznych FPGA [71, 74].

Autor wykorzystał układ FPGA do realizacji drugiej części sterowania platformą. Wartości optymalnych sygnałów sterujących generowanych w sterowniku cRIO zamieniane są drogą sprzętową (w FPGA) na modulowaną falę (w takt zmian sygnału sterującego) PWM o częstotliwości 20 kHz. Dodatkowo, w FPGA autor zrealizował pomiar wartości natężenia prądu, prędkości i położenia wału silników oraz statusu poszczególnych sterowników.



Rys.5.4. Budowa sterownika cRIO NI – 9076

5.2.1. Generowanie optymalnego sygnału sterującego w systemie Real Time sterownika cRIO

Kod analizowanego algorytmu, którego postać matematyczna została wyprowadzona w rozdziale 4, został zaimplementowany do systemu LabVIEW. Wykorzystano przy tym standardowe bloki dostępne z bibliotek wewnętrznych języka graficznego G środowiska. Wykorzystano również bloki dostępne w module CD&S, realizujące całkowanie, różniczkowanie, a także – pamięć buforową realizującą jedno-iteracyjne opóźnienie (macierze **M** oraz **L** wyznaczane są w chwili poprzedniej $t_{i\text{-}1}$, zamiast w chwili obecnej t_i).

W celu lepszego dostępu do kodu programu, a także – w celu wyeliminowania błędnych połączeń, system sterowania platformą został podzielony na bloki (podsystemy). Blok algorytmu, tj. finalnego bloku generującego optymalne sygnały sterujące, został dołączony do innych bloków realizujących rozwiązania równań różniczkowych kinematyki oraz dynamiki projektowanego układu. Zrealizowany w środowisku LabVIEW system sterowania (tj. zaimplementowany algorytm) został przedstawiony na rys. 5.5 (ze względu na rozbudowany program pokazano ogólną architekturę programu, obrazującą zastosowaną koncepcję, na bazie której powstało szczegółowe rozwiązanie).

W celu uzyskania determinizmu czasowego oraz nadania najwyższego priorytetu realizowanemu procesowi, kod programu stworzony w CD&S został umieszczony w pętli Real Time – *Timed Loop*. Obie pętle (CD&S oraz RT) zostały zsynchronizowane wspólnym zegarem systemowym o częstotliwości 1 kHz [65, 77]. Przyjęcie kroku całkowania ustala szybkość wykonywania pętli CD&S. W analizowanym przypadku poszczególna iteracja (wyliczanie odpowiedniej wartości sygnału sterującego) odbywa się z częstotliwością 200 Hz. Pętla RT dostosowuje swoją prędkość (dzięki uprzednio nałożonej synchronizacji czasowej obu pętli) do pętli CD&S.

W celu weryfikacji czasu obliczania optymalnego sygnału sterującego, na bieżąco dokonywany jest pomiar czasu dwóch wewnętrznych zegarów – zegara symulacji oraz zegara rzeczywistego (rys. 5.5). Zakłada się, że w przypadku realizacji systemu Real Time występuje znak równości w wartościach opisywanych zegarów (autor dzięki przeprowadzonemu badaniu oraz mechatronicznej optymalizacji systemu spełnił ten warunek).

Realizowane trajektorie ruchu platformy ustala się poprzez umieszczenie odpowiedniego podsystemu „trajektoria” w pętli CD&S.

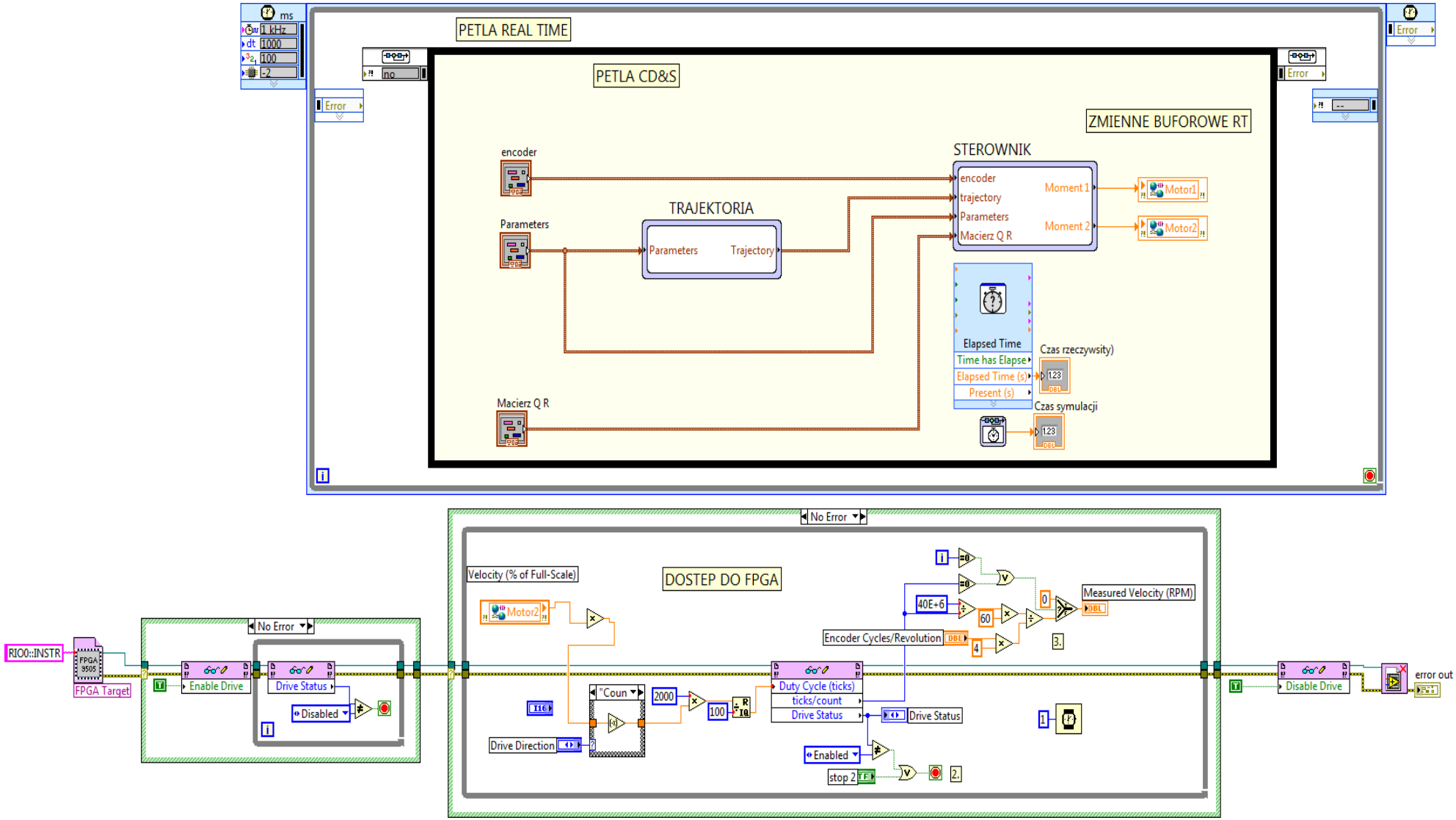
Dostęp do układu FPGA z poziomu systemu sterowania zrealizowanego na cRIO odbywa się poprzez otwarcie tzw. referencji FPGA – utworzonego wcześniej pliku FPGA – w tym przypadku zrealizowanego programu sterownika silnika DC. Procedurę dostępu do układu FPGA w celu optymalizacji kodu umieszczono w osobnej pętli (rys. 5.5), podrzędnej w stosunku do omawianej powyżej pętli systemu sterowania. Ze względu na większą czytelność programu autor na rysunku zamieścił jedynie pętle dla jednego momentu. Dla drugiego (niepokazanego) pętla została zrealizowana identycznie. Sygnały generowane z pętli nadrzędnej (podsystem „sterownik”) przesyłane są do pętli otwierającej FPGA poprzez zmienne buforowe czasu rzeczywistego (Moment 1, Moment 2) [72, 73, 79]. Wielkość bufora w zrealizowanej aplikacji została dobrana drogą eksperymentalną. Otwarcie interfejsu FPGA pozwala kontrolerowi cRIO w czasie rzeczywistym przesłać sygnały do programowalnych układów logicznych FPGA, w celu ich sprzętowej zamiany na odpowiednio zmodulowaną falę PWM. Dodatkowo, otwarty interfejs umożliwia odczyt również w czasie rzeczywistym, prędkości i pozycji wału napędowego silnika, a także – podgląd danych statusowych modułów sterowników NI – 9505.

Warto przypomnieć, że utworzony system sterowania został wykorzystany do przeprowadzenia techniki wirtualnego prototypowania (rozdział 4). W tym przypadku pożądane przebiegi parametrów obiektu badań zostały zobrazowane na wykresach powstałych na skutek połączenia wyjść danego bloku z blokiem środowiskowego grafu. Należy zaznaczy, że docelowo, mając na uwadze optymalne wykorzystanie pamięci operacyjnej kontrolera oraz zwiększenie operacyjności procesora, bloki grafów z finalnego kodu systemu sterowania zostały usunięte.

Dużą zaletą wykorzystanego środowiska LabVIEW jest fakt, iż tworzony program (plik z rozszerzeniem *vi*) jest na bieżąco kompilowany. Pozwala to na bieżące śledzenie wadliwych połączeń lub użytych funkcji. Skompilowany program może być uruchomiony na maszynie z zainstalowanym LabVIEW, bądź też zapisany do jednostki cRIO.

Utworzony kod programu (algorytm sterowania optymalnego, którego zadaniem jest wygenerowanie sygnałów sterujących (momentów) dla silników DC) został umieszczony w dedykowanej pętli modułu CD&S. Realizacja systemu nadzorowania czasu rzeczywistego narzuciła wymagania związane z wyborem metody

oraz kroku całkowania. Zaistniały tu konflikt pomiędzy zastosowaną mocą obliczeniową procesora (szybkością obliczeń) oraz wydajnością energetyczną układu, a dokładnością przeprowadzanych metod numerycznych, rozwiązywany został drogą kompromisu, czyli optymalizacji energetycznej. W tym przypadku zastosowano odpowiedni kroku całkowania 0,005 s, co umożliwiło spełnienie determinizmu czasowego, oraz wprowadzono prędkości korygujące. Omówienie tej analizy zostało zamieszczone w rozdziale 4.



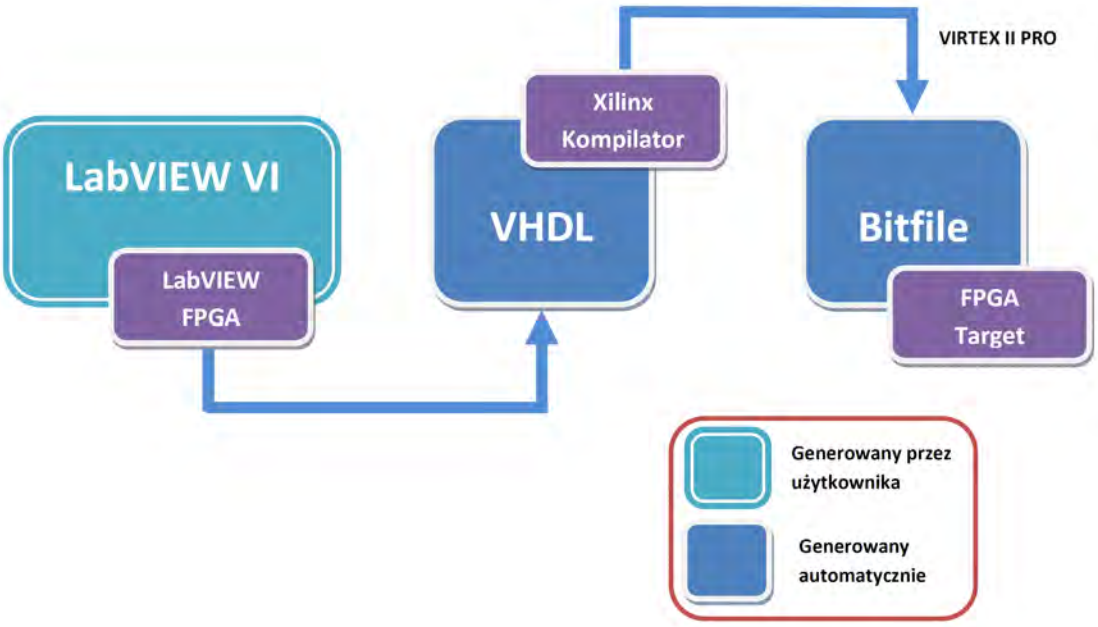
Rys. 5.5. Algorytm sterowania zaimplementowany do sterownika NI – 9076 (realizacja w programie LabVIEW)

5.2.2. Układ FPGA sterownika cRIO. Optymalizacja systemu sterowania

Wykorzystany do budowy systemu nadzorowania platformą mobilną kontroler cRIO NI – 9076 został wyposażony w reprogramowalny układ logiczny FPGA. Zastosowano układ scalony z serii Spartan – 6 LX45 firmy Xilinx. Układ rezyduje w obudowie sterownika, łącząc jego układy wejścia/wyjścia (w tym przypadku moduły NI – 9505). Komunikacja pomiędzy procesorem RT a układem FPGA odbywa się poprzez szybką magistralę komunikacyjną *PCI – BUS*. Koncepcja układu została zaprezentowana na rys. 5.4.

Programowanie układu FPGA, składającego się z trzech głównych komponentów: bloków logicznych, programowalnych połączeń oraz bloków wejść/wyjść, odbywa się poprzez odpowiednio zaprogramowane układy logiczne (w ogólności, połączenie np. bramek AND) oraz zdefiniowanie połączeń pomiędzy tak określonymi blokami, a ich odpowiednimi blokami układów wejść/wyjść.

Do programowania rozpatrywanego układu logicznego autor wykorzystał moduł LabVIEW FPGA, oferujący dedykowany dla jednostek programowalnych zestaw możliwych do wykorzystania funkcji logiczno – matematycznych (języka G). Kompilacji kodu jest dość złożona. Tworzony program kompilowany jest przez LabVIEW, który generuje kod VHDL i transferuje go następnie do kompilatora układu Xilinx. Tutaj odbywa się kolejna kompilacja (zamiana) kodu VHDL do plików bitowych. Ostatecznie pliki bitowe wysyłane są do FPGA w celu odpowiedniego ustawienia układów cyfrowych. Ułatwieniem tego procesu jest jego pełna automatyzacja oraz możliwość wglądu w raporty generowane podczas trwania programowania (kompilacji). Schematycznie proces kompilacji został przedstawiony na rys. 5.6.



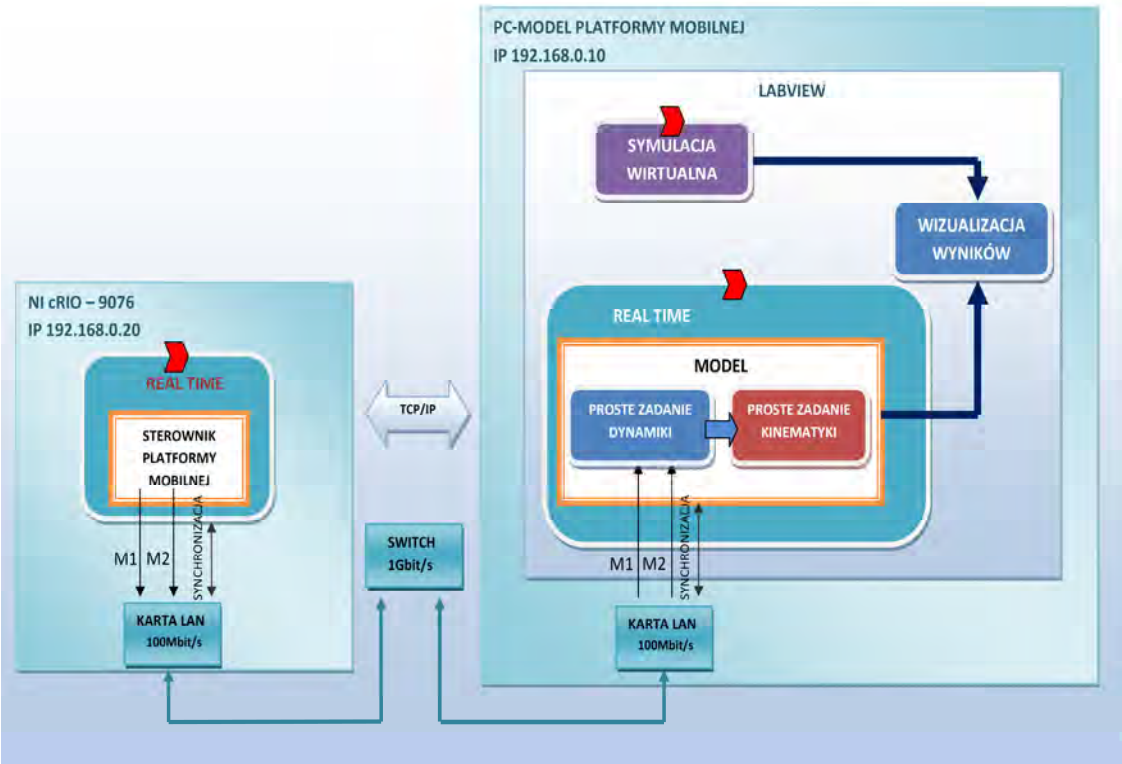
Rys.5.6. Proces kompilacji kodu do FPGA

Autor wykorzystał układ FPGA do osiągnięcia najwyższej wydajności systemu sterowania platformy mobilnej. Połączenie dwóch układów procesora RT oraz układu FPGA pozwoliło na odseparowanie procesów, których realizacja wymaga obliczeń zmiennoprzecinkowych (dla generowania rozwiązań równań różniczkowych) od tych, dla których najważniejszy jest czas obliczeń i precyzja czasowa (w tym przypadku generowanie fali PWM o częstotliwości 20 kHz – rekomendowanej dla zastosowanych modułów NI – 9505). O ile pętla RT systemu cRIO synchronizowana jest do zegara systemowego, to w przypadku układu FPGA czas wykonania pętli ściśle koresponduje z zastosowanym zegarem układu FPGA (w tym przypadku 40 MHz). W niniejszej pracy częstotliwość zegara systemowego wynosi 1 kHz. Istnieje też możliwość synchronizacji pętli do zegara 1 MHz. Jednak dla bardzo złożonych obliczeń realizowanych w pracy, synchronizacja do tego zegara przebiegła niepomyślnie. Procesor nie jest w stanie w czasie RT generować sygnałów z tak zadaną częstotliwością.

Program generujący modulowaną falę PWM w takt zmian sygnału sterującego (momenty silników) został podzielony na pętle FPGA realizujące poszczególne podzadania sterowania napędem platformy, tj. generowanie fali, obsługa enkodera, obsługa zdarzeń (statusu) sterownika (czy aktywny, uszkodzony, przeciążony, za wysoka temperatura modułu itp.).

5.3. Symulacja systemu sterowania optymalnego przy energetycznym wskaźniku jakości w czasie rzeczywistym

Koncepcja zrealizowanego testu została schematycznie zademonstrowana na rys. 5.8. Przeprowadzenie testów czasu rzeczywistego HILS wymagało zapewnienia równoczesności rozpoczęcia generowania sygnału w pętli sterownika oraz generowania odpowiedzi w pętli modelu, czyli rozwiązania w kolejności zadania prostego dynamiki oraz kinematyki. Ważne jest, aby czas RT był wspólny i jednoczesny dla obu realizowanych pętli badań. Ze względu na brak możliwości rozsyłania sygnału zegara, autor zdecydował, aby sygnał synchronizacji był generowany przez sterownik i przesyłany jako zmienna globalna czasu rzeczywistego do pętli obiektu. W celu porównania odpowiedzi uzyskanych w trakcie symulacji wirtualnych z metodą HILS, dodatkowo na komputerze PC generowane są (bez wpływu na przeprowadzone testy HILS) pożądane trajektorie ruchu platformy i ich zestawiane na wspólnym wykresie obrazującym występujące różnice (realizowany jest tu identyczny proces, jak podczas techniki wirtualnego prototypowania opisanej w rozdziale 4).

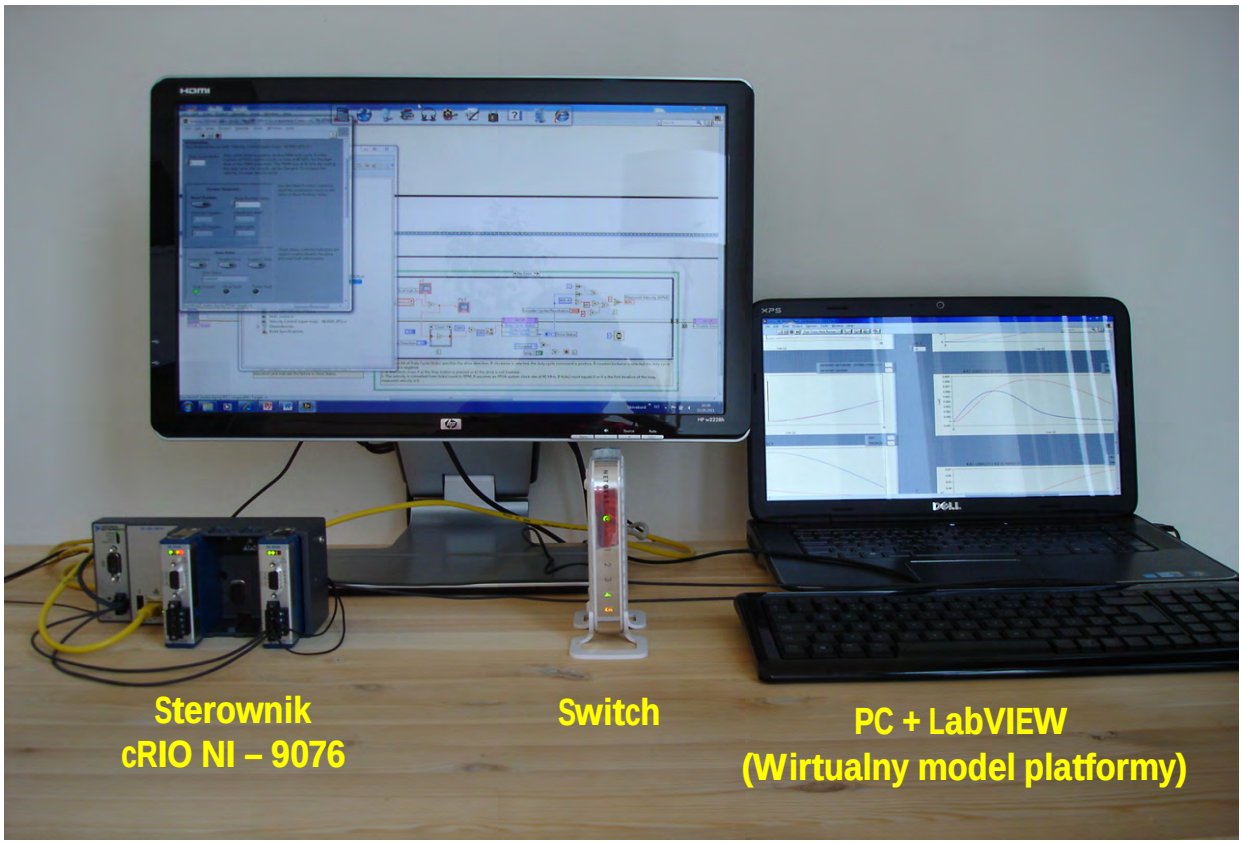


Rys.5.8. Zastosowana koncepcja konfiguracji testu HILS

Przeprowadzono testy dla trzech rozpatrywanych poprzednio typów trajektorii ruchu platformy mobilnej, tj. „sinus”, „parabola” oraz „okrąg”.

Każdy przeprowadzony test wymagał indywidualnej konfiguracji. Główne prace skupiły się wokół konfiguracji jednostki cRIO, do której należało zapisać algorytm oraz dokonać niezbędnych nastaw wzmocnień i współczynników macierzy **R** i **Q**. Zmienne globalne czasu rzeczywistego o charakterze kolejek FIFO [77, 79, 80], proces synchronizacji pętli sterownika oraz obiektu, a także – model matematyczny platformy, pozostały wspólne dla wszystkich trajektorii.

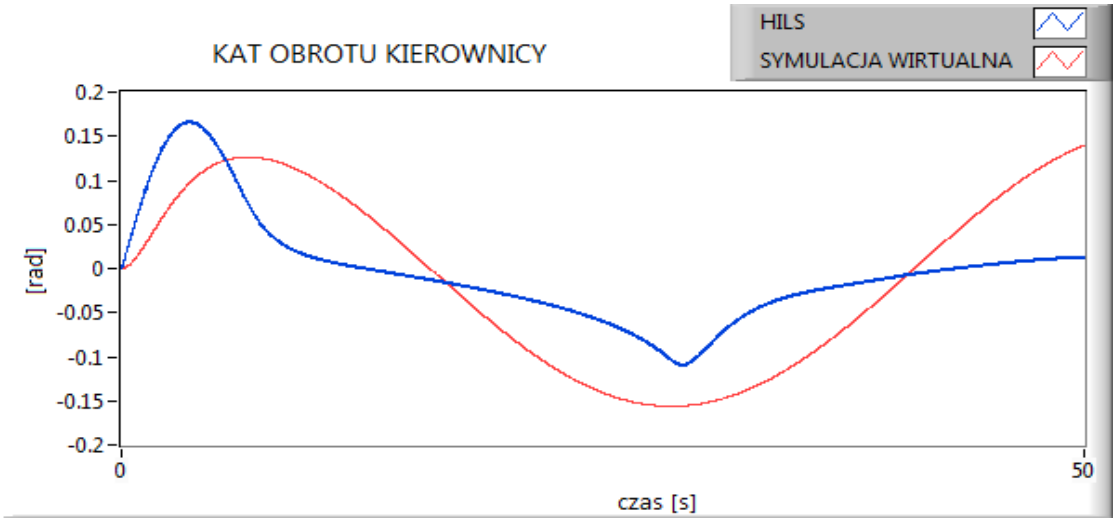
Autor, o czym wspominał powyżej, na komputerze emulującym badany obiekt przeprowadził w czasie rzeczywistym dodatkowe referencyjne symulacje wirtualne. Rezultaty obu testów (HILS oraz odniesienia) dla rozpatrywanych parametrów (rozwiązań zadania prostego dynamiki i kinematyki) zostały zestawione na wspólnych wykresach. Stanowisko do przeprowadzenia testów techniką HILS ilustruje rys. 5.9.



Rys.5.9. Widok stanowiska do przeprowadzania testów techniką HILS

5.3.1. Test HILS dla trajektorii typu „sinus”

Zgodnie z zamieszczonym rys. 5.8, model obliczeniowy emulowanej platformy mobilnej bazuje na równania dynamiki Lagrange’a II rodzaju z mnożnikami. Przeprowadzona analiza prostego zadania dynamiki oraz prostego zadania kinematyki, pozwoliła ma wyznaczenie parametrów ruchu platformy, będącej pod wpływem momentów (sygnałów sterujących) pochodzących ze sterownika cRIO. W sterowniku, jak i modelu platformy mobilnej, krok całkowania (w teście HILS dla wszystkich trajektorii) został ustawiony na 0,005 s. Parametry trajektorii, które zostały ustalone w trakcie realizacji symulacji komputerowych, pozostały w niezmienionej formie. Wyniki testów HILS, czyli parametrów otrzymanych drogą analizy obu zadań (kąt obrotu kierownicy φ, kąt obrotu platformy β oraz prędkości kątowe kół α_1, α_2, α_3), dla trajektorii typu „sinus”, zostały zilustrowane na poniższych rysunkach.



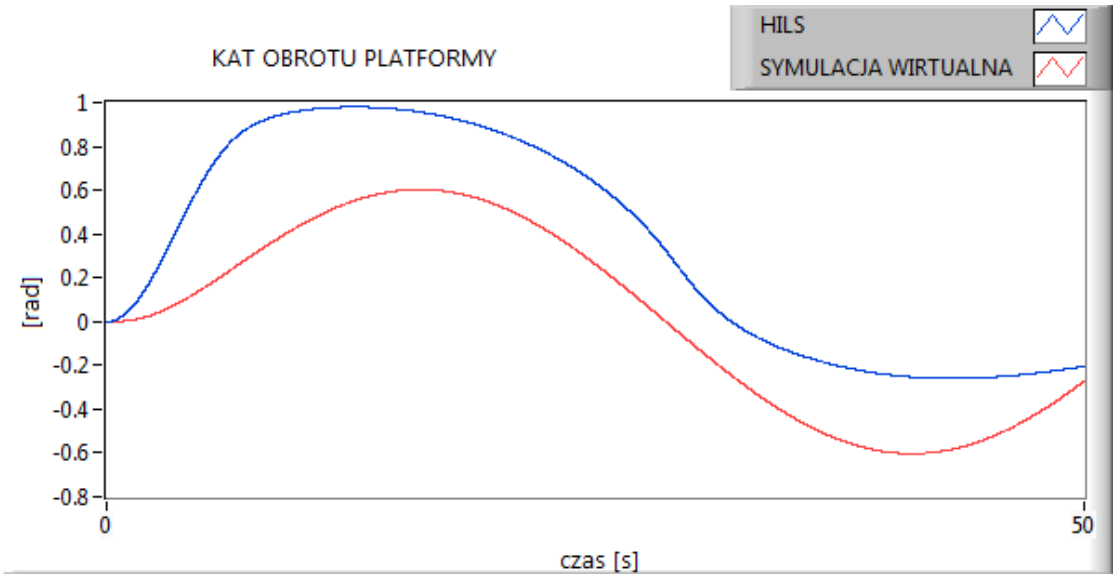
Rys. 5.10. Kąt obrotu kierownicy dla trajektorii typu „sinus”

Należy jeszcze raz podkreślić (o czym była mowa w rozdziale 3), że krzywa „sinus” jest bardzo selektywna i wymagająca (spośród wszystkich zastosowanych w niniejszej pracy) zarówno dla systemu nadzorowania, który musi w czasie rzeczywistym analizować sygnały wejściowe (z enkoderów) oraz generować optymalne sygnały sterujące, jak i dla platformy. Chociaż platforma porusza się po łukach, realizowane są dwa zakręty, które z uwagi na silną nieliniowość macierzy bezwładności (zależność od kąta obrotu kierownicy) mogą być przyczyną powstania ewentualnych odchyłek od planowego toru przejazdu oraz lokalnych przyspieszeń

prowadzących do wystąpienia poślizgów i co za tym idzie – utraty stabilności. System nadzorowania w tym przypadku wykonuje pracę nad wyliczaniem sygnału sprowadzającego sterowany obiekt na właściwą drogę.

Rys. 5.10 przedstawia odpowiedź emulowanego modelu (w tym przypadku, chwilowa wartość kąta obrotu kierownicy φ). Zakłada się, że idealna sytuacja została przedstawiona podczas realizacji symulacji wirtualnych, w której wartości poszczególnych amplitud w obu półokresach krzywej są niemal identyczne. W tym przypadku emulowany obiekt podążał po krzywej o „charakterze" krzywej „sinus", jednak obserwuje się znaczące przesterowania w chwilowych wartościach tego kąta.

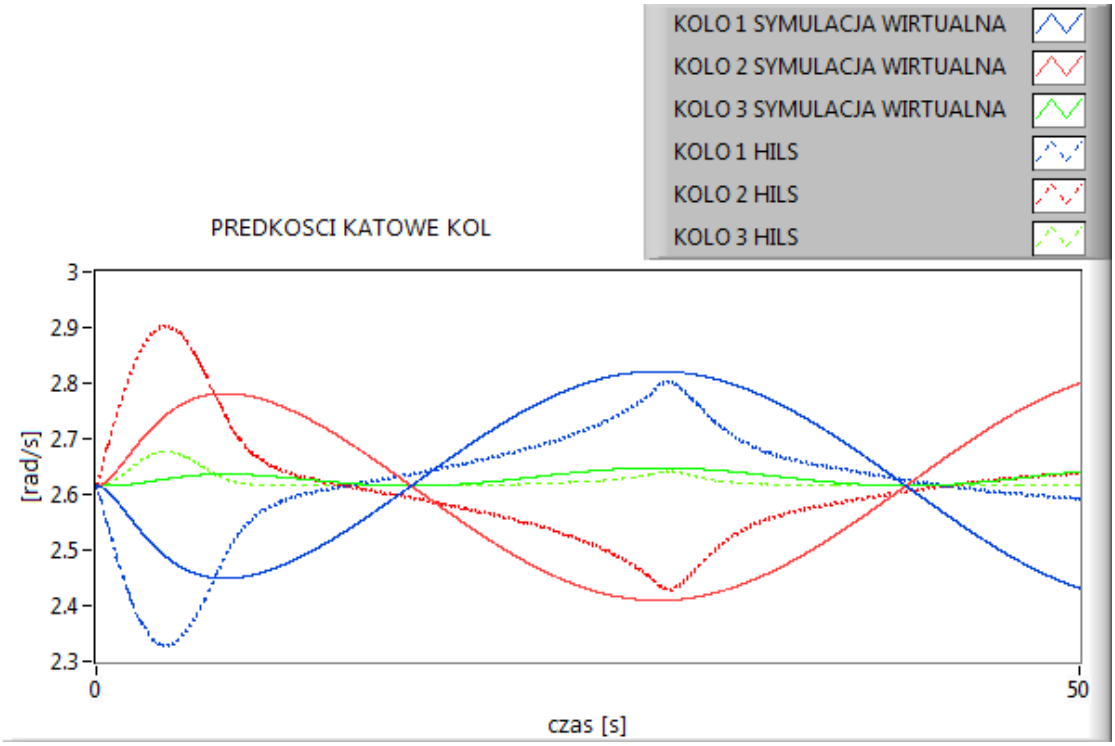
Otrzymany w teście HILS przebieg (krzywa dla kąta φ) ściśle koresponduje z krzywą przedstawioną na rys. 5.11, na której wykreślono kąt obrotu platformy β. Zbyt duże wysterowanie kąta obrotu kierownicy emulowanego modelu powoduje przeregulowanie wartości kąta obrotu ramy. Zbyt mała wartość nastawy kąta φ w drugim półokresie powoduje, iż platforma porusza się po łuku o mniejszym promieniu.



Rys. 5.11. Kąt obrotu platformy dla trajektorii typu „sinus"

Prędkości poszczególnych kół platformy (rys. 5.12), ze względu na swoją bezpośrednią matematyczną zależność od kąta φ, wykazują również tendencje do podsterowności w pierwszym półokresie krzywej „sinus". W drugim półokresie ze względu na mniejsze wartość kąta, również i w tym przypadku wartość maksymalna prędkości jest poniżej wartości oczekiwanej (domniema się, że wartością oczekiwaną jest wartość otrzymana drogą techniki wirtualnej). Warto zwrócić uwagę, że szybkie zmiany prędkości w 28 s mogą być przyczyną powstania poślizgów. Mniejsze

wartości prędkości, szczególnie w drugim półokresie są pochodną mniejszych zmian kąta wychylenia ramy β.



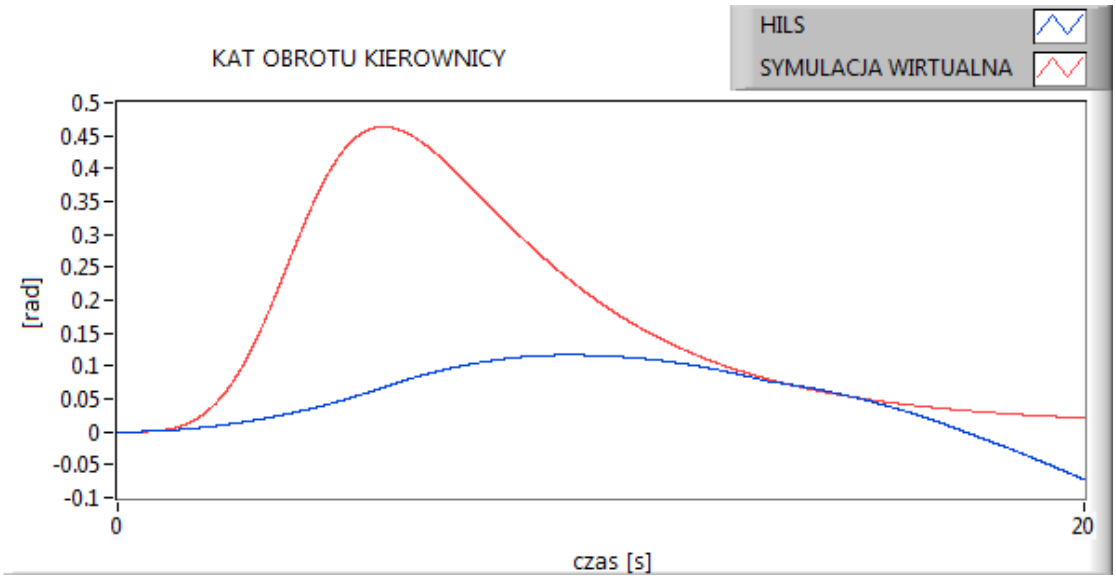
Rys. 5.12. Prędkości kątowe kół platformy dla trajektorii typu „sinus”

5.3.2. Test HILS dla trajektorii typu „parabola”

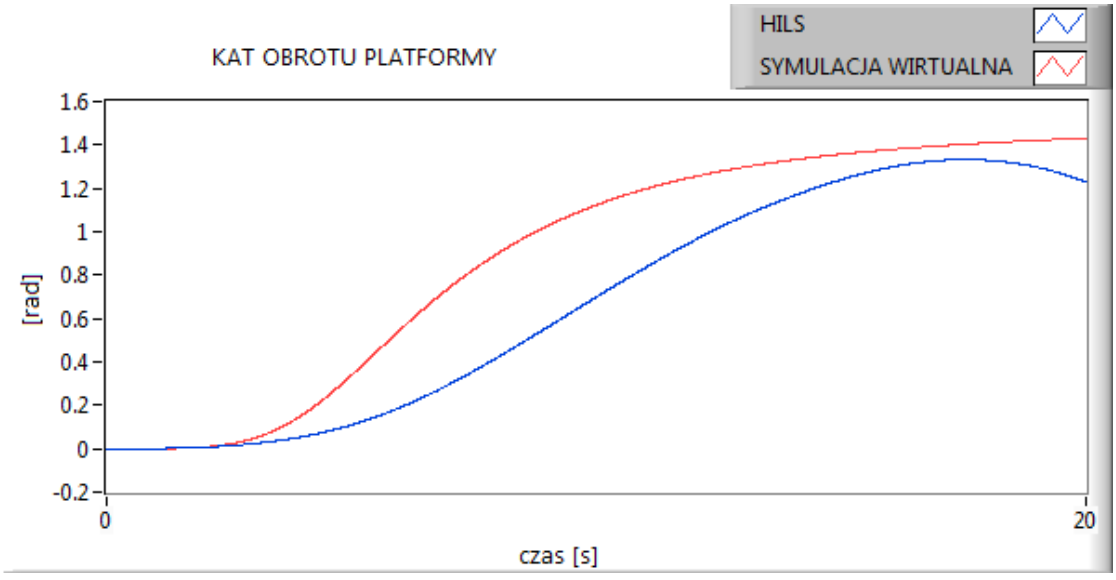
Przeprowadzenie testów HILS dla drugiej drogi przejazdu wymagało zmiany podsystemu „trajektoria” z algorytmu głównego sterownika oraz przeprowadzenie ponownego procesu kompilacji (wraz z implementacją kodu do sterownika cRIO). Zmianie uległy również parametry i ustawienia symulacji wirtualnej, będącej w tym przypadku (o czym autor wspominał) symulacją referencyjną do realizowanej. W celu sprawdzenia zachowania się tego samego obiektu badań (modelu matematycznego) podczas realizacji różnych trajektorii testowych nie dokonywano zmian w ustawieniach w emulowanym modelu.

Ze względu na fakt, iż charakter krzywej parabolicznej jest mniej wymagający dla systemu sterowania (jeden łuk) oraz dla samej platformy mobilnej, uzyskano zależności bardzo silnie powiązane z przebiegami z symulacji wirtualnych. O ile realizacja drugiego zakrętu (w przypadku trajektorii typu „sinus”) powodowała, że

chwilowe wartości kąta obrotu kierownicy φ zmieniały charakter z „ za duży” na „ za mały”, to w przypadku przejazdu po krzywej parabolicznej sytuacja taka nie występuje. Kąt φ emulowanej platformy (rys. 5.13) jest znacznie mniejszy od wartości wynikających z przeprowadzonych symulacji wirtualnych, jednakże zachowuje przy tym poprawny kształt i charakter zmian.



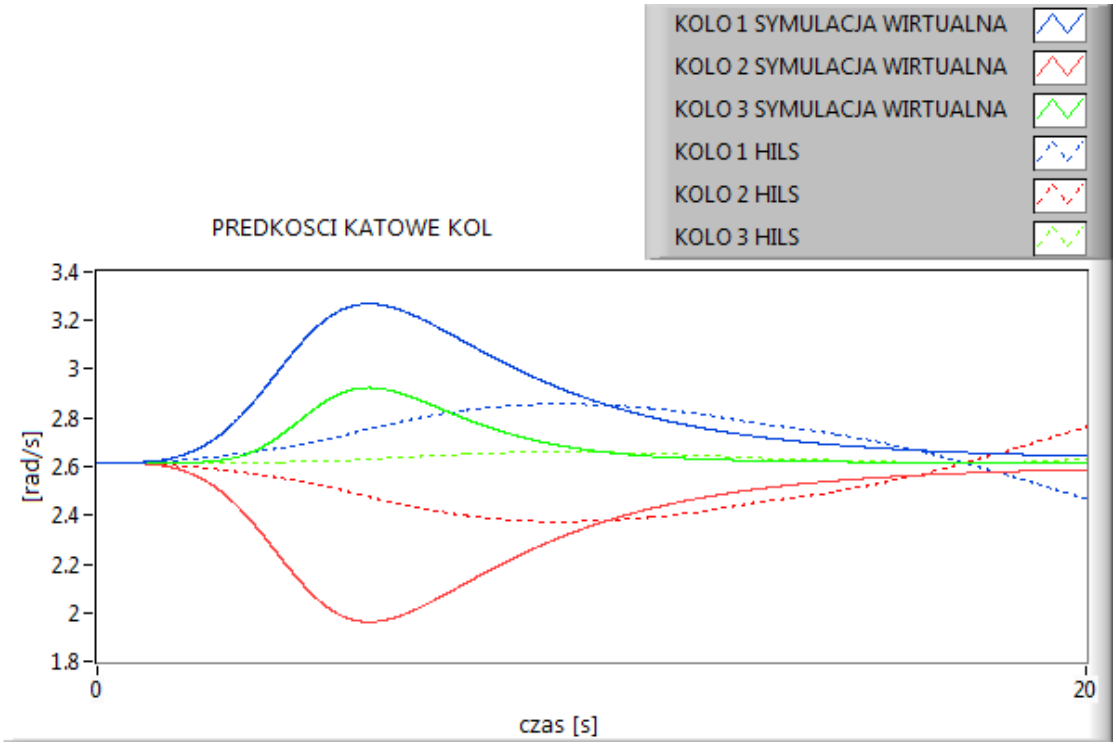
Rys.5.13. Kąt obrotu kierownicy dla trajektorii typu „parabola”



Rys. 5.14. Kąt obrotu platformy dla trajektorii typu „parabola”

Mniejszy w stosunku do symulacji wirtualnej kąt obrotu kierownicy powoduje, iż kąt obrotu platformy bezpośrednio zależny od tej wartości, jest również mniejszy (rys. 5.14). Na uwagę zasługuje wyraźny obrót platformy w drugiej części symulacji. Podobna uwaga dotyczy (co jest wynikiem nieoczekiwanego skrętu) symulacji

prędkości kątowych kół (rys. 5.15), które do około 14 s zachowują podobną tendencję. W późniejszym okresie czasu do końca trwania symulacji emulowana platforma po wyjściu z własnej dla siebie trajektorii ruchu dokonuje nieoczekiwanego skrętu w przeciwną stroną. Jest to wyraźny błąd, jednak o czym autor będzie pisał w podrozdziale 5.3.4, całkowicie wytłumaczalny.



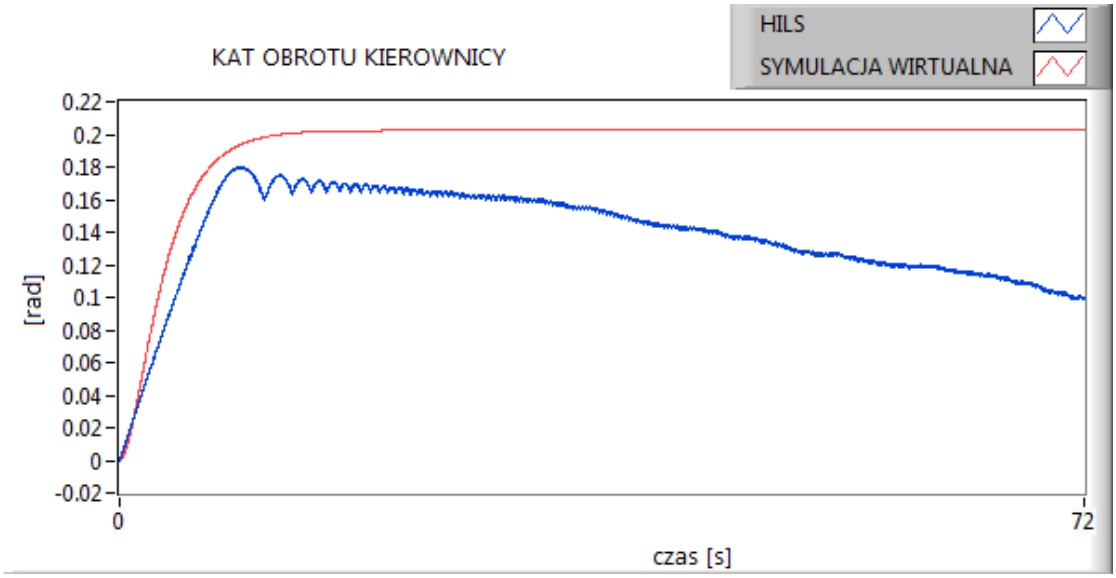
Rys. 5.15. Prędkości kątowe kół platformy dla trajektorii typu „parabola”

5.3.3. Test HILS dla trajektorii typu „okrąg”

Podobnie, jak to miało miejsce w przypadku trajektorii typu „parabola”, dokonano zamiany podsystemu „trajektoria” z algorytmu głównego sterownika oraz przeprowadzono ponowny proces kompilacji oraz implementacji kodu do sterownika cRIO.

Charakterystyka otrzymanych zależności dla kąta obrotu koła kierownicy φ (rys. 5.16) jest zbieżna z zależnościami otrzymanymi techniką symulacji wirtualnych. Różnica w bardzo nierównej wartości kąta obrotu kierownicy, dostrzegalna jest w drugiej części przejazdu emulowanej platformy. Niezwykle istotnym czynnikiem wpływającym na występujące oscylacje (oprócz występujących błędów numerycznych) jest charakter zastosowanej trajektorii. Platforma w tym przypadku

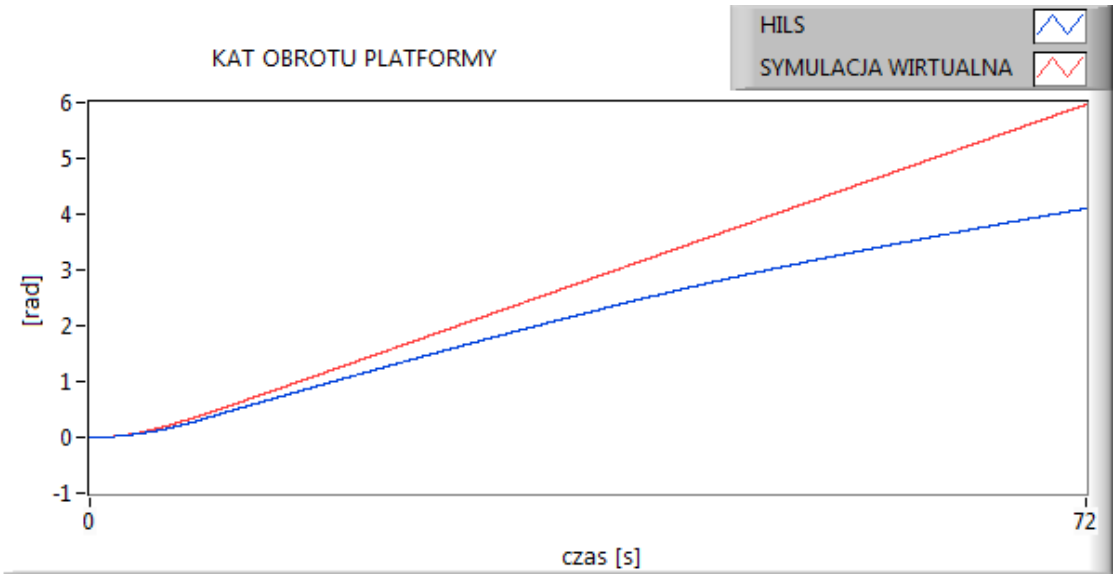
porusza się po trajektorii, której wartości na osi x oraz y układu kartezjańskiego opisane są równaniami nieliniowymi zależnymi od kąta θ.



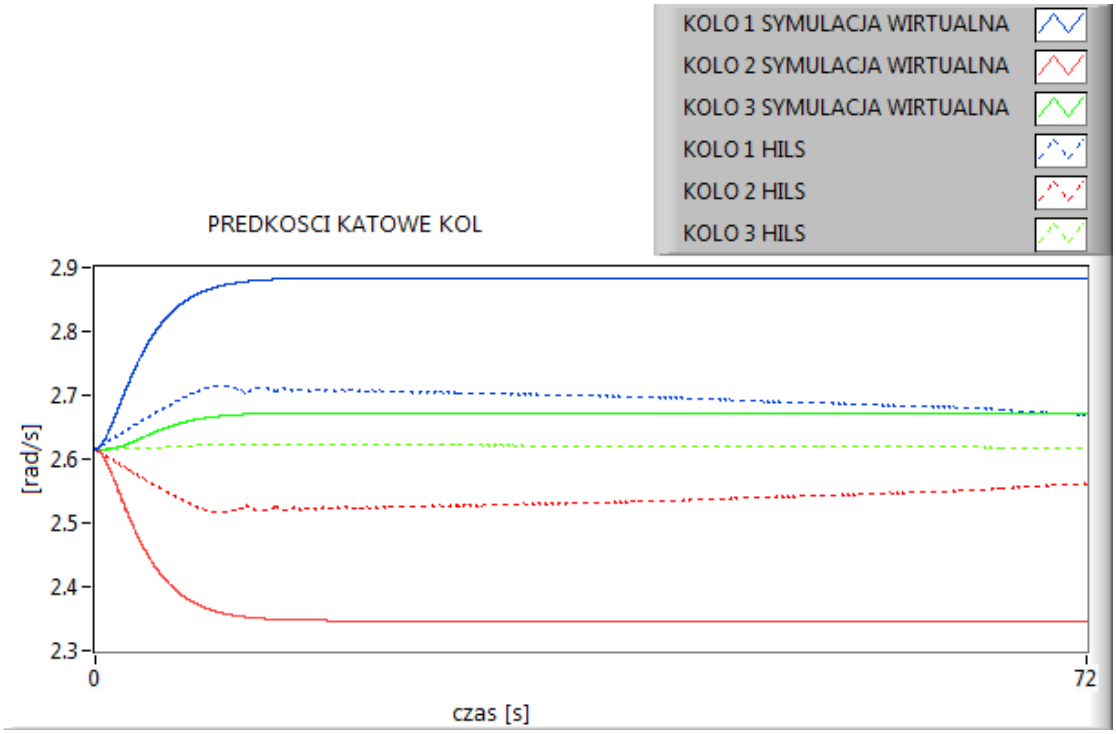
Rys. 5.16. Kąt obrotu koła kierownicy dla trajektorii typu „okrąg”

Mniejszy kąt obrotu kierownicy platformy emulowanej ma bezpośredni wpływ na mniejszy kąt obrotu platformy (rys. 5.17) oraz, co za tym idzie, mniejsze wartość prędkości poszczególnych kół napędowych (rys. 5.18), które również z powodu występujących oscylacji kierownicy platformy, zachowują w drugiej części przejazdu podobny charakter. Warto zauważyć, że pomimo występujących oscylacji kierownicy, nie oddziałują one na wartości kąta obrotu platformy.

Przyprowadzone badania HILS dla trajektorii „okrąg” wybitnie dowiodły (uzyskano bardzo dobrą zbieżność rezultatów techniki HILS z wynikami dla techniki wirtualnej), że realizowany sterownik powinien z bardzo dużym prawdopodobieństwem spełniać stawiane wymagania dotyczące realizacji procesu sterowania optymalnego silnie nieliniowym obiektem docelowym.



Rys. 5.17. Kąt obrotu platformy dla trajektorii typu „okrąg”



Rys. 5.18. Prędkości kątowe kół platformy dla trajektorii typu „okrąg”

5.3.4. Podsumowanie testów HILS

Przeprowadzanie symulacji w czasie rzeczywistym przy wykorzystaniu rzeczywistego sterownika dowiodło poprawności realizacji algorytmu sterowania (szczególnie dla trajektorii „okrąg") oraz wyboru architektury sprzętowej. Niezwykle cennym (jak się w późniejszej fazie okazało, także koniecznym) elementem przeprowadzonych testów jest możliwość weryfikacji, podczas realizacji jednego z etapów (iteracji) techniki wirtualnego prototypowania, długości kroku całkowania. Jak dowiodły testy, ustalony we wcześniejszej fazie realizowanych badań krok całkowania 0,001 s musiał na skutek niespełnienia warunku determinizmu czasowego zostać zwiększony (przy zachowaniu poziomu błędów na niezmienionym poziomie) do wartości 0,005 s.

Ze względu na specyfikę testu HILS, w którym adoptuje się wirtualny model obliczeniowy obiektu (w tym przypadku trójkołowej platformy mobilnej), wystąpiły różnice pomiędzy odpowiedziami układu na przyłożone sygnały sterujące generowane przez rzeczywisty sterownik, a odpowiedziami w przypadku symulacji wirtualnych. Głównym powodem zaistniałej dewiacji jest skończona dokładność obliczeniowa komputerów (błędy numeryczne), co skutkuje pojawianiem się znacznych błędów w trakcie rozwiązywania równań różniczkowych dynamiki platformy. Należy zaznaczyć, że końcowym rezultatem testu HILS jest sygnał poddany czterem przekształceniom różniczkowo-całkowym, tj. odwrotnemu zadaniu kinematyki, odwrotnemu zadaniu dynamiki (niezbędne do wyznaczenia optymalnego sygnału sterującego generowanego w sterowniku cRIO), prostemu zadaniu dynamiki oraz prostemu zadaniu kinematyki (niezbędnemu dla zbadania odpowiedzi modelu).

Dodatkowo, niezwykle istotnym a zarazem negatywnie wpływającym na rezultaty przeprowadzonych testów jest fakt kontrolowania obiektu silnie nieliniowego (macierz bezwładności uzależniona od kąta φ). Małe odchyłki kąta obrotu kierownicy φ w stosunku do wartości pożądanych sprawiają pojawienie się znacznych rozbieżności w przebiegach finalnych. Należy podkreślić, że pomimo występujących różnic, model wirtualny (emulowany) platformy porusza się po prawidłowych trajektoriach, a silnikom nadane są właściwe prędkości oraz kąty wychyleń (silnik sterujący kołem kierowniczym).

W przyjętym modelu obliczeniowym platformy pominięto wpływ dynamiki napędu (silników DC), co niewątpliwie będzie skutkowało w zmianach doboru wartości współczynników macierzy **R** oraz **Q** w sygnale sterującym oraz nastaw wzmocnienia sterowników silnikami (modułami NI – 9505).

Zastosowanie metody HILS umożliwiło autorowi optymalizację kodu algorytmu sterowania. Część kodu została umieszczona w osobnych pętlach, natomiast bloki danych funkcji pogrupowano w podsystemy.

Zastosowany proces optymalizacji znacznie usprawnił przepływ sygnału wewnątrz systemu sterowania, co sprawiło znaczne zmniejszenie wykorzystania pamięci sterownika cRIO). Proces obliczeniowy realizowany w czasie rzeczywistym jest na tyle absorbujący zasoby jednostki CPU (głównie ALU) sterownika, że pomimo realizacji systemu *embedded* (tj. wbudowanego – sterownik obsługuje tylko jedną aplikację) nie udało się znacząco zmniejszyć wykorzystania obciążenia procesora.

Z uwagi na silną nieliniowość układu proces doboru współczynników macierzy **R** oraz **Q** wymagał przeprowadzenia bardzo dużej liczby testów. Niezwykle pomocnym w ich doborze, o czym autor wspominał w treści wcześniejszych akapitów pracy, było zastosowanie środowiska LabVIEW oraz utworzenie interfejsu mechatronicznego umożliwiającego wizualizację wyników przeprowadzanych symulacji, a także szybką korektę parametrów i ustawień.

Podczas przeprowadzania techniki wirtualnego prototypowania, a także testów HILS autor zauważył, że z uwagi na silną nieliniowość układu, sterowalność platformy jest bardzo czuła na wszelkie zmiany wartości momentów bezwładności poszczególnych części układu (szczególnie zespołu układu kierownicy). Wyjaśnienie tego zjawiska leży w równaniu dynamiki Lagrange'a II rodzaju opisującym ruch platformy, w którym każdy ze współczynników jest zależny od momentu bezwładności tej części (kierownicy), a także – od kąta φ.

6. Implementacja systemu nadzorowania ruchu na rzeczywistej platformie mobilnej

Autor realizując przyjętą metodykę projektowania systemu nadzorowania dla trójkołowej platformy mobilnej, zastosował podejście równolegle (kompleksowe), spełniając tym samym jednocześnie jeden z postulatów projektowania mechatronicznego. Obrany kierunek był podyktowany uwzględnianiem w przeprowadzanych badaniach licznych sprzężeń zwrotnych wymuszających wprowadzanie zmian, czy też powodujących całkowite odstąpienie od założonej pierwotnie koncepcji.

Fazą końcową przeprowadzonego projektowania (opisaną we wcześniejszych rozdziałach pracy), weryfikującą przyjęte założenia oraz rozwiązania techniczne (drogą eksperymentalną poprzez realizacje przejazdów platformy po uzgodnionych trajektoriach ruchu) była zastosowana technika szybkiego prototypowania na obiekcie docelowym (rzeczywistej trójkołowej platformie mobilnej powstałej równocześnie w ramach prowadzonych prac rozwojowych systemu nadzorowania).

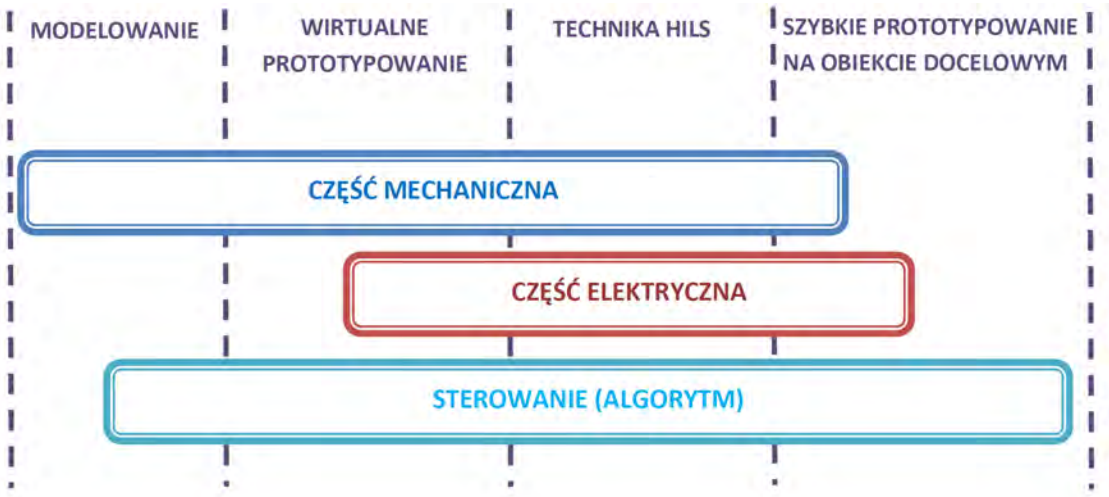
Fazą poprzedzającą rozważane szybkie prototypowanie była technika wirtualnego prototypowania, która w realizowanej pracy umożliwiła osiągnięcie wstępnej optymalizacji, w której dokonano wyboru jednostki docelowej sterownika. Zdefiniowany sterownik posłużył w pierwszej kolejności do przeprowadzania testów HILS, aby w końcowej fazie zostać zamontowanym na rzeczywistej platformie, gdzie w ramach prototypowania na obiekcie docelowym dokonano jednocześnie procesu prototypowania sterownika – algorytmu, a także jego implementacji (integracji) z rzeczywistym obiektem badań. Wspomniana technika HILS pozwoliła autorowi na dokonanie procesu integracji projektowanego algorytmu z rzeczywistym sterownikiem (szczegóły opisano w rozdziale 5). Proces implementacji pociągnął za sobą również konieczność rozbudowy interfejsu mechatronicznego użytkowania, umożliwiającego drogą symulacji komputerowej dobór macierzy **R** oraz **Q**. Należy zaznaczyć, że w systemie docelowym (zaprojektowanym) zastosowano tylko takie rozwiązania sprzętowe i programistyczne, których parametry techniczne umożliwiają generowanie sygnału z zadaną częstotliwością (szybkość transferu danych pomiędzy realizowanymi procesami, odczyt lub zapis stanów wejść/wyjść musiał być szybszy, niż narzucona szybkość generowania sygnałów).

W trakcie realizacji projektu część decyzji, np. wybór koncepcji sterowania kołem kierowniczym, musiała zapaść dość wcześnie. Stąd istniała konieczność określenia sposobu jego sterowania (typu układu), a tym samym – skoncentrowania uwagi nad ustaleniem typu sterownika. Wybór sterownika wiązał się z koniecznością analizy algorytmu sterowania, przy jednoczesnym spełnieniu kryterium optymalizacji energetycznej. Podjęcie decyzji (wybór sterownika oraz dedykowanego modułu sterownika silników DC) umożliwiło przeprowadzenie badań w czasie rzeczywistym realizując testy HILS, podczas których optymalizacji poddany zostaje algorytm oraz ustala się końcową architekturę systemu sterowania. Dodatkowo, dokonuje się podziału wykonania algorytmu sterowania na dwa układy (procesor Real Time oraz układ FPGA), ustala się długość kroku całkowania oraz przeprowadza weryfikację współczynników macierzy **R** i **Q**.

Implementacji proponowanego wskaźnika jakości dokonano w pierwszej kolejności na komputerze PC (realizując technikę wirtualnego prototypowania), aby poprzez realizację techniki HILS w fazie docelowej było ono realizowane na platformie rzeczywistej (sterowniku cRIO zintegrowanym z obiektem badań). Wszystkie ustawienia (wybór trajektorii, ustawienia macierzy **R** i **Q**, metoda oraz krok całkowania) dokonane na komputerze użytkowania są transferowane bezpośrednio do sterownika. Sterownik po uruchomieniu działa według zapisanego algorytmu opisanego zależnością (4.1), a ściślej – zależnością (4.9) uwzględniającą wprowadzone prędkości korygujące. Generowane sygnały (momenty) są zamieniane w układzie FPGA na odpowiednio ukształtowaną falę PWM, sterującą poprzez moduły NI–9505 silnikami DC (szczegóły zostały opisane w rozdziale 5). Należy zaznaczyć (o czym autor wspominał powyżej), że w trakcie realizacji projektu, część badań na czas realizacji innego badania cząstkowego była wstrzymywana. W niejednym przypadku wyniki cząstkowe przeprowadzonych badań stanowiły w następnej fazie warunki brzegowe dla innej analizy. Tak było i w tym przypadku, w którym autor w celu wypracowania optymalnych warunków energetycznych systemu, sprawdzał, podczas realizacji techniki wirtualnego prototypowania, warunek determinizmu czasowego. Realizował on równolegle technikę HILS – fazę „wstępną", w której istotą jest przyjęcie (w tym przypadku, w drodze badań mechatronicznych) takiego kroku całkowania (warunku brzegowego), dla którego spełniony jest warunek czasu rzeczywistego.

Zgodnie ze zrealizowaną koncepcją projektowania mechatronicznego, autor ze względu na nierozerwalną zależność projektowanych systemów, stanowiących funkcjonalną całość architektury platformy mobilnej, przez większość faz projektu realizował wiele zadań równocześnie. Zależność użytych do budowy elementów, a także przyjęta koncepcja (założenia konstrukcyjne) wpłynęła na konieczność, od samego początku trwania projektu, uwzględniania szeregu szczegółów niezbędnych w danej chwili badań.

Na rys. 6.1 przedstawiono schematycznie fazy rozpatrywanego projektu mechatronicznego oraz czasoprzestrzeń zrealizowanych zadań, a także przeprowadzonych optymalizacji.



Rys. 6.1. Fazy projektowania mechatronicznego

Określony charakter platformy, a także konieczność zapewnienia poruszania się platformy bez poślizgów, determinowały przyjętą konstrukcję geometryczną oraz rozwiązania systemów napędowego i kierowniczego. Zastosowany algorytm, którego postać umożliwia minimalizację w trybie *on-line* powstałych odchyłek, a także bardzo silnia nieliniowość projektowanej platformy, narzuciły konieczność stosowania bardzo wydajnej jednostki obliczeniowej systemu sterowania, rozwiązującej w czasie rzeczywistym, przy ustalonym drogą badań mechatronicznych kroku całkowania, równania różniczkowe (w celu wyznaczenia optymalnego sygnału sterującego).

Niezwykle istotnym elementem, zastosowanej (mechatronicznej) koncepcji projektowania jest użyte środowisko LabVIEW oraz jego zintegrowane możliwości

sprzętowe [40]. Na fundamencie tego środowiska, realizując technikę wirtualnego prototypowania powstała architektura platformy oraz systemu sterowania (algorytmu). Możliwości LabVIEW w zakresie stosowania techniki HILS, zostały wykorzystane do optymalizacji przyjętego rozwiązania oraz zastosowanych koncepcji badań. Warto zaznaczyć (o czym autor wspominał w poprzednich rozdziałach), że do realizacji systemu sterowania platformy użyto jednostkę czasu rzeczywistego cRIO, która została zaprogramowana również na bazie wykorzystanego środowiska LabVIEW.

6.1. Opis obiektu badań

Główna koncepcja oraz mechaniczna architektura systemu kierowniczego i przeniesienia napędu projektowanej platformy została pokazana na rys. 6.2. Określenie modelu matematycznego przyjętej koncepcji, przegląd dostępnej literatury oraz publikacji pozwoliły na zawężenie obszaru poszukiwań parametrów geometrycznych platformy, a także – na wstępną analizę (parametrów) dostępnych części mechanicznych.

Pozyskane wyniki z obu zastosowanych technik projektowania mechatronicznego umożliwiły optymalny wybór parametrów geometrycznych platformy oraz jej komponentów. Dobór systemu napędowego oraz kierowniczego został dokonany przy dodatkowym uwzględnieniu parametrów i możliwości zastosowanej jednostki sterownika cRIO.

Głównym czynnikiem w doborze silników okazał się zastosowany moduł sterownika NI – 9505. Parametrem decydującym okazała się rekomendowana przez producenta częstotliwość fali PWM, a także maksymalny prąd rozruchu dla tego modułu. Należy zaznaczyć, że w przypadku zastosowania sterowania impulsowego, jakim jest PWM, prąd rozruchu silnika jest identyczny z prądem nominalnym. Autor jednak na podstawie zdobytego doświadczenia, a także kierując się dobrą praktyką inżynierską podjął decyzję o wyborze silników spełniających wymagania tzw. *worst case scenerio* i zawęził swoje poszukiwania do silników, których prąd rozruchu wynosi maksymalnie 12 A, a także – posiadających możliwość sterowania falą o częstotliwości 20 kHz. Autor, w celu zapewnienia poprawnej pracy jednostki napędowej i kierującej oraz bezpiecznej pracy modułu sterownika silników, doposażył moduły NI–9505 w końcówkę mocy NI–9931, która rozszerzyła zakres dopuszczalnej

ciągłej obciążalności prądowej z 5 A do 7,3 A przy szczycie prądowym rzędu 12 A w czasie mniejszym, niż 2 sekundy.

Przeprowadzone badania na określonych trasach przejazdu wirtualnej platformy (trajektoriach) pozwoliły na wypracowanie koncepcji kształtu i rozmiaru platformy. Ze względu na podjęcie decyzji, iż do sterowania układem kierowniczym zostanie wykorzystany motoreduktor DC, istniała konieczność zaprojektowania systemu prostego o jak najmniejszej liczbie elementów, które mogły ulec szybkiemu zużyciu. Zdecydowano, że silnik będzie bezpośrednio sterował kierownicą oraz będzie usytuowany pionowo. Decyzja montażu motoreduktora, ustaliła jego górną granicę długości tak, aby ewentualne przeniesienie środka ciężkości do góry nie powodowało utraty stabilności w trakcie rzeczywistych przejazdów. Na tym etapie, autor podjął również decyzję, iż systemem sterowania platformy (mając na uwadze wcześniej rozważane środowisko LabVIEW oraz jego pełną integrację z oferowanym również sprzętem) będzie bazował na środowisku LabVIEW, z którego będą generowane sygnały sterujące. Dodatkowo, przyjął silnik prądu stałego (motoreduktor), jako aktor systemu napędowego oraz kierowniczego. Określił również architekturę napięciową, w której przyjął 12V DC jako obowiązujący standard zasilania. Warto, zauważyć, że w tej fazie nie rozważano sterownika (generatora fali PWM) silnika DC. Mając na uwadze wymagania wynikające z koncepcji systemu sterowania (algorytmu) autor poszukiwał możliwości implementacji środowiska LabVIEW do struktur platformy mobilnej. Rozpatrywanym przez dość długi okres czasu był komputer jednoukładowy PC104, którego procesor był na tyle wydajny, że istniała możliwość generowania sygnału w czasie rzeczywistym.

Wstępna selekcja komponentów platformy określiła górną granicę masy projektowanego obiektu (10 kg). W tym miejscu nastąpiło sprzężenie zwrotne, które umożliwiło wybór odpowiedniego do masy platformy systemu mechanizmu różnicowego. Zastosowano system pochodzący od ważącego 15 kg zdalnie sterowanego samochodu benzynowego typu Off Road Buggy firmy *HiMoto*. Wyspecyfikowany układ mechanizmu różnicowego oraz maksymalna masa platformy zawęziły obszar poszukiwań odpowiednich silników DC. Swoja uwagę autor skierował na silniki firmy *MicroMotors*, które charakteryzują się doskonałymi parametrami, przy niewygórowanej cenie.

W tym samym czasie autor na podstawie przeprowadzonych badań (m.in. kinematyki oraz dynamiki układu, wirtualnego prototypowania), analizy (m.in.

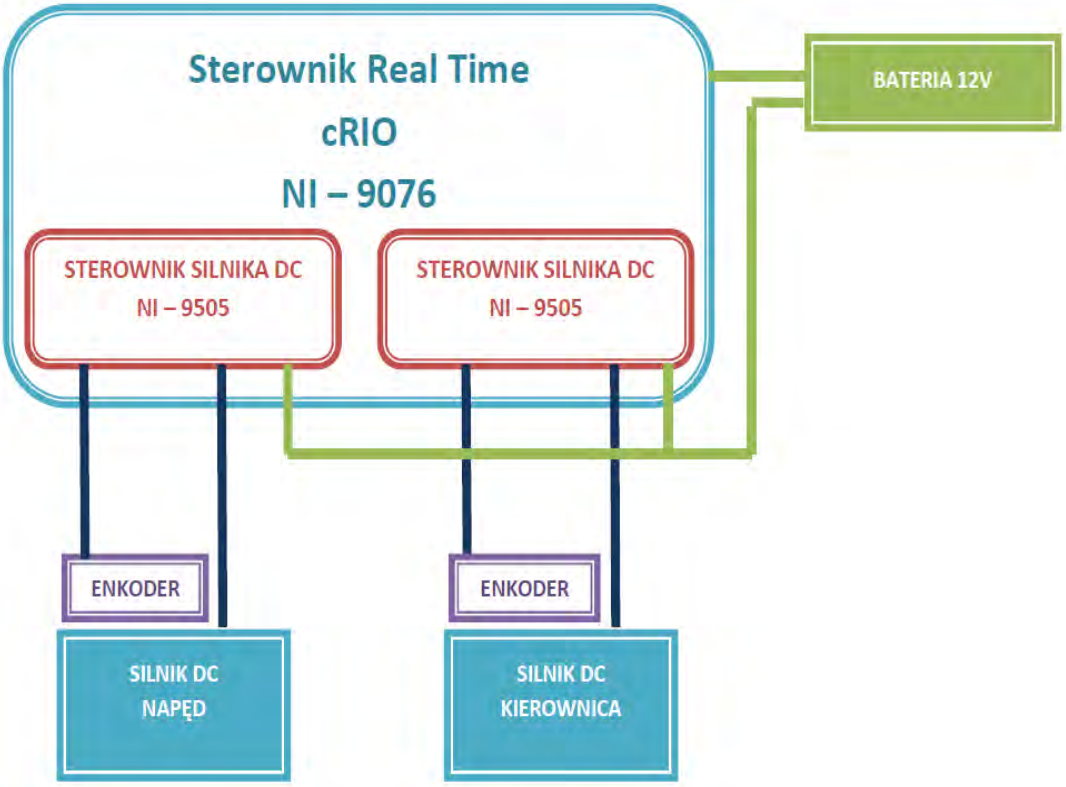
mnożników Lagrange'a) oraz możliwości układu napędowego (prędkość silnika DC przy maksymalnym obciążeniu, odpowiedni moment silnika, stopień przełożenia układu mechanizmu różnicowego, średnica dostępnych kół itp.) zweryfikował maksymalną prędkość, z jaką platforma będzie mogła się poruszać (0,17 m/s). Należy zauważyć, że proces definiowania prędkości był dość długi. Przeprowadzane analizy oraz badania były realizowane w ciągłej pętli sprzężenia zwrotnego, które ostatecznie ustaliły przyjęte rozwiązanie. W trakcie przeprowadzonych eksperymentów autor zweryfikował ustaloną w trakcie analizy teoretycznej prędkość poruszania się platformy. Okazało się, że skonstruowana platforma porusza się dwa razy szybciej, niż zakładano tj. z prędkością 0,35 m/s. Podłożem zaistniałej rozbieżności stał się fakt przyjęcia przez autora, podczas fazy projektowania systemu napędowego platformy, prędkości maksymalnej silnika przy pełnym obciążeniu. Eksperymenty ostatecznie zweryfikowały przyjęte założenie oraz wykazały brak wpływu obciążenia platformy na prędkość pracy silnika.

Równolegle z procesem ustalenia prędkości poruszania się platformy opracowywano metody określania innych istotnych parametrów jej pracy.

Niezmiernie ważnym elementem wpływającym na poprawność realizacji założeń konstrukcyjnych stała się weryfikacja możliwości systemu nadzorowania. Przeprowadzone badania (w tym analiza wydajności energetycznej) zdefiniowały ostatecznie wymagania stawiane temu systemowi. Autor dokonując analizy koncepcji systemu nadzorowania bazującej na wspomnianym wcześniej komputerze jednoukładowym, zweryfikował możliwości stosowania systemu czasu rzeczywistego pracującego na bazie Windows (warunek nie realizowalny w przypadku stosowania tego systemu). Dodatkowo, przegląd oferty rynkowej sterowników silnika DC (istniała konieczność, aby sterownik silnika pracował z co najmniej równą lub wyższą częstotliwością, co generowany optymalny sygnał sterujący) oraz układu wejść enkoderów (które również musiały być odczytywane z właściwą częstotliwością tzn. równą lub większą generowanemu sygnałowi sterującemu) spowodowały podjęcie decyzji o zaniechaniu realizacji pierwotnej koncepcji systemu nadzorowania oraz opracowaniu jego nowej architektury. Postanowiono, że zdefiniowana ponownie architektura będzie bazowała na sterowniku cRIO (wraz modułami sterowników silnika DC – generatorów fali PWM), do którego zaimplementowano rozważany algorytm sterowania. Wybór jednostki sterującej (szeroko opisanej w rozdziale 5) pozwolił na przeprowadzenie w czasie rzeczywistym testów HILS, gdzie stosując

metody optymalizacji energetycznej autor ustalił końcowy krok całkowania równań różniczkowych, a tym samym – częstotliwość generowania optymalnego sygnału sterującego (200 Hz). Wspomniane powyżej enkodery zostały zamocowane na wale silnika, a nie motoreduktora. Ustalony w trakcie badań mechatronicznych krok całkowania narzucił pośrednio rozdzielczość niezbędnego w budowie platformy enkodera (rozdział 5). Istniała, bowiem konieczność, aby w czasie 1 s liczba zmian stanów enkodera był równa lub większa przyjętej częstotliwości sygnału sterującego. Dla najwolniejszego motoreduktora platformy, tj. kierownicy liczba obrotów na minutę przy pełnym obciążeniu wynosi 58. Wynika z tego, że (w przybliżeniu) w trakcie sekundy motoreduktor wykonuje jeden obrót. Generowany sygnał sterujący w ciągu rozważanego czasu tj. jednej sekundy, jest w stanie ustalić pozycję silnika 200–krotnie. Fakt ten ustala dolną granicę rozdzielczości enkodera, która powinna wynosić 200 impulsów na obrót (cpr). Autor sugerując się wcześniejszymi założeniem o częstotliwości generowania sygnału na poziomie 1 kHz, dokonał specyfikacji enkodera o rozdzielczości 1000 impulsów na obrót. Późniejsza weryfikacja możliwości sterownika oraz zmiana częstotliwości generowania sygnału z 1 kHz do 200 Hz, nie spowodowała faktu poszukiwania enkodera o innych parametrach. Ostatecznie do platformy zastosowano enkodery optyczne o wcześniej wyspecyfikowanej rozdzielczości, tj. 1000 impulsów na obrót.

Ze względu na zapewnienie platformie mobilności, układ napędowy oraz sterowania zasilany jest najcięższą dopuszczalną dla projektowanej platformy baterią, tj. akumulatorem żelowym 12 V. Rys. 6.2 przedstawia schematycznie system elektryczny zaprojektowanej platformy.



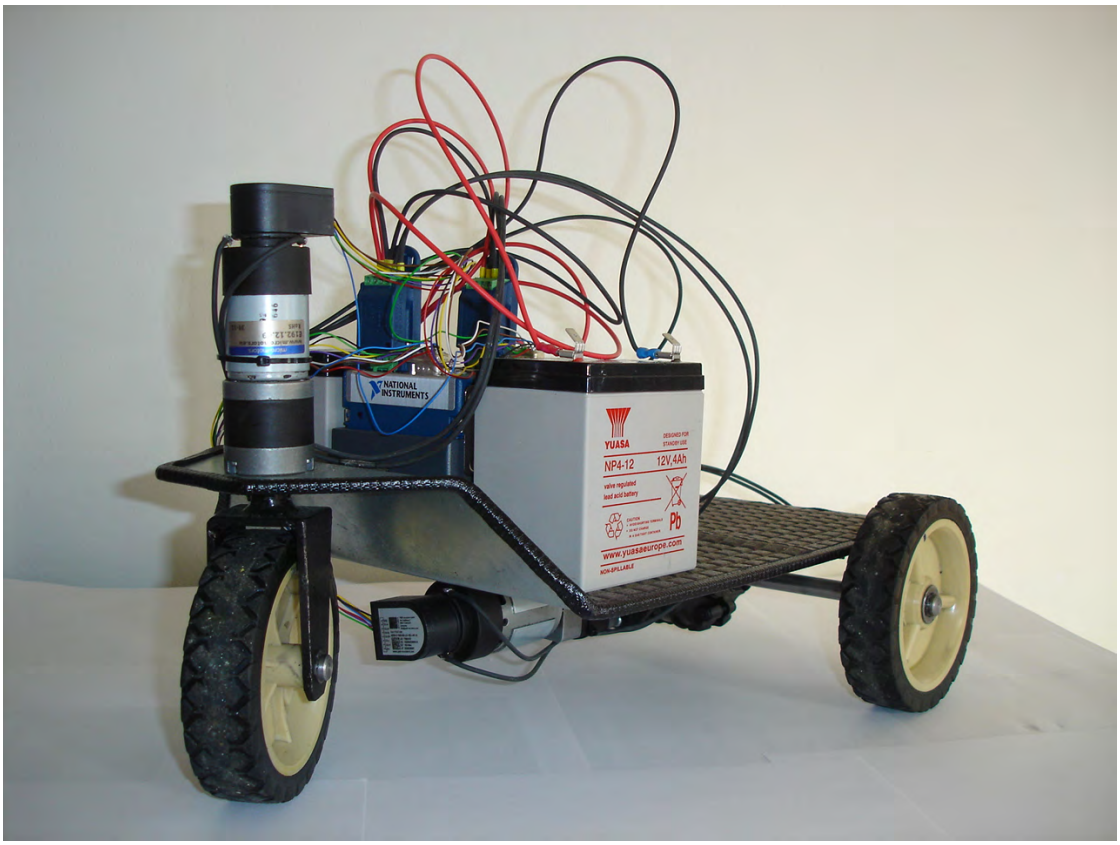
Rys. 6.2. System elektryczny trójkołowej platformy mobilnej

W całym tym cyklu realizowano szereg sprzężeń zwrotnych, w których poddawano modyfikacji geometrię platformy i typy silników. Rozważano przełożenia przekładni planetarnych tak, aby uzyskać właściwą prędkość oraz moment obrotowy. Istotnym stał się również parametr prądu rozruchu silników, który nie mógł, z uwagi na zastosowane końcówki mocy wyjść sterownika, przekroczyć 12 A, a także –system sterowania.

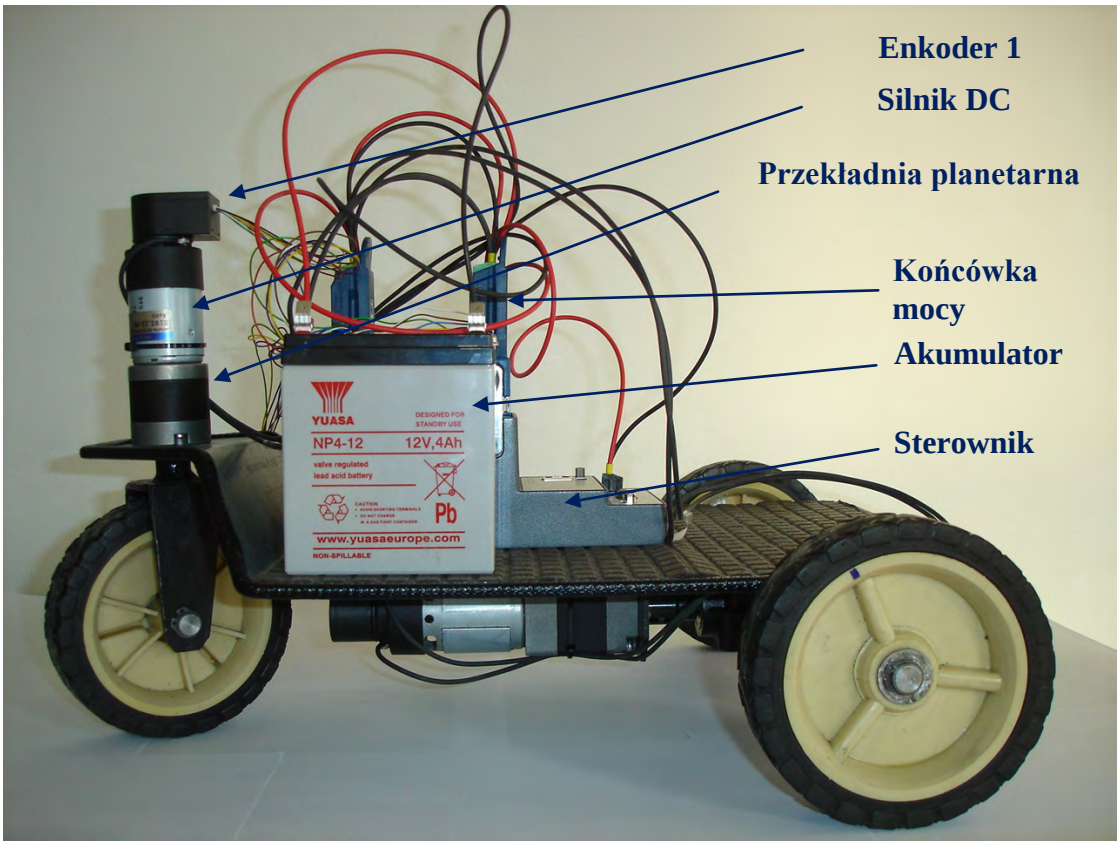
Mechatroniczne podejście w projektowaniu platformy mobilnej spowodowało, iż od wczesnych faz trwania tego procesu, wszystkie rozważane systemy (mechaniczny, elektryczny oraz sterowania) musiały być rozpatrywane jako nierozłączna całość o strukturze połączeń wzajemnie na siebie oddziaływujących. Podejście mechatroniczne znacznie skróciło realizację projektu oraz wyeliminowało dość częsty zakup niewłaściwych elementów w przypadku stosowania podejścia szeregowego. Niezwykle istotnym i wartym podkreślenia jest fakt użytego mechatronicznego środowiska projektowego LabVIEW, którego możliwości oraz zalety zostały wykorzystane do realizacji badań nad implementacją rozpatrywanego algorytmu (energetycznego wskaźnika jakości) do struktury trójkołowej platformy mobilnej.

Należy zaznaczyć, że projekt badawczy realizowany był przez jedna osobę, która obejmowała całość tematyczną przedsięwzięcia. W przypadku bardziej złożonych projektów, w których zaangażowana jest większa liczba osób (mechatroników), tylko częsta dyskusja, swobodna wymiana myśli, otwartość, weryfikacja zadań cząstkowych i większych części, a także – podział ról i wspólny cel mogą przynieść założony efekt.

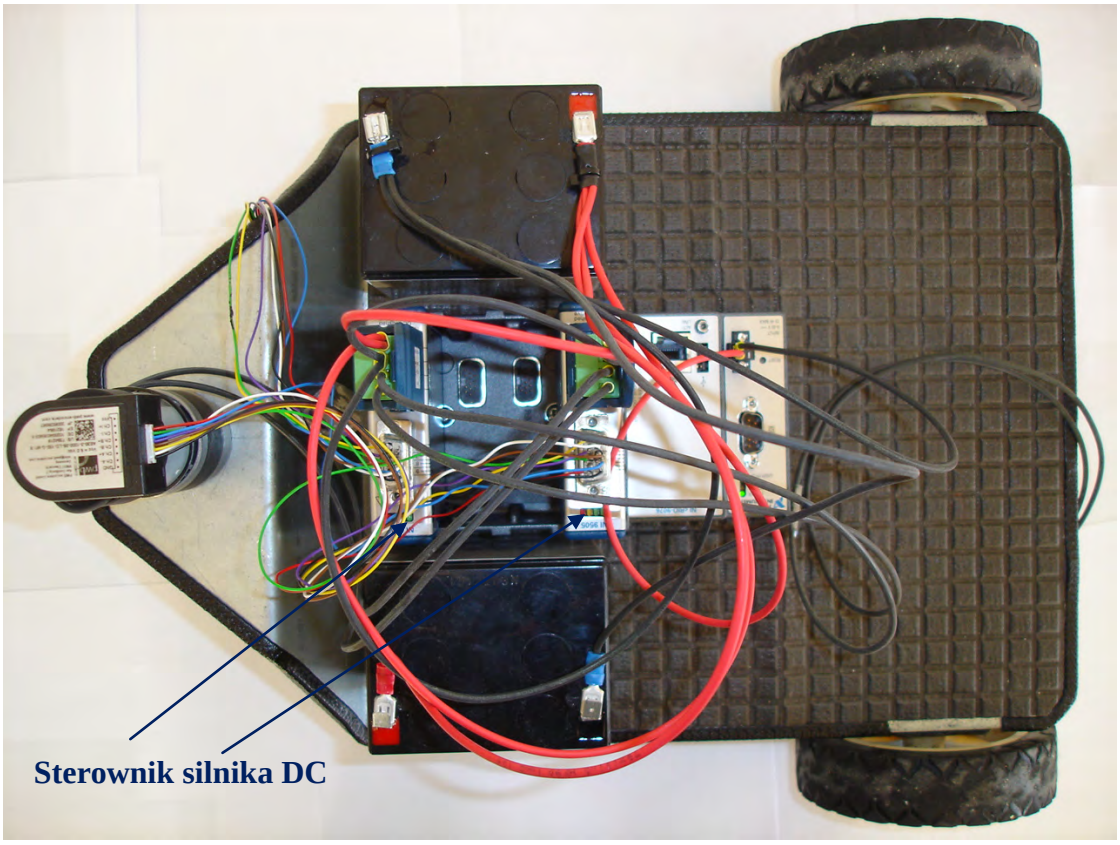
Na rys. 6.3, 6.4, 6.5, 6.6 oraz 6.7 zamieszczono zdjęcia zbudowanej w Katedrze Mechaniki i Mechatroniki PG, przy wydatnym współdziałaniu autora, dla potrzeb realizacji badań trójkołowej platformy mobilnej.



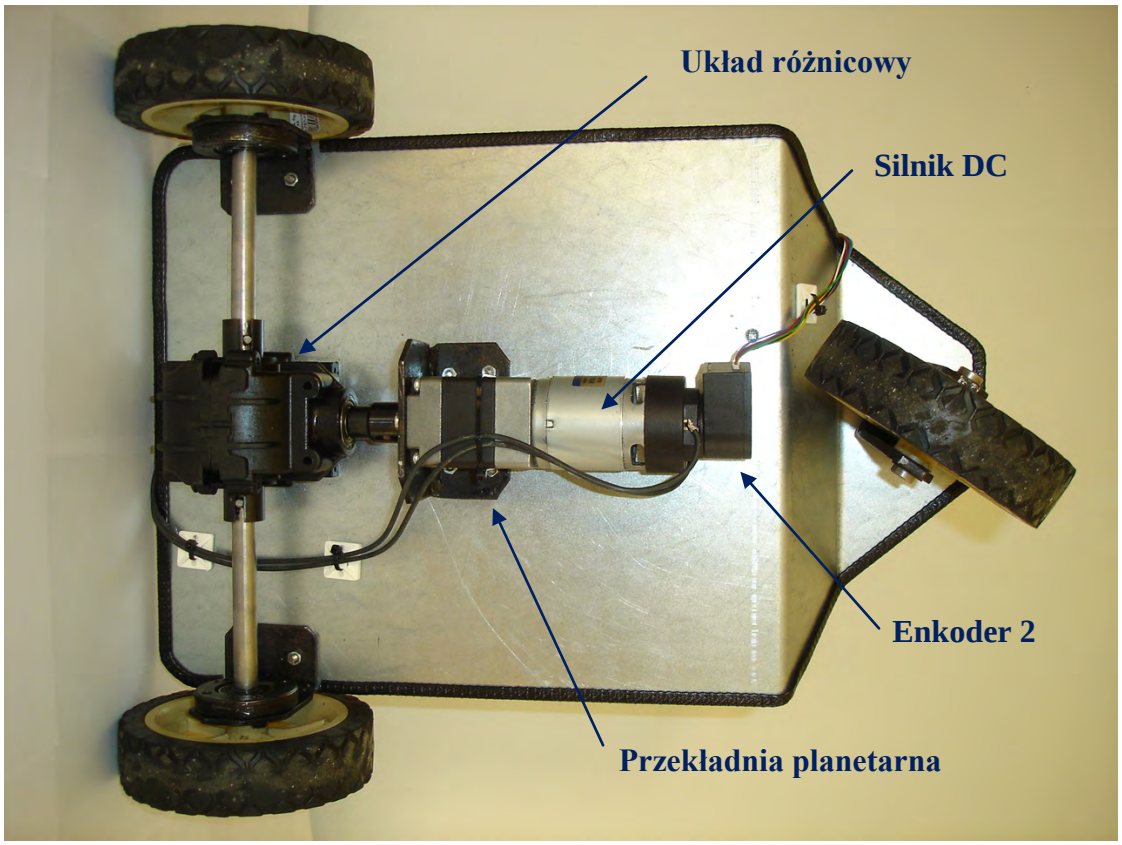
Rys. 6.3. Trójkołowa platforma mobilna



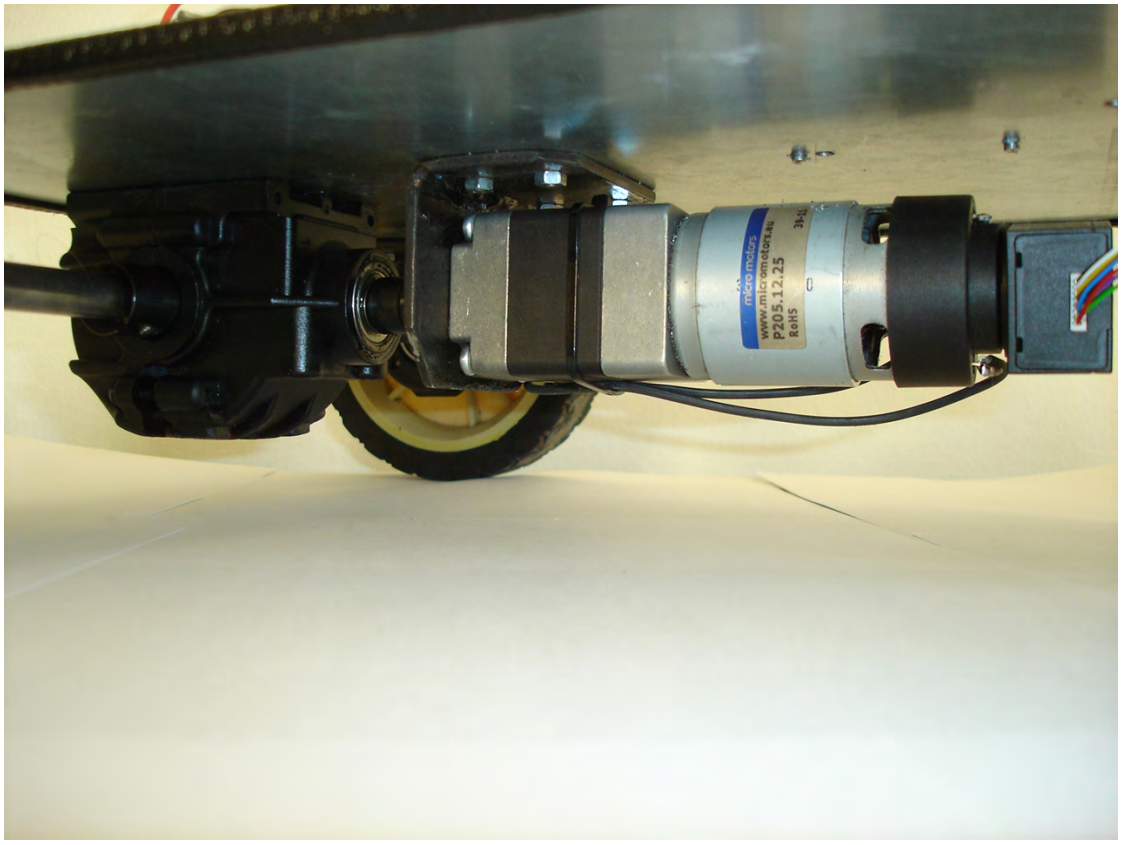
Rys. 6.4. Trójkołowa platforma mobilna



Rys. 6.5. Trójkołowa platforma mobilna



Rys. 6.6. Trójkołowa platforma mobilna



Rys. 6.7. Trójkołowa platforma mobilna (układ napędowy – motoreduktor, układ różnicowy)

6.1.1. Parametry trójkołowej platformy mobilnej

Autor w poniższych tabelach przedstawił najważniejsze parametry zbudowanej platformy. Wyodrębniono części: definiujące parametry platformy, mechaniczną oraz elektryczną (w tym, sterowania).

Tab. 6.1 Parametry konstrukcyjne trójkołowej platformy mobilnej

Oznaczenie zgodne z przyjętym modelem obliczeniowym platformy (rys.3.2)	l [m]	l_1 [m]	l_3 [m]	r [m]	v_A [m/s]	m_1 [kg]	m_2 [kg]	m_3 [kg]	m_5 [kg]	f [m]
Wartość	0,325	0,12	0,12	0,064	0,17	0,4	0,4	0,4	8	0,001

Oznaczenie zgodne z przyjętym modelem obliczeniowym platformy (rys.3.2)	N_1 [N]	N_2 [N]	N_3 [N]	I_{x1}, I_{x2}, I_{x3} [kgm^2]	I_{z1}, I_{z2}, I_{z3} [kgm^2]	I_{x3} [kgm^2]	I_{z5} [kgm^2]	M_0 [Nm]
Wartość	22	22	55	0,00055	0,00082	0,2125	2,25	0,42

Tab. 6.2. Parametry techniczne systemu napędowego trójkołowej platformy mobilnej

Element	**Cecha**
Motoreduktor układu napędowego	MicroMotors P205.12.25 250 Ncm; 110rpm@max.moment, 54 W
Motoreduktor układu sterującego	MicroMotors E192.12.49 160 Ncm; 58rpm@max.moment, 19,2 W
Układ różnicowy	HiMoto 1:3,33
Enkoder	PWB Encoders AE30 Optyczny 1000 impulsów/obrót

Tab. 6.3. Parametry techniczne systemu sterującego trójkołowej platformy mobilnej

Element	**Cecha**
Jednostka sterująca	cRIO NI-9076, 400 MHz, 256DRAM, FPGA Xilinx Spartan6 LX-45
Sterownik silnika	NI-9505 Servo Drive, PWM 40 kHz, Odświeżanie wyjść 20 MHz, szczytowe natężenie prądu 12 A
Końcówka mocy	NI-9931
System operacyjny	LabVIEW Real Time z modułami FPGA, Control System & Design
Częstotliwość sygnału sterującego	200 Hz

6.2. Cel badań

Celem przeprowadzanych badań jest potwierdzenie przez autora drogą eksperymentalną skuteczności sterowania optymalnego przy energetycznym wskaźniku jakości podczas nadzorowania ruchu trójkołowej platformy mobilnej, zrealizowanego w wyniku zastosowania technik projektowania mechatronicznego.

Badania zostały przeprowadzone w wyniku realizacji przejazdów rozważanej platformy mobilnej dla trzech typów trajektorii ruchu, które zostały wzięte pod uwagę podczas prowadzonych prac badawczych nad systemem nadzorowania tj. „sinus", „parabola" oraz „okrąg". Osiągana większa prędkość przez platformę (niż to wynikało z wcześniejszych założeń) zweryfikuje przydatność (elastyczność) proponowanej metody sterowania dla innych konfiguracji systemu (próby przejazdów w ramach danej trajektorii zostaną przeprowadzone dla dwóch prędkości poruszania się platformy).

Autor na podstawie wcześniejszych badań [59] wykazujących nieprzydatność sterowania nieliniowym obiektem przy użyciu sterowania PID założył, że proponowana metoda sterowania optymalnego w przypadku silniej nieliniowych obiektów (macierze **M** oraz **L** silnie nieliniowe) jest skuteczniejsza niż metody sterowania z wykorzystaniem regulatorów PID i zrezygnował z ich porównywania.

6.3. Opis stanowiska

Weryfikacja systemu nadzorowania przy energetycznym wskaźniku jakości, zrealizowanego przy zastosowaniu technik projektowania mechatronicznego (wirtualne prototypowanie, HILS), została przeprowadzona (o czym autor wspominał) drogą badań doświadczalnych na rzeczywistym obiekcie (trójkołowa platforma mobilna własnej konstrukcji). W ten sposób zrealizowano ostatnią z wcześniej opisanych technik projektowania mechatronicznego, tj. szybkie prototypowanie w systemie docelowym.

Stanowisko do badań doświadczalnych (oraz prototypowania) składało się z elementów składowych użytych podczas realizacji techniki wirtualnego prototypowania oraz techniki HILS. Komputer klasy PC, na którym zainstalowano oprogramowanie LabVIEW (szczegóły w puncie 5.2), podłączono poprzez zewnętrzny przełącznik sieciowy LAN do sterownika zintegrowanego ze zbudowaną platformą mobilną. Kabel sieciowy pomiędzy przełącznikiem a sterownikiem był na tyle długi, że istniała możliwość swobodnego poruszania się obiektu w miejscu przeprowadzania badań. Należy zaznaczyć, że skonstruowany robot mobilny jest jednostką całkowicie autonomiczną, poruszającą się według zapisanego programu sterownika (w tym przypadku, generującego optymalne sygnały sterujące dla rozpatrywanych trajektorii przejazdu). Z uwagi na konieczność dokonywania ustawień wzmocnień sterownika (kompilacji programu na PC oraz jego wczytania do sterownika), ustawień rozpoczynających realizację trajektorii (konieczność zapewnienia w całym cyklu badań tych samych warunków początkowych przejazdu platformy – ustawienie w pozycji zero koła kierownicy), a także zbierania danych pomiarowych, autor postanowił nie odłączać kabla od sterownika (robot w trakcie całego cyklu badań był połączony kablem). Użyte w trakcie prac rozwojowych nad systemem nadzorowania parametry trajektorii ze względów lokalowych, w których przeprowadzano badania uległy zmodyfikowaniu tak, aby realizacja przejazdów była możliwa. Ostatecznie zmianie uległ promień okręgu, którego rozmiary zmniejszono do 1,5 m (autor postanowił dodatkowo sprawdzić trajektorię typu „okrąg”, dla której promień wyniósł 1 m). Dodatkowo, autor postanowił zmodyfikować parametry trajektorii typu „sinus” tak, aby platforma pokonywała dwa pełne okresy tej krzywej oraz była realna możliwość wykazania skuteczności działania realizowanego algorytmu (w tym,

realizacja pomiarów). W tym przypadku zmianie uległ okres oraz amplituda krzywej sinus.

Ze względu na wspomnianą w podpunkcie 6.1 większą osiągalną prędkość poruszania się platformy, autor postanowił dodatkowo sprawdzić dla każdej trasy przejazdu dwie prędkości (tj. 0,17 m/s oraz 0,31 m/s, stanowiącą 90% wartości maksymalnej platformy), a tym samym potwierdzić elastyczność stosowania rozważanej metody nadzorowania ruchu. W różnych częściach realizowanej trajektorii obrano punkty pomiarowe, w których poprzez pomiar odchyłki realizowanej trasy przejazdu platformy od wartości zadanej zweryfikowano skuteczność nadzorowania obiektu. Pomiaru dokonano mierząc współrzędne punktu charakterystycznego platformy od środka układu współrzędnych kartezjańskich umieszczonego w obranym punkcie pomiarowym. Dla każdego z przyjętych punktów pomiarowych (szczegóły rozmieszczenia punktów dla uzgodnionych trajektorii omówiono w podpunkcie prezentującym rezultaty badań dla danej trasy) wykonano 10 przejazdów jeden po drugim.

Proces końcowej implementacji algorytmu wiązał się (główne prace nad rozwojem systemu nadzorowania zostały przeprowadzone podczas realizacji technik wirtualnego prototypowania oraz HILS) z ustalaniem eksperymentalnym wzmocnień sterownika oraz macierzy **R** i **Q**. Jako kryterium oceny w doborze powyższych parametrów wzięto pod uwagę błędy położenia oraz charakter przejazdu platformy (konieczność zapewnienia łagodnego przejazdu).

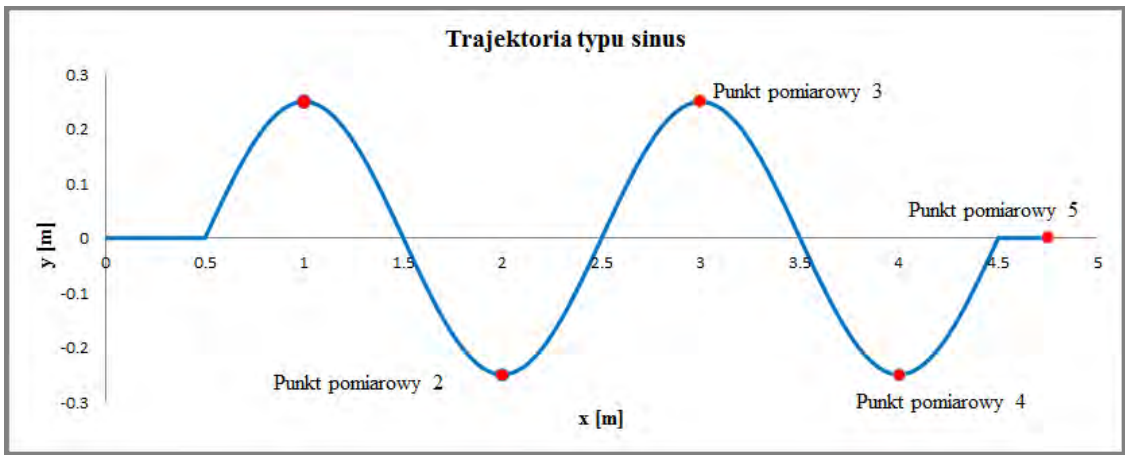
W trakcie trwania eksperymentów niezwykle ważnym, ale też bardzo wymagającym procesem stało się zapewnienie, aby koło kierownicy w momencie startu (rozpoczęcia realizacji zadanej trajektorii) znajdowało się w pozycji zero (aby platforma poruszała się na wprost). Ze względu na fakt, iż pozycja enkodera jest resetowana po ponownym załadowaniu programu sterownika („miękki” reset), autor każdorazowo przeprowadzał fazę kalibracji położenia koła (próbna jazda do przodu).

Autor przeprowadzając proces szybkiego prototypowania na platformie docelowej, przeprowadzał wiele testów poprawności pokonywania ustalonych tras oraz parametrów jezdnych platformy. Z przeprowadzonych wstępnych eksperymentów nasuwał się jeden wniosek, że platforma porusza się bez jakichkolwiek poślizgów. Nagłe przyspieszenie od prędkości zerowej do prędkości maksymalnej bądź w kierunku przeciwnym – do całkowitego zatrzymania, nie powoduje powstania efektu utraty przyczepności kół z podłożem. Jest to

niezaprzeczalna zaleta układu jezdnego, którego zachowanie z góry, z bardzo dobrą dokładnością można przewidzieć i poddać procesowi sterowania.

6.4. Sterowanie optymalne przy energetycznym wskaźniku jakości dla trajektorii typu „sinus”

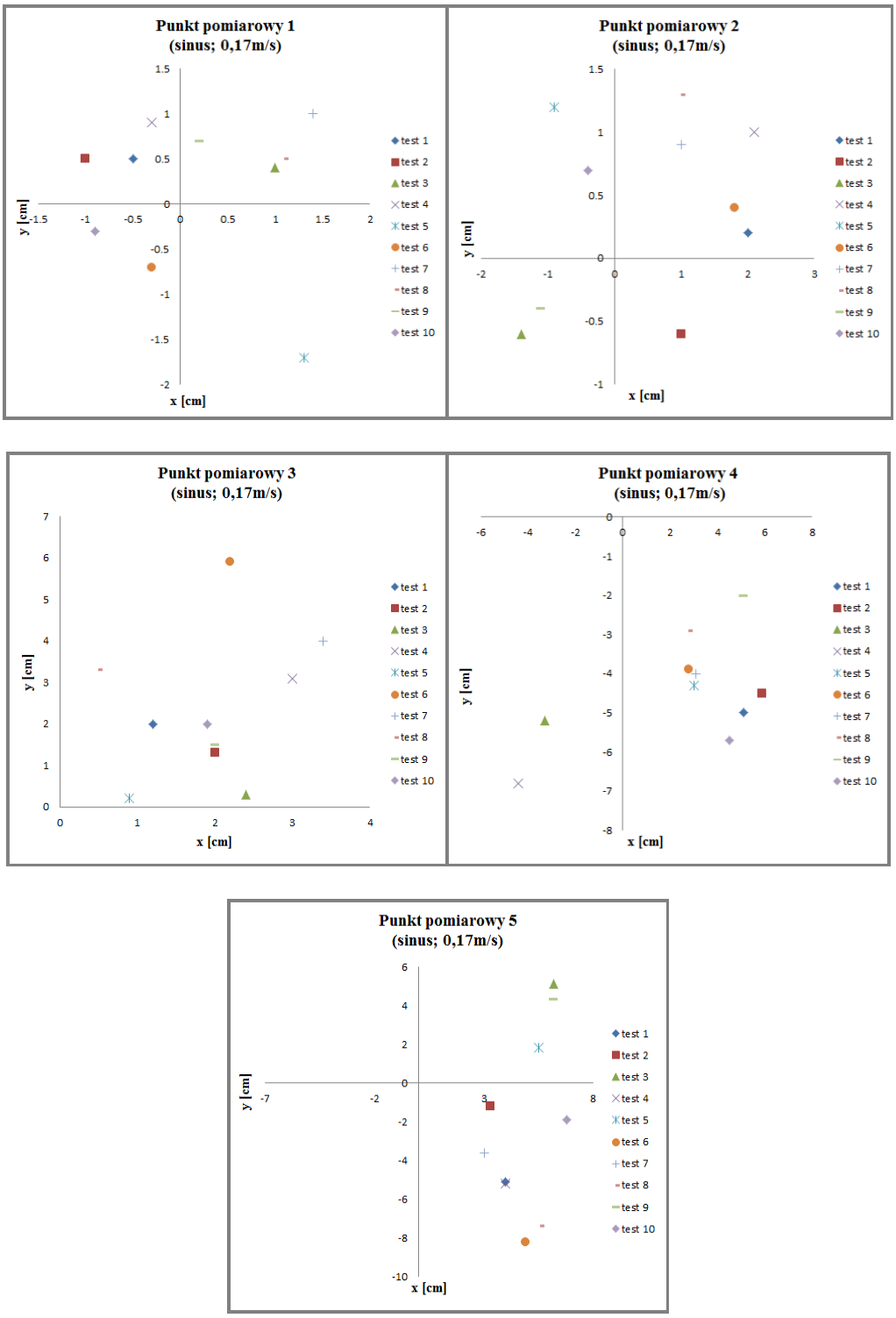
Eksperymenty zostały rozpoczęte od przejazdów rzeczywistej platformy mobilnej po trajektorii typu „sinus”. Jej przebieg wraz z punktami (punkt pomiarowy 1, 2, 3, 4 i 5) w których dokonano pomiaru odchyłki od położenia zadanego, został wykreślony na rys. 6.8. W celu eksperymentalnego wykazania skuteczności sterowania optymalnego, dla każdego punktu pomiarowego, w którym dokonano pomiaru punktu charakterystycznego platformy mobilnej, zrealizowano 10 przejazdów. Następnie wyniki zestawiono w tab. 6.4 oraz 6.5, odpowiednio dla prędkości 0,17 m/s oraz 0,31 m/s. Reprezentacja graficzna przeprowadzonych badań została zamieszczona na rys. 6.9 oraz na rys. 6.12. Autor postanowił dla przejazdu piątego wykreślić dane pomiarowe pochodzące z enkoderów platformy oraz porównać otrzymane wyniki z wartościami zadanymi (rys. 6.10, rys. 6.11, rys. 6.13 oraz rys. 6.14). Zauważalne nieregularności w przebiegach wartości umieszczonych na platformie enkoderów tłumaczy się niedokładnością konstrukcji mechanicznej zbudowanej platformy mobilnej, a także zastosowanego typu enkodera, gdzie przy dużej liczbie impulsów na obrót (1000), rejestrowane są najmniejsze odchyłki czy też nierówności.



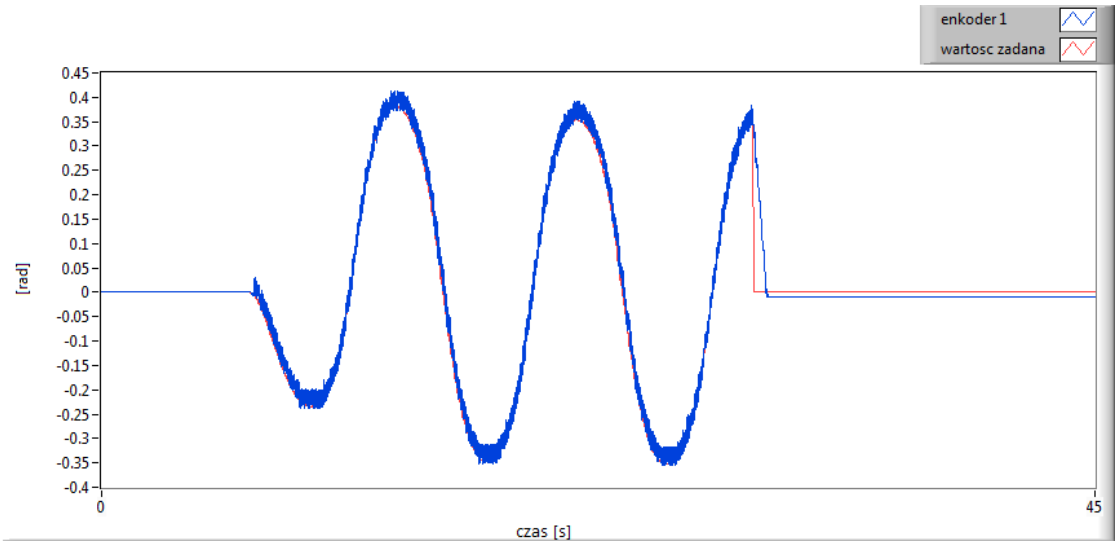
Rys. 6.8. Zadana trajektoria ruchu platformy mobilnej typu „sinus” (punktu charakterystycznego) wraz z naniesionymi punktami pomiarowymi

Tab. 6.4. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „sinus”. Prędkość przejazdu platformy 0,17 m/s

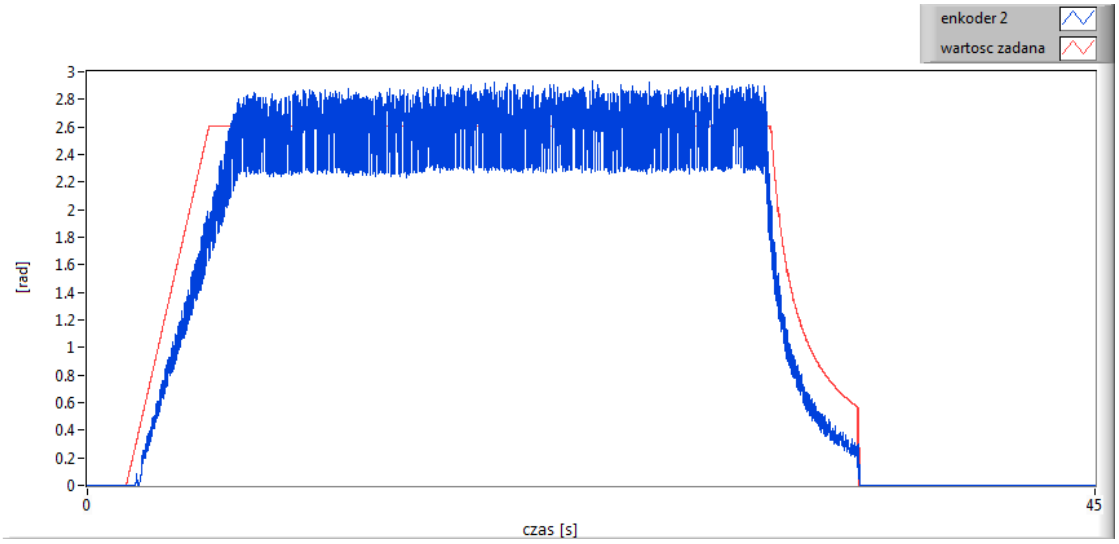
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4		PUNKT 5	
	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]
test 1	-0,5	0,5	2	0,2	1,2	2	5,1	-5	4	-5,1
test 2	-1	0,5	1	-0,6	2	1,3	5,9	-4,5	3,3	-1,2
test 3	1	0,4	-1,4	-0,6	2,4	0,3	-3,3	-5,2	6,2	5,1
test 4	-0,3	0,9	2,1	1	3	3,1	-4,4	-6,8	4	-5,2
test 5	1,3	-1,7	-0,9	1,2	0,9	0,2	3	-4,3	5,5	1,8
test 6	-0,3	-0,7	1,8	0,4	2,2	5,9	2,8	-3,9	4,9	-8,2
test 7	1,4	1	1	0,9	3,4	4	3,1	-4	3	-3,6
test 8	1,1	0,5	1	1,3	0,5	3,3	2,8	-2,9	5,6	-7,4
test 9	0,2	0,7	-1,1	-0,4	2	1,5	5,1	-2	6,2	4,3
test 10	-0,9	-0,3	-0,4	0,7	1,9	2	4,5	-5,7	6,8	-1,9
wartość min. (bezwzględna) [cm]	0,2	0,3	0,4	0,2	0,5	0,2	2,8	2	3	1,2
wartość średnia (bezwzględna) [cm]	0,8	0,72	1,27	0,73	1,95	2,36	4	4,43	4,95	4,38
wartość max. (bezwzględna) [cm]	1,4	1,7	2,1	1,2	3,4	5,9	5,9	6,8	6,8	8,2
wartość średnia dla trajektorii [cm]	2,594	2,524								



Rys. 6.9. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „sinus" przy prędkości 0,17 m/s dla poszczególnych punktów pomiarowych



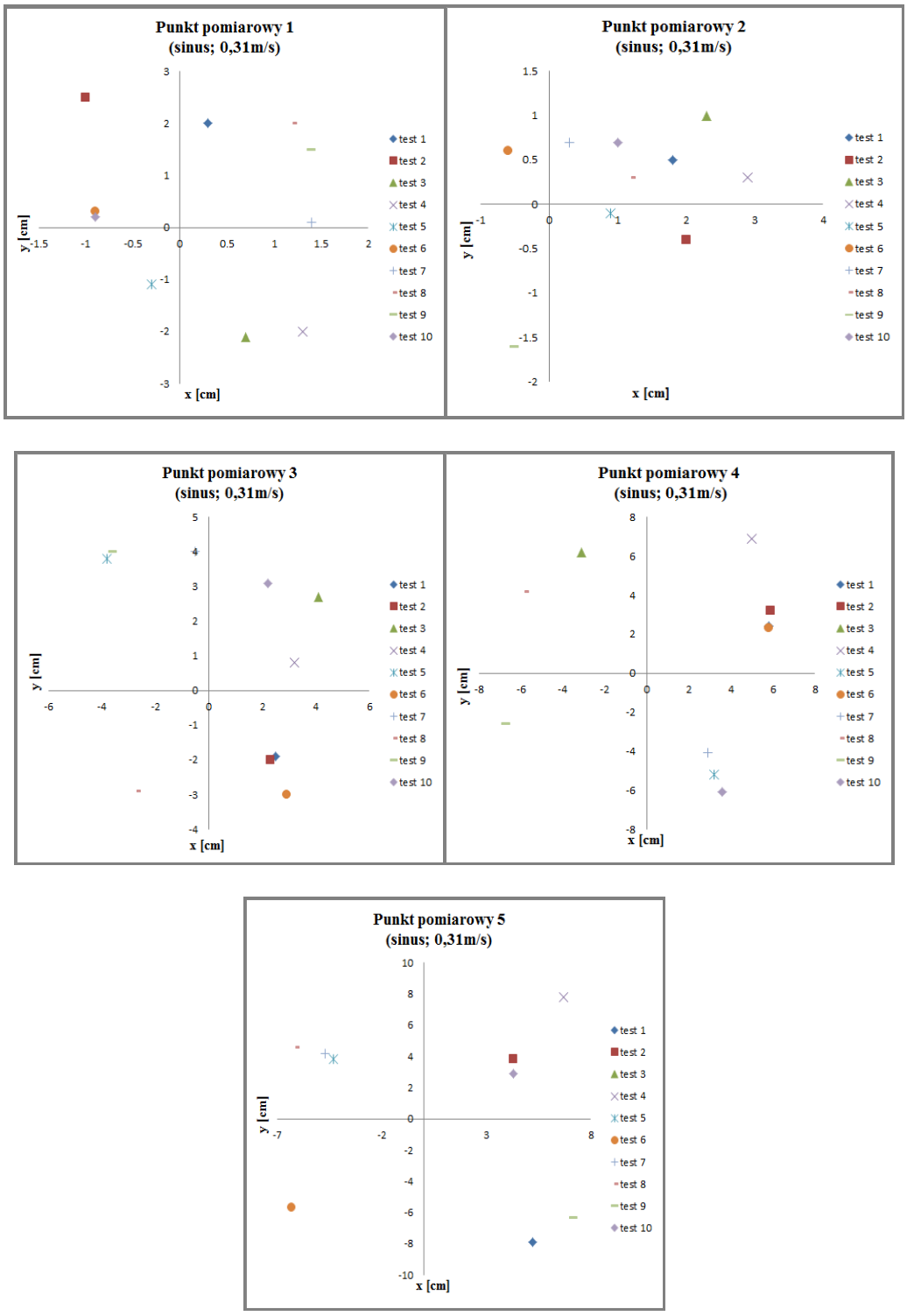
Rys. 6.10. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 1) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „sinus" przy prędkości 0,17 m/s



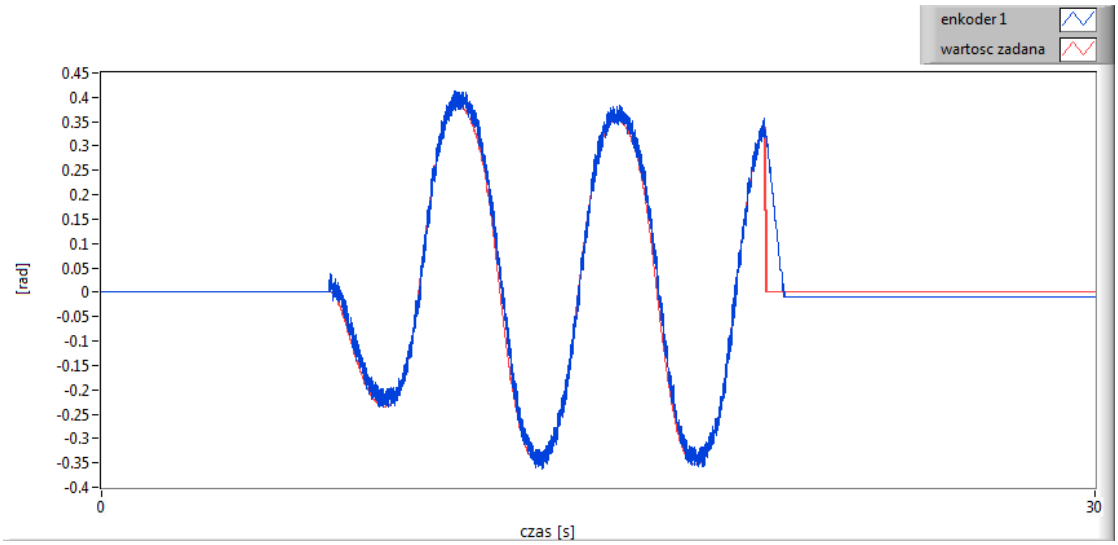
Rys. 6.11. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „sinus" przy prędkości 0,17 m/s

Tab. 6.5. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „sinus”. Prędkość przejazdu platformy 0,31 m/s

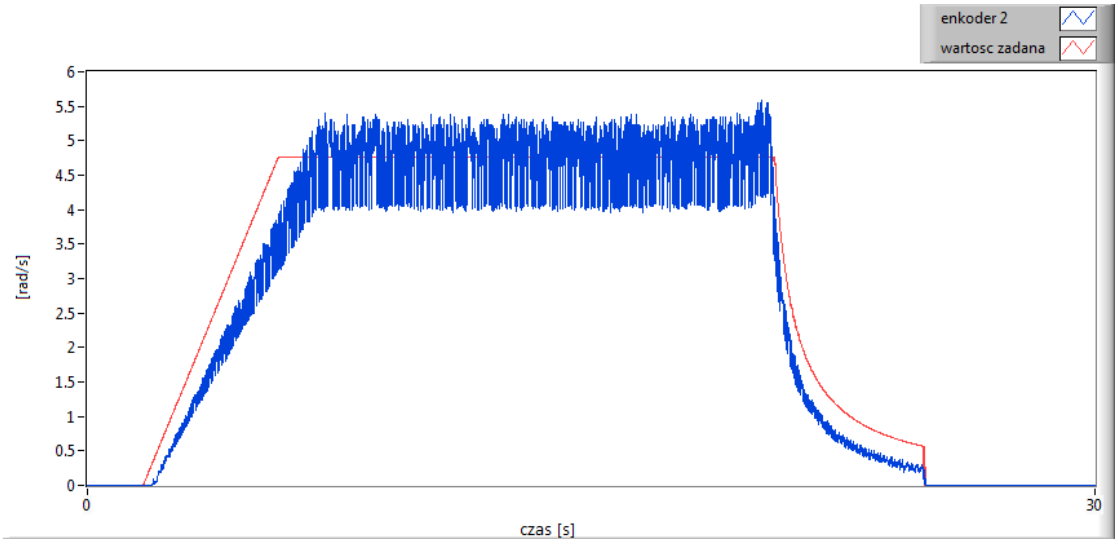
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4		PUNKT 5	
	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]
test 1	0,3	2	1,8	0,5	2,5	-1,9	5,8	2,4	5,2	-7,9
test 2	0,8	2,5	2	-0,4	2,3	-2	5,9	3,2	4,3	3,8
test 3	0,7	-2,1	2,3	1	4,1	2,7	-3,1	6,2	-8,2	-6,4
test 4	1,3	-2	2,9	0,3	3,2	0,8	5	6,9	6,7	7,8
test 5	-0,3	-1,1	0,9	-0,1	-3,8	3,8	3,2	-5,2	-4,3	3,8
test 6	-0,9	0,3	-0,6	0,6	2,9	-3	5,8	2,3	-6,3	-5,7
test 7	1	0,1	0,3	0,7	-0,5	4	2,9	-4,1	-4,7	4,2
test 8	1,2	2	1,2	0,3	-2,7	-2,9	-5,8	4,2	-6,1	4,6
test 9	1,4	1,5	-0,5	-1,6	-3,6	4	-6,7	-2,6	7,2	-6,3
test 10	-0,9	0,2	1	-0,9	2,2	3,1	3,6	-6,1	4,3	2,9
wartość min. (bezwzględna) [cm]	0,3	0,1	0,3	0,3	0,5	0,8	2,9	2,3	4,3	2,9
wartość średnia (bezwzględna) [cm]	0,88	1,38	1,35	0,64	2,78	2,82	4,78	4,32	5,73	5,34
wartość max. (bezwzględna) [cm]	1,4	2,5	2,9	1,6	4,1	4	6,7	6,9	8,2	7,9
wartość średnia dla trajektorii [cm]	3,104	2,9								



Rys. 6.12. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „sinus” przy prędkości 0,31 m/s dla poszczególnych punktów pomiarowych



Rys. 6.13. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „sinus" przy prędkości 0,31 m/s



Rys. 6.14. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – czerwony, wartość zamierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „sinus" przy prędkości 0,31 m/s.

Z uwagi na wymagający dla platformy mobilnej charakter krzywej typu „sinus", można zauważyć (choć nie dotyczy to wszystkich przypadków) generalną tendencję do nieznacznego wzrostu średniej wartości błędu wraz ze wzrostem długości przejazdu platformy po trajektorii. Wytłumaczenie tej zależności leży po pierwsze w sterowaniu silnie nieliniowym obiektem, a także dużą, o czym autor wspomniał, złożonością samej trajektorii oraz skończonej precyzji wykonania elementów mechanicznych. Należy podkreślić, że występujące odchyłki (wartość średnia błędu

dla całej trajektorii wyniosła odpowiednio Δx=2,594 cm i Δy=2,524 cm dla prędkości 0,17 m/s oraz Δx=3,104 cm i Δy=2,9 cm dla 0,31 m/s) są stosunkowe niewielkie w zestawieniu z długością pokonanej trasy, a także wielkością obiektu badań. Warto zwrócić uwagę na fakt występowania stosunkowo niewielkich różnic w wartościach błędów dla trasy pokonanej z prędkością mniejszą (tj. 0,17 m/s) oraz większą (tj. 0,31 m/s). Pomimo sterowania silnie nieliniowym obiektem na wymagającej trasie (częste zmiany kąta obrotu koła kierownicy mają znaczący wpływ na chwilowe wartości współczynników macierzy **M** oraz **L** rozpatrywanej platformy) platforma porusza się zgodnie z wcześniejszymi prognozami (tj. wirtualne prototypowanie oraz technika HILS), w których założono ruch platformy bez jakichkolwiek szarpnięć czy poślizgów. Dodatkowo, dla danego punktu pomiarowego obserwuje się dużą powtarzalność otrzymanych rezultatów.

Z zamieszczonych wykresów wartości uzyskanych dwóch enkoderów (pierwszego usytuowanego na kierownicy platformy i dokonującego pomiaru kąta obrotu kierownicy oraz drugiego będącego elementem wyposażenia układu napędu platformy, a dokonującego pomiaru prędkości silnika i koła zastępczego) w trakcie trwania eksperymentów wynika, że platforma poruszała się z pożądaną prędkością oraz koło kierownicy dokonywało skrętów w odpowiednim momencie z pożądanym kątem. Występujące różnice (wahania), tłumaczy się, o czym autor rozważał, faktem istnienia skończonej precyzji wykonania elementów mechanicznych (szczególnie kół, których ogumienie musiało zostać poddane przez autora dodatkowej obróbce mechanicznej), a także – bardzo dużą czułością zastosowanych enkoderów. Dodatkowo warto wspomnieć o działaniu samego sygnału sterującego oraz występującej nieliniowej macierzy bezwładności **M** układu platformy mobilnej. Wykreślone charakterystyki (rys. 6.11 oraz rys. 6.14) potwierdzają w sposób graficzny przejazd platformy bez poślizgów (brak gwałtownych zmian w odczytach stanu enkoderów). Dużą rolę w zapewnieniu płynności poruszania się platformy odgrywa precyzyjny układ przeniesienia napędu, bazujący na scalonym mechanizmie różnicowym. Powyższe stwierdzenia są obowiązujące dla wszystkich wykresów (dla rozważanych trajektorii), na których zamieszczono odczyty wartości enkoderów.

Z przeprowadzonych eksperymentów wynika, że nieznaczne błędy powstałe w trakcie przejazdu platformy po torze typu „sinus”, a także skuteczne sterowanie dla prędkości niemalże dwukrotnie większej, niż wynikało to z wartości dla której układ sterowania został zaprojektowany, wskazują na wysoce skuteczną technikę

nadzorowania przy energetycznym wskaźniku jakości dla obiektów również silnie nieliniowych, a także potwierdzają poprawność stosowania technik projektowania mechatronicznego.

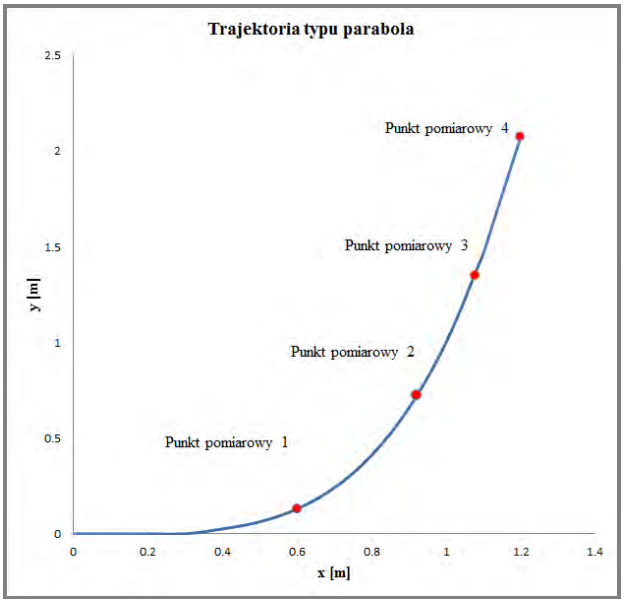
6.5. Sterowanie optymalne przy energetycznym wskaźniku jakości dla trajektorii typu „parabola”

Wykazanie skuteczności sterowania optymalnego przy energetycznym wskaźniku jakości, a także – potwierdzenie poprawności zastosowanych technik projektowania mechatronicznego zostało przeprowadzone drogą eksperymentalną dla drugiej rozpatrywanej trajektorii, tj. dla „paraboli”. Trasa przejazdu platformy mobilnej została zademonstrowana na rys. 6.15. Na zilustrowaną trasę przejazdu naniesiono również punkty pomiarowe, w których dokonano pomiaru odchylenia toru zrealizowanego od zadanego. Podobnie, jak to miało miejsce dla trajektorii typu „sinus”, autor wykonał dwie serie testów tj. dla prędkości 0,17 m/s dla której przeprowadzono projektowanie systemu, oraz dla prędkości 0,31 m/s. Wyniki przeprowadzonych eksperymentów zestawiono (dla każdego punktu pomiarowego dokonano 10-ciu przejazdów) w tab. 6.6 oraz tab. 6.7, a także – przedstawiano w postaci graficznej na rys. 6.16. oraz rys. 6.19.

Na rys. 6.17, 6.18, 6.20 oraz 6.21. wykreślono przebiegi wartości chwilowych z enkoderów, zarejestrowanych podczas przejazdu platformy po omawianej trasie. Z zamieszczonych przebiegów wynika, że charakter pracy układu napędowego podczas przejazdu po krzywej typu „parabola” jest tożsamy z identycznym przebiegiem dla trajektorii typu „sinus”. Można wyraźnie wyróżnić fazy rozpędzania, przejazdu oraz hamowania. Dostrzegalne chwilowe różnice notuje się natomiast dla koła kierowniczego. Należy zauważyć, że w przypadku układów złożonych (sterowaniu podlegają dwa silniki), kombinacja wielu wartości (m.in. położenia, prędkości oraz przyspieszenia), a nie – ich poszczególnych składowych, będą determinowały jakość sterowania, wynikającą z minimalizacji energii pomiędzy trajektorią założoną a realizowaną w danej chwili czasu t. Dlatego, w tym przypadku za ocenę jakości sterowania przyjmuje się efekt końcowy (powstające błędy położenia platformy, które w tym przypadku są niewielkie).

Analiza przeprowadzonych eksperymentów dowodzi prawdziwości postawionej tezy pracy o możliwości sterowania obiektem nieliniowym przy

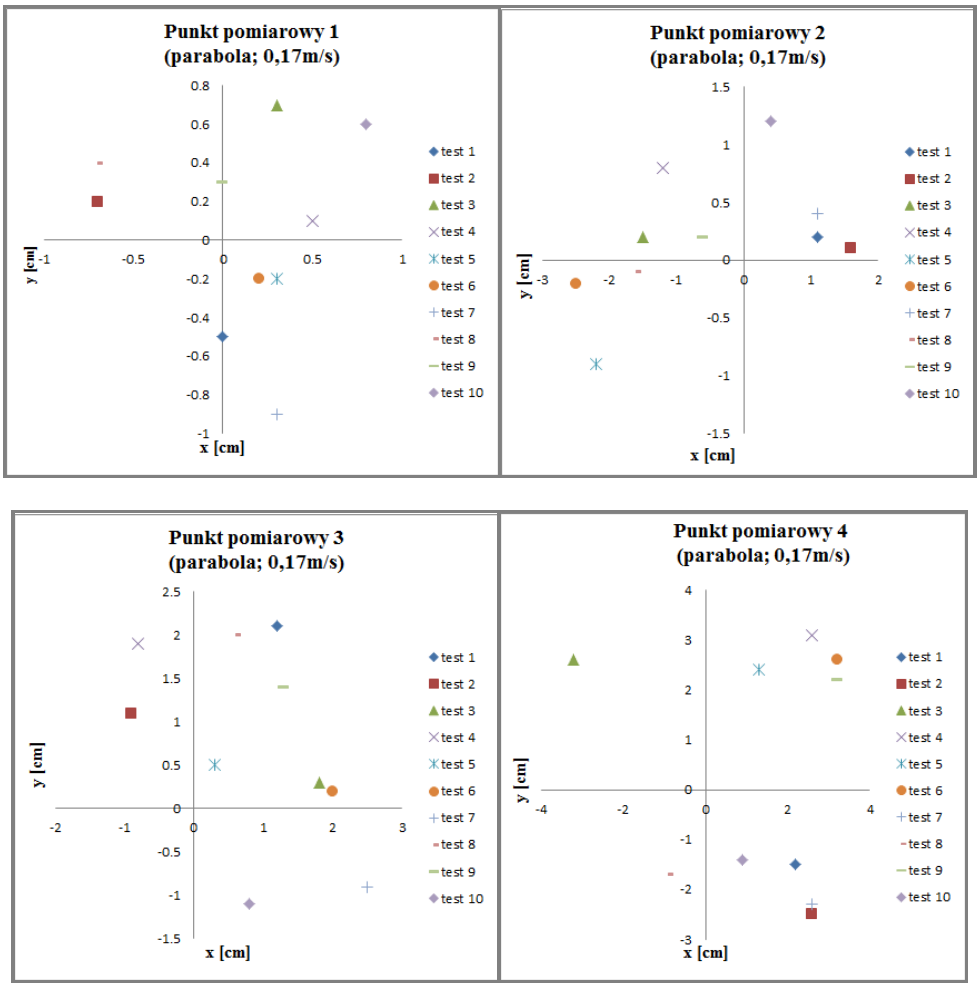
energetycznym wskaźniku jakości. Dla dwóch rozpatrywanych prędkości poruszania się platformy, przeprowadzono 40 prób przejazdu, gdzie wartość średnia błędu dla całej trajektorii wyniosła odpowiednio Δx=1,3125 cm i Δy=1,055 cm dla prędkości 0,17 m/s oraz Δx=1,76 cm i Δy=1,6975 cm dla 0,31 m/s. Podobnie, jak to miało miejsce dla trajektorii typu „sinus", tak i w tym przypadku wydłużenie trasy przejazdu jest przyczyną wzrostu średniego błędu dla danego punktu pomiarowego. Zważywszy na złożoność mechaniczną obiektu oraz układu sterowania, oszacowany rezultat predestynuje do sformułowania, co jest podtrzymaniem wniosków wynikających z eksperymentów dla trajektorii typu „sinus", twierdzenia o przydatności proponowanej metody do sterowania obiektami nieliniowymi, a także wykorzystanych technik projektowania mechatronicznego.



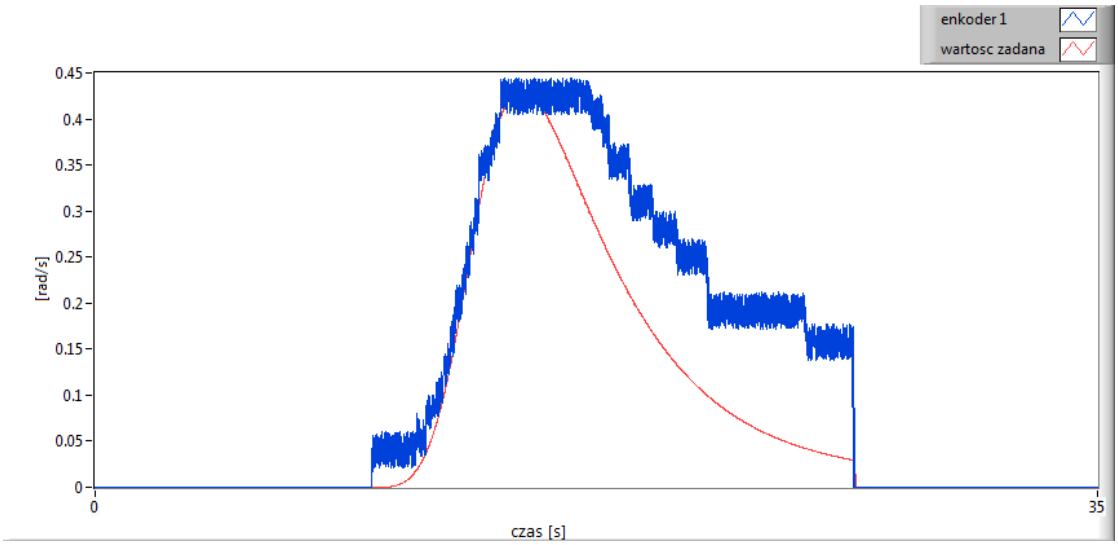
Rys. 6.15. Zadana trajektoria ruchu platformy mobilnej typu „parabola" (punktu charakterystycznego) wraz z naniesionymi punktami pomiarowymi

Tab. 6.6. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „parabola". Prędkość przejazdu platformy 0,17 m/s.

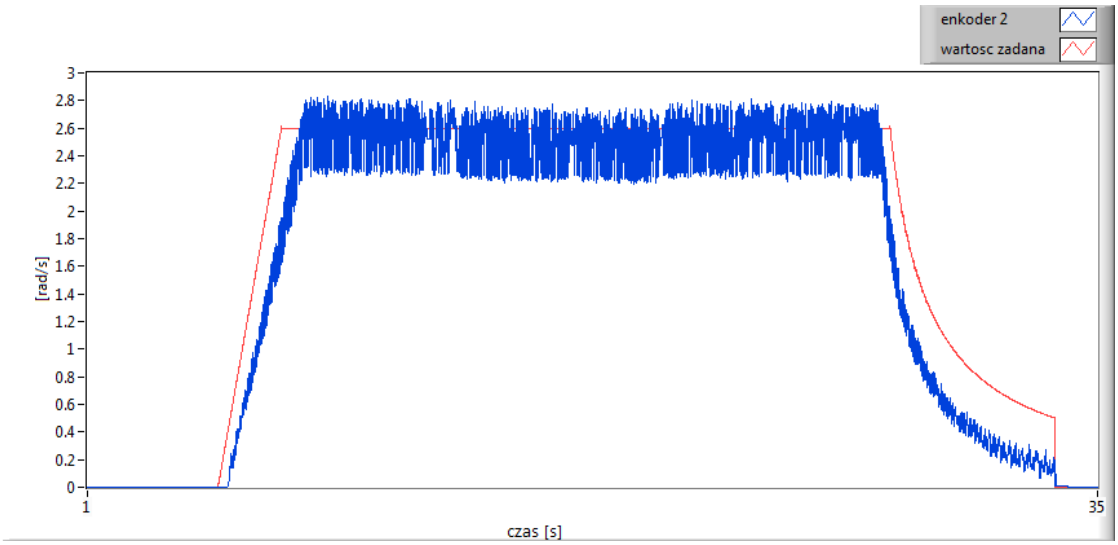
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4	
	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]
test 1	0	-0,5	1,1	0,2	1,2	2,1	2,2	-1,5
test 2	-0,7	0,2	1,6	0,1	-0,9	1,1	2,6	-2,5
test 3	0,3	0,7	-1,5	0,2	1,8	0,3	-3,2	2,6
test 4	0,5	0,1	-1,2	0,8	-0,8	1,9	2,6	3,1
test 5	0,3	-0,2	-2,2	-0,9	0,3	0,5	1,3	2,4
test 6	0,2	-0,2	-2,5	-0,2	2	0,2	3,2	2,6
test 7	0,3	-0,9	1,1	0,4	2,5	-0,9	2,6	-2,3
test 8	-0,7	0,4	-1,6	-0,1	0,6	2	-0,9	-1,7
test 9	0	0,3	-0,6	0,2	1,3	1,4	3,2	2,2
test 10	0,8	0,6	0,4	1,2	0,8	-1,1	0,9	-1,4
wartość min. (bezwzględna) [cm]	-0,7	-0,9	-2,5	-0,9	-0,9	-1,1	-3,2	-2,5
wartość średnia (bezwzględna) [cm]	0,38	0,41	1,38	0,43	1,22	1,15	2,27	2,23
wartość max. (bezwzględna) [cm]	0,8	0,7	1,6	1,2	2,5	2,1	3,2	3,1
wartość średnia dla trajektorii [cm]	1,3125	1,055						



Rys. 6.16. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „parabola" przy prędkości 0,17 m/s dla poszczególnych punktów pomiarowych



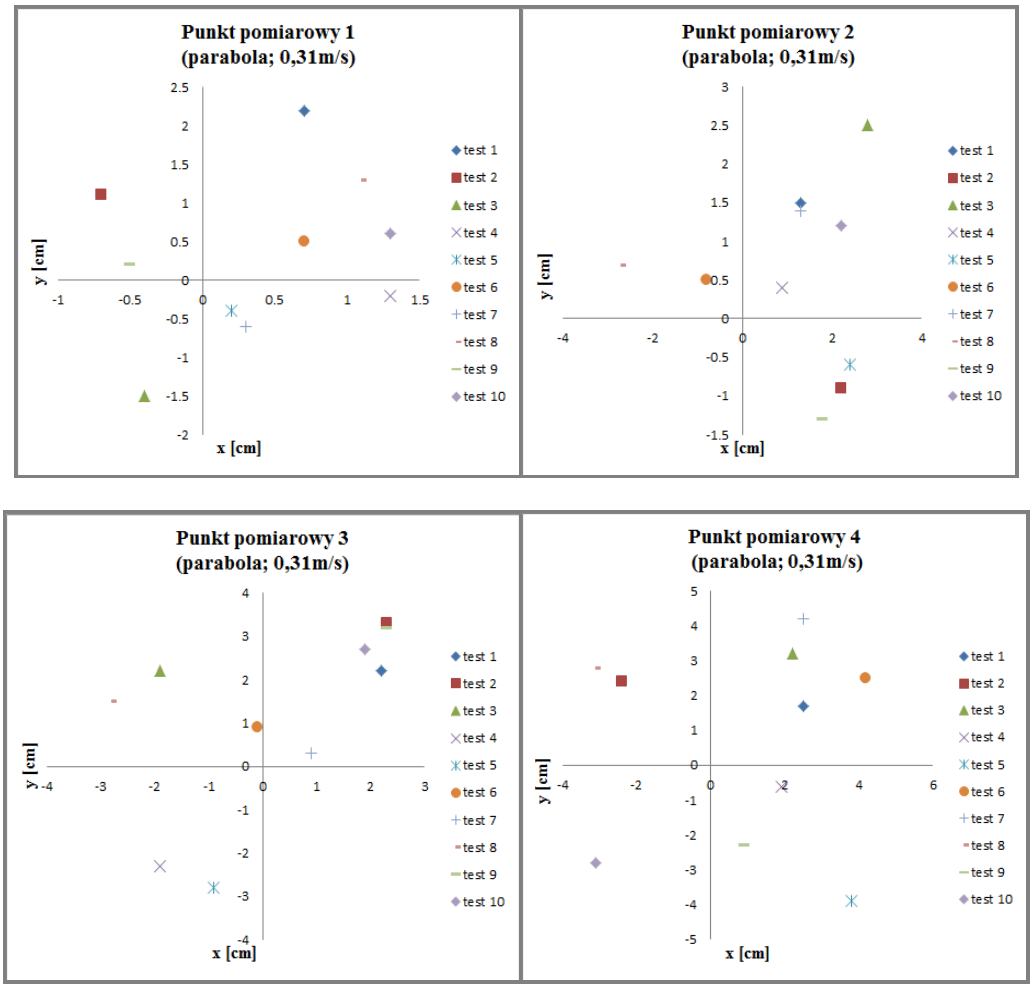
Rys. 6.17. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „parabola" przy prędkości 0,17 m/s



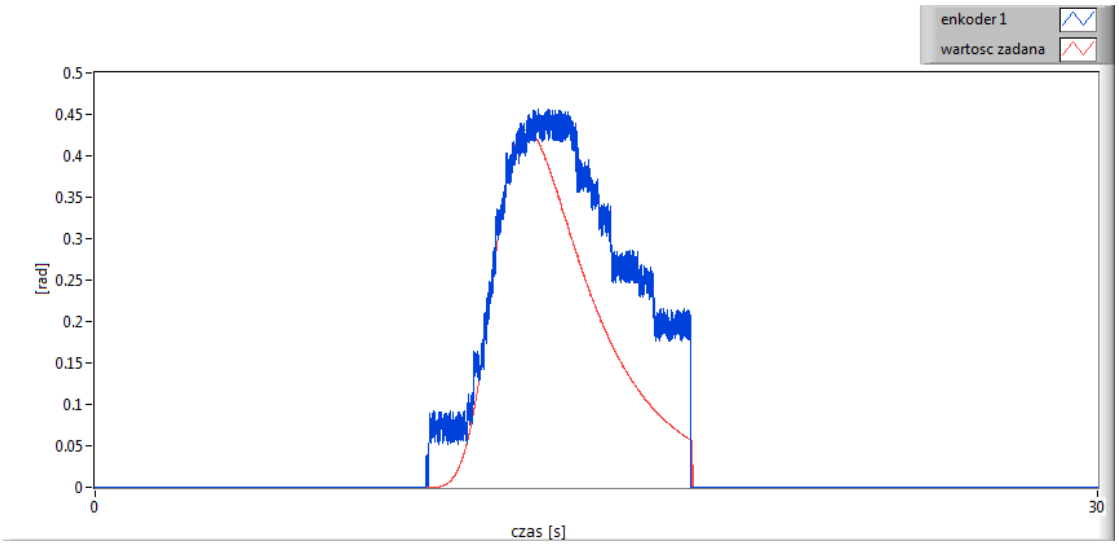
Rys. 6.18. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „parabola" przy prędkości 0,17 m/s

Tab. 6.7. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „parabola”. Prędkość przejazdu platformy 0,31 m/s

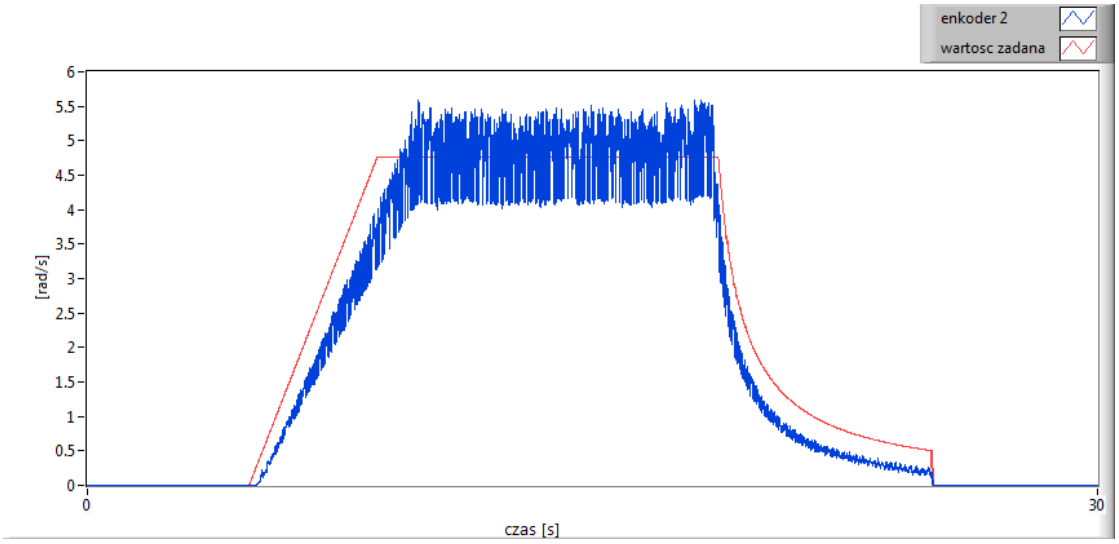
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4	
	x [cm]	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**
test 1	0,7	2,2	1,3	1,5	2,2	2,2	2,5	1,7
test 2	0,8	1,1	2,2	-0,9	2,3	3,3	-2,4	2,4
test 3	-0,4	-1,5	2,8	2,5	-1,9	2,2	2,2	3,2
test 4	1,3	-0,2	0,9	0,4	-1,9	-2,3	1,9	-0,6
test 5	0,2	-0,4	2,4	-0,6	-0,9	-2,8	3,8	-3,9
test 6	0,7	0,5	-0,8	0,5	-0,1	0,9	4,2	2,5
test 7	1,2	-0,6	1,3	1,4	0,9	0,3	2,5	4,2
test 8	1,1	1,3	-2,7	0,7	-2,8	1,5	-3,1	2,8
test 9	-0,5	0,2	1,8	-1,3	2,3	3,2	0,9	-2,3
test 10	1,3	0,6	2,2	1,7	1,9	2,7	-3,1	-2,8
wartość min. (bezwzględna) [cm]	-0,5	-1,5	-2,7	-1,3	-2,8	-2,8	-3,1	-3,9
wartość średnia (bezwzględna) [cm]	0,82	0,86	1,84	1,15	1,72	2,14	2,66	2,64
wartość max. (bezwzględna) [cm]	1,3	2,2	2,8	2,5	2,3	3,3	4,2	4,2
wartość średnia dla trajektorii [cm]	1,76	1,6975						



Rys. 6.19. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „parabola" przy prędkości 0,31 m/s dla poszczególnych punktów pomiarowych



Rys. 6.20. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „parabola" przy prędkości 0,31 m/s



Rys. 6.21. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski podczas przejazdu platformy mobilnej po trajektorii typu „parabola" przy prędkości 0,31 m/s

6.6. Sterowanie optymalne przy energetycznym wskaźniku jakości dla trajektorii typu „okrąg”

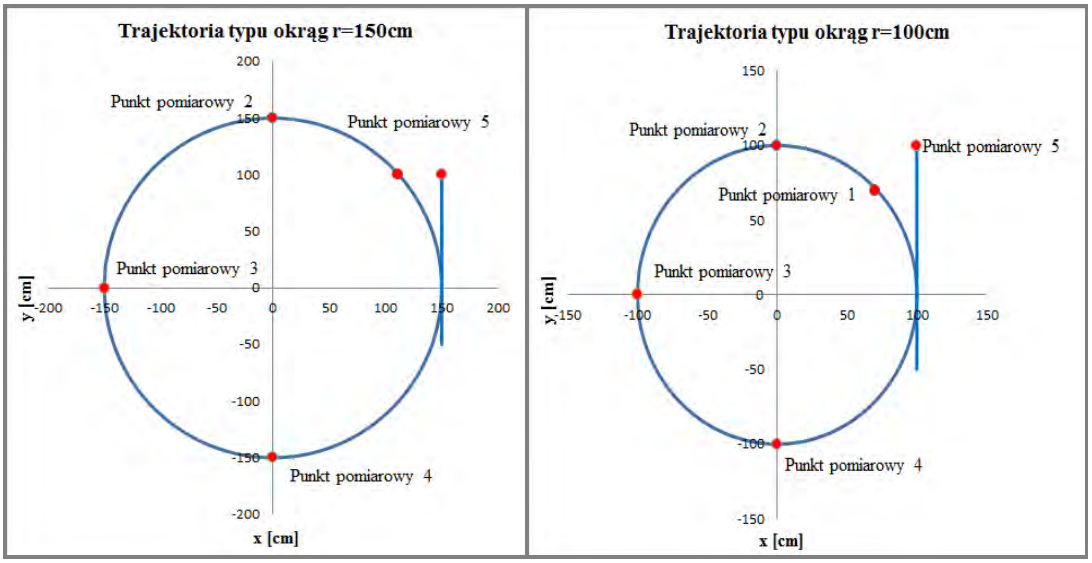
Weryfikacja tezy pracy o skuteczności sterowania optymalnego przy energetycznym wskaźniku jakości, a także wykazanie zalet stosowania technik projektowania mechatronicznego, zostały przeprowadzone drogą eksperymentalną również dla trajektorii typu „okrąg”. Warunki przeprowadzonych badań eksperymentalnych zmusiły autora do dokonania modyfikacji kształtu trajektorii (zmniejszono promień okręgu). Dodatkowo, celem potwierdzenia elastyczności stosowania przyjętej koncepcji sterowania, jako parametr autor (oprócz dwóch prędkości przejazdu) przyjął promień okręgu. W sumie badania zostały wykonane dla dwóch trajektorii (promień 1,5 m oraz 1 m) oraz dla dwóch prędkości przejazdu platformy. Trasy przejazdu platformy wraz z punktami, w których dokonano pomiarów położenia (błędów) platformy, zostały zilustrowane na rys. 6.22. Rezultaty przeprowadzonych badań zamieszczono w tab. 6.8, 6.9, 6.10 oraz 6.11, a ich graficzną interpretację wykreślono na rys. 6.23, 6.26, 6.29 oraz 6.32.

Analiza otrzymanych rezultatów zarówno dla okręgu o promieniu 1 m jak i o promieniu 1,5 m dla obu prędkości przejazdu, potwierdza wcześniejsze prognozy o możliwości realizacji systemu nadzorowania ruchu w układzie silnie nieliniowym za pomocą omawianej metody sterowania optymalnego przy energetycznym wskaźniku jakości.

Większa prędkość przejazdu (tj. 0,31 m/s), a także – dłuższa trasa przejazdu (promień 1,5 m) intensyfikuje (w niewielkim stopniu) średni błąd położenia platformy. Dla dwóch rozpatrywanych promieni okręgu omawianej trajektorii w trakcie 40 prób przejazdu, średnia wartość błędu dla całej trajektorii wyniosła: dla r=1,5 m: Δx=1,62 cm i Δy=1,55 cm przy prędkości 0,17 m/s oraz Δx=3,06 cm i Δy=3.51 cm przy prędkości 0,31 m/s; dla r=1,0 m: Δx=1,65 cm i Δy=1,51 cm przy prędkości 0,17 m/s oraz Δx=1,91 cm i Δy=1,99 cm przy prędkości 0,31 m/s.

Pomimo kompensacyjnego charakteru stosowanego algorytmu (minimalizacja energii) system sterowania na skutek zawiłego matematycznie modelu platformy (w tym również skończonej precyzji wykonania elementów mechanicznych), a także na skutek wymogu realizacji systemu Real Time pracuje z błędem wynikającym z ustalenia kroku całkowania Δt=0,005 s tak, aby spełnić warunek determinizmu czasowego. Rekompensatą tego faktu jest osiągnięcie (drogą optymalizacji)

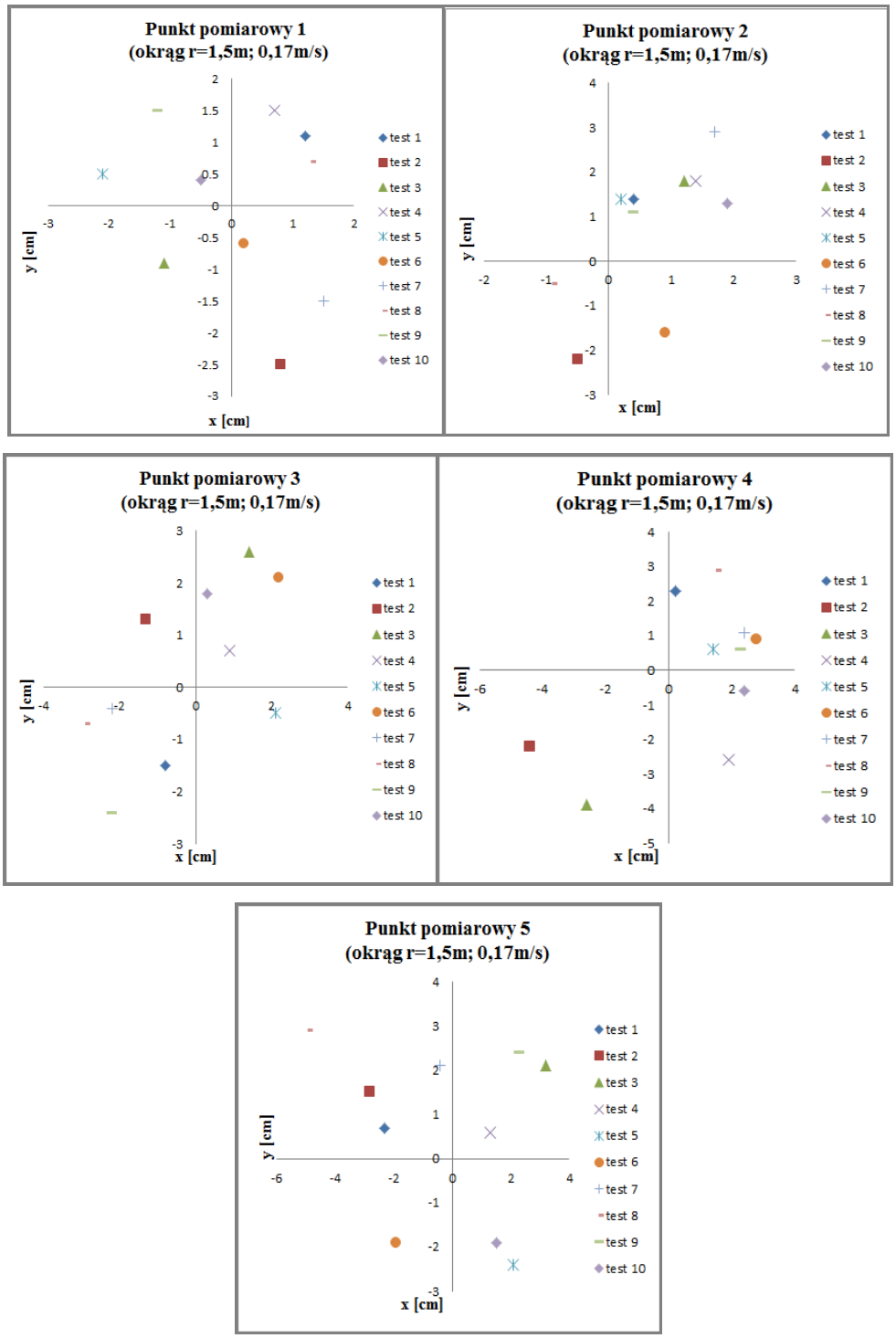
determinizmu czasowego oraz zachowania płynności ruchu przejazdu. Podobnie, jak to miało miejsce dla omawianych powyżej trajektorii, tak i w tym przypadku przeprowadzona analiza otrzymanych przebiegów wartości chwilowych enkoderów (rys. 6.24, 6.25, 6.27, 6.28, 6.30, 6.31, 6.33 oraz 6.34.) potwierdza brak jakichkolwiek oznak wystąpienia poślizgów, czy utraty stabilności ruchu platformy.



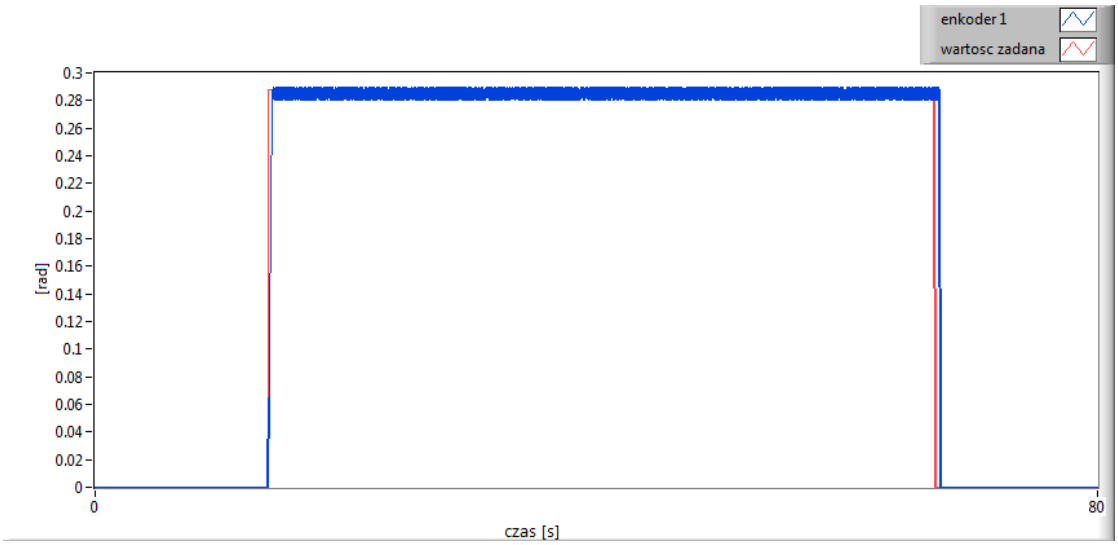
Rys. 6.22. Zadana trajektoria ruchu platformy mobilnej typu „okrąg" (punktu charakterystycznego) wraz z naniesionymi punktami pomiarowymi. Wykres lewy r=1,5 m. Wykres prawy r=1,0 m

Tab. 6.8. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „okrąg" r=1,5 m. Prędkość przejazdu platformy 0,17 m/s

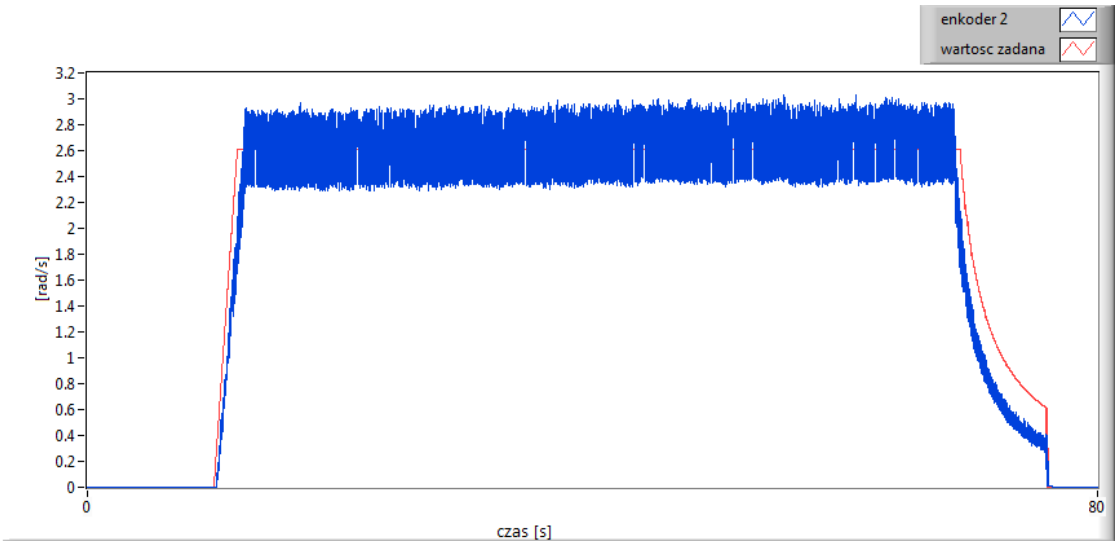
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4		PUNKT 5	
	x [cm]	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**
test 1	1,2	1,1	0,4	1,4	-0,8	-1,5	0,2	2,3	-2,3	0,7
test 2	0,8	-2,5	-0,5	-2,2	-1,3	1,3	-4,4	-2,2	-2,8	1,5
test 3	-1,1	-0,9	1,2	1,8	1,4	2,6	-2,6	-3,9	3,2	2,1
test 4	0,7	1,5	1,4	1,8	0,9	0,7	1,9	-2,6	1,3	0,6
test 5	-2,1	0,5	0,2	1,4	2,1	-0,5	1,4	0,6	2,1	-2,4
test 6	0,2	-0,6	0,9	-1,6	2,2	2,1	2,8	0,9	-1,9	-1,9
test 7	1,5	-1,5	1,7	2,9	-2,2	-0,4	2,4	1,1	-0,4	2,1
test 8	1,3	0,7	-0,9	-0,5	-2,9	-0,7	1,5	2,9	-4,9	2,9
test 9	-1,2	1,5	0,4	1,1	-2,2	-2,4	2,3	0,6	2,3	2,4
test 10	-0,5	0,4	1,9	1,3	0,3	1,8	2,4	-0,6	1,5	-1,9
wartość min. (bezwzględna) [cm]	0,2	0,3	0,4	0,2	0,5	0,2	2,8	2	3	1,2
wartość średnia (bezwzględna) [cm]	1,06	1,12	0,95	1,6	1,63	1,4	2,19	1,77	2,27	1,85
wartość max. (bezwzględna) [cm]	1,5	1,7	2,1	1,2	2,2	5,9	5,9	6,8	3,2	8,2
wartość średnia dla trajektorii [cm]	1,62	1,55								



Rys. 6.23. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,5 m, przy prędkości 0,17 m/s dla poszczególnych punktów pomiarowych



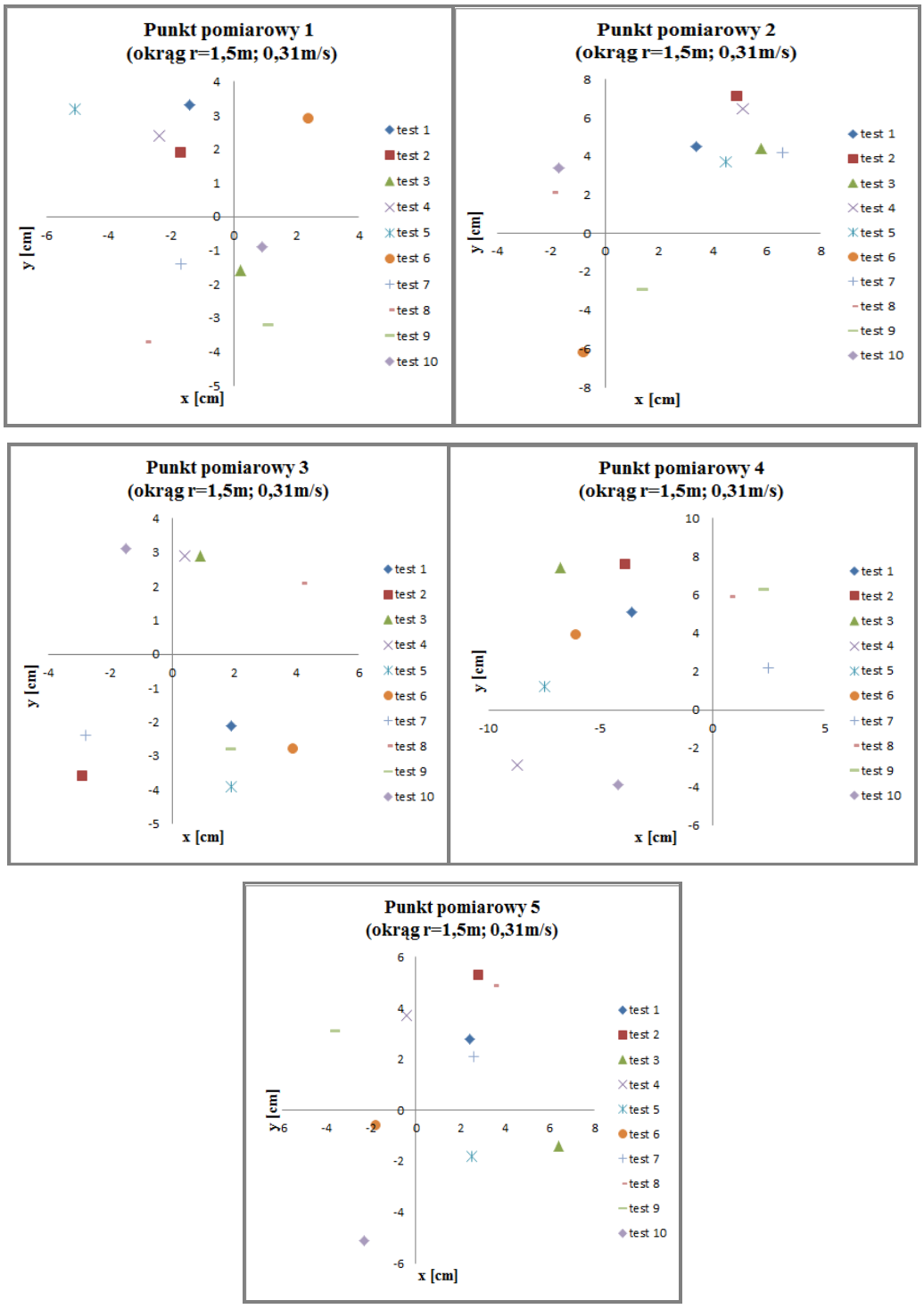
Rys. 6.24. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,50 m przy prędkości 0,17 m/s



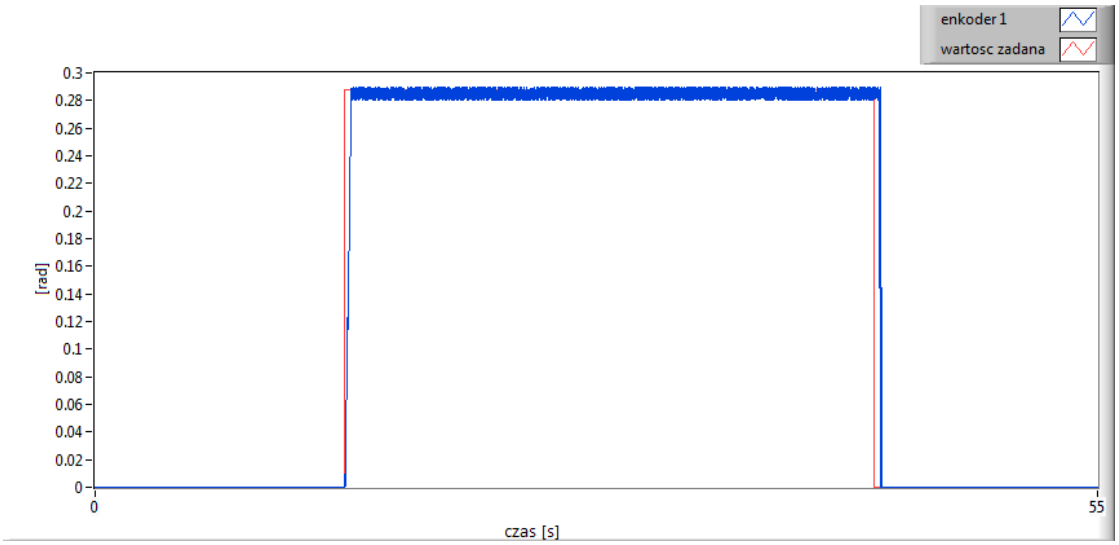
Rys. 6.25. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,50 m przy prędkości 0,17 m/s

Tab. 6.9. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „okrąg” r=1,50 m. Prędkość przejazdu platformy 0,31 m/s

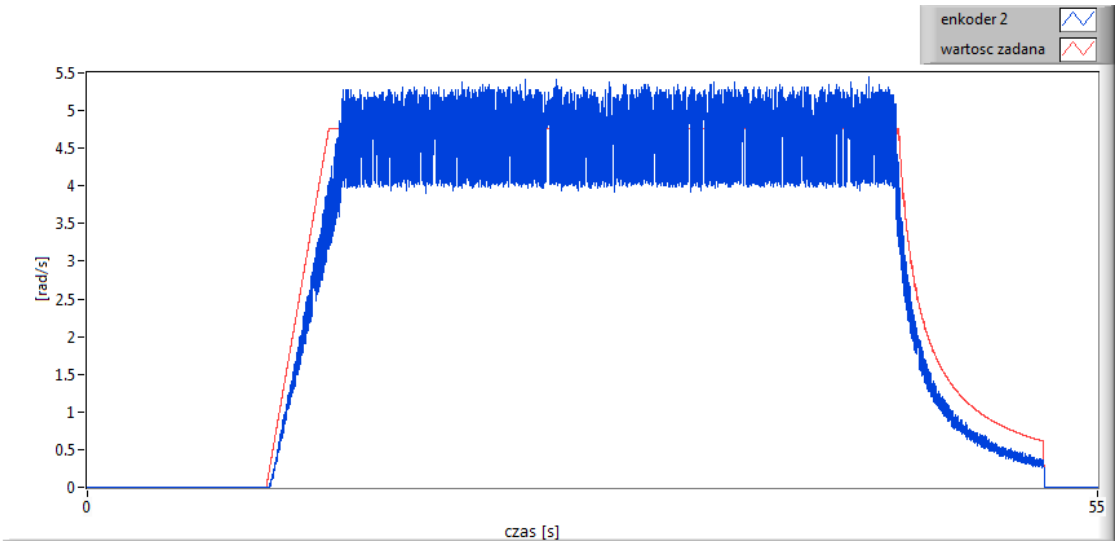
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4		PUNKT 5	
	x [cm]	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**
test 1	-1,4	3,3	3,4	4,5	1,9	-2,1	-3,6	5,1	2,4	2,8
test 2	-1,7	1,9	4,9	7,1	-2,9	-3,6	-3,9	7,6	2,8	5,3
test 3	0,2	-1,6	5,8	4,4	0,9	2,9	-6,8	7,4	6,4	-1,4
test 4	-2,4	2,4	5,1	6,5	0,4	2,9	-8,7	-2,9	-0,4	3,7
test 5	-5,1	3,2	4,5	3,7	1,9	-3,9	-7,5	1,2	2,5	-1,8
test 6	2,4	2,9	-0,8	-6,2	3,9	-2,8	-6,1	3,9	-1,8	-0,6
test 7	-1,7	-1,4	6,6	4,2	-2,8	-2,4	2,5	2,2	2,6	2,1
test 8	-2,8	-3,7	-1,9	2,1	4,2	2,1	0,8	5,9	3,5	4,9
test 9	1,1	-3,2	1,4	-2,9	1,9	-2,8	2,3	6,3	-3,6	3,1
test 10	0,9	-0,9	-1,7	3,4	-1,5	3,1	-4,2	-3,9	-2,3	-5,1
wartość min. (bezwzględna) [cm]	0,2	0,3	0,4	0,2	0,5	0,2	2,8	2	3	1,2
wartość średnia (bezwzględna) [cm]	1,97	2,45	3,61	4,5	2,23	2,86	4,64	4,64	2,83	3,08
wartość max. (bezwzględna) [cm]	2,4	1,7	2,1	1,2	4,2	5,9	5,9	6,8	6,4	8,2
wartość średnia dla trajektorii [cm]	3,06	3,51								



Rys. 6.26. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r= r=1,50 m, przy prędkości 0,31m/s dla poszczególnych punktów pomiarowych



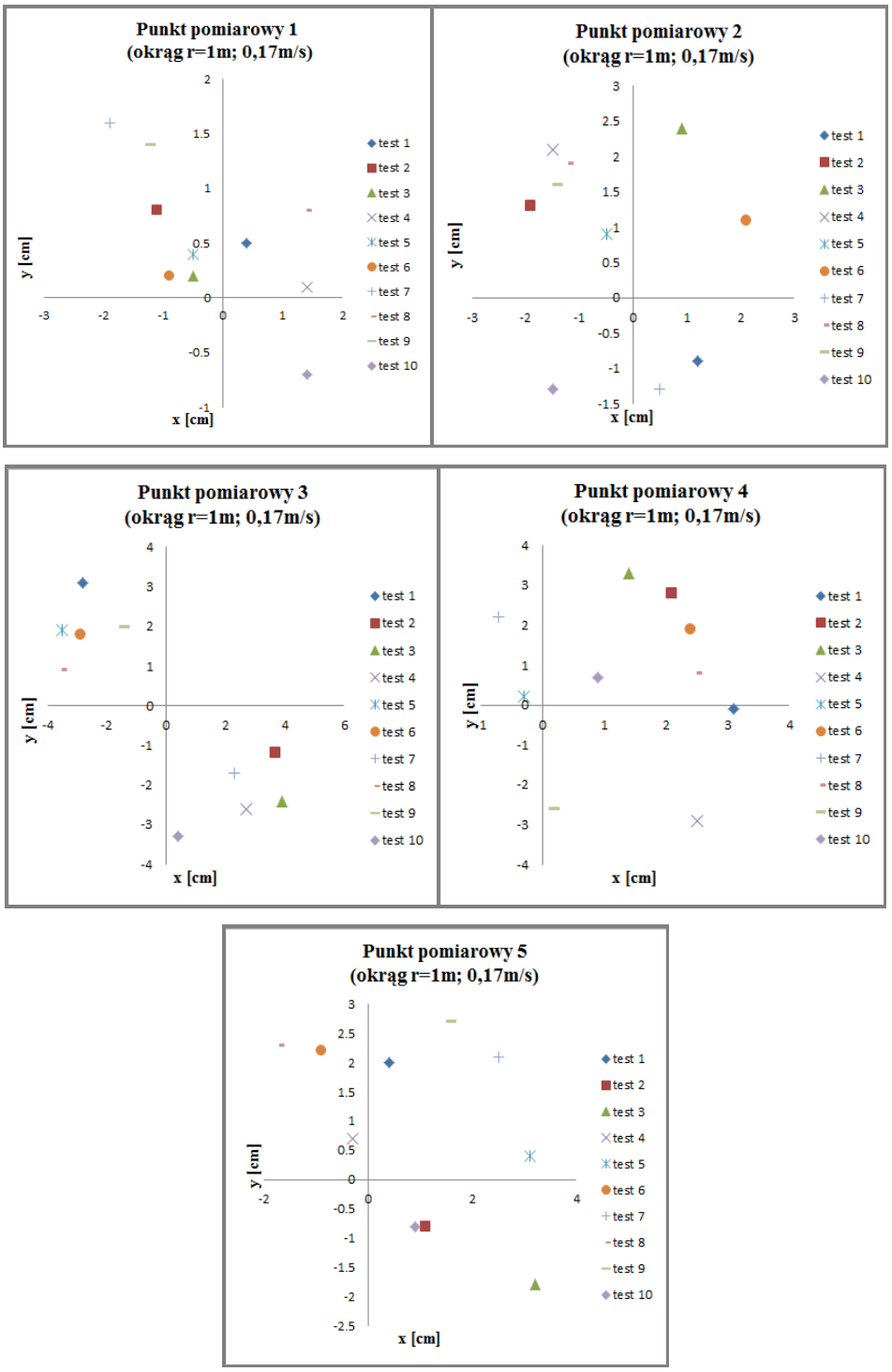
Rys. 6.27. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,50 m przy prędkości 0,31 m/s



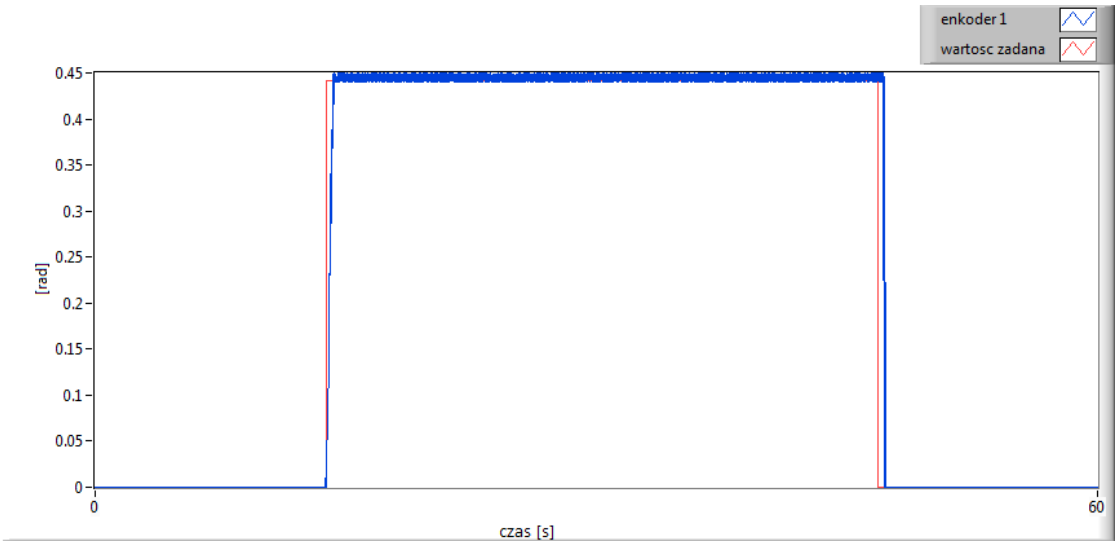
Rys. 6.28. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,50 m przy prędkości 0,31 m/s

Tab. 6.10. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „okrąg” r=1,00 m. Prędkość przejazdu platformy 0,17 m/s.

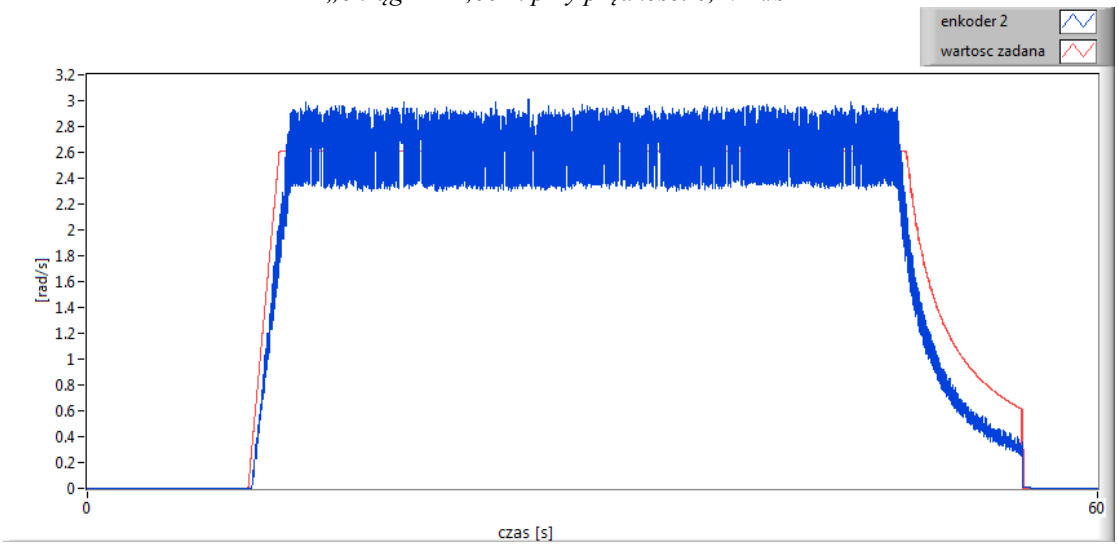
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4		PUNKT 5	
	x [cm]	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**	**x [cm]**	**y [cm]**
test 1	0,4	0,5	1,2	-0,9	-2,8	3,1	3,1	-0,1	0,4	2
test 2	-1,1	0,8	-1,9	1,3	3,7	-1,2	2,1	2,8	1,1	-0,8
test 3	-0,5	0,2	0,9	2,4	3,9	-2,4	1,4	3,3	3,2	-1,8
test 4	1,4	0,1	-1,5	2,1	2,7	-2,6	2,5	-2,9	-0,3	0,7
test 5	-0,5	0,4	-0,5	0,9	-3,5	1,9	-0,3	0,2	3,1	0,4
test 6	-0,9	0,2	2,1	1,1	-2,9	1,8	2,4	1,9	-0,9	2,2
test 7	-1,9	1,6	0,5	-1,3	2,3	-1,7	-0,7	2,2	2,5	2,1
test 8	1,4	0,8	-1,2	1,9	-3,5	0,9	2,5	0,8	-1,7	2,3
test 9	-1,2	1,4	-1,4	1,6	-1,4	2	0,2	-2,6	1,6	2,7
test 10	1,4	-0,7	-1,5	-1,3	0,4	-3,3	0,9	0,7	0,9	-0,8
wartość min. (bezwzględna) [cm]	0,2	0,3	0,4	0,2	0,5	0,2	2,8	2	3	1,2
wartość średnia (bezwzględna) [cm]	1,07	0,67	1,27	1,48	2,71	2,09	1,61	1,75	1,57	1,58
wartość max. (bezwzględna) [cm]	1,4	1,7	2,1	1,2	3,9	5,9	5,9	6,8	3,2	8,2
wartość średnia dla trajektorii [cm]	1,65	1,51								



Rys. 6.29. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,00 m, przy prędkości 0,17 m/s dla poszczególnych punktów pomiarowych



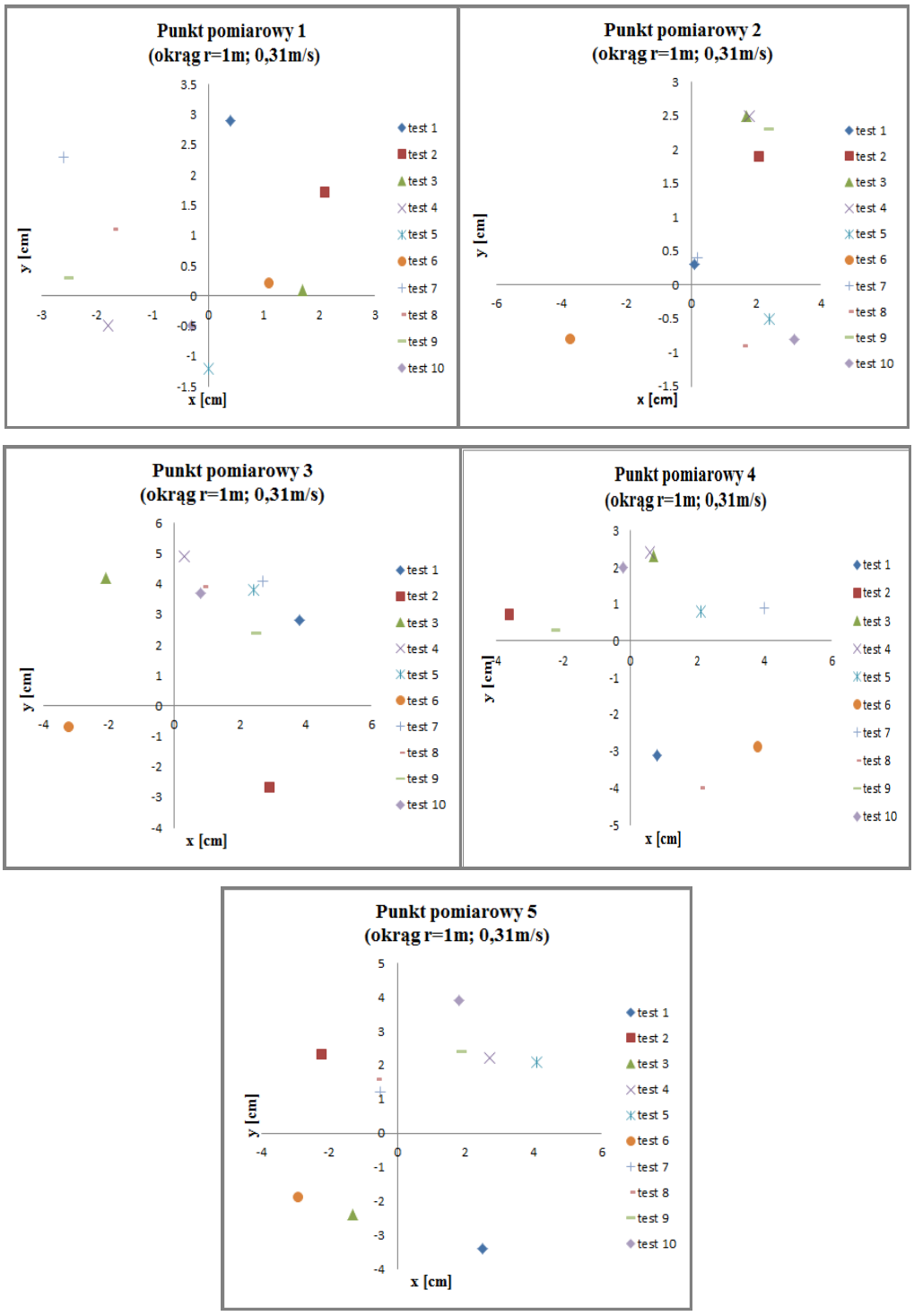
Rys. 6.30. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,00 m przy prędkości 0,17 m/s



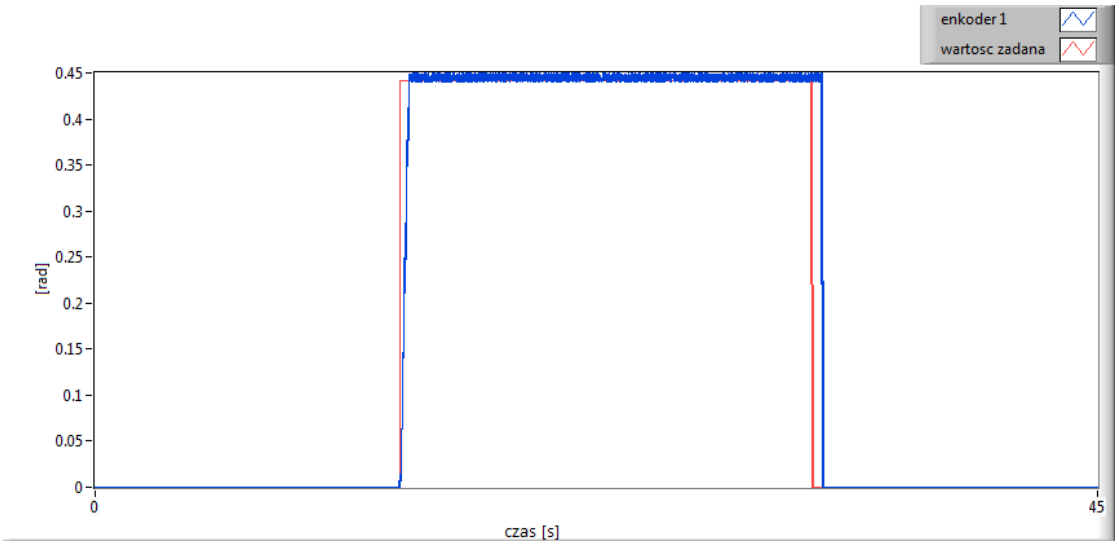
Rys. 6.31. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,00 m przy prędkości 0,17 m/s

Tab. 6.11. Wartości błędów sterowania podczas ruchu platformy mobilnej po trajektorii typu „okrąg” r=1,00 m. Prędkość przejazdu platformy 0,31 m/s

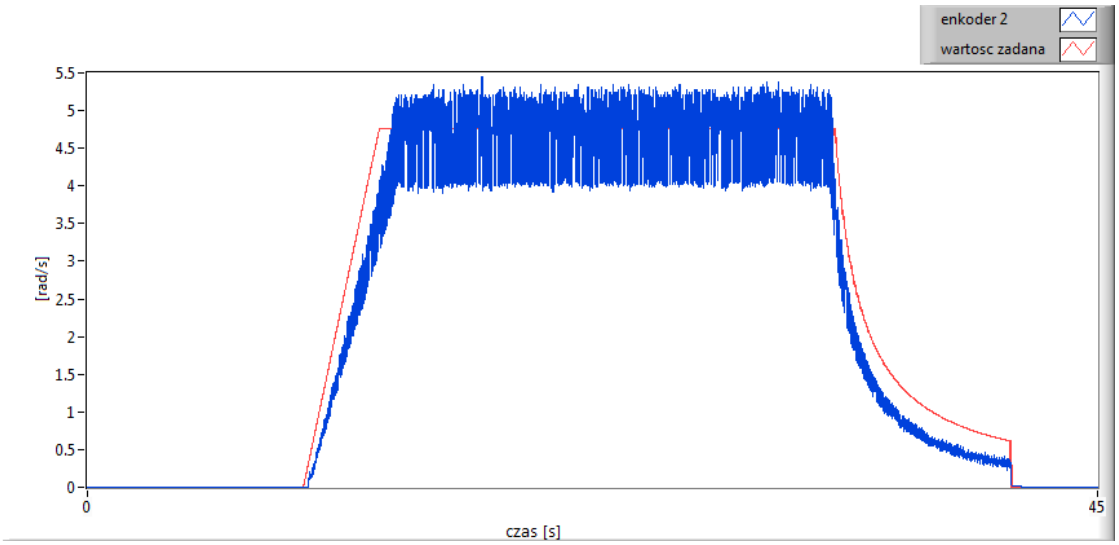
TEST	PUNKT 1		PUNKT 2		PUNKT 3		PUNKT 4		PUNKT 5	
	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]	x [cm]	y [cm]
test 1	0,4	2,9	0,1	0,3	3,8	2,8	0,8	-3,1	2,5	-3,4
test 2	2,1	1,7	2,1	1,9	2,9	-2,7	-3,6	0,7	-2,2	2,3
test 3	1,7	0,1	1,7	2,5	-2,1	4,2	0,7	2,3	-1,3	-2,4
test 4	-1,8	-0,5	1,8	2,5	0,3	4,9	0,6	2,4	2,7	2,2
test 5	0	-1,2	2,4	-0,5	2,4	3,8	2,1	0,8	4,1	2,1
test 6	1,1	0,2	-3,7	-0,8	-3,2	-0,7	3,8	-2,9	-2,9	-1,9
test 7	-2,6	2,3	0,2	0,4	2,7	4,1	4	0,9	-0,5	1,2
test 8	-1,7	1,1	1,6	-0,9	0,9	3,9	2,1	-4	-0,6	1,6
test 9	-2,5	0,3	2,4	2,3	2,5	2,4	-2,2	0,3	1,9	2,4
test 10	-0,3	-0,5	3,2	-0,8	0,8	3,7	-0,2	2	1,8	3,9
wartość min. (bezwzględna) [cm]	0,2	0,3	0,4	0,2	0,5	0,2	2,8	2	3	1,2
wartość średnia (bezwzględna) [cm]	1,42	1,08	1,92	1,29	2,16	3,32	2,01	1,94	2,05	2,34
wartość max. (bezwzględna) [cm]	2,1	1,7	2,1	1,2	3,8	5,9	5,9	6,8	4,1	8,2
wartość średnia dla trajektorii [cm]	1,91	1,99								



Rys. 6.32. Błędy położenia platformy mobilnej podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,00 m, przy prędkości 0,31 m/s dla poszczególnych punktów pomiarowych



Rys. 6.33. Chwilowa wartość kąta obrotu kierownicy. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,00 m przy prędkości 0,31 m/s



Rys. 6.34. Chwilowa wartość prędkości kąta obrotu koła zastępczego. Wartość zadana – kolor czerwony, wartość zmierzona (enkoder 2) – kolor niebieski, podczas przejazdu platformy mobilnej po trajektorii typu „okrąg" r=1,00 m przy prędkości 0,31 m/s

6.7. Podsumowanie wyników pomiarów

Rezultaty przeprowadzonych eksperymentów dowodzą skuteczności metody sterowania optymalnego przy energetycznym wskaźniku jakości w zastosowaniu do nadzorowania ruchu silnie nieliniowego obiektu, jakim jest trójkołowa platforma mobilna. Przyjęta w pracy metodologia prowadzenia badań, bazująca na zastosowaniu technik projektowania mechatronicznego, potwierdza swoją zaletę. Umożliwia ona realizację systemu nadzorowania ruchu spełniającego stawiane wymagania w znacznie krótszym czasie, a także – końcową walidację tego systemu.

Przeprowadzone wirtualne prototypowanie, późniejsze testy HILS, a także końcowe prototypowanie na docelowym obiekcie, umożliwiły osiągnięcie optymalnego systemu (będącego integralną częścią powstałego równolegle obiektu badań), którego funkcjonalność umożliwia sterowanie w czasie rzeczywistym nieliniowym obiektem podczas jego ruchu po różnych trajektoriach oraz z różnymi prędkościami. W szczególności, pozytywne rezultaty uzyskane podczas przejazdu platformy z większą prędkością są wykładnikiem możliwości sterowania optymalnego przy energetycznym wskaźniku jakości, a także potwierdzeniem uzyskania zarówno optymalnego systemu sterowania, jak i konstrukcji mechanicznej. Dodatkowo, wprowadzona wariantowość w realizacji przejazdów platformy po różnych typach trajektorii, uświadamia o posługiwaniu się metodą sterowania o sporej elastyczności, pozwalającej przy innych środkach programowo-sprzętowych na realizację złożonych procesów sterowania maszyn.

Dużym wyzwaniem w całym cyklu badań była realizacja trajektorii ruchu po krzywej typu „sinus”. Podczas realizacji badań, platforma pomimo pokonania trasy składającej się z dwóch okresów przebiegu tej krzywej, nie doznała znaczących odchyłek dyskwalifikujących przyjętą koncepcję sterowania. Zestawiając te rezultaty z rezultatami dla pozostałych krzywych, tj. „paraboli” oraz „okręgu” można zauważyć, że uzyskane średnie wartości błędów położenia nie przekraczają 4 cm.

Na wszystkich zrealizowanych trasach przejazdu platformy uzyskano dobrą powtarzalność otrzymywanych rezultatów. Pojawiające się błędy należy tłumaczyć faktem sterowania obiektem opisanym za pomocą silnie nieliniowych równań dynamiki, których rozwiązanie numeryczne w czasie rzeczywistym staje się kłopotliwe.

Drogą przeprowadzonej optymalizacji (szczególnie przy wykorzystaniu techniki HILS) uzyskano kompromis pomiędzy występującymi błędami numerycznymi, a spełnieniem warunku determinizmu czasowego. Pomimo, że implementacja algorytmu jest operacją wymagającą, udało się uzyskać możliwość uodpornienia metody sterowania na zmienne w czasie trwania przejazdu bieguny układu (platformy mobilnej). Zalety tej nie posiada sterowanie obiektami nieliniowymi z wykorzystaniem sterownika PID, co zostało wykazane w pracy [59]

Dużą niedogodnością w całym cyklu badań stała się konieczność każdorazowego ustawiania kierownicy w pozycji „zero" tak, aby platforma poruszała się na wprost. Kolejna wersja platformy, bądź też inna koncepcja bazująca na sterowanym kole skrętnym powinna wyeliminować tę niedogodność.

Osiągnięta optymalizacja układu (wybór kroku całkowania, stosowanie prędkości korygujących) pozwoliła na realizację systemu umożliwiającego sterowanie obiektem z założonym z góry błędem (numerycznym), skutkującym powstaniem podczas realizacji trasy przejazdu błędu położenia. W tym miejscu należy wymienić inne czynniki wpływające na wzrost wartości błędu (np. skończona precyzja wykonania elementów mechanicznych, konieczność ustawiania platformy w pozycji „zero"), które podczas działania sygnału sterującego są kompensowane. Ostatecznie, średni błąd dla wszystkich zrealizowanych trajektorii jest mały, natomiast w przyszłych pracach badawczych może być poddany próbie sukcesywnej eliminacji. Wiąże się to głównie z podniesieniem wydajności procesora Real Time (celem zwiększania częstotliwości generowania optymalnego sygnału sterującego), bądź realizacją całkowitego sterowania (implementacja algorytmu) w układzie FPGA.

7. Wnioski

Przedstawiona w pracy koncepcja nadzorowania ruchu (bazująca na energetycznym wskaźniku jakości) silnie nieliniowego obiektu (trójkołowa platforma mobilna) powstała na skutek zastosowania technik projektowania mechatronicznego, a także późniejsza analiza wyników przeprowadzonych badań doświadczalnych (walidacja systemu podczas przejazdu obiektu badań po wyznaczonych trajektoriach ruchu) upoważniają do sformułowania wniosków dotyczących skuteczności proponowanej metody. Wnioski są obrazem doświadczeń autora z przeprowadzonego procesu badań, a także odzwierciedlają obecną tendencję do stosowania metod projektowania równoległego (mechatronicznego) podczas realizacji systemów sterowania oraz budowy złożonych urządzeń. Ponadto, należy wspomnieć o zastosowanym oprogramowaniu (LabVIEW), mającym pozytywny wpływ na ostateczne powodzenie realizacji niniejszej pracy.

1. W przeprowadzonych badaniach doświadczalnych potwierdzono skuteczność nadzorowania ruchu obiektu silnie nieliniowego za pomocą sterowania optymalnego przy energetycznym wskaźniku jakości. Skuteczność proponowanej metody sterowania w układzie nieliniowym została wykazana drogę eksperymentalną, w której zbudowana dla potrzeb badań trójkołowa platforma mobilna własnej konstrukcji, wykazała dużą dokładność oraz powtarzalność w trakcie przejazdu na wybranych trajektoriach.
2. Z przeprowadzonych badań wynika, że wykorzystana trójkołowa platforma mobilna z uwagi na silną nieliniowość (macierze **M** oraz **L** zależne od kąta skrętu kierownicy) jest trudna w sterowaniu. Jednak przyjęta metodyka projektowania bazująca na zastosowaniu technik projektowania mechatronicznego przy znaczącym udziale środowiska LabVIEW, dowiodła swojej skuteczności podczas realizacji systemu nadzorowania, a także – zadecydowała o końcowym sukcesie przeprowadzonego przedsięwzięcia naukowego. Osiągnięto optymalny energetycznie system czasu rzeczywistego, w skuteczny sposób realizujący postawione zadania, a jednocześnie umożliwił znaczną redukcję czasu realizacji projektu.
3. Stosowanie techniki wirtualnego prototypowania umożliwiło wstępną weryfikację zastosowanych rozwiązań implementowanego algorytmu oraz

przyjętych koncepcji architektury systemu. Poprzez eliminację błędów oraz poprawę przepływu sygnałów, dokonano optymalizacji procesu zapisu algorytmu w środowisku Labview. Zawężono obszar poszukiwań odpowiednich nastaw macierzy **R** oraz **Q**, a także – skrócono czas weryfikacji proponowanych rozwiązań oraz przyjętej architektury rozwiązania.

4. Szczególne rezultaty osiągnięto stosując technikę HILS, w której wykorzystano rzeczywisty sterownik czasu Real Time. Konfiguracja testu, a także – zakres przeprowadzonych badań (koncepcja techniki zastosowana również przy realizacji (w jednej z faz) techniki wirtualnego prototypowania) umożliwiła poprzez weryfikację warunku determinizmu czasowego (dla zastosowanego sterownika cRIO NI-9076), ustalenie optymalnej częstotliwości generowania sygnału. Drogą eliminacji błędów oraz reorganizacji wykonania algorytmu (wykonanie części algorytmu odpowiedzialnej za formowanie fali PWM, obsługę sterowników silników DC oraz enkoderów powierzono układowi FPGA) dokonano kolejnej optymalizacji w ramach architektury systemu.
5. Proces integracji systemu nadzorowania ze zbudowaną dla potrzeb realizacji badań platformą mobilną, a także w późniejszej fazie końcowej walidacji, przeprowadzono stosując technikę szybkiego prototypowania na platformie docelowej, dzięki której wykazano skuteczność proponowanego algorytmu sterowania. Walidacja zintegrowanego obiektu badań potwierdziła możliwości proponowanej metody (energetyczny wskaźnik jakości), której funkcjonalność umożliwia sterowanie w czasie rzeczywistym silnie nieliniowym obiektem, podczas jego ruchu po różnych torach oraz z różnymi prędkościami przejazdu platformy. Pozytywne rezultaty uzyskane podczas przejazdu platformy z większą prędkością są wykładnikiem możliwości stosowania rozpatrywanego wskaźnika, a także faktem posługiwania się metodą sterowania o sporej elastyczności, pozwalającej przy zastosowaniu innych rozwiązań programowo-sprzętowych na realizację złożonych procesów sterowania maszyn.
6. Zastosowane techniki projektowania mechatronicznego, a także – seria badań doświadczalnych, pozwoliły na osiągnięcie optymalizacji układu oraz opracowanie metodologii jej stosowania. Warunek optymalizacji energetycznej układu osiągnięto poprzez dobór kroku całkowania równań różniczkowych (częstotliwości generowania optymalnego sygnału sterującego), dla którego

spełniony jest warunek czasu rzeczywistego, a występujące błędy są na satysfakcjonującym poziomie. Proces mechatronicznego doboru długości kroku całkowania, okazał się niezbędnym elementem w budowie systemu nadzorowania obiektu nieliniowego. Optymalizacja kroku (kompromis pomiędzy występującymi błędami, a wydajnością CPU sterownika) oraz wprowadzenie prędkości korygujących pozwoliło na poprawę wydajności sterowania oraz sprawności energetycznej układu, a także przyczyniło się do poprawy stabilności przejazdu platformy (poprzez eliminację fluktuacji – lokalnych przyspieszeń).

7. Powstała równolegle (wraz z systemem nadzorowania) na skutek stosowania technik projektowania mechatronicznego, trójkołowa platforma mobilna własnej konstrukcji z powodzeniem została zintegrowana z proponowanym systemem nadzorowania. Zrealizowana konstrukcja wykazała swoją zaletę w trakcie realizacji badań eksperymentalnych. System napędowy bazujący na układzie różnicowym, w skuteczny sposób umożliwił platformie poruszanie się po zadanych trajektoriach bez poślizgów. Również układ kierowniczy, dzięki swojej małej złożoności mechanicznej (bezpośrednie połącznie koła kierowniczego z motoreduktorem) zapewnił w trakcie przejazdu platformy precyzyjne ustawienie kąta obrotu kierownicy.
8. Zastosowane w pracy spójne środowisko programistyczne LabVIEW zostało wykorzystane przy realizacji technik projektowania mechatronicznego, do powstania systemu nadzorowania trójkołowej platformy mobilnej. W dalszej części, możliwości środowiska w zakresie integracji z dedykowanym sprzętem (sterownik cRIO NI-9076) pozwoliły na łatwą transformację kodu programu, (zaimplementowanego algorytmu systemu nadzorowania) tworzonego dla potrzeb realizacji technik projektowania, do postaci zapisywanej do fizycznego sterownika czasu rzeczywistego platformy mobilnej. Wykorzystany moduł programu LabVIEW Real Time umożliwił realizację koncepcji deterministycznego sterowania bazującą na algorytmie *on-line*. Możliwości zastosowanego sterownika, optymalizacja algorytmu, a także nawiązanie do obecnych trendów w budowie mechatronicznych systemów sterowania, zostały osiągnięte, dzięki wykorzystaniu dodatkowo modułu LabVIEW FPGA. Należy wspomnieć o wykorzystanym w pierwszej fazie realizacji projektu środowisku Maple, które umożliwiło autorowi, poprzez operacje symboliczne,

rozwiązanie zawiłych równań różniczkowych. Otrzymane rozwiązanie zostały najpierw zweryfikowane, by w późniejszej fazie zostać w postaci parametrycznej zaimplementowane do środowiska LabVIEW.

9. Wybór jednostki sterownika NI-9076 umożliwił autorowi osiągnięcie pełnej integracji fazy projektowania (podczas realizacji techniki HILS) oraz walidacji systemu w ramach jednej platformy programistyczno-sprzętowej. Dedykowany sprzęt dla środowiska LabVIEW, zapewnił realizację skutecznego systemu nadzorowania czasu rzeczywistego. Wbudowane możliwości diagnostyczne w zakresie monitorowania przebiegu procesu wykonania kodu, wykorzystania zasobów procesora oraz pamięci operacyjnej spowodowały wypracowanie systemu optymalnego pod względem energetycznym. Dzięki zastosowaniu modułów NI-9505 osiągnięto możliwość skutecznej kontroli pracy silników DC z poziomu układów FPGA, a także funkcji pomiarowych w zakresie dostępnego interfejsu zewnętrznego enkodera. Dodatkowo, końcówka mocy NI-9031 rozszerzyła możliwości stosowania silników prądu stałego.

8. Wskazówki dotyczące dalszych badań

Zastosowane techniki projektowania mechatronicznego pozwoliły na wypracowanie metodyki realizacji projektu, której charakter może być stosowany podczas prowadzenia innych przedsięwzięć naukowych. Elastyczne podejście do tworzenie modelu oraz interfejsów mechatronicznych, równoległość realizacji zadań, a także wykorzystane oprogramowanie (LabVIEW, Maple) mogą być łatwo adoptowane do szeregu przyszłych pomysłów oraz idei.

Zebrane przez autora doświadczenia wynikające z własnych przemyśleń oraz prezentowanych w literaturze trendów światowych, prowadzą do zdefiniowania propozycji przyszłych badań nad rozwojem energetycznego wskaźnika jakości, jako algorytmu nadzorowania robotyką mobilną.

1. Autor proponuje rozważenie platformy czterokołowej, jako doskonałego rozwiązania dla robotów poruszających się w terenie (przy eksploracji kosmosu lub głębin morskich). Z uwagi na ruch robota w terenie, wydaje się rozsądnym realizowanie napędu czterokołowego, mogącego sprostać większym wzniesieniom oraz pokonaniu bardziej wymagającego terenu. Budowa platformy czterokołowej będzie wymagać rozważenia połączenia układu kierowniczego z napędowym. Proponuje się realizację układu napędu platformy czterokołowej bazującej na dwóch silnikach napędzających niezależnie osie platformy (tylna i przednią), bądź jednej wydajnej jednostki (np. silnika DC) usytuowanej centralnie. Niewątpliwie, koncepcja układu napędu bazująca na dwóch silnikach, z punktu widzenia konstrukcji mechanicznej oraz układu sterowania, będzie bardziej zaawansowana (wymagająca) technicznie (potrzeba realizacji niezbędnej synchronizacji układu, modyfikacja systemu sterowania), jednak wpłynie to na uelastycznienie realizacji napędu oraz możliwości stosowaniu trybów pracy takiego układu (rozdział mocy napędu pomiędzy obie osie, całkowite wyłączanie i realizowanie tylko napędu na przód lub tył, itp.). Do wyzwań konstruktora układu platformy czterokołowej będzie należało, zaprojektowanie układu kierowniczego takiego układu, łączącego w sobie układ napędowy. Autor proponuje wykorzystanie wzorców (gotowych rozwiązań) stosowanych w samochodach zdalnie sterowanych. Należy zauważyć, że przypadek robota

poruszającego się w terenie wymaga rozpatrzenia przestrzeni trójwymiarowej w opisie kinematyki układu, a także uwzględnienia energii potencjalnej w opisie dynamiki i realizacji algorytmu sterowania (składnika również podlegającego minimalizacji). Z uwagi na większą złożoność opisu modelu matematycznego takiego układu, niewątpliwie, konieczne stanie się stosowanie wyrafinowanych jednostek obliczeniowych (systemu sterowania). Pomocnym w realizowaniu tej koncepcji na pewno będzie wykorzystane w pracy środowisko LabVIEW wraz ze sterownikiem cRIO, który dostępny jest również ostatnio (premiera wrzesień, 2011) jako wersja dwurdzeniowa Dual Core 1,3GHz. Dodatkowo, autor proponuje rozważanie koncepcji stosowania amortyzatorów, jako elementu minimalizującego drgania konstrukcji, a tym samym tłumiącego ewentualne zakłócenia zewnętrzne mogące być przyczyną utraty stabilności (wzrost zużycia energii będącej przedmiotem optymalizacji w zespołach napędowych robotów mobilnych).

2. Kolejnym krokiem w rozwoju rozważanego wskaźnika jakości w robotyce mobilnej jest realizacja platformy kołowej (najlepiej czterokołowej) z przyczepą. W pracach związanych z eksploracja kosmosu, realizacji zadań podwodnych lub lądowych często może dojść do sytuacji, w której platforma mobilna ma za zadanie transport różnych materiałów czy narzędzi wyposażenia pracy robota, np. chwytak, podnośnik, zespół spawalniczy, materiałów niebezpiecznych, bądź innych niezbędnych podczas realizacji zadania danego typu. W proponowanych badaniach należałoby rozważyć ponownie konstrukcje platformy pod kątem jej kinematyki oraz dynamiki. Zadanie sprowadziłoby się do określenia modelu obliczeniowego dla pojazdu wielkokołowego, którego rama jest w jednym punkcie skręcana. Należy zauważyć, że przypadek realizacji zadania, w którym uwzględnia się stosowanie przyczepy, mógłby posłużyć w celu zbadania wpływu zmiany ciężaru ładunku (bądź samego obiektu w wyniku wykorzystania zasobów energii w postaci paliwa płynnego) na proces sterowania obiektem. Zmiana ciężaru platformy, z uwagi na nieliniowość macierzy bezwładności (platforma trójkołowa i czterokołowa) będzie determinowała jakość sterowania oraz jego odporność na zmiany dynamiki układu. Rozważany w pracy algorytm uwzględnia zmiany biegunów układu w trakcie jego przemieszczania. Stąd,

zbadanie zmian ciężaru obiektu powinno być interesującym doświadczeniem projektanta.

3. W niniejszej pracy rozważono platformę, która porusza się z prędkością 0,17 m/s oraz 0,31 m/s. Przyjęta prędkość wiązała się z możliwościami zastosowanego silnika prądu DC, który musiał mieć odpowiedni moment obrotowy. Wzrost tej wartości obarczony był (dla danego typu silnika) spadkiem prędkości obrotowej (stosowano większe przekładnie). Niewątpliwie, roboty mobilne wykonujące pracę na dużych przestrzeniach np. w halach produkcyjnych, muszą poruszać się z maksymalnie możliwą prędkością. Autor proponuje, aby rozważyć konstrukcję trójkołową (zastosowaną w niniejszej pracy), w której dokonanie niezbędnej rekonstrukcji pozwoli na uzyskanie prędkość docelowej na poziomie 2-3 m/s (około 10 km/h). W realizacji tego typu zadania będzie istniała konieczność dokonania zmian systemu napędowego (większy silnik) oraz prawdopodobnie będzie wiązało się to z koniecznością zastosowania szybszej jednostki obliczeniowej systemu sterowania. Niewątpliwie może okazać się koniecznym zastosowanie bardziej wydajnych (bądź bardziej pojemnych) źródeł energii. Projektant przyszłej platformy, może rozważyć również, stosowanie silników prądu stałego oferowanych przez firmę Maxon Motors, charakteryzujących się największą na rynku gęstością mocy (obarczone jest to niestety bardzo wysoką ceną).
4. Kierunek rozwoju autor upatruje również w stosowaniu układu FPGA, mogącego stanowić jednoukładowy systemem nadzorowania platformą mobilną dowolnej konstrukcji. Stosowanie jednoukładowych, programowalnych jednostek znacznie przyspieszy proces generowania sygnału sterującego, a także umożliwi dodatkową optymalizację algorytmu. Autor przewiduje podniesienie sprawności energetycznej projektowanej platformy poprzez redukcję powstałych błędów (możliwość częstszego generowania sygnału będzie skutkowała krótszym czasem, w którym platforma pozostaje bez sygnału sterującego), a także znacznie szybszą odpowiedź systemu na powstające ewentualne dewiacje. Realizacja jednostki sterującej wyłącznie na bazie układu FPGA rozszerzyłaby liczbę potencjalnych dostawców takiego układu (autor proponuje układy firmy Xilinix lub dSpace wraz z dostarczanymi kompilatorami), a także pozwoliła na zastosowanie do realizacji badań

mechatronicznych inne alternatywne środowisko deweloperskie, niż wykorzystane LabVIEW.

Należy zaznaczyć, że doskonałą alternatywą dla użytego w pracy oprogramowania LabVIEW byłby niewątpliwie (użyty w niniejszej pracy) również w nieco mniejszym zakresie pakiet Maple. Środowisko Maple, wraz z modułem SIM, pozwala na bazie wcześniej zdefiniowanego modelu obliczeniowego, na tworzenie wirtualnych modeli mechatronicznych obiektów (w tym, rozważanej platformy mobilnej), a tym samym – na realizację wirtualnych przejazdów (animacja komputerowa pozwalająca na śledzenie zachowania się obiektu badań w trakcie pokonywania zdefiniowanej trajektorii wraz z szeregiem niezbędnych danych pomiarowych otrzymywanych w postaci grafów lub tabelarycznej). Zastosowana metoda oraz przeprowadzone badania dopełniłyby wykorzystane w pracy projektowanie mechatroniczne. Proponuje się zastosowanie Maple (wraz z SIM) w tych badaniach, w których takie podejście byłoby zasadne oraz podniosłoby efektywność przeprowadzonych badań mechatronicznych.

5. Propozycją wartą rozważania, zmieniającą częściowo koncepcję rozpatrywanego sterowania, jest realizacja sterowania scentralizowanego (dla większej liczby robotów realizujących w jednym miejscu określone zadania). W takim rozwiązaniu platforma mobilna nie jest wyposażona w sterownik, a jedynie w układy wykonawcze, sensory oraz moduły komunikacyjne realizujące transmisję radiową z komputerem centralnym, gdzie dokonywane są obliczenia (sterowanie według przyjętego algorytmu, tutaj energetycznego wskaźnika jakości) Scentralizowane sterowanie może być chętnie wykorzystywane w halach produkcyjnych, gdzie stosuje się kilkanaście robotów mobilnych. W takim przypadku, bardzo silny komputer oblicza w czasie rzeczywistym sygnały dla tych platform, które w danej chwili są aktywne (wykonują określone zadanie). Proponowana koncepcja może znacznie zredukować koszt wdrożenia takiego systemu (platforma nie jest wyposażona w system sterowania), a także – znacznie zmniejszyć koszt w przypadku jego rozbudowy (oczywiście przy założeniu sterowania dużą grupa robotów wystąpi tzw. efekt skali). Rozważana konfiguracja wiązałaby się z większym skupieniem uwagi konstruktora na realizacji systemu sterowania grupą robotów (nadawania priorytetów), obsługą w czasie rzeczywistym

kolejek do wspólnego zasobu (komputera) oraz samą techniką transmisji danych (np. GSM/UMTS, WLAN, WiMax, itp.).

9. Literatura

[1] Balint D.: *Maple Manual for Engineering Mechanics*: *Dynamics*. Thompson Engineering, 2007. ISBN-13: 978-0495296102.

[2] Bauer, R.: Leung, W., and Barfoot, T.: *Development of a Dynamic Simulation Tool for the ExoMars Rover*. i-SAIRAS, Monachium, Niemcy, 2005.

[3] Bouscayrol A.: *Different types of hardware-in-the-loop simulation for electric drives*. IEEE ISIE, Cambridge, Anglia, Czerwiec, 2008, s. 2146–2151.

[4] Brzózka J.: *Regulatory i układy automatyki*. Warszawa: Mikom 2004. ISBN 83-7279-380-8.

[5] Budnicki Z.: *Teoria i algorytmy sterowania*. Warszawa: PWN 2005. ISBN 83-01-14414-9.

[6] Cheli F., Concas A., Giangiulio E., Sabbioni E.: A simplified ABS numerical model*: Comparison with HIL and full scale experimental tests*. Computers and Structures 2008 (86), s. 1494–1502.

[7] Chruściel M.: *Labview w Praktyce*. Legionowo, BTC, 2008, ISBN 978-83-60233-32-0.

[8] Czemplik A.: *Modele dynamiki układów fizycznych dla inżynierów*. WNT: Warszawa 2008. ISBN 978-83-204-3388-3

[9] Galewski M., Kaliński K.: *Nadzorowanie drgań przy frezowaniu szybkościowym smukłymi narzędziami ze zmienną prędkością obrotową*. Gdańsk: Wyd. Pol. Gdańskiej 2009. ISBN 978-83-7348-251-7.

[10] Gans N.R., Dixon W.E., Lind R., Kurdila A.: *A hardware in the loop simulation platform for vision-based control of unmanned air vehicles*. Mechatronics, vol. 19, 2009, s. 1043–1056.

[11] Gibbesch, A., Schaefer, B., Michaud S.: *3D Simulation and Validation of RCL-E and MER Rover Types Mobility*. The 9th ESA Workshop on Advanced Space Technologies for Robotics and Automation, Nordwijk, 2006, Holandia.

[12] Giergiel J., Hendzel Z., Żylski W.: *Symboliczne generowanie równań kinematyki mobilnego robota Pioneer-2DX*. Przegląd Mechaniczny 2000, z. 19-20, s. 26–31.

[13] Giergiel J., Hendzel Z., Żylski W.: *Symboliczne generowanie równań dynamiki mobilnego robota Pioneer-2DX*. Przegląd Mechaniczny 2001, z. 11, s. 24–29.

[14] Giergiel J., Hendzel Z., Żylski W.: *Synteza zadania odwrotnego kinematyki mobilnego robota Pioneer-2DX przy użyciu procesora symbolicznego*. Zeszyty Naukowe AGH 2000, Mechanika, Tom 19, z. 4, s. 397–409.

[15] Giergiel J., Hendzel Z., Żylski W.: *Zagadnienia kinematyki mobilnych robotów kołowych*. Przegląd Mechaniczny1999, nr 14, s. 20–25.

[16] Giergiel J., Kurc K., Giergiel M.: *Mechatroniczme projektowanie robotów inspekcyjnych*. Rzeszów: Oficyna Wyd. Pol. Rzeszowskiej 2010. ISBN 978-83-7199-638-2.

[17] Giergiel J., Małka P.: *Modelowanie kinematyki i dynamiki mobilnego robota. Wybrane zagadnienia kinematyki i dynamiki mobilnych robotów kołowych*. Modelowanie Inżynierskie, Gliwice 2006, s. 157–162.

[18] Giergiel J., Żylski W.: *Dynamika mobilnych robotów kołowych*. Zeszyty Naukowe AGH 2000, Mechanika, Tom 16, z. 4, s. 461–490.

[19] Giergiel J., Żylski W.: *Identyfikacja równań ruchu mobilnych robotów kołowych*. Zeszyty Naukowe AGH 1998, tom 17, Mechanika z. 1, s. 75–104.

[20] Giergiel M., Hendzel Z., Żylski W.: *Modelowanie i sterowanie mobilnych robotów kołowych*. Warszawa: PWN 2002. ISBN 83-01-13789-4.

[21] Górecki H. : *Optymalizacja i sterowanie systemów dynamicznych*. Kraków: AGH 2006. ISBN 83-7464-021-9.

[22] Grabowski R.: *Kształtowanie geometryczne krzywych przejściowych w drogach kołowych, kolejowych i trasach wodnych*. Białystok: Wydawnictwa Pol. Białostockiej 1996. ISBN 0867-096X.

[23] Guzman J., Fernandez G.,Gutierrez V.: *Design of a Programmable Platform for the Control of Mobile Robots Based on a Multi-FPGA architecture*. Proceedings of the 7th International Caribbean Conference on Devices, Circuits and Systems, Meksyk, Kwiecień 2008. s. 28-30.

[24] Hahenberger P., Poltschak F., Zeman K., Amrhein W.: *Hierarchical design models in mechatronic product development process of synchronous machines*. Mechatronics 20, 2010 s. 864–875.

[25] Hanselman H.: *Hardware-in-the-loop Simulation Testing and its Integration into a CACSD Toolset*. IEEE International Symposium on Computer-Aided Control System Design, 1996, s. 15–18.

[26] Heiskanen P., Heikkila S., Halme A.: *Development of a dynamic mobile robot simulator for astronaut assistance*. 10th ESA Workshop on Advanced Space Technologies for Robotics and Automation, Holandia, Noordwijk 2008.

[27] Hendzel Z., Żylski W., Burchardt A.: *Autonomiczne mobilne roboty kołowe*. Rzeszów: Oficyna Wyd. Pol. Rzeszowskiej 2008. ISBN 978-83-7199-524-8.

[28] Henselmann H.: *Hardware in the loop simulations as a standard approach for development, customisation and production test*. SAE Information Congress, Detroit ISA, USA, 1993.

[29] Hewit H. R.: *Mechatronic design – The key to performance enhancement*. Robotics and Autonomus Systems, 1996, vol. 19, s. 135–142.

[30] Honczarenko J.: *Roboty przemysłowe*. Warszawa: WNT 2004. ISBN 978-83-204-3656-3.

[31] Howard T., Kelly A.: *Optimal rough terrain trajectory generation for wheeled mobile robots*. International Journal of Robotic Research 2007, Vol. 26, no. 2 (Luty), s. 141–166.

[32] Iagnemma K., Shibly H., Dubovsky S.: *On-line terrain parameter estimation for planetary rovers*. International Conferances on Robotics and Automation, Waszyngton, USA, 2002.

[33] Kaczorek T.: *Teoria układów regulacji automatycznej*. Warszawa: WNT 1977. ISBN 978-83-204-3556-6.

[34] Kaczorek T., Dzieliński A., Dąbrowski W., Łopatka R.: *Podstawy Teorii Sterowania*. Warszawa: WNT 2003. ISBN 978-83-204-3556-6.

[35] Kaliński K.: *Nadzorowanie drgań układów mechanicznych modelowanych dyskretnie*. Gdańsk: Wyd. Pol. Gdańskiej 2001. ISBN 83-88007-83-1.

[36] Kaliński K., Buchholz C.: *Trajectory optimal control of three wheeled mobile platform at time changeable energy performance index*. 10th Conference Active Noise and Vibration Control Methods, Kraków, 2011.

[37] Kaliński K., Buchholz C.: *Mobile platform power optimization by control command at time changeable energy performance index*. Annual International Workshop 2011-Dynamic Behaviour of Structures and Materials, Interaction and Friction, Metz, Francja, 2011.

[38] Kaliński K., Buchholz C.: *Error minimisation in orientation and localization by correction velocities for three-wheeled mobile platform at time changeable*

energy performance index. 16th International Conference on Methods and Models in Automation and Robotics, Międzyzdroje, 2011.

[39] Kaliński K., Buchholz C.: *HILS for the design of three-wheeled mobile platform motion surveillance system with use of energy performance index*. 8th International Conference on Mechatronic Systems and Materials, Białystosk, 2012.

[40] Kaliński K., Buchholz C.: *Three wheeled mobile platform powered by LabVIEW at energy performance index*. International Conference Mechatronics, Warszawa, 2012.

[41] Kaliński K. J., Buchholz C.: *LABVIEW in mechatronic design of control system for three wheeled mobile platform at time changeable performance index*. W: *Projektowanie mechatroniczne. Zagadnienia wybrane*. Red. T. Uhl. Kraków: Katedra Robotyki i Mechatroniki AGH 2011, s. 47–55. ISBN 978-83-7789-072-1.

[42] Kozłowski K., Dutkiewicz P., Wróblewski W.: *Modelowanie i sterowanie robotów*. Warszawa: PWN 2003. ISBN-83-01-14081-X.

[43] Krebs, A., Thueer, T., Carrasco, E., and Siegwart, R.: *Towards torque control of the CRAB rover*. International Symposium on Artificial Intelligence, Robotics and Automation in Space i-SAIRAS, Pasadena, USA, 2008.

[44] Kucherenko, V., Bogatchev, A., and Van Winnendael, M.: *Chassis Concepts for the ExoMars Rover*. The 8th ESA Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA'04), Nordwijk, Holandia, 2004.

[45] Kurc K.: *Mechatronika w projektowaniu robota*. Rzeszów: Oficyna Wyd. Pol. Rzeszowskiej 2010. ISBN 978-83-7199-642-9.

[46] Lamon P., Krebs A.: *Wheel torque control for a rough terrain rover*. IEEE 2004 International Conference on Robotics and Automation, New Orleans, LA, USA, s. 4682–4687.

[47] Lamon P., Siegwart R.: *Wheel Torque Control in Rough Terrain - Modelling and Simulation*. Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Hiszpania, 2005.

[48] Ledin J.: *Hardware-in-the-loop simulation*. Embedded Systems Programming 1999, Vol. 12, No. 2, s. 42–53.

[49] Lee Y., Toon M., Sunwoo M.: *A cost and time effective hardware in the loop simulation platform for automative engine control systems*. Proc. Instn Mech. Engrs, vol. 217, part D, no. 7, 2003, s. 41–52.

[50] Leitner J.: *Space Technology Transition Using Hardware in the Loop Simulation*. Proceedings of the 1996 Aerospace Applications Conference, Vol. 2, Luty 1996, s. 303–311.

[51] Lentijo S., D'Arco S., Monti A., S.: *Comparing the Dynamic Performances of Power Hardware-in-the-Loop Interfaces*. IEEE Transactions on industrial electronics, Vol. 57, No. 4, Kwiecień 2010, s. 1195–1207.

[52] Leonardo M. P., Dias A. L., Massaro L.C., Augusto de Paula Caurin G.: *Dynamic Modelling and Hardware-in-the-loop Simulation applied to a Mechatronic Project*. ABCM Symposium Series in Mechatronics, Vol. 3, Rio de Janerio, Brazylia, 2008, s. 96–105.

[53] Lin C., Zhang L.: *Hardware-in-the-loop Simulation and Its Application in Electric Vehicle Development*. IEEE Vehicle Power and Propulsion Conference (VPPC), Wrzesień 3-5, 2008, Harbin, Chiny, s. 1–6.

[54] Lin W. Ch.; Zheng Ch.: *Energy management of fuel cell/batterry/supercapacitor hybrid power system using an adaptive optimal-control method*. Journal of Power Source 2011 (196), s. 3280–3289.

[55] Linjama M., Virvalo T., Gustafsson J., Lintula J, Aaltonen V., Kivikoski M., *Hardware-in-the-loop Environment for Servo System Controller Design, Tuning, and Testing*. Microprocessors and Microsystems 2000, Vol. 24, No. 1, s. 13–21.

[56] Maclay D.: *Simulation gets into the loop*. IEE Rev. 1997, vol. 43, no. 3, s. 109–112.

[57] Martin A., Emami M. R.: *An Architecture for Robotic Hardware in- the-Loop Simulation*. IEEE International Conference on Mechatronics and Automation, Session WP1-9 Control Architecture, HeNan, Chiny, Czerwiec 2006.

[58] Martin A., Emami M. R.: *Analysis of Robotic Hardware-in-the-Loop Simulation Architecture*. IEEE/RSJ International Conference on Intelligent Robots and Systems San Diego, USA, 2007.

[59] Mazur M.: *Nadzorowanie ruchu 2-kołowej platformy mobilnej z zastosowaniem sterowania optymalnego przy energetycznym wskaźniku jakości*. Praca doktorska. Politechnika Gdańska Wydział Mechaniczny, 2010.

[60] Meriam J. L.: *Engineering Mechanics-dynamics: WITH Solving Dynamics Maple*. Ohio: John Wiley & Sons 2007. ISBN-13 978-0-470-09920-9.

[61] Michaloski J., Wheatley T.: *Design Principles for a Real Time Robot Control Systems*. IEEE International conference on system engineering, Pittsburgh, USA, 1990.

[62] Michaud, S., Gibbesch, A., Thueer, T., Krebs, A., Lee, C., Despont, B., Schaefer, B., and Slade, R.: *Development of the ExoMars Chassis and Locomotion Subsystem*. International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS), Pasadena, USA, 2008.

[63] Montelo H., Furukawa C. M: *Hardware In The Loop Simulation Applied To Semi-Autonomous Underwater Vehicles*. 9th IEEE/IAS International Conference on Indystry applications, 2010, s. 1–6.

[64] Murphey Y.Ch.; Chen Z.; Kiliaris L.; Masrur M.A.: *Intelligent power management in a vehicular system with multiple power sources*. Journal of Power Source 2011 (196), s. 835–846.

[65] National Instrument, *Building networked applications with the LabWindows/CVI TCP Support Library*. Tutorial, 2008, Dostępny w internecie: www.ni.com.

[66] National Instrument, *CompactRIO developers guide*. National Instrument Tutorial, 2009, Dostępny w internecie: www.ni.com.

[67] National Instrument, *Command based communication using simple TCP/IP messaging*. National Instrument Tutorial, 2008, Dostępny w internecie: www.ni.com.

[68] National Instrument, *Creating Hardware-in-the-Loop (HIL) Test System Applications*. National Instrument Tutorial, 2011, Dostępny w internecie: www.ni.com.

[69] National Instrument, *Hardware-in-the-Loop (HIL) Test System Architectures*. National Instrument Tutorial, 2011, Dostępny w internecie: www.ni.com.

[70] National Instrument, *HIL (Hardware-in-the-Loop) Evaluation of Vehicle Components with LabVIEW Real-Time and CarSim/TruckSim*. National Instrument Tutorial, 2008, Dostępny w internecie: www.ni.com.

[71] National Instrument, *LabVIEW FPGA in Hardware-in-the-Loop Simulation Applications*. National Instrument Tutorial, 2008, Dostępny w internecie: www.ni.com.

[72] National Instrument, *Lossless Communication with network streams: components, architecture and performance*. National Instrument Tutorial, 2011, Dostępny w internecie: www.ni.com.

[73] National Instrument, *Network variable technical overview*. National Instrument Tutorial, 2009, Dostępny w internecie: www.ni.com.

[74] National Instrument, *Offloading signal processing with Labview FPGA*. National Instrument Tutorial, 2011, Dostępny w internecie: www.ni.com.

[75] National Instrument, *Real Time high performance computing with NI Labview*. National Instrument Tutorial, 2010, Dostępny w internecie: www.ni.com.

[76] National Instrument, *Remotely monitoring I/O*. National Instrument Tutorial, 2011, Dostępny w internecie: www.ni.com.

[77] National Instrument, *Suggestions for using execution systems and priorities*. National Instrument Tutorial, 2008, Dostępny w internecie: www.ni.com.

[78] National Instrument, *Using LabVIEW for Rapid Control Prototyping and Hardware-in-the-Loop Simulation*. National Instrument Tutorial, 2008, Dostępny w internecie: www.ni.com.

[79] National Instrument, *Using the Labview shared variables*. National Instrument Tutorial, 2010, Dostępny w internecie: www.ni.com.

[80] National Instrument, *Using the right networking protocol*, National Instrument Tutorial, 2010, Dostępny w internecie: www.ni.com.

[81] Ombach J.: *Wykłady z równań różniczkowych wspomagane komputerowo - Maple*. Kraków: Wyd. Uniw. Jagiellońskiego 1999. ISBN 83-233-1162-5.

[82] Petko M.: *Wybrane metody projektowania mechatronicznego*. Radom: Instytut Technologii Eksploatacji – PIB, 2008. ISBN 978-83-7204-709-0.

[83] Priya T.K, Rejash Kumar P., Shridharan K.: *A Harware efficient scheme and FPGA realization for computation of a single shortest path for a mobile automation*. Microprocesosors and Microsystems 2006, s. 413–424.

[84] Prochowski L.: *Mechanika ruchu*. Warszawa: WKL 2008. ISBN 978-83-206-1701-6.

[85] Ren, W. Steurer, M. Woodruff, S.: *Progress and challenges in real time hardware-in-the loop simulations of integrated ship power systems*. Power Engineering Society General Meeting, IEEE, Vol. 1, 2006, s. 534–537.

[86] Stoeppler G., Menzel T., Douglas S.: *Hardware-in-the-loop Simulation of Machine Tools and Manufacturing Systems*. Computing and Control Engineering Journal, Vol. 16, No. 1, 2005, s. 10–15.

[87] Sun Z., Reif J.: *On Energy-minimizing Path on Terrains for a Mobile Robot*. Proceedings of the 2003 IEEE Internatonal Conference on Robtics and Automation, Taipei, Tajwan, Wrzesień 14-19, 2003, s. 3782–3788.

[88] Tchoń K. : *Manipulatory i roboty mobilne*. Warszawa: Akademicka Oficyna Wyd. 2000. ISBN 83-7101-427-9.

[89] Thounthong P., Chunkag V., Sethakul P., Sikkabut S., Peierfederici S., Davat B.: *Energy management of fuel call/batterry/supercapacitor hybrid power*. Journal of Power Source 2011 (196), s. 313–324.

[90] Thounthong P.; Real S.; Devat B.: *Energy management of fuel cell/batterry/supercapacitor hybrid power source for vehicle applications*. Journal of Power Source 2009 (193), s. 376–385.

[91] Thueer, T., Lamon, P., Krebs, A., and Siegwart, R.: *CRAB - Exploration Rover with Advanced Obstacle Negotiation Capabilities*. The 9th ESA Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA'06), Nordwijk, Holandia, 2006.

[92] Tłaczala W.: *Środowisko Labview w eksperymencie wspomaganym komputerowo*. Warszawa: WNT 2002. ISBN 83-204-2754-1.

[93] Tomizuka M.: *Mechatronics: from the 20th to 21st century*. Control Engineering Practice 2002, vol. 10, s. 877–886.

[94] Vibet C.: *Symbolic modelling of robot kinematics and dynamics*. Robotics and Autonomous System 1995, Vol. 14, s. 301-314.

[95] Wang J., Wang W., Jin L., Song C.: *Independent wheel torque of 4WD electric vehicle for differential drive assisted steering*. Mechatronics 2011, 21, s. 63–76.

[96] Wolf D. F., Holanda J.A, Bonato V., Peron R., Marques E.: *An FPGA based mobile platform robot controller*. Programmable Logic 2007, s. 119–123.

[97] Wong J.Y: *Theory of ground vehicles*. New Jersey: John Wiley and Sons Inc. 2008. ISBN 978-0-470-17038-0.

[98] Zhang W.; Hu J.; Lu Y.*: Optimal solution to a class of power management problems in mobile robots*. Automatica 2009 (45), s. 980–996.

[99] Żylski W.: *Opis ruchu mobilnego robota kołowego*. Przegląd Mechaniczny 1997, z. 14, s. 12-18.

[100] Żylski W.: *Kinematyka i dynamika mobilnych robotów kołowych*. Rzeszów: Oficyna Wyd. Pol. Rzeszowskiej 1996. ISBN 83-86705-51-5.

10. Załączniki

10.1. Środowisko LabVIEW

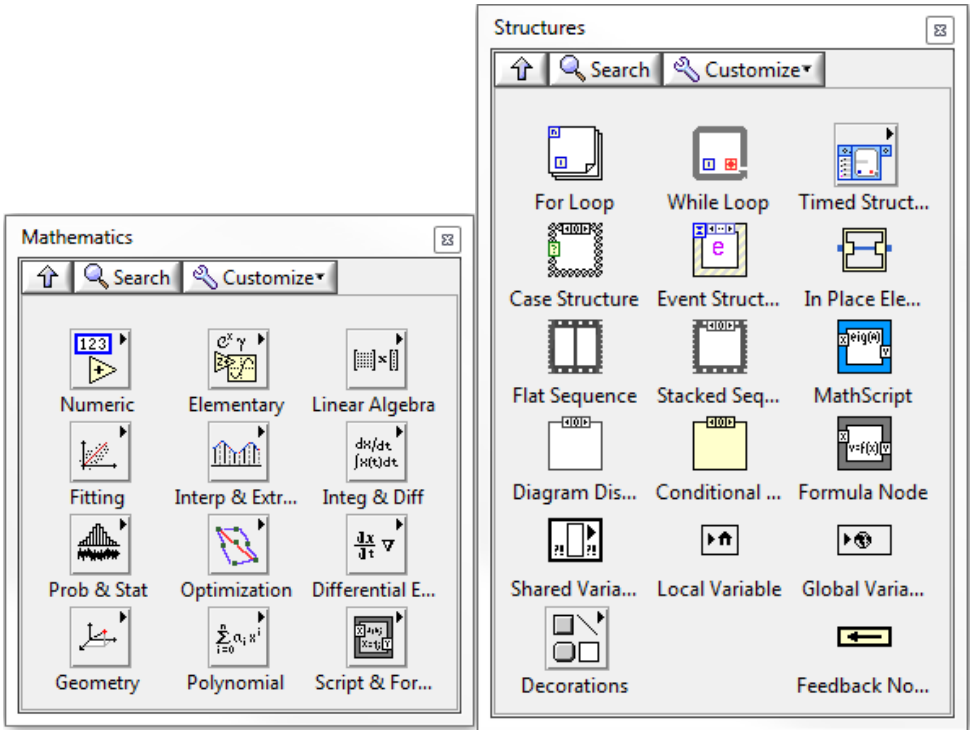
Zintegrowane środowisko LabVIEW jest narzędziem, które przy wykorzystaniu języka programowania G (graficznego) umożliwiającym (w pierwotnej koncepcji) tworzenie wirtualnych urządzeń pomiarowych, symulacji, zbieranie danych oraz ich przetwarzanie, wykorzystywane jest coraz częściej (dzięki rozbudowanym interfejsom komunikacyjnym, a także bogatej ofercie sprzętowej) do budowania m.in. systemów automatyki, sterowania oraz aplikacji czasu rzeczywistego. W połączeniu z dedykowanym sprzętem, pozwala na uruchamianie zaawansowanych aplikacji sterujących maszynami oraz robotami.

Zastosowane w pracy środowisko LabVIEW zostało wykorzystane do przeprowadzenia projektowania mechatronicznego systemu nadzorowania trójkołową platformą mobilną (w środowisku LabVIEW przeprowadzono symulacje komputerowe kinematyki oraz dynamiki układu, wirtualne prototypowanie, HILS oraz szybkie prototypowanie na obiekcie rzeczywistym). Zastosowanie sterownika cRIO systemu czasu rzeczywistego umożliwiło, w połączeniu z środowiskiem LabVIEW, osiągnięcie pełnej integracji projektowo-sprzętowej w ramach jednego obiektu, (trójkołowej platformy mobilnej), poruszającej się według proponowanej metody sterowania.

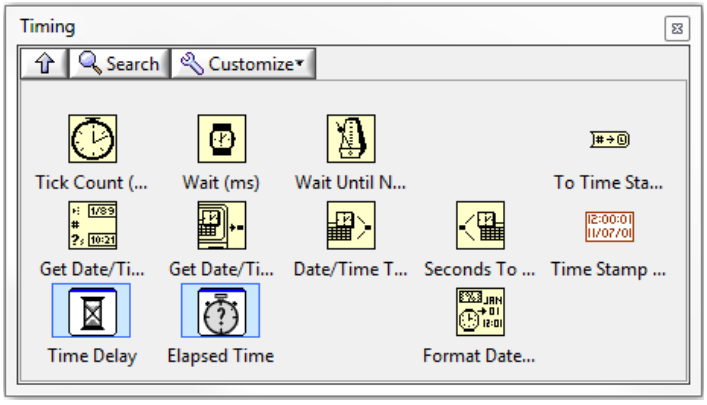
Konfiguracja programistyczna

W pracy wykorzystano środowisko programistyczne LabVIEW Professional Development w wersji 2010. Standardowa konfiguracja oprogramowania, poprzez bogatą bibliotekę elementów, umożliwia tworzenie wirtualnych systemów pomiarowych oraz innych aplikacji wspomagających pracę użytkownika. Tworzone oprogramowanie powstaje równolegle na dwóch płaszczyznach jednocześnie (*Front Panel* będącego interfejsem HMI oraz *Block Diagram*, w którym tworzona jest aplikacja), które w czasie rzeczywistym podlegają procesowi kompilacji. Wspomniana biblioteka elementów (dostępna z poziomu *Block Diagram*), stanowi odzwierciedlenie

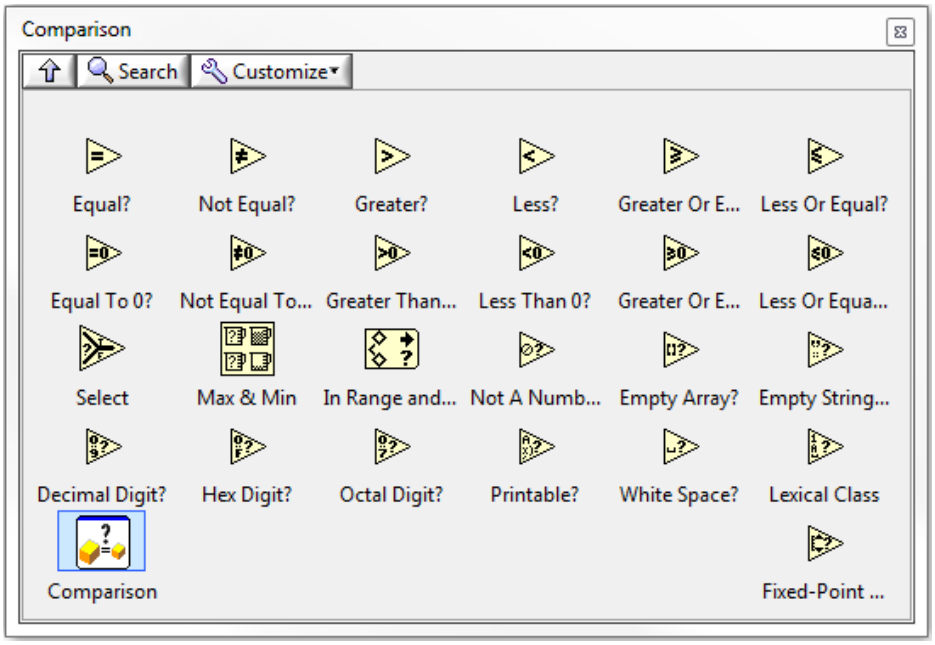
klasycznych, znanych z innych środowisk programistycznych (języków) grup funkcji, struktur czy rozkazów (rys. 10.1, 10.2, 10.3 i 10.4)



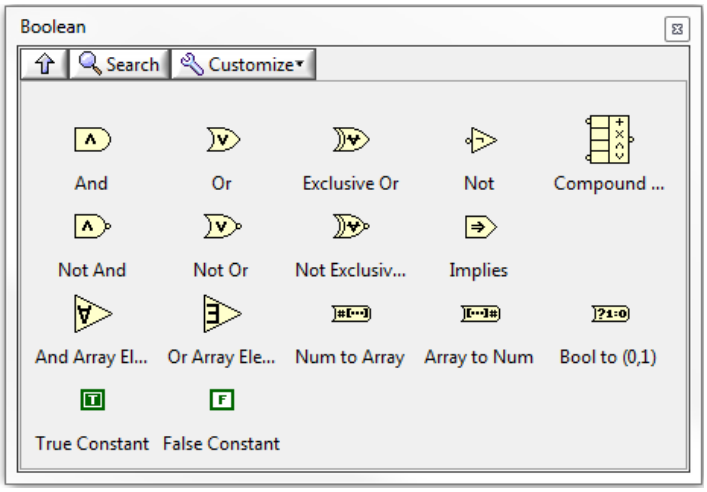
Rys. 10.1 Bloki funkcji matematycznych (lewy) oraz struktur (prawy)



Rys. 10.2. Bloki „obsługujące” czas aplikacji

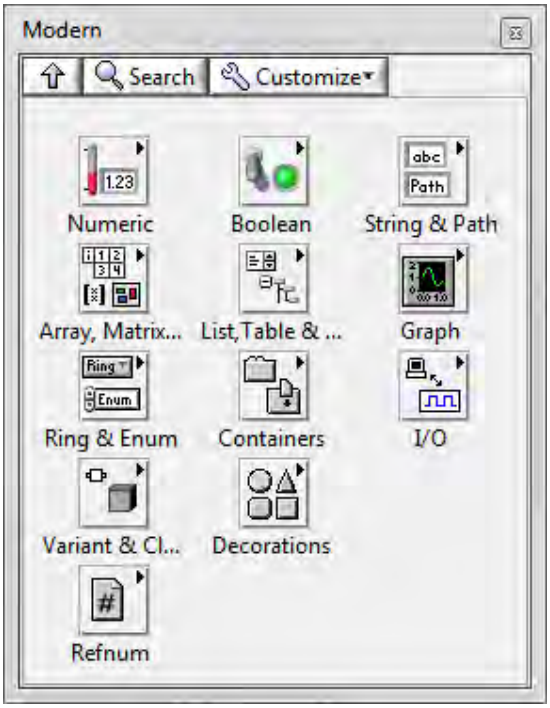


Rys. 10.3. Bloki funkcji porównania

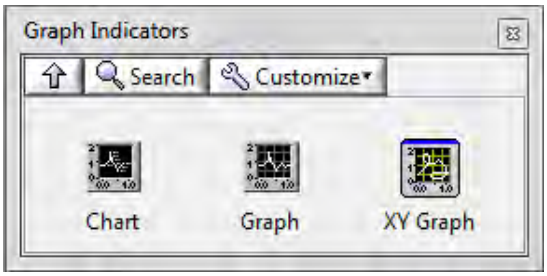


Rys. 10.4. Bloki funkcji logicznych

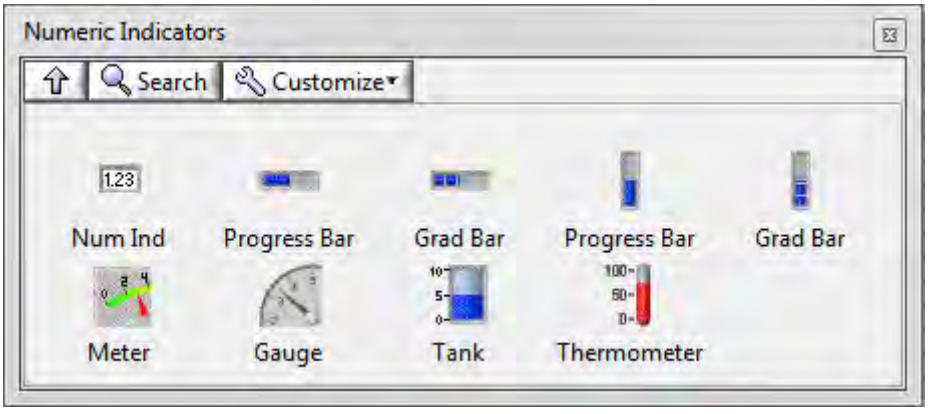
Elementy dostępne (wstawiane do programu z tzw. palety) z poziomu *Front Panel* umożliwiają tworzenie zaawansowanych interfejsów HMI, służących m.in. do wizualizacji pomiarów-wykresy, odczytu danych pomiarowych, zadawania wartości wejściowych danemu procesowi (rys. 10.5, 10.6, 10.7 i 10.9).



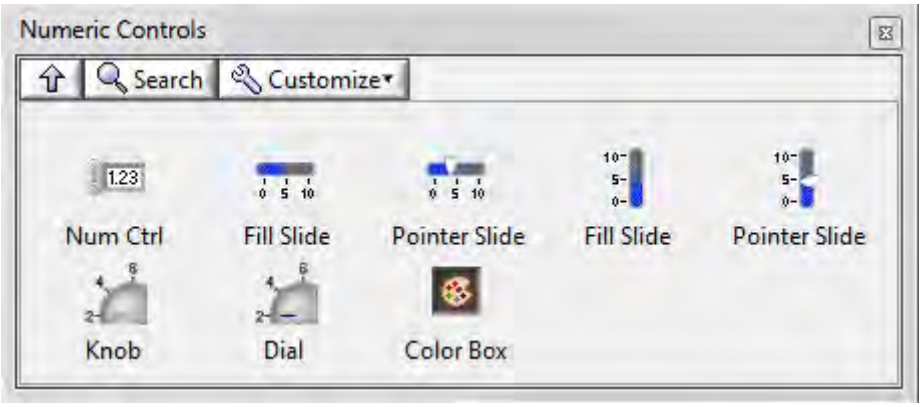
Rys. 10.5. Paleta grup elementów wejścia/wyjścia



Rys. 10.6. Paleta dostępnych bloków umożliwiających wizualizację wyników przeprowadzanych symulacji oraz pomiarów



Rys. 10.7. Elementy tworzące współczesny interfejs HMI



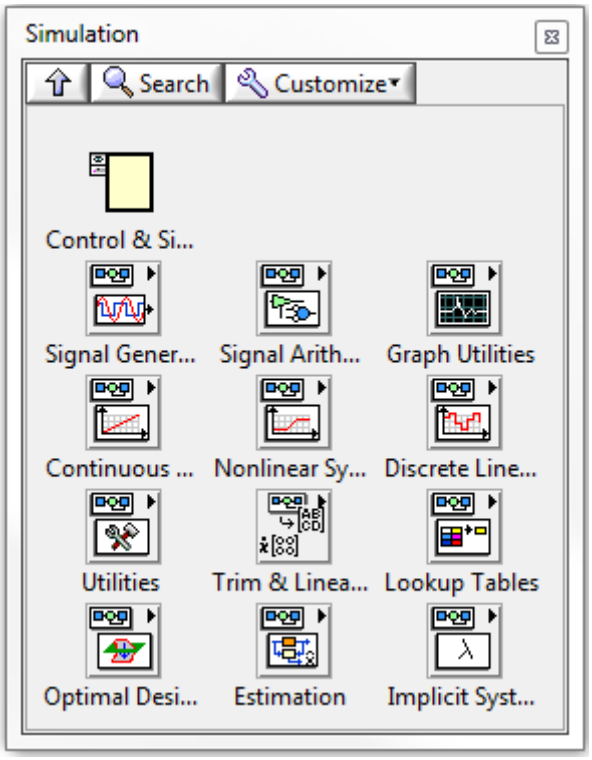
Rys. 10.8. Zestaw instrumentów kontrolnych, wzbogacających interfejs użytkownika

Tworzona aplikacja (wszystkie łączone bloki), zazwyczaj (choć można tworzyć również aplikację jednoiteracyjną) budowana jest na bazie pętli *While Loop*, wykonywanej do momentu wystąpienia zewnętrznego przerwania, czyli komendy *Stop*. Standardowe środowisko może zostać uzupełnione o dodatkowe (w zależności od potrzeb danej aplikacji) moduły programistyczne, wspierające dedykowany sprzęt bądź proces. Instalacja dodatkowego oprogramowania rozszerza paletę dostępnych elementów, specyficznych dla danego modułu.

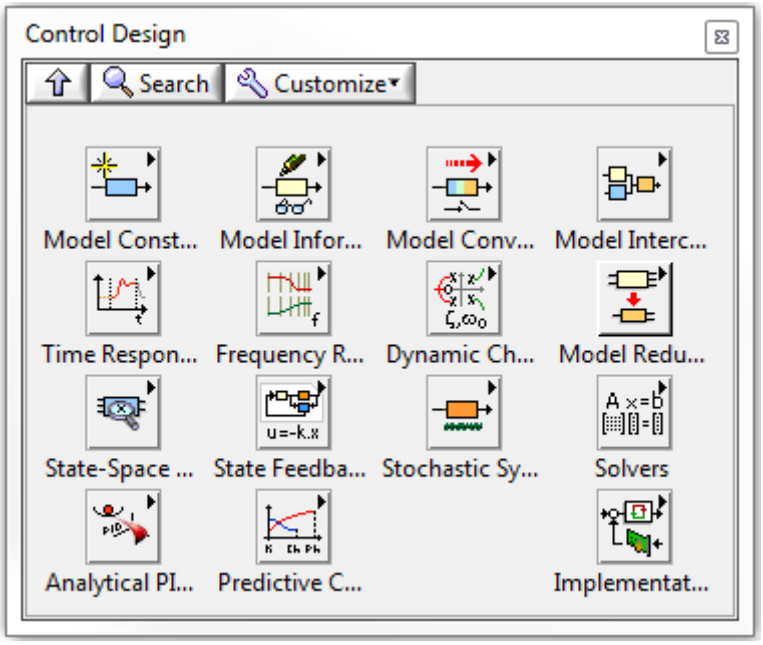
Dla potrzeb niniejszej pracy, środowisko LabVIEW zostało wyposażone w dodatkowe moduły umożliwiające realizację postawionych celów naukowych badań. I tak, wykorzystano:

Control Design & Simulations (CD&S)

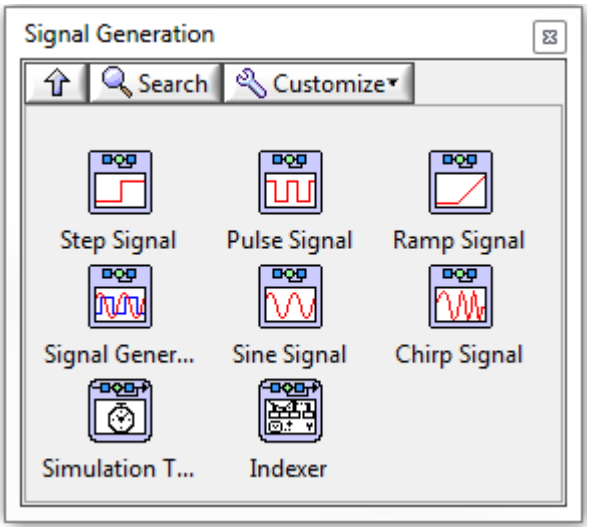
Potrzeba w pierwszej kolejności rozwiązania równań różniczkowych opisujących model obliczeniowy platformy mobilnej, a także w późniejszym czasie – implementacja docelowego systemu nadzorowania oraz przeprowadzona optymalizacja energetyczna układu, spowodowały konieczność stosowania oprogramowania, którego funkcjonalność umożliwia (dla danego kroku całkowania) generację rozwiązań przy wykorzystaniu danej metody numerycznej.
Instalacja modułu (CD&S) umożliwia, poprzez swoją bogatą paletę elementów, zamodelowanie dowolnego zjawiska czy układu oraz przeprowadzanie procesu jego weryfikacji dla dowolnych sygnałów wejściowych (rys. 10.9, 10.10, 10.11, 10.12 i 10.13).



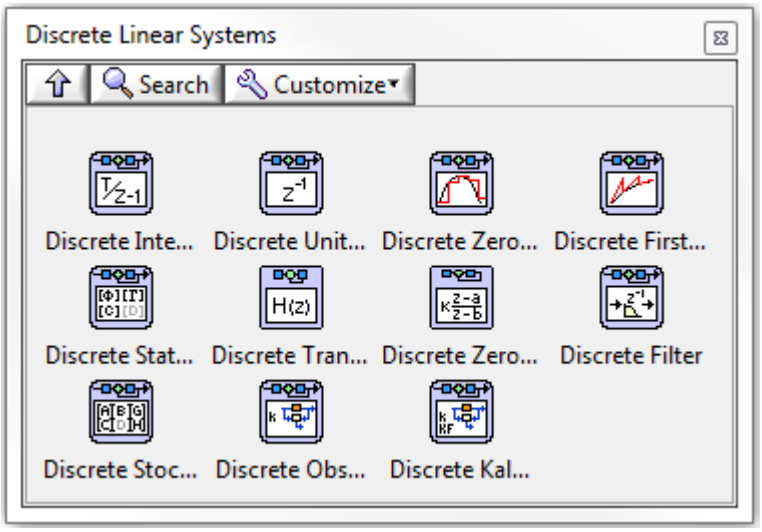
Rys. 10.9. Główna paleta modułu Control Design & Simulations



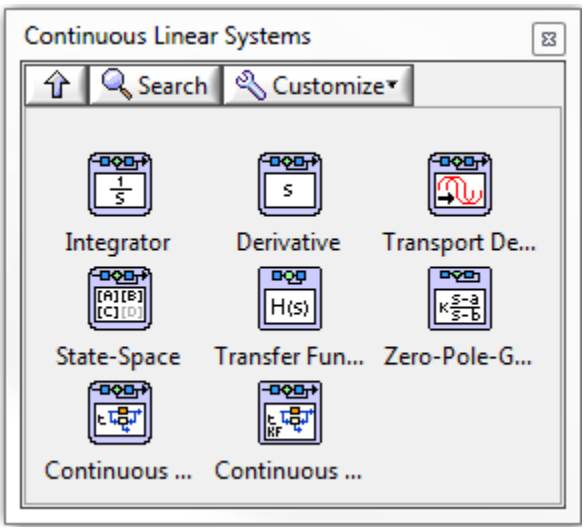
Rys. 10.10. Bloki umożliwiające tworzenie wyrafinowanych systemów sterowania



Rys.10.11. Zestaw generatorów sygnałów wejściowych

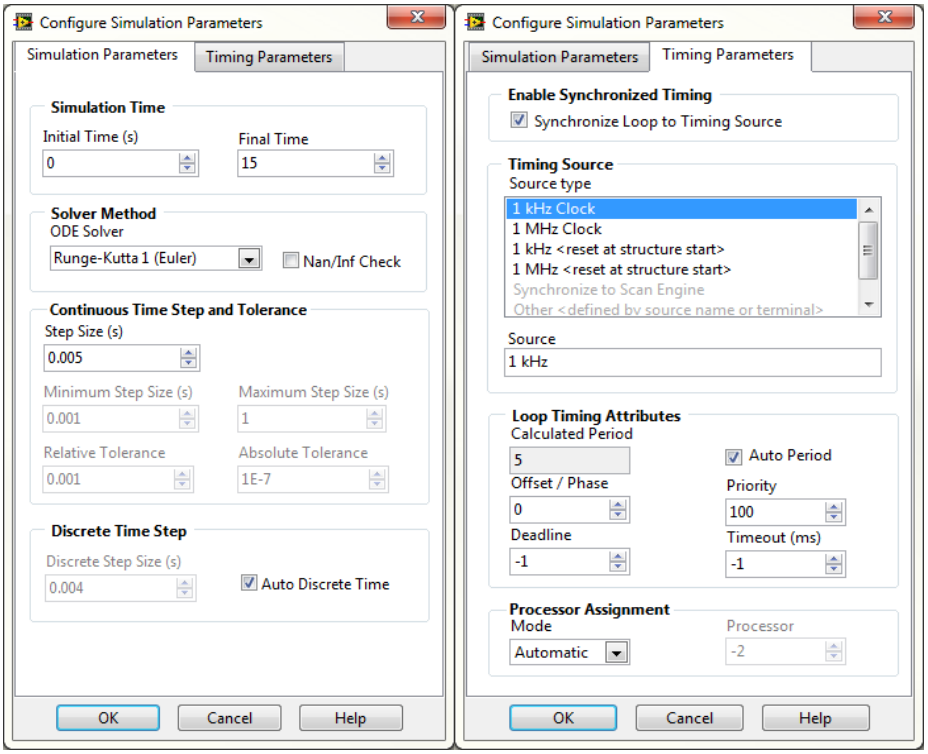


Rys. 10.12. Bloki układów dyskretnych



Rys. 10.13. Paleta bloków umożliwiających budowę modelu w dziedzinie Laplace'a

Zasada tworzenia aplikacji (w CD&S) jest identyczna, jak standardowe oprogramowanie LabVIEW. Jedyna różnica polega na konieczności umieszczenia dostępnych (dodatkowych dla danego modułu) elementów (z których powstaje aplikacja) w oddzielnej pętli *Control & Simulation Loop*. Dedykowana pętla dla tego modułu określa warunki przeprowadzania symulacji, tzn. metodę całkowania, długość symulacji, krok całkowania (jeśli jest to metoda stałokrokowa), bądź minimalną oraz maksymalną wartość kroku jeśli jest to metoda o zmiennym kroku (rys. 10.14)



Rys. 10.14. Okno konfiguracyjne procesu symulacji tworzonej aplikacji (z lewej strony). Okno konfiguracyjne procesu synchronizacji i wyboru zegara odniesienia (z prawej strony)

Niezwykle cennym elementem (który został wykorzystany w pracy) jest możliwość synchronizacji omawianej pętli z zegarem czasu rzeczywistego. Dla systemu Windows istnieje możliwość synchronizacji do zegara 1 kHz. W przypadku posiadania jednostki Real Time (w tym przypadku jednostki cRIO) - możliwość ta poszerza się o dodatkowy zegar 1 MHz).

Real Time

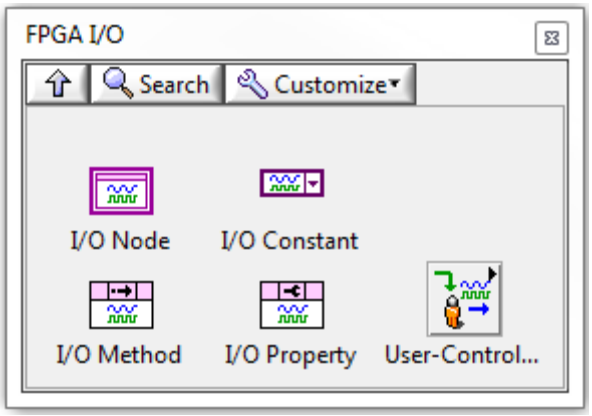
Aplikacje, w których istotnym aspektem jest przewidywalność wykonania danego zadania, bazują na rozwiązaniach deterministycznych, pozwalających na dotrzymanie reżimu czasowego, a w konsekwencji – na wykonanie danego zadania w określonej czasoprzestrzeni przyczynowo–skutkowej. Proponowana metoda nadzorowania bazującą na algorytmie *on-line* spowodowała konieczność stworzenia aplikacji czasu rzeczywistego, co w konsekwencji doprowadziło do konieczności zastosowania modułu Real Time. Instalacja modułu RT, podobnie jak to miało miejsce dla modułu CD&D, powoduje pojawienie się dodatkowej palety programistycznej, wspomagającej za pomocą dostępnych bloków tworzenie aplikacji czasu rzeczywistego. Kreacja aplikacji w module Real Time wymaga umieszczenia wszystkich bloków (bądź pętli) w pętli *Timed Loop* zewnętrznej (do pozostałych). W pracy stworzona aplikacja (system nadzorowania) głównie za pomocą modułu CD&D (aplikacja dla jednostki FPGA zapisywana jest oddzielnie) została całkowicie umieszczona w pętli czasu rzeczywistego (patrz rys. 5.5) oraz z nią zsynchronizowana. Właściwości *Timed Loop* umożliwiają nastawę czasu trwania wykonania pętli, okresu oczekiwania pomiędzy wykonaniem poszczególnej pętli, zwłokę oraz ustawienia priorytetów, jeżeli w ramach aplikacji wykonywanych jest więcej niż jedna pętla.

FPGA

Zastosowany przy budowie sterownik cRIO, ze względu na swoją konstrukcję posiada możliwość wykorzystania wewnętrznego, integralnego modułu FPGA, firmy Xilinx. Układ logiczny sterownika, a także moduł FPGA LabVIEW pozwolił, dzięki wydzieleniu części wykonywanego kodu algorytmu (wykonywanego w procesorze Real Time oraz układzie FPGA) na osiągnięcie ostatecznej optymalizacji systemu.

Programowanie jednostki FPGA sterownika odbywa się poprzez stworzenie nowej aplikacji w ramach LabVIEW FPGA (nie odbywa się to w ramach stosowania poprzednio opisanych modułów), również w ramach dostępnych pętli m.in. *Timed Loop, While Loop*. Ze względu na specyfikę układu logicznego, instalacja nowego typu sprzętu powoduje pojawienie się na dostępnej palecie programisty nowych elementów (bloków) specyficznych dla danej jednostki sterownika oraz modułów wejść/wyjść. Jedną z główniejszych palet (z racji pełnionych funkcji) jest FPGA I/O

(rys. 10.15), w której dostępne są elementy bezpośrednio odpowiedzialne za dostęp do wyjść danego układu (w tym przypadku NI-9505) oraz umożliwiające w czasie rzeczywistym odczyt stanów wejść układu (NI-9505, enkoder).



Rys. 10.15. Bloki obsługi wejścia/wyjścia

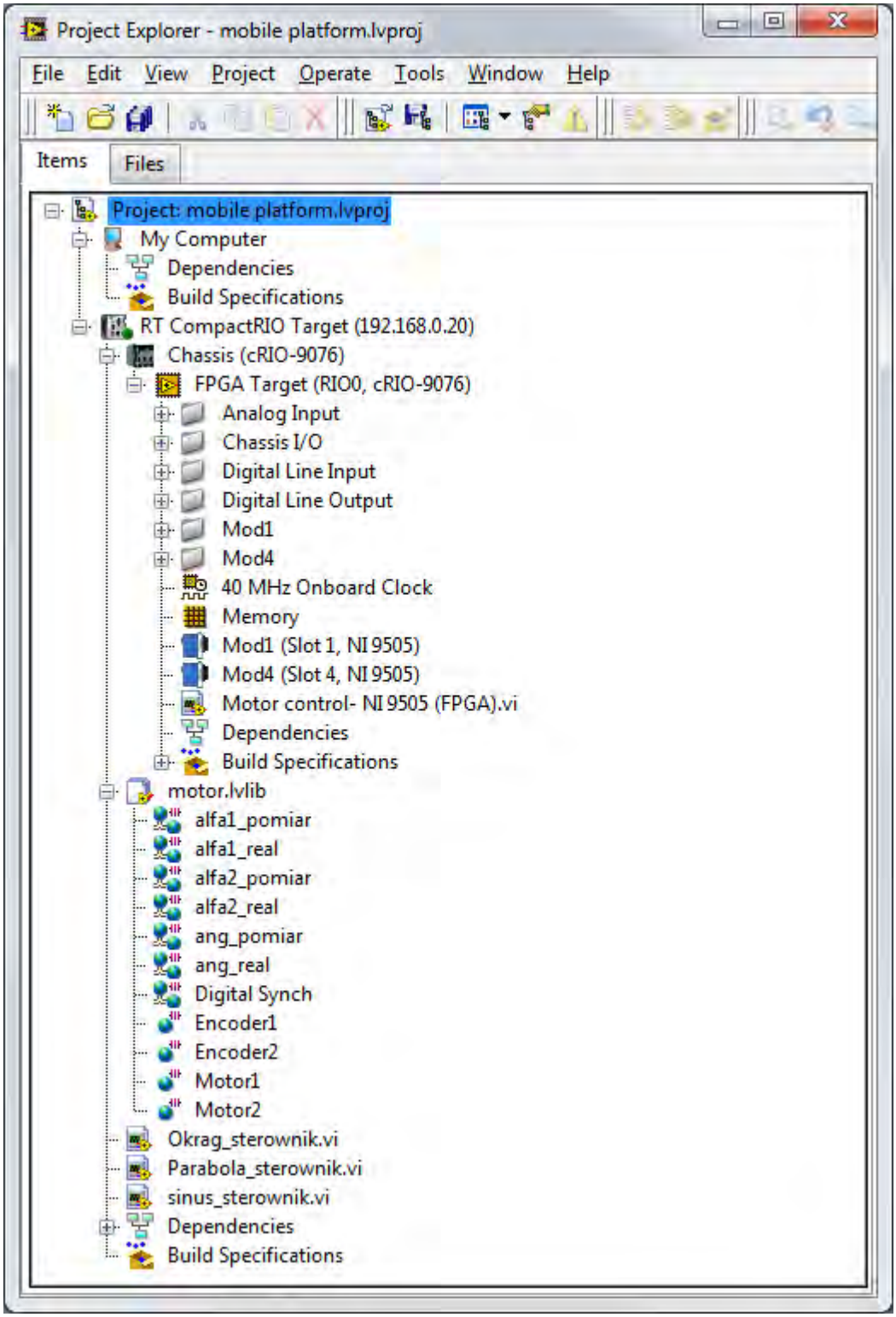
Kompilacja kodu (przedstawiona schematycznie na rys. 5.6), powoduje powstanie pliku *Bitfile*, który może zostać wywołany z poziomu aplikacji nadrzędnej (w tym przypadku, aplikacją nadrzędną jest zaimplementowany algorytm). Schemat tego wywołania został zademonstrowany na rys. 5.5. W czasie trwania kompilacji, oprócz procesu tworzenia pliku *Bitfile*, stworzony program zapisywany jest do układu i tam wykonywany.

Struktura projektu

Proces rozwoju oprogramowania, programowanie sterownika cRIO oraz końcowa walidacja systemu (platformy mobilnej) odbywała się w ramach stworzonego projektu (warunek konieczny) – rys. 10.16. Tworzenie projektu polega na dodawaniu kolejnych jego części tak, aby w końcowej fazie powstała jednolita programistyczno sprzętowa struktura powiązana wzajemnymi zależnościami.

Pierwszym krokiem jest zdefiniowanie tzw. *Targetu* czasu rzeczywistego – sterownika, w którym wykonywany jest program (w tym przypadku aplikacja sterująca platformą mobilną). Komunikacja ze sterownikiem odbywa się poprzez sieć Ethernetową, dlatego niezbędne jest określenie adresu IP jednostki (proces konfiguracji odbywa się poprzez oprogramowanie *Measurement & Automation*

dostarczane wraz ze sterownikiem). W tym przypadku jednostce cRIO-9076 nadano adres 192.168.0.20. Kolejnym krokiem jest dodanie obudowy sterownika – *Chassis*. Należy zaznaczyć, że firma National Instrument oferuje różne moduły sterownika, oraz różne obudowy, które w zależności od potrzeb aplikacji można ze sobą łączyć. W tym przypadku autor zastosował sterownik zintegrowany (mechanicznie połączony sterownik z obudową). Dodatkowo wartym jest podkreślenia, że FPGA rezyduje w obudowie, a nie w module sterownika. W niniejszej pracy dodanie obudowy spowodowało automatycznie dodanie układu FPGA. W ramach obudowy dodawane są kolejno moduły wejścia/wyjścia (NI-9505) oraz aplikacja stworzona w ramach realizowanego badania (w tym przypadku *Motor Control NI-9505 (FPGA)*, a odpowiedzialną za formowanie fali PWM, obsługę modułów sterownika silników oraz enkoderów). Jednym z ostatnich etapów jest dodanie rozwijanego oprogramowania sterownika (*Okrag_sterownik*, *sinus_sterownik oraz Parabola_sterownik*). Z tego również poziomu odbywa się proces zapisu (*deployment*) danego programu do sterownika. Dla zapewnienia swobodnego przepływu danych pomiędzy poszczególnymi modułami tworzonego oprogramowania, dokonuje się definicji zmiennych (w gałęzi motor).



Rys. 10.16. Struktura projektu systemu nadzorowania platformy mobilnej

10.2. Środowisko Maple

Równania różniczkowe opisujące model obliczeniowy przyjętego obiektu sterowania (tj. platformy mobilnej, rys. 3.2) zostały rozwiązane (oraz zweryfikowane w procesie porównawczym z wynikami uzyskanymi w środowisku LabVIEW) w pakiecie Maple (wersja 13).

Głównym założeniem wykorzystanego oprogramowania jest proces wspomagania użytkownika w obliczeniach numerycznych oraz poszukiwaniu rozwiązań złożonych równań i nierówności. Istotną zaletą jest możliwość zapisu równań w postaci symbolicznej (notacja matematyczna) oraz uzyskaniu rozwiązania również w takiej samej postaci (tj. parametrycznej). Oprogramowanie dzięki swojej rozbudowanej funkcjonalności, umożliwia również tworzenie zaawansowanych interfejsów użytkownika (HMI), a także – generowanie rozwiązań w postaci graficznej, również wykorzystanych w pracy.

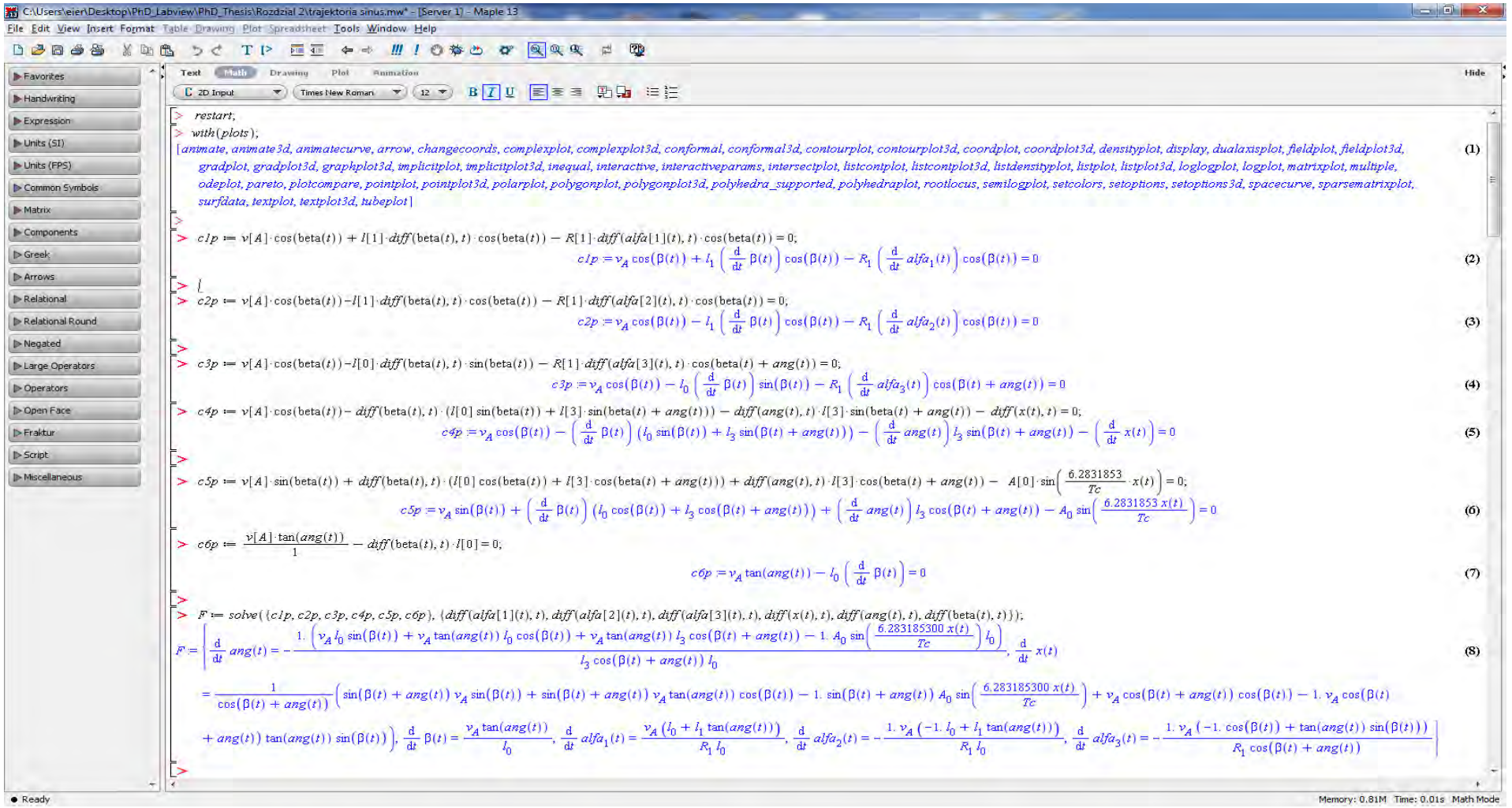
Zapis i rozwiązywanie równań

W pracy wykorzystano dwie zasadnicze funkcje programu. Po pierwsze, rozwiązano układ równań różniczkowych, a w następnej kolejności metodami numerycznymi, przy zadanym kroku całkowania wygenerowano rozwiązania w postaci graficznej).

Praca z programem polega na wpisywaniu ciągu komend (w arkuszu dialogu), które po zatwierdzeniu klawiszem klawiatury *Enter*, są zapisywane do pamięci komputera. Jeżeli komendy i dane wprowadzane są poprawnie oraz w odpowiedniej kolejności, program generuje w trybie *on-line* rozwiązania (na podstawie wcześniej wprowadzonych danych). W przypadku wystąpienia konieczności korekty wprowadzonych wcześniej danych (w dowolnym miejscu tworzenia kodu – arkusza dialogu), program umożliwia modyfikacje tego fragmentu kodu, późniejsze jego wskazanie oraz ponowne przeprowadzenie procesu wykonawczego – *Execute selected groups* (tylko dla zmodyfikowanego/wskazanego fragmentu) Istnieje, również możliwość ponownego wykonania całego „arkusza dialogu" - *Execute the entire worksheet*.

W celu identyfikacji danego równania bądź złożonej instrukcji, nadaje im się unikatowe dla całego arkusza nazwy (zmienne).

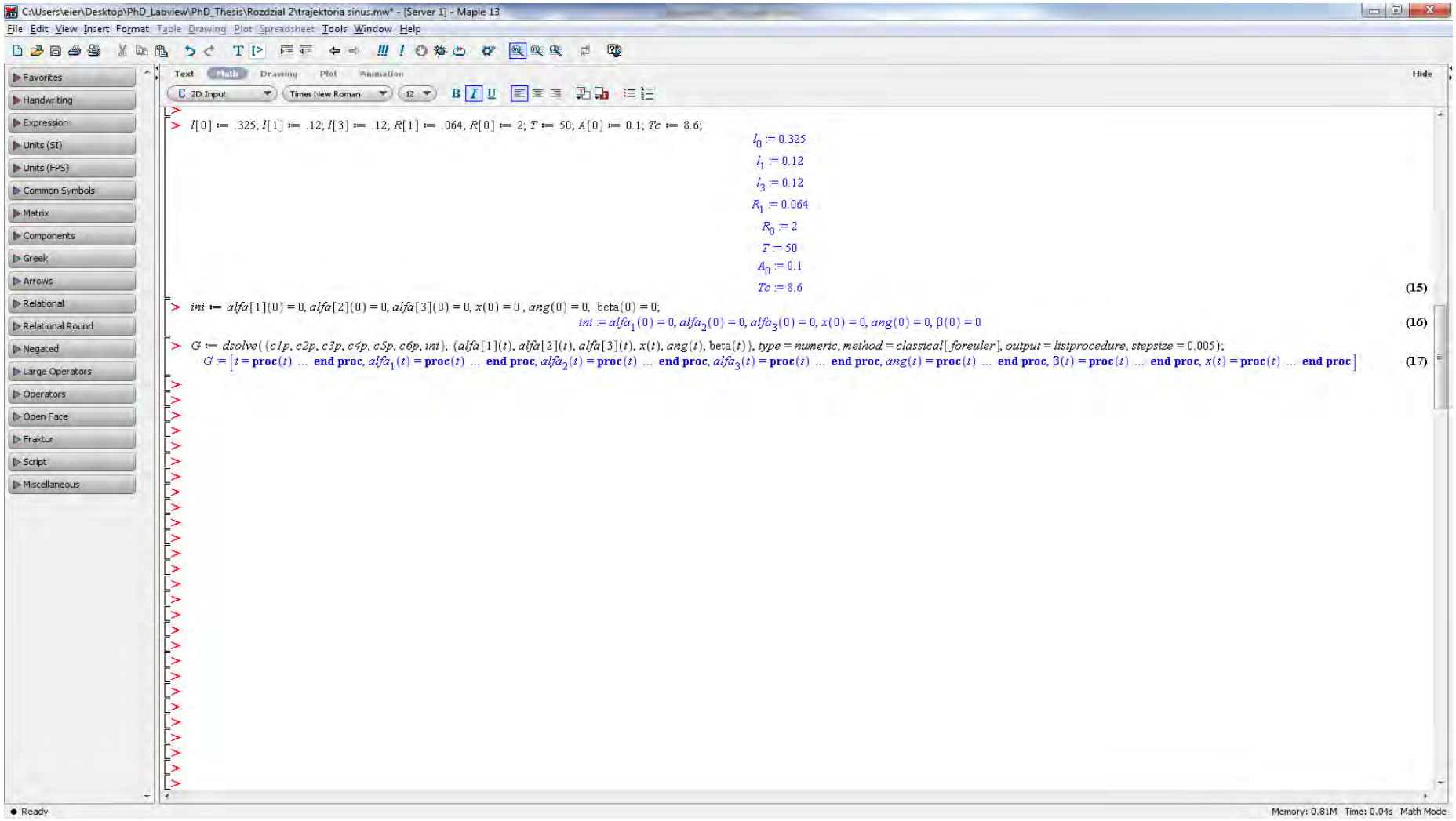
W przypadku równań różniczkowych opisujących ruch trójkołowej platformy mobilnej po trajektorii typu sinus (zależności (3.5), (3.13), oraz (3.14)) zapis równań do programu Maple został przedstawiony na rys. 10.17. Należy zauważyć, że autor każdemu z równań nadał unikatową zmienną c1p, c2p, c3p, c4p, c5p oraz c6p. Program po zinterpretowaniu wprowadzonych danych, odpowiada zapisem równania w postaci matematycznej (wprowadzenie danych odbywa się metodą symboliczną np. zapis pochodnej kąta β odpowiada instrukcji *diff(beta(t),t)*). Należy zaznaczyć, że parametrem jest tu zmienna *t*, dlatego pochodne obliczane są względem tego parametru. Rozwiązanie równań (tą postać zaimplementowano do programu LabVIEW) możliwe jest poprzez wywołania procedury *solve()*. W treści wywołania procedury zamieszcza się nazwę (zmienną) wskazującą na dane równanie, oraz zmienną, która z danego równania ma być wyznaczona.



Rys. 10.17. Procedura zapisu oraz rozwiązania równań w środowisku Maple

Generacja rozwiązań metodami numerycznymi

Funkcjonalność programu Maple pozwoliła autorowi na wygenerowanie rozwiązań (przy pomocy dostępnych metod numerycznych) dla zawiłych równań różniczkowych opisujących nieliniowy obiekt badań (platformę mobilną). W programie do rozwiązania równań różniczkowych c1p, c2p, c3p, c4p, c5p oraz c6p, użyto procedury *dsolve()* (należy znaczyć, że wpisanie procedury poprzedza nazwa zmiennej dla danego wywołania – w tym przypadku *G*, rys. 10.18). W celu przeprowadzenia procedury numerycznego rozwiązania zdefiniowanych równań, należy dla każdego stałego współczynnika określić wartości liczbowe, a także o ile to konieczne, również warunki początkowe (*ini*). Parametrami wywoływanej procedury są m.in. metoda całkowania (*method*), a także krok całkowania (*stepsize*). Wywołanie procedury z tak określonymi parametrami, rozpoczyna proces numerycznego rozwiązania równań różniczkowych. Rozwiązania przechowywane są w pamięci komputera i aby je odczytać, należy zastosować kolejne procedury umożliwiające wizualizację otrzymanych wyników obliczeń. Czas numerycznego generowania rozwiązań jest ściśle uzależniony od liczby równań oraz ich stopnia skomplikowania, a także – od długości kroku całkowania, metody oraz wydajności maszyny liczącej (komputera PC, jego procesora oraz wielkości pamięci operacyjnej).

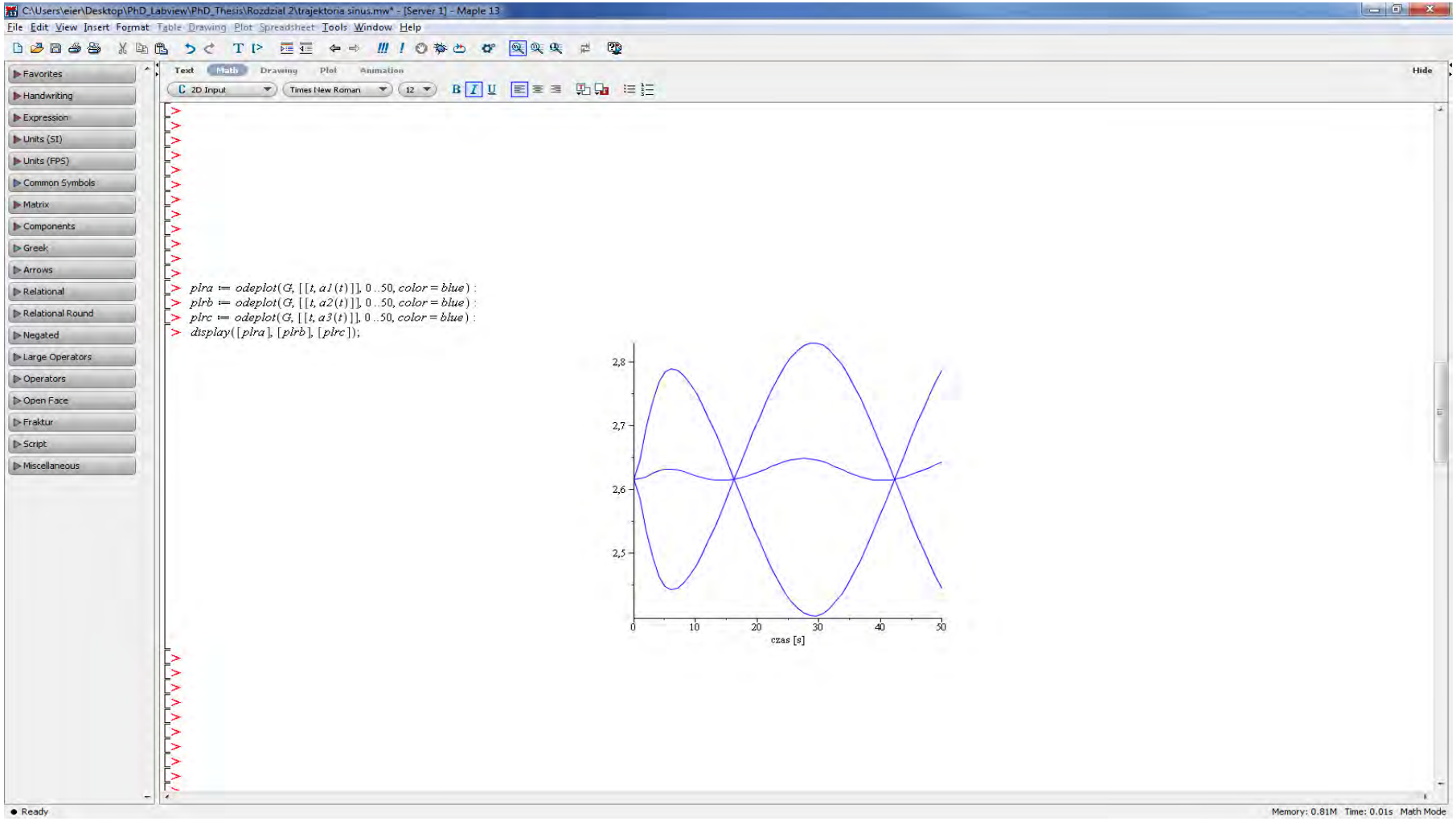


Rys. 10.18. Procedura rozwiązania równań różniczkowych w środowisku Maple

Wizualizacja

Otrzymane rozwiązania równań różniczkowych (które w niniejszej pracy zostały wygenerowane w celu porównania wyników z rozwiązaniami otrzymanymi w programie LabVIEW) wywołuje się (z pamięci komputera) procedurami umożliwiającymi na ekranie komputera generowanie wykresów dla zmiennych danych. Pierwszy etap generacji rozwiązań wiąże się z procesem przygotowania takiego rozwiązania. Odbywa się to za pomocą procedury *odeplot()* wywoływanej z parametrami. W tym przypadku parametrami są: nazwa zmiennej dla której przeprowadzono obliczenia numeryczne, nazwa zmiennej dla której mają być wykreślone rozwiązania, przedział zmienności wywoływanej funkcji (w tym przypadku parametru platformy mobilnej) oraz kolor, w którym rysowana jest krzywa. Należy zaznaczyć (jak to miało miejsce poprzednio), że dla tak przygotowanej procedury istnieje konieczność nadania nazwy (przykładowo dla prędkości poszczególnych kół są to *plra*, *plrb* oraz *plrc*, rys. 10.19).

Kolejnym etapem jest wizualizacja przygotowanych rozwiązań. Odbywa się to poprzez wywołania procedury *display()*, z parametrami, którymi w tym przypadku są unikatowe nazwy przygotowanych rozwiązań.



Rys. 10.19. Wizualizacja rozwiązań w środowisku Maple