POLITECHNIKA POZNAŃSKA

WYDZIAŁ AUTOMATYKI, ROBOTYKI I ELEKTROTECHNIKI

Instytut Automatyki i Robotyki

Praca dyplomowa inżynierska

AUTOMATYCZNA STABILIZACJA RUCHU WIROWEGO RAKIETY SPORTOWEJ

Natalia Wiśniewska, 144558

Andrzej Rafalski, 144502

Promotor

dr hab. inż. Maciej Marcin Michałek, prof. PP

POZNAŃ 2023

Streszczenie

Celem pracy dyplomowej jest zaprojektowanie oraz weryfikacja działania układu automatycznej stabilizacji ruchu wirowego rakiety sportowej. Praca zawiera implementację symulatora ruchu rakiety w sześciu stopniach swobody w programie MATLAB, projekt układu sterowania prędkością rotacji w osi podłużnej (ang. *roll*), projekt komputera pokładowego oraz wyniki eksperymentalnej weryfikacji układu sterowania.

Zaprojektowany układ sterowania wykorzystuje sterownik ADRC (ang. *Active Disturbance Rejection Control*). Jego zadaniem jest zerowanie prędkości rotacji rakiety w osi podłużnej. Symulacyjna weryfikacja układu sterowania została przeprowadzona przy użyciu zaprojektowanego modelu rakiety w programie MATLAB.

Następnie skuteczność zaprojektowanego oraz zaimplementowanego układu sterowania została zweryfikowana podczas eksperymentu przeprowadzonego w tunelu aerodynamicznym. W celu weryfikacji działania układu w stałych oraz zmiennych warunkach przepływu powietrza zrealizowano dwa scenariusze testowe. Wyniki testów potwierdziły skuteczność działania układu sterowania. Pozwoliły także na wyciągnięcie wniosków co do dalszego rozwoju projektu.

Spis treści

Rozdział 1

Wprowadzenie

1.1 Cel i zakres pracy

System sterowania prędkością rotacji w kontekście rakiety pozwala na zadanie zerowej prędkości wokół osi podłużnej (ang. *roll*), równoległej do korpusu rakiety [41]. Pojęcie rakiety sportowej rozumiane jest w kontekście rakiety badawczej, która jest wykorzystywana podczas sportowych zawodów rakietowych.

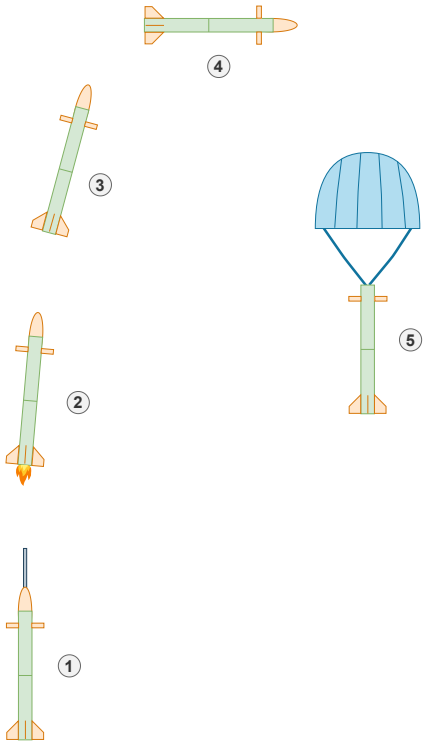
Praca dyplomowa obejmuje implementację symulatora ruchu rakiety w sześciu stopniach swobody, projekt płytki drukowanej komputera pokładowego oraz enkodera magnetycznego, przygotowanie oprogramowania dla komputera pokładowego, zaprojektowanie sterownika do układu sterowania prędkością ruchu wirowego, zaprojektowanie mocowania elektroniki do stanowiska testowego oraz eksperymentalną weryfikację działania układu sterowania. Projekt płytki drukowanej komputera pokładowego i modułu enkodera magnetycznego oraz implementację modelu rakiety w programie MATLAB wykonał Andrzej Rafalski. Za wykonanie oprogramowania, projekt mocowania elektroniki oraz projekt sterownika odpowiedzialna była Natalia Wiśniewska. Praca dyplomowa została wykonana we współpracy z kołem naukowym PUT Rocketlab [8].

1.2 Znaczenie tematu pracy

Typowa misja rakiety badawczej składa się z pięciu etapów widocznych na rys. 1.1. Kolejno fazy misji można określić jako: start rakiety, faza lotu napędzana silnikiem, lot po wypaleniu silnika, osiągnięcie apogeum wysokości oraz faza odzysku rakiety (np. z wykorzystaniem spadochronu). W zależności od charakteru misji w poszczególnych etapach lotu wykonywane są dodatkowe zadania.

Podczas drugiego i trzeciego etapu widocznych na rys. 1.1 rakieta, wznosząc się, obraca się wokół osi podłużnej. Bezpośrednią przyczyną powstawania prędkości rotacji są naturalne występujące niesymetryczności rakiety. Można do nich zaliczyć przekrzywienie stateczników względem osi podłużnej wynikające z błędu montażu, niesymetrycznie rozłożona masa czy niesymetryczna głowica [22]. Dokonana analiza danych telemetrycznych dla rakiet ARAV-B wykazała, że wprowadzenie przekrzywienia lotek stabilizujących o kąt $0,5°$ prowadzi do prędkości rotacji $540\frac{°}{s}$ dla wartości prędkości około 1 liczby Macha (wyjaśnienie pojęcia w rozdziale 2) [41][27]. Z raportu dla rakiety dwustopniowej Sidewinder-Arcas [28], dla wychylenia stateczników o $0,2°$ przy wartości prędkości około 2,5 liczby Macha, prędkość rotacji wyniosła $1260\frac{°}{s}$. Małe kąty błędu montażu lotek stabilizujących prowadzą, więc do znacznych prędkości rotacji. Wytwarzanie rakiety badawczej w reżimie precyzyjnego montażu jest czasochłonne i kosztowne. Oprócz tego niemożliwe jest całkowite wyeliminowanie niedokładności wykonania. W związku z tym istnieje potrzeba zaprojektowania układu stabilizacji ruchu wirowego rakiety.

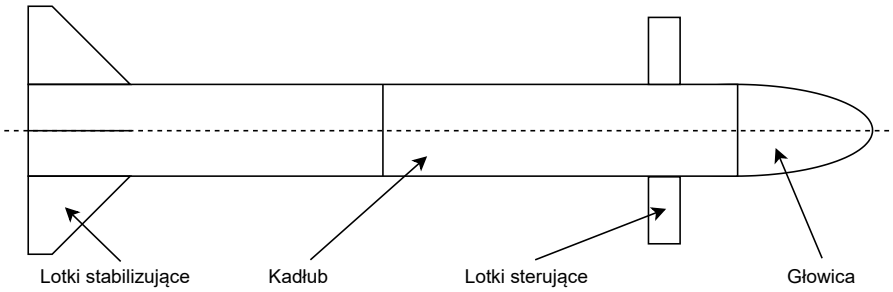
Istnieją dwie możliwości sterowania prędkością rotacji. Pierwszą z nich jest zerowanie prędkości kątowej, która pozwala na zebranie dokładniejszych odczytów z czujników oraz uzyskanie obrazu z kamer



Rysunek 1.1: Ogólny schemat misji rakiety badawczej. Gdzie: 1 — start rakiety, 2 — lot napędzany silnikiem, 3 — lot po wypaleniu silnika (bezwładnościowy), 4 — osiągnięcie apogeum wysokości, 5 — faza odzysku rakiety (np. z wykorzystaniem spadochronu).

o lepszej jakości. Ma to również zastosowanie w przypadku fazy odzysku rakiety badawczej. Redukcja prędkości rotacji zapobiega niechcianym zachowaniom spadochronu odzyskowego. Drugą możliwością jest uzyskanie niezerowej prędkości rotacji, pomagającej w stabilizacji lotu. W tej pracy za główny cel zostało postawione zadanie zerowania prędkości rotacji. Korzyści, które przynosi realizacja wybranego celu sterowania, pozwala na rozwój rakiety badawczej pod względem fuzji danych z czujników (ang. *sensor fusion*), algorytmów przetwarzania obrazów oraz zwiększa szanse powodzenia misji.

Istnieje kilka rozwiązań pozwalających na sterowanie prędkością rotacji rakiety. Podejścia te opierają się na kombinacji różnych konfiguracji lotek. Do często spotykanych układów lotek należy konfiguracja składająca się z tylnych powierzchni sterowych (ang. *tails*), umieszczonych na ogonie rakiety oraz lotek sterujących, umieszczonych w konfiguracji kaczki (ang. *canards*), to znaczy przed środkiem ciężkości rakiety (rys. 1.2) [41, 22, 36, 13].



Rysunek 1.2: Poglądowy widok rakiety z wybraną kombinacją lotek

Tylne ruchome powierzchnie sterowe często poprzedzane dodatkowymi lotkami są bardziej efektywne dla dużych kątów natarcia (wyjaśnienie pojęcia w rozdziale 2). Wywołują one mniejszy moment siły w

osi podłużnej oraz powodują mniejsze obciążenia dla systemu wykonawczego sterującego wychyleniem tylnych lotek [22]. Zastosowanie lotek sterujących w konfiguracji kaczki wymaga nieruchomych lotek umieszczonych na tylnej części korpusu rakiety zwanych statecznikami, które zapewniają stabilność trajektorii rakiety. Dostosowywanie powierzchni lotek obu rodzajów oraz położenie ruchomych powierzchni sterowych względem czubka rakiety, pozwala natomiast na zapewnienie odpowiedniej manewrowości (zapewnia efektywność lotek sterujących) oraz stabilności (wyjaśnienie pojęcia w rozdziale 2). W przypadku zastosowania tej kombinacji do sterowania prędkością rotacji wystarczą tylko dwie ruchome powierzchnie sterowe, jednak są one efektywne tylko dla małych kątów natarcia rakiety [22].

W ramach pracy dyplomowej został zaprojektowany układ sterowania dla rakiet poddźwiękowych (nieprzekraczających wartości prędkości 0,8 liczby Macha), statycznych (wyjaśnienie pojęcia w rozdziale 2), których profil misji najczęściej polega na wyniesieniu ładunku badawczego na określony pułap nieprzekraczający górnej warstwy troposfery (ok. 11 km wysokości) [18]. Planowana trajektoria lotu rakiety nie obejmuje, więc dużych kątów natarcia tzn. wykraczających poza $\pm 15^\circ$ [22].

Wobec powyższych założeń została wybrana konfiguracja kaczki, przedstawiona na rys. 1.2, zapewniająca uproszczony montaż oraz projektowanie podzespołów. Ponadto istnieje możliwość rozbudowy wybranej konfiguracji lotek zapewniającej efektywność dla większych kątów natarcia [22]. Same lotki sterujące działają na zasadzie wychylenia przeciwsobnego.

1.3 Struktura pracy dyplomowej

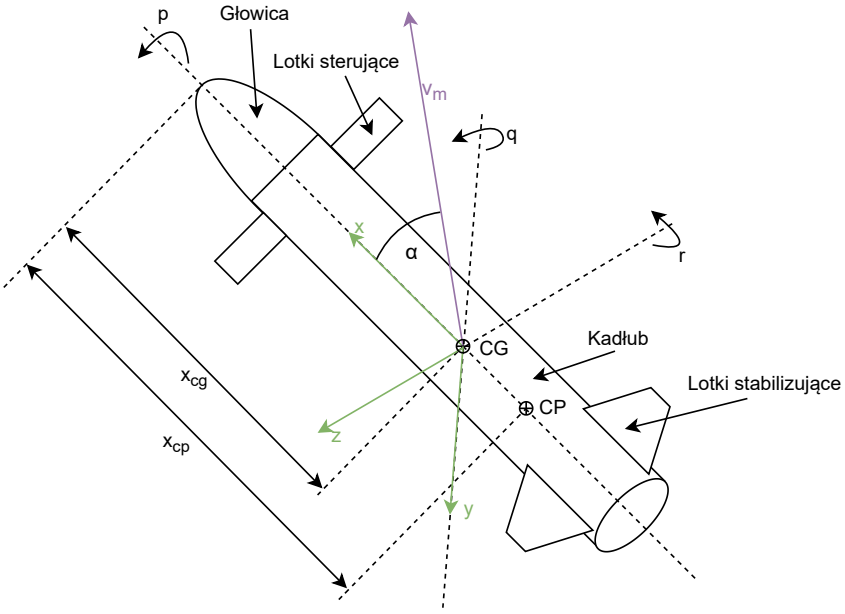
Praca kolejno zawiera wprowadzenie teoretyczne, które obejmuje równania oraz przyjęte uproszczenia dla analizowanego ruchu rakiety, a także wyjaśnienie projektu układu sterowania ADRC (ang. *active disturbance rejection control*) oraz wyprowadzenie adekwatnych równań dla obiektu sterowania. Następnie przedstawione są: implementacja symulatora ruchu rakiety 6 DoF, sterownika w programie MATLAB/Simulink oraz wyniki symulacji. W dalszej kolejności zaprezentowane są wyniki projektu, w tym uruchomienie płytki drukowanej, integracja podzespołów oraz test działania układu sterowania w tunelu aerodynamicznym. Praca kończy się podsumowaniem.

Rozdział 2

Równania ruchu rakiety

2.1 Fizyczne podstawy równań modelu

Równania opisujące modelu rakiety zostały wyprowadzone na podstawie [17] oraz [34]. W celu matematycznego opisania modelu rakiety wprowadzono szereg oznaczeń wielkości fizycznych.



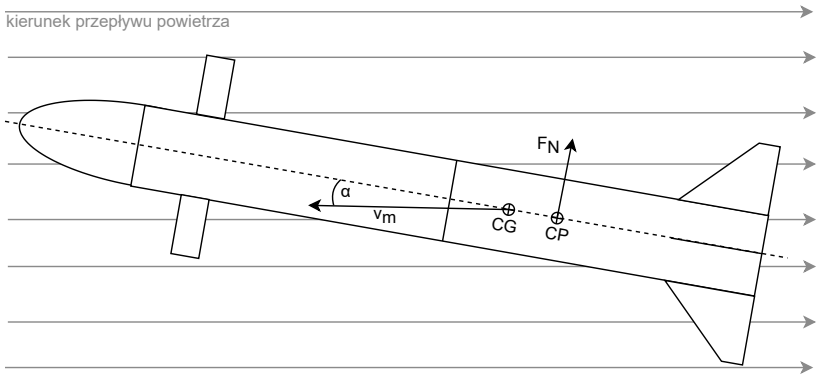
Rysunek 2.1: Lokalny układ współrzędnych rakiety oraz podstawowe wielkości

Jak przedstawiono na rysunku 2.1, przyjęto, że lokalny układ współrzędnych rakiety (osie x, y i z) jest prawoskrętnym układem współrzędnych. Oś x skierowana jest wzdłuż osi podłużnej rakiety w kierunku jej czubka (ang. *nose cone*). Prędkości rotacji rakiety wokół osi x, y oraz z (inaczej prędkości kątowe w osiach podłużnej (ang. *roll*), poprzecznej (ang. *pitch*) oraz pionowej (ang. *yaw*) [17]) oznaczono odpowiednio przez p, q, r. Jedną z wielkości fizycznych, która jest używana do opisu ruchu obiektów latających, jest *kąt natarcia* α (ang. *angle of attack*) [36]. Jest to kąt pomiędzy wektorem wypadkowej prędkości ruchu punktu ciężkości rakiety a jej osią podłużną.

Jako punkt referencyjny, względem którego rozpatrywano ruch rakiety, przyjęto *środek ciężkości* (ang. *center of gravity*) oznaczony jako CG na rys. 2.1. Założono, że rakieta jest symetryczna względem osi podłużnej. Zatem to na niej znajduje się środek ciężkości rakiety. Położenie punktu referencyjnego zostało opisane jako jego odległość od czubka rakiety x_{cg}.

Ważnym pojęciem jest także *środek parcia* (ang. *center of pressure*) oznaczony jako *CP* na rys. 2.1. Jest to punkt względem, którego rozpatruje się działanie na rakietę wypadkowej siły aerodynamicznej [36, 42]. Z tego względu jest to także punkt, względem którego momenty sił aerodynamicznych są równe 0.

Położenia środków parcia oraz ciężkości są wykorzystywane do analizy stabilności lotu rakiety. Rakieta jest stabilna, jeśli punkt parcia znajduje się w większej odległości od czubka rakiety niż środek ciężkości [34]. Wynika to z tego, że przy takim położeniu środka parcia oraz niezerowym kącie natarcia, siła

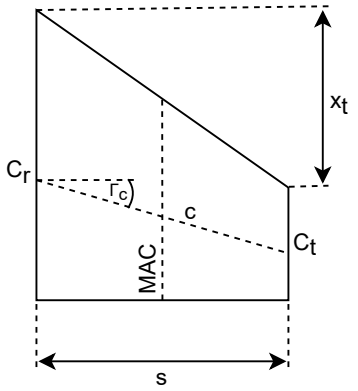


Rysunek 2.2: Ilustracja działania siły normalnej przy niezerowym kącie natarcia

normalna przyłożona do kadłuba rakiety generuje moment siły starający się ustawić rakietę równolegle do kierunku przepływu powietrza.

W przypadku rozpatrywania ruchu obiektów latających często stosuje się *liczbę Macha* jako miarę prędkości w ruchu postępowym. Opisuje ona stosunek prędkości postępowej obiektu do prędkości rozchodzenia się dźwięku w powietrzu.

Konieczne jest także oznaczenie istotnych parametrów geometrycznych powierzchni sterowych rakiety tj. lotek sterujących i stateczników. Przyjęto, że wszystkie powierzchnie sterowe mają kształt tra-



Rysunek 2.3: Parametry opisujące geometrię lotki (za pracą [34])

pezu o długościach podstaw równych C_r oraz C_t tak jak to zostało przedstawione na rys. 2.3. C_r jest długością podstawy, która przymocowana jest to korpusu rakiety, a C_t — podstawy zewnętrznej lotki. Odcinek oznaczony na rys. 2.3 jako *MAC* (ang. *mean aerodynamic chord*) jest równoległy do podstaw lotki. Określa on pozycję, w jakiej należałoby umieścić bok prostokątnej lotki, stykający się z kadłubem, tak aby miała ona taki sam wpływ na generowanie siły aerodynamicznej, jak rozpatrywana lotka [17]. Istotnym parametrem jest także odległość x_t, o jaką przesunięte są początki podstaw lotki (patrząc od

czubka rakiety), a także kąt Γ_c pomiędzy odcinkiem c łączącym środki postaw a odcinkiem prostopadłym do podstawy lotki.

Na potrzeby wyprowadzenia modelu rakiety założono, że powierzchnie sterowe mogą być wychylane od pionu o małe wartości kąta tj. nie większe niż $\pm 10^\circ$ [22]. Przyjęto, że silnik zamocowany jest tak, aby jego oś podłużna pokrywała się z osią podłużną rakiety. Co za tym idzie siła ciągu działa wyłącznie wzdłuż wspomnianej osi rakiety.

W celu zamodelowania dynamiki ruchu rakiety wykorzystana została tzw. metoda Barrowmana [11]. Pozwala ona na numeryczne wyznaczanie położenia środka parcia, jak i sił oraz momentów sił działających na rakietę podczas lotu. Dzięki temu stosowana jest do analizy stabilności lotu rakiety. Może być także wykorzystana do wyprowadzenia równań opisujących jej ruch [34]. W celu wykorzystania metody Barrowmana do analizy położenia środka parcia należy podzielić rakietę na mniejsze komponenty. W przypadku klasycznego wykorzystania tej metody są to:

- głowica,
- kadłub rakiety,
- zestaw lotek stabilizujących.

Jednak ze względu na większą liczbę lotek w analizowanej rakiecie (dodatkowe lotki sterujące) dokonano jej podziału na:

- głowicę,
- lotki sterujące,
- lotki stabilizujące
- dwie połowy kadłuba rakiety (do każdej z nich przymocowane są lotki jednego typu).

2.2 Parametry modelu rakiety - wyprowadzenie zależności

Jedną z wielkości, od której zależne są siły oraz momenty sił aerodynamicznych jest *ciśnienie dynamiczne* [36], opisywane wzorem:

$$Q = \frac{1}{2}\rho v_m^2, \tag{2.1}$$

gdzie:
ρ - gęstość powietrza $\left[\frac{kg}{m^3}\right]$,
v_m - wartość prędkości przepływu powietrza względem rakiety $\left[\frac{m}{s}\right]$.

Siły aerodynamiczne i momenty sił przez nie generowane opisywane są jako iloczyn ciśnienia dynamicznego (obliczanego na podstawie równania (2.1)), umownego pola referencyjnego (oraz długości referencyjnej w przypadku momentów sił) i współczynnika wyznaczonego analitycznie lub doświadczalnie. Można je wyznaczyć za pomocą równań

$$F = QC_F A_{ref}, \tag{2.2}$$

$$L = QC_L A_{ref} d_{ref}, \tag{2.3}$$

gdzie:
F - obliczana siła aerodynamiczna $[N]$,
L - obliczany moment siły aerodynamicznej $[Nm]$,
Q - ciśnienie dynamiczne $[Pa]$,

C_F - współczynnik obliczanej siły aerodynamicznej $[\cdot]$,
C_L - współczynnik obliczanego momentu siły aerodynamicznej $[\cdot]$,
A_{ref} - powierzchnia referencyjna $\left[m^2\right]$,
d_{ref} - długość referencyjna $[m]$.

Współczynniki służące do wyznaczania sił oraz momentów mogą być obliczane w sposób analityczny lub doświadczalny. Jako powierzchnię i długość referencyjną w przypadku metody Barrowmana uznaje się pole przekroju poprzecznego kadłuba rakiety oraz jego średnicę w najszerszym punkcie.

Występują dwa rodzaje sił aerodynamicznych działających na rakietę podczas lotu. Pierwszym z nich jest opór aerodynamiczny działający przeciwnie do kierunku lotu rakiety tj. przeciwnie do osi x układu lokalnego rakiety. Siła oporu aerodynamicznego (ang. *drag force*) opisywana jest analogicznie do wzoru (2.2)

$$F_{a_x} = -QC_aA_{ref} \tag{2.4}$$

gdzie:
C_a - współczynnik siły oporu aerodynamicznego $[\cdot]$.

Drugim rodzajem jest siła normalna (ang. *normal force*) generowana przez powierzchnie sterowe, jak i kadłub rakiety. Działa ona w punkcie parcia rakiety i jest skierowana prostopadle do korpusu rakiety. Wartość siły normalnej także można obliczyć analogicznie do wzoru (2.2):

$$F_n = QC_nA_{ref}, \tag{2.5}$$

gdzie:
C_n - współczynnik siły normalnej $[\cdot]$.

Założono, że pochodna współczynnika siły normalnej obliczana po kącie natarcia jest liniowa dla małych wartości kątów natarcia (ok. $\pm 10°$) [22]. Zatem współczynnik siły normalnej użyty we wzorze (2.5) może być opisany jako

$$C_n = C_{n_\alpha} \cdot \alpha, \tag{2.6}$$

gdzie C_{n_α} jest pochodną współczynnika siły normalnej obliczaną po kącie natarcia rakiety $\left[\frac{1}{rad}\right]$.

Pochodną współczynnika siły normalnej użytą we wzorze (2.6) można obliczyć jako sumę pochodnych współczynników siły normalnej dla każdego komponentu rakiety

$$C_{n_\alpha} = C_{n_{\alpha_n}} + C_{n_{\alpha_{b_c}}} + C_{n_{\alpha_{b_s}}} + C_{n_{\alpha_c}} + C_{n_{\alpha_s}}, \tag{2.7}$$

gdzie:
$C_{n_{\alpha_n}}$ - pochodna wsp. siły normalnej obliczona dla głowicy rakiety $\left[\frac{1}{rad}\right]$,
$C_{n_{\alpha_{b_c}}}$ - pochodna wsp. siły normalnej obliczona dla części kadłuba związanej z lotkami sterującymi $\left[\frac{1}{rad}\right]$,
$C_{n_{\alpha_{b_s}}}$ - pochodna wsp. siły normalnej obliczona dla części kadłuba związanej z lotkami stabilizującymi $\left[\frac{1}{rad}\right]$,
$C_{n_{\alpha_c}}$ - pochodna wsp. siły normalnej obliczona dla lotek sterujących $\left[\frac{1}{rad}\right]$,
$C_{n_{\alpha_s}}$ - pochodna wsp. siły normalnej obliczona dla lotek stabilizujących $\left[\frac{1}{rad}\right]$.
Pochodne współczynników siły normalnej każdego z komponentów są możliwe do wyznaczenia w sposób analityczny. W przypadku głowicy rakiety przyjmuje ona postać [34, 11]

$$C_{n_{\alpha_n}} = 2 \cdot \frac{\sin\alpha}{\alpha}, \tag{2.8}$$

a dla fragmentów kadłuba ma on postać [34]

$$C_{n_{\alpha_b}} = 1.1\frac{d_{ref}}{A_{ref}}l \cdot \frac{\sin\alpha}{\alpha}, \tag{2.9}$$

gdzie l $[m]$ jest długością danej sekcji kadłuba, a dla zestawu powierzchni sterowych można ją obliczyć ze wzoru [34]

$$C_{n_{\alpha_f}} = \left(1 + \frac{r_t}{s + r_t}\right) \begin{cases} \left(\sum_{k=1}^{N} \sin^2 \Lambda_k\right) C_{n_{\alpha_1}}, & N = 1,\, 2 \\ 1.5 C_{n_{\alpha_1}} \left(1 - 0.15 \cos^2\left(\frac{3}{2}\Lambda_1\right)\right), & N = 3 \\ 2 C_{n_{\alpha_1}} \left(1 - 0.06 \cos^2\left(2\Lambda_1\right)\right), & N = 4 \end{cases}, \tag{2.10}$$

gdzie:
$C_{n_{\alpha_1}}$ - pochodna wsp. siły normalnej obliczana dla pojedynczej lotki $\left[\frac{1}{rad}\right]$,
N - liczba lotek danego typu $[\cdot]$,
Λ_k - kąt pomiędzy przepływem powietrza prostopadłym do kadłuba a k-tą lotką $[rad]$,
r_t - promień kadłuba rakiety $[m]$.

Wartość współczynnika $C_{n_{\alpha_1}}$ wykorzystana we wzorze (2.10) jest zależna m.in. od prędkości rakiety wyrażonej w liczbie Macha. Można ją wyznaczyć, korzystając z zależności [34]

$$C_{n_{\alpha_1}} = \frac{2\pi \frac{s^2}{A_{ref}}}{1 + \sqrt{1 + \left(s^2 \frac{\sqrt{|M^2 - 1|}}{A_{fin} \cos \Gamma_c}\right)^2}}, \tag{2.11}$$

gdzie:
M - liczba Macha, $[\cdot]$,
A_{fin} - powierzchnia lotki $\left[m^2\right]$.

Liczbę Macha definiuje się jako stosunek prędkości rakiety względem powietrza do aktualnej prędkości dźwięku

$$M = \frac{v_m}{v_s}, \tag{2.12}$$

gdzie v_s $\left[\frac{m}{s}\right]$ jest aktualną prędkością dźwięku.

Aktualna prędkość dźwięku została wyznaczona, korzystając ze standardowego modelu atmosfery ISA [18]. Zależy ona od aktualnej temperatury powietrza i można ją opisać wzorem

$$v_s = v_{s_0} \sqrt{\frac{T}{T0}}, \tag{2.13}$$

gdzie:
$v_{s_0} = 340,3\, \frac{m}{s}$ - wartość prędkości dźwięku w atmosferze standardowej na wysokości morza,
$T_0 = 288,15\, K$ - temperatura w atmosferze standardowej na wysokości morza,
T - temperatura powietrza $[K]$.
Temperatura powietrza T na danej wysokości w modelu atmosfery standardowej może zostać wyznaczona jako

$$T = T_0 + \gamma h, \tag{2.14}$$

gdzie:
$\gamma = -6.5 \cdot 10^{-3}\, \frac{K}{m}$ - pionowy gradient temperaturowy do wysokości 11 km w modelu atmosfery standardowej.

Przyjęcie liniowej zależności temperatury od wysokości lotu jest prawdziwe tylko do wysokości ok. 11 km tj. do osiągnięcia górnej granicy troposfery [18]. Nie jest to jednak ograniczenie na potrzeby tej pracy, ponieważ rozważany typ rakiet nie wznosi się na tak duże wysokości.

Wykorzystując model atmosfery standardowej ISA, wyznaczona została także zależność gęstość powietrza w zależności od wysokości [18]:

$$\rho = \rho_0 \left(1 - \frac{\gamma}{T_0} h\right)^{\frac{g}{R \cdot \gamma} - 1}, \tag{2.15}$$

gdzie:
$g = 9,81\ \frac{m}{s^2}$ - przyspieszenie grawitacyjne,
$R = 287,0529\ \frac{m^2}{s^2}K$ - stała gazowa,
$\rho_0 = 1,225\ \frac{kg}{m^3}$ - gęstość powietrza na poziomie morza.

Mając obliczone pochodne współczynników siły normalnej dla każdego komponentu rakiety, możliwe jest wyznaczenie punktu przyłożenia tej siły tj. środka parcia rakiety. Odległość środka parcia od czubka rakiety jest średnią ważoną odległości środków parcia poszczególnych komponentów od czubka rakiety, gdzie wagami są wartości pochodnych współczynników siły normalnej dla danych komponentów (obliczone na podstawie wzorów (2.8)–(2.10)) [34]

$$x_{cp} = \frac{x_{cp_n} C_{n_{\alpha_n}} + x_{cp_{b_c}} C_{n_{\alpha_{b_c}}} + x_{cp_{b_s}} C_{n_{\alpha_{b_s}}} + x_{cp_c} C_{n_{\alpha_c}} + x_{cp_s} C_{n_{\alpha_s}}}{C_{n_{\alpha_n}} + C_{n_{\alpha_{b_c}}} + C_{n_{\alpha_{b_s}}} + C_{n_{\alpha_{b_s}}} + C_{n_{\alpha_c}}}, \tag{2.16}$$

gdzie:
x_{cp_n} - odległość środka parcia głowicy rakiety od czubka $[m]$,
$x_{cp_{b_c}}$ - odległość środka parcia części korpusu rakiety związanej z lotkami sterującymi od czubka $[m]$,
$x_{cp_{b_s}}$ - odległość środka parcia części korpusu rakiety związanej ze statecznikami od czubka $[m]$,
x_{cp_c} - odległość środka parcia lotek sterujących od czubka $[m]$,
x_{cp_s} - odległość środka parcia stateczników od czubka $[m]$.

Odległość środka parcia głowicy rakiety od jej czubka możliwa jest do wyznaczenia za pomocą wzoru [34]

$$x_{cp_n} = l_n - \frac{V_n}{\pi r_t^2}, \tag{2.17}$$

gdzie:
l_n - długość głowicy $[m]$,
V_n - objętość głowicy $\left[m^3\right]$.
Założono, że głowica rakiety ma kształt połowy elipsoidy o długościach półosi równych r_t, r_t oraz l_n. Zatem objętość głowicy wykorzystana we wzorze (2.17) można wyrazić jako

$$V_n = \frac{1}{2} \cdot \frac{4}{3} \pi l_n r_t r_t = \frac{2}{3} \pi l_n r_t^2. \tag{2.18}$$

Odległość środka parcia od początku każdej z części kadłuba rakiety znajduje się w połowie ich długości [34]. Zatem uwzględniając położenie fragmentów kadłuba można zapisać odległość ich środków parcia od czubka rakiety jako

$$x_{cp_{b_c}} = l_n + \frac{1}{2} l_c, \tag{2.19}$$

$$x_{cp_{b_s}} = l_n + l_c + \frac{1}{2} l_s, \tag{2.20}$$

gdzie l_c $[m]$ oraz l_s $[m]$ są długościami części kadłuba związanych z odpowiednio lotkami sterującymi i statecznikami.

Odległość środka parcia ruchomych powierzchni sterowych i stateczników od początków ich podstaw stykających się kadłubem można wyznaczyć ze wzoru [34]

$$x_{cp_f} = \frac{x_t}{3} \frac{C_r + 2C_t}{C_r + C_t} + \frac{1}{6} \frac{C_r^2 + C_t^2 + C_r C_t}{C_r + C_t},$$

a uwzględniając położenie lotek względem czubka uzyskamy zależności

$$x_{cp_c} = l_n + x_c + \frac{x_{t_c}}{3} \frac{C_{r_c} + 2C_{t_c}}{C_{r_c} + C_{t_c}} + \frac{1}{6} \frac{C_{r_c}^2 + C_{t_c}^2 + C_{r_c} C_{t_c}}{C_{r_c} + C_{t_c}}, \tag{2.21}$$

$$x_{cp_s} = l_n + l_c + x_s + \frac{x_{t_s}}{3} \frac{C_{r_s} + 2C_{t_s}}{C_{r_s} + C_{t_s}} + \frac{1}{6} \frac{C_{r_s}^2 + C_{t_s}^2 + C_{r_s} C_{t_s}}{C_{r_s} + C_{t_s}}, \tag{2.22}$$

gdzie:
x_{t_c}, x_{t_s} - odległość x_t odpowiednio dla lotek sterujących i stateczników $[m]$,
C_{r_c}, C_{r_s} - długość podstawy C_r odp. dla lotek sterujących i stateczników $[m]$,
C_{t_c}, C_{t_s} - długość podstawy C_t odp. dla lotek sterujących i stateczników $[m]$.

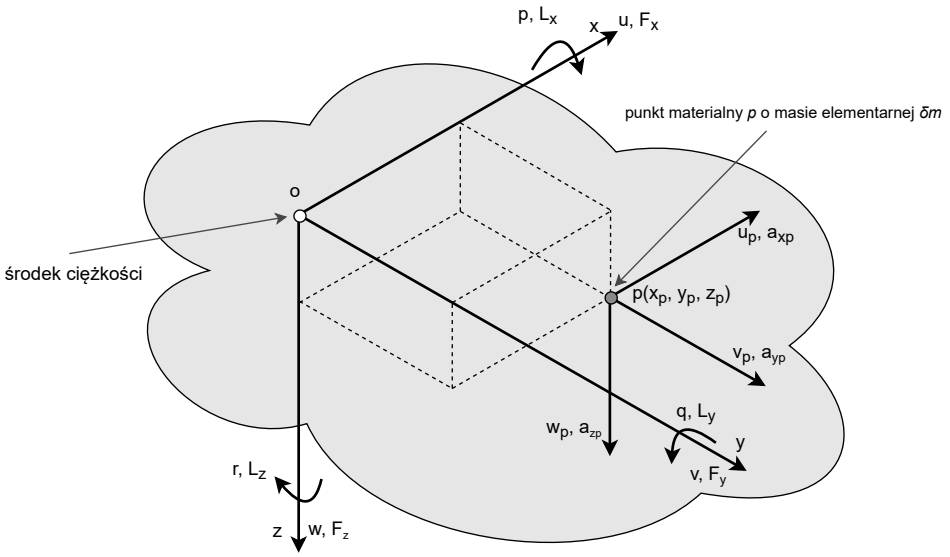
2.3 Wyprowadzenie równań dla ruchu postępowego

Równania opisujące ruch postępowy rakiety podczas lotu wyprowadzono na podstawie [17]. Można je opisać, korzystając z II zasady dynamiki Newtona opisanej zależnością

$$F = ma, \tag{2.23}$$

gdzie:
F - wypadkowa siła działająca na ciało w danej osi $[N]$,
a - wypadkowe przyspieszenie ciała wzdłuż danej osi $\left[\frac{m}{s^2}\right]$,
m - masa rozpatrywanego ciała $[kg]$.

Ruch ciała sztywnego można rozpatrzeć jako ruch nieskończenie wielu punktów materialnych, z których składa się to ciało. Rozpatrzono ruch przykładowego punktu materialnego p o współrzędnych (x_p, y_p, z_p) w układzie bryły sztywnej z rys. 2.4 i masie elementarnej δm. Prędkości poruszania się



Rysunek 2.4: Ruch bryły sztywnej opisany w jej lokalnym układzie współrzędnych (za pracą [17])

ciała wzdłuż osi x, y, z układu zostały oznaczone odpowiednio jako u, v, w, a siły działające wzdłuż tych osi jako F_x, F_y, F_z. Prędkości kątowe obrotu wokół tych samych osi oznaczono za pomocą p, q, r, a momenty sił jako L_x, L_y, L_z. Wartości składowych prędkości poruszania się punktu p względem układu współrzędnych ciała zostały oznaczone jako u_p, v_p, w_p, a przyspieszenia jako a_{x_p}, a_{y_p}, a_{z_p}.

Wartości prędkości u_p, v_p, w_p można zdefiniować jako sumę prędkości poruszania się punktu p względem osi układu współrzędnych ciała oraz prędkości wynikającej z obracania się rozpatrywanego ciała

wokół osi x, y, z

$$\begin{cases} u_p = \dot{x}_p - ry_p + qz_p, \\ v_p = \dot{y}_p - pz_p + rx_p, \\ w_p = \dot{z}_p - qx_p + py_p. \end{cases}$$

Jednak ze względu na założenie, że rozpatrywane ciało jest bryłą sztywną, można uznać, że p nie porusza się względem środka ciężkości, zatem

$$\dot{x}_p = \dot{y}_p = \dot{z}_p = 0,$$

więc

$$\begin{cases} u_p = -ry_p + qz_p, \\ v_p = -pz_p + rx_p, \\ w_p = -qx_p + py_p. \end{cases} \tag{2.24}$$

W sposób analogiczny można zdefiniować wartości składowych przyspieszenia punktu p

$$\begin{cases} a_{x_p} = \dot{u}_p - rv_p + qw_p, \\ a_{y_p} = \dot{v}_p - pw_p + ru_p, \\ a_{z_p} = \dot{w}_p - qu_p + pv_p. \end{cases} \tag{2.25}$$

Uwzględniając poruszanie się ciała względem układu lokalnego, można zapisać, że wartości składowych prędkości punktu p w układzie ustalonym są równe

$$\begin{cases} u'_p = u + u_p = u - ry_p + qz_p, \\ v'_p = v + v_p = v - pz_p + rx_p, \\ w'_p = w + w_p = w - qx_p + py_p. \end{cases} \tag{2.26}$$

Zatem wartości składowych przyspieszeń w układzie ustalonym na podstawie wzoru (2.25) są równe

$$\begin{cases} a_x = \dot{u}'_p - rv'_p + qw'_p, \\ a_y = \dot{v}'_p - pw'_p + ru'_p, \\ a_z = \dot{w}'_p - qu'_p + pv'_p. \end{cases} \tag{2.27}$$

Różniczkując obustronnie równanie (2.26), pamiętając, że rozpatrywane ciało jest bryłą sztywną otrzymujemy

$$\begin{cases} \dot{u}'_p = \dot{u} - \dot{r}y_p + \dot{q}z_p, \\ \dot{v}'_p = \dot{v} - \dot{p}z_p + \dot{r}x_p, \\ \dot{w}'_p = \dot{w} - \dot{q}x_p + \dot{p}y_p. \end{cases} \tag{2.28}$$

Podstawiając równania (2.26) i (2.28) do równania (2.27) otrzymujemy

$$\begin{cases} a_x = \dot{u} - rv + qw - x_p\left(q^2 + r^2\right) + y_p\left(pq - \dot{r}\right) + z_p\left(pr + \dot{q}\right), \\ a_y = \dot{v} - pw + ru + x_p\left(pq + \dot{r}\right) - y_p\left(p^2 + r^2\right) + z_p\left(qr - \dot{p}\right), \\ a_z = \dot{w} - qu + pv + x_p\left(pr - \dot{q}\right) + y_p\left(qr + \dot{p}\right) - z_p\left(p^2 + q^2\right). \end{cases} \tag{2.29}$$

Korzystając z II zasady dynamiki Newtona (2.23) można zapisać, że wypadkowe siły działające na ciało w osiach x, y i z są sumą sił działających w tych osiach na każdy punkt materialny tego ciała

$$\begin{cases} F_x = \sum \delta m a_x, \\ F_y = \sum \delta m a_y, \\ F_z = \sum \delta m a_z. \end{cases} \tag{2.30}$$

Po podstawieniu wartości przyspieszeń z równania (2.29) do równania (2.30) oraz zakładając, że układ współrzędnych ciała ma swój początek w środku ciężkości

$$\sum \delta m x_p = \sum \delta m y_p = \sum \delta m z_p = 0, \tag{2.31}$$

otrzymano

$$\begin{cases} F_x = m\left(\dot{u} - rv + qw\right), \\ F_y = m\left(\dot{v} - pw + ru\right), \\ F_z = m\left(\dot{w} - qu + pv\right). \end{cases} \tag{2.32}$$

Po przekształceniu równań (2.32) otrzymujemy równania opisujące ruch postępowy rakiety

$$\begin{cases} \dot{u} = \dfrac{F_x}{m} - qw + rv, \\ \dot{v} = \dfrac{F_y}{m} - ru + pw, \\ \dot{w} = \dfrac{F_z}{m} - pv + qu. \end{cases} \tag{2.33}$$

Siła wypadkowa działająca na rakietę składa się z siły aerodynamicznej, siły ciągu generowanej przez silnik oraz siły grawitacji. Siła aerodynamiczna składa się z siły oporu aerodynamicznego wyznaczanego za pomocą równania (2.4) oraz siły normalnej wyznaczonej za pomocą (2.5). Składowe siły wypadkowej są zatem równe

$$\begin{aligned} F_x &= F_{a_x} + F_{p_x} + F_{g_x}, \\ F_y &= F_{n_y} + F_{p_y} + F_{g_y}, \\ F_z &= F_{n_z} + F_{p_z} + F_{g_z}, \end{aligned} \tag{2.34}$$

gdzie:
F_{n_y}, F_{n_z} - składowe siły normalnej działające w osiach y i z $[N]$,
F_{p_x}, F_{p_y}, F_{p_z} - składowe siły ciągu silnika $[N]$,
F_{g_x}, F_{g_y}, F_{g_z} - składowe siły grawitacji $[N]$.

Wartości składowych siły normalnej F_{n_y}, F_{n_z} można wyznaczyć rzutując siłę normalną F_n na osie y oraz z układu lokalnego rakiety

$$\begin{aligned} F_{n_y} &= -F_n \frac{v}{\sqrt{v^2 + w^2}}, \\ F_{n_z} &= F_n \frac{w}{\sqrt{v^2 + w^2}}. \end{aligned} \tag{2.35}$$

Założono, że silnik jest umiejscowiony współosiowo z kadłubem rakiety. Można zatem przyjąć, że siła ciągu generowana przez silnik działa na rakietę wyłącznie w osi x, zatem

$$\begin{aligned} F_{p_x} &= F_p, \\ F_{p_y} &= 0\,N, \\ F_{p_z} &= 0\,N. \end{aligned} \tag{2.36}$$

Wektor siły grawitacji został zdefiniowany za pomocą przekształcenia siły grawitacji działającej w układzie globalnym do układu lokalnego rakiety

$$\begin{bmatrix} F_{g_x} \\ F_{g_y} \\ F_{g_z} \end{bmatrix} = T_l^g \begin{bmatrix} F_{g_{x_g}} \\ F_{g_{y_g}} \\ F_{g_{z_g}} \end{bmatrix}, \tag{2.37}$$

gdzie T_l^g jest macierzą obrotu pomiędzy układem globalnym a układem lokalnym rakiety. Macierz T_l^g jest ortogonalna, a więc $T_l^g = \left(T_g^l\right)^T$. Macierz obrotu T_g^l jest złożeniem obrotów o wartości kątów Eulera, jakie należy wykonać, aby orientację układu lokalnego rakiety przekształcić w orientację układu globalnego [17]

$$T_g^l = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

$$T_g^l = \begin{bmatrix} \cos\theta\cos\psi & \sin\varphi\sin\theta\cos\psi - \cos\varphi\sin\psi & \cos\varphi\sin\theta\cos\psi + \sin\varphi\sin\psi \\ \cos\theta\sin\psi & \sin\varphi\sin\theta\sin\psi + \cos\varphi\cos\psi & \cos\varphi\sin\theta\sin\psi - \sin\varphi\cos\psi \\ -\sin\theta & \sin\varphi\cos\theta & \cos\varphi\cos\theta \end{bmatrix}. \tag{2.38}$$

Korzystając z wartości prędkości rotacji rakiety wokół osi lokalnego układu współrzędnych, opisanych za pomocą pochodnych wartości kątów Eulera [17]:

$$\begin{cases} p = \dot{\varphi} - \dot{\psi}\sin\theta, \\ q = \dot{\theta}\cos\varphi + \dot{\psi}\sin\varphi\cos\theta, \\ r = \dot{\psi}\cos\varphi\cos\theta - \dot{\theta}\sin\varphi, \end{cases}$$

co można przekształcić do postaci (zakładając, że $\cos\theta \neq 0$):

$$\begin{cases} \dot{\varphi} = p + \mathrm{tg}\theta\left(q\sin\varphi + r\cos\varphi\right), \\ \dot{\theta} = q\cos\varphi - r\sin\varphi, \\ \dot{\psi} = \dfrac{q\sin\varphi + r\cos\varphi}{\cos\theta}. \end{cases} \tag{2.39}$$

Całkując obustronnie równanie (2.39) można uzyskać równania opisujące wartości kątów Eulera wykorzystanych w równaniu (2.38):

$$\begin{cases} \varphi = \displaystyle\int_0^t p + \mathrm{tg}\theta\left(q\sin\varphi + r\cos\varphi\right)\ d\tau, \\ \theta = \displaystyle\int_0^t q\cos\varphi - r\sin\varphi\ d\tau, \\ \psi = \displaystyle\int_0^t \frac{q\sin\varphi + r\cos\varphi}{\cos\theta}\ d\tau. \end{cases} \tag{2.40}$$

2.4 Wyprowadzenie równań dla ruchu obrotowego

Równania dla ruchu obrotowego zostały, analogicznie do równań ruchu postępowego, wyznaczone na podstawie [17]. Wypadkowe momenty sił działające na ciało względem osi x, y i z można wyznaczyć jako sumę momentów sił generowanych przez wypadkowe siły działające na punkty materialne

$$\begin{aligned} L_x &= \sum \delta m a_z y_p - \sum \delta m a_y z_p, \\ L_y &= \sum \delta m a_x z_p - \sum \delta m a_z x_p, \\ L_z &= \sum \delta m a_y x_p - \sum \delta m a_x y_p, \end{aligned}$$

a po podstawieniu wartości składowych przyspieszeń z równania (2.29), uwzględnieniu równania (2.31) oraz wykonaniu przekształceń otrzymujemy

$$\begin{aligned}
L_x &= \dot{p}\sum \delta m\left(y_p^2+z_p^2\right)+qr\sum \delta m\left(y_p^2-z_p^2\right)+\left(r^2-q^2\right)\sum \delta m y_p z_p\\
&\quad-(pq+\dot{r})\sum \delta m x_p z_p+(pr-\dot{q})\sum \delta m x_p y_p,\\
L_y &= \dot{q}\sum \delta m\left(x_p^2+z_p^2\right)+pr\sum \delta m\left(x_p^2-z_p^2\right)+\left(p^2-r^2\right)\sum \delta m x_p z_p\\
&\quad-(qr+\dot{p})\sum \delta m x_p y_p+(pq-\dot{r})\sum \delta m y_p z_p,\\
L_z &= \dot{r}\sum \delta m\left(x_p^2+y_p^2\right)-pq\sum \delta m\left(x_p^2-y_p^2\right)+\left(q^2-p^2\right)\sum \delta m x_p y_p\\
&\quad-(pr+\dot{q})\sum \delta m y_p z_p+(qr-\dot{p})\sum \delta m x_p z_p.
\end{aligned} \tag{2.41}$$

Można zauważyć, że wyrażenia w równaniu (2.41) z symbolem sumy $\sum$ są momentami bezwładności. Zatem można zapisać równanie (2.41) w uproszczonej formie

$$\begin{aligned}
L_x &= \dot{p}I_x+qr\left(I_y-I_z\right)+\left(r^2-q^2\right)I_{xy}-\left(pq+\dot{r}\right)I_{xz}+\left(pr-\dot{q}\right)I_{xy},\\
L_y &= \dot{q}I_y+pr\left(I_x-I_z\right)+\left(p^2-r^2\right)I_{xz}-\left(qr+\dot{p}\right)I_{xy}+\left(pq-\dot{r}\right)I_{yz},\\
L_z &= \dot{r}I_z+pq\left(I_y-I_x\right)+\left(q^2-p^2\right)I_{xy}-\left(pr+\dot{q}\right)I_{yz}+\left(qr-\dot{p}\right)I_{xz},
\end{aligned} \tag{2.42}$$

gdzie:
I_x, I_y, I_z - momenty bezwładności $\left[kg\,m^2\right]$,
I_{xy}, I_{xz}, I_{yz} - momenty dewiacyjne $\left[kg\,m^2\right]$.
Zakładając, że ruch rakiety rozważany jest względem jej środka ciężkości, a osie układu współrzędnych rakiety pokrywają się głównymi osiami bezwładności, można pominąć momenty dewiacyjne [17]

$$\begin{aligned}
L_x &= \dot{p}I_x+qr\left(I_y-I_z\right),\\
L_y &= \dot{q}I_y+pr\left(I_x-I_z\right),\\
L_z &= \dot{r}I_z+pq\left(I_y-I_x\right).
\end{aligned} \tag{2.43}$$

Zatem równania opisujące ruch obrotowy rakiety można zapisać jako

$$\begin{cases}
\dot{p} = \dfrac{L_x-qr(I_z-I_y)}{I_x},\\
\dot{q} = \dfrac{L_y-rp(I_x-I_z)}{I_y},\\
\dot{r} = \dfrac{L_z-pq(I_y-I_x)}{I_z}.
\end{cases} \tag{2.44}$$

Momenty sił L_x, L_y, L_z powstają w wyniku działania sił aerodynamicznych i siły ciągu silnika

$$\begin{aligned}
L_x &= L_{a_x}+L_{p_x},\\
L_y &= L_{a_y}+L_{p_y},\\
L_z &= L_{a_z}+L_{p_z},
\end{aligned}$$

gdzie L_{a_x}, L_{a_y}, L_{a_z} są momentami sił aerodynamicznych, a L_{p_x}, L_{p_y}, L_{p_z} to momenty sił wywołane przez siłę ciągu. Ze względu na współosiowe umiejscowienie silnika rakiety względem jej kadłuba momenty sił wywołane siłą ciągu są równe 0. Założono, że momenty bezwładności względem osi y i z dla analizowanej rakiety są sobie równe

$$I_y = I_z.$$

Zatem ostatecznie można zapisać

$$\begin{cases} \dot{p} = \dfrac{L_{a_x}}{I_x}, \\ \dot{q} = \dfrac{L_{a_y} - rp(I_x - I_z)}{I_y}, \\ \dot{r} = \dfrac{L_{a_z} - pq(I_y - I_x)}{I_z}. \end{cases} \tag{2.45}$$

Moment siły L_{a_x} w równaniu (2.45) tj. moment siły aerodynamicznej działający w osi podłużnej można opisać jako różnicę momentu wymuszającego ruch obrotowy, generowanego przez każdy z zestawów lotek oraz momentów tłumiących ten ruch. Można go obliczyć analogicznie do równania (2.3) korzystając ze współczynników wspomnianych momentów sił [34]

$$L_{a_x} = Q\left(C_{l_f} - C_{l_d}\right) A_{ref} d_{ref}, \tag{2.46}$$

gdzie:
$C_{l_f} = C_{l_{f_c}} + C_{l_{f_s}}$ - suma współczynników momentów forsujących w osi podłużnej obu zestawów lotek $[\cdot]$,
$C_{l_d} = C_{l_{d_c}} + C_{l_{d_s}}$ - suma współczynników momentów tłumiących w osi podłużnej obu zestawów lotek $[\cdot]$.

Współczynniki forsujących momentów sił w równaniu (2.46) generowanych przez lotki sterujące i stabilizujące opisać można równaniem [34]

$$C_{l_f} = \frac{N\left(y_{MAC} - r_t\right) C_{n_{\alpha_1}} \delta}{d_{ref}}, \tag{2.47}$$

gdzie:
N - liczba lotek danego typu $[\cdot]$,
y_{MAC} - długość średniego ramienia aerodynamicznego $[m]$,
δ - kąt wychylenia lotki $[rad]$.
Długość średniego ramienia aerodynamicznego y_{MAC} (tj. odcinka MAC na rys. 2.3) w przypadku lotek o kształcie trapezu można wyznaczyć na podstawie długości ich podstaw oraz ich rozpiętości [34]

$$y_{MAC} = \frac{s}{3} \cdot \frac{C_r + 2C_t}{C_r + C_t}. \tag{2.48}$$

Współczynniki tłumiących momentów sił w osi podłużnej są zależne m.in. od aktualnej prędkości rotacji rakiety p. Można je wyznaczyć, korzystając z równania [34]

$$C_{l_d} = \frac{2\pi N p}{A_{ref} d_{ref} v_m \sqrt{|M^2 - 1|}} \left(\frac{C_r + C_t}{2} r_t^2 s + \frac{C_r + 2C_t}{3} r_t s^2 + \frac{C_r + 3C_t}{12} s^3 \right). \tag{2.49}$$

Momenty działające w osi poprzecznej i pionowej są wynikiem działania składowych siły normalnej (2.35) w osiach y i z oraz momentu siły tłumiącego ruch obrotowy wokół tych osi. Można opisać je za pomocą równań

$$\begin{aligned} L_{a_y} &= -F_{n_z}\left(x_{cp} - x_{cg}\right) + L_{d_y}, \\ L_{a_z} &= -F_{n_y}\left(x_{cp} - x_{cg}\right) + L_{d_z}. \end{aligned} \tag{2.50}$$

gdzie L_{d_y} i L_{d_z} $[Nm]$ są tłumiącymi momentami sił w osiach pionowej oraz poprzecznej, które mają zawsze przeciwny znak do prędkości rotacji w odpowiadającej im osi. Długość ramienia, na jakim działają te siły jest równa różnicy odległości środka parcia x_{cp} i środka ciężkości x_{cg} od czubka rakiety (tj. odległości środka parcia od środka ciężkości).

Na momenty sił L_{d_y} i L_{d_z} składają się tłumiące momenty sił generowane przez kadłub rakiety, jak i tłumiące momenty sił generowane przez same powierzchnie sterowe. Tłumiące momenty sił generowane przez kadłub rakiety można opisać równaniami [34]

$$\begin{aligned} L_{d_{y_b}} &= 0,275 \cdot \rho r_t l^4 q^2, \\ L_{d_{z_b}} &= 0,275 \cdot \rho r_t l^4 r^2, \end{aligned} \tag{2.51}$$

gdzie l $[m]$ jest długością całego kadłuba rakiety. Natomiast tłumiące momenty sił generowane przez powierzchnie sterowe można wyznaczyć, korzystając ze współczynników [34]

$$\begin{aligned} L_{d_{y_f}} &= QC_{d_y}A_{ref}d_{ref}, \\ L_{d_{z_f}} &= QC_{d_z}A_{ref}d_{ref}, \end{aligned} \tag{2.52}$$

gdzie:

$$\begin{aligned} C_{d_y} &= 0.6 \cdot \frac{NA_{fin}\kappa^3}{A_{ref}d_{ref}} \cdot \frac{q^2}{v_m^2}, \\ C_{d_z} &= 0.6 \cdot \frac{NA_{fin}\kappa^3}{A_{ref}d_{ref}} \cdot \frac{r^2}{v_m^2}, \end{aligned} \tag{2.53}$$

przy czym κ $[m]$ jest odległością danego zestawu lotek od środka ciężkości rakiety. Momenty $L_{d_{y_f}}$ i $L_{d_{z_f}}$ są obliczane dla każdego z zestawu lotek (sterujących i stabilizujących) osobno, a następnie są one dodawane.

Rozdział 3

Projekt układu sterowania

3.1 Definicja zadania sterowania

Za obiekt sterowania uważany jest model ruchu wirowego rakiety wraz z układem wykonawczym. W związku z tym obiekt jest złożeniem dwóch członów.

Uwzględniając równanie (2.1) w (2.46) oraz podstawiając do równania przyspieszenia rotacji $\dot{p}$ (2.45) zostaje otrzymane:

$$\dot{p} = \frac{\rho v_m^2 \left(C_{l_f} - C_{l_d}\right) A_{ref} d_{ref}}{2I_x}. \tag{3.1}$$

Widoczne jest, że przyspieszenie rotacji zależy nieliniowo od prędkości rakiety. Poza tym jest zależne od nieznanych błędów montażu stateczników, nieliniowych pochodnych współczynników siły normalnej liczonej dla kąta natarcia zdefiniowanej dla pojedynczej wybranej lotki (2.11) i współczynników momentów tłumiących (2.49). Ponadto przyspieszenie kątowe obliczane jest na podstawie parametrów geometrycznych rakiety i lotek (2.47, 2.48) oraz zakłada się, że momenty bezwładności względem osi y i z układu współrzędnych rakiety są sobie równe, dzięki czemu zależności skrośne prędkości dla pionowej i poprzecznej osi są pomijane. Model ruchu wirowego rakiety posiada, więc dużą niepewność strukturalną i parametryczną.

Zaprojektowany układ wykonawczy poruszający lotkami sterującymi jest inercją pierwszego rzędu z opóźnieniem czasowym. W rezultacie człon wykonawczy jest nieminimalnofazowy oraz rozszerza rząd obiektu sterowania do drugiego rzędu.

W konsekwencji człon sterowania (3.1) wraz z elementem wykonawczym jest nieliniowy, a na obiekt sterowania działa zmienne zaburzenie. Jako zadanie sterowania zdefiniowano stabilizację zerowej prędkości rotacji. Jest to, więc zadanie regulacji stałowartościowej. Zakres prędkości rotacji, dla którego uznawane jest, że sterownik spełnił zadanie sterowania wynosi $\pm 20\frac{^\circ}{s}$ wokół wartości zadanej. Został on wyznaczony na podstawie efektywności żyrolotek, czyli mechanicznego odpowiednika sterowania prędkością rotacji. Wspomniana alternatywna metoda sterowania pozwala na uzyskanie minimalnej prędkości rotacji równej $\pm 20\frac{^\circ}{s}$ [10] [29]. Założony czas ustalania prędkości rotacji dla lotu rakiety ze sterowaniem włączonym od momentu startu wynosi $T_s \leq 2s$, gdzie T_s jest czasem ustalania odpowiedzi układu regulacji. Został on wyznaczony na podstawie znajomości czasu wypalenia silnika.

3.2 Uzasadnienie wyboru algorytmu sterowania

W związku z charakterem obiektu sterowania oraz działających zakłóceń stwierdzono, że podstawowe sterowniki (np. P, PI, PD, PID) mogą zapewnić nieakceptowalną jakość regulacji. Postanowiono wykorzystać adaptacyjną metodę sterowania. Podejście adaptacyjne dla wyżej określonego zadania sterowania wykorzystywane jest w rozwiązaniach technologicznych stosowanych w sektorze wojskowym [36]

[21] oraz lotniczym [13]. W celu uzasadnienia wyboru algorytmu sterowania porównana została metoda szeregowania parametrów ze sterownikiem ADRC [41].

Szeregowanie parametrów jest metodą wyznaczania zestawów nastaw sterownika (np. P, PI, PD, PID) dla wielu punktów pracy [26]. Wspomniane nastawy przestrajane są w zależności od aktualnego punktu pracy układu. W konsekwencji konieczne jest wcześniejsze wyznaczenie dokładnych wartości parametrów projektowych sterownika. Jedną z metod jest symulacyjny dobór nastaw sterownika. Wymaga to jednak dokładnego symulatora.

ADRC jest sterownikiem działającym w torze zamkniętym, odpornym na niepewność strukturalną i parametryczną obiektu oraz zakłócenia. Pozwala na adaptację do zmieniającej się dynamiki i zewnętrznych zaburzeń [46].

Symulując rakietę, przyjmuje się określone uproszczenia i założenia, które nie odzwierciedlają dokładnie rzeczywistości. Jednym z takich założeń jest to, że zakłada się określoną wartość przekrzywienia stateczników, która w rzeczywistości jest trudna lub niemożliwa do zmierzenia. Kolejnym przykładem jest założenie stałego kierunku przepływu wiatru. Algorytm szeregowania parametrów wymaga wcześniejszego wyznaczenia nastaw dla określonych punktów pracy. Można to wykonać symulacyjnie [44] lub poprzez testy w tunelu przeskalowanego modelu rakiety [13]. W aspekcie różnych projektów rakiet wprowadza to duży nakład pracy. W kontekście zmiennej szeregującej nastawy, którą w tym przypadku byłaby liczba Macha lub prędkość rakiety oraz patrząc z perspektywy zadania sterowania, wykorzystanie tego regulatora wymusza dokładny pomiar prędkości przepływu (np. za pomocą rurki Pitota). Dodatkowo w zależności od dynamiki możliwe jest wyznaczenie od kilkunastu do kilkudziesięciu pakietów nastaw. Powyższe wymogi komplikują implementacje oraz architekturę podzespołów elektronicznych. Sterownik ADRC nie wymaga takiego nakładu pracy. Z racji tego, że pracuje on w torze zamkniętym oraz dostaje aktualną wartość prędkości rotacji w czasie lotu rakiety, estymuje w czasie rzeczywistym działające całkowite zaburzenie. Daje to elastyczność w projektowaniu różnych modeli rakiet przy użyciu kilku parametrów projektowych [25] [41]. W związku z tym postanowiono wykorzystać sterownik ADRC.

3.3 Podstawy działania ADRC

Wybrany sterownik potrafi efektywnie kompensować wpływ niepewności modelu, jak i zaburzeń zewnętrznych. Metodą uporania się ze wspomnianymi trudnościami jest estymacja całkowitego zaburzenia. Dzięki temu można tak zaprojektować sygnał sterujący, by całkowite zaburzenie było efektywnie kompensowane.

Przykładową dynamikę SISO (ang. *single input single output*) można rozpisać jako [31] :

$$y^{(n)} + g(\cdot) + F(y, \dot{y}...y^{(n-1)}, t, p_o) = b_o u + d, \quad b_0 \neq 0, \tag{3.2}$$

gdzie: y - mierzalne wyjście, u - dostępne wejście sterujące, $g(\cdot)$ - dokładnie znany element modelu, d - niemierzalne zewnętrzne zakłócenie, $F(\cdot)$ - liniowa lub nieliniowa funkcja o nieznanej strukturze i zależna od nieznanych/niepewnych parametrów, b_0 - nieznany/niepewny, rzeczywisty współczynnik skalujący w torze wejścia sterującego. Następnie równanie (3.2) można zapisać następująco:

$$y^{(n)} + g(\cdot) = f(y, \dot{y}...y^{(n-1)}, t, p_o, u, d) + \hat{b}u, \tag{3.3}$$

gdzie: $\hat{b}$ - zgrubna estymata wartości b_0. Natomiast całkowite zaburzenie zdefiniowane jest jako:

$$f(y, \dot{y}...y^{(n-1)}, t, p_o, u, d) := -F(y, \dot{y}...y^{(n-1)}, t, p_o) + d + (b_0 - \hat{b}u). \tag{3.4}$$

W rezultacie działające zakłócenia oraz niepewności zebrane są w jedną funkcję $f(\cdot)$, która jest estymowana w czasie rzeczywistym.

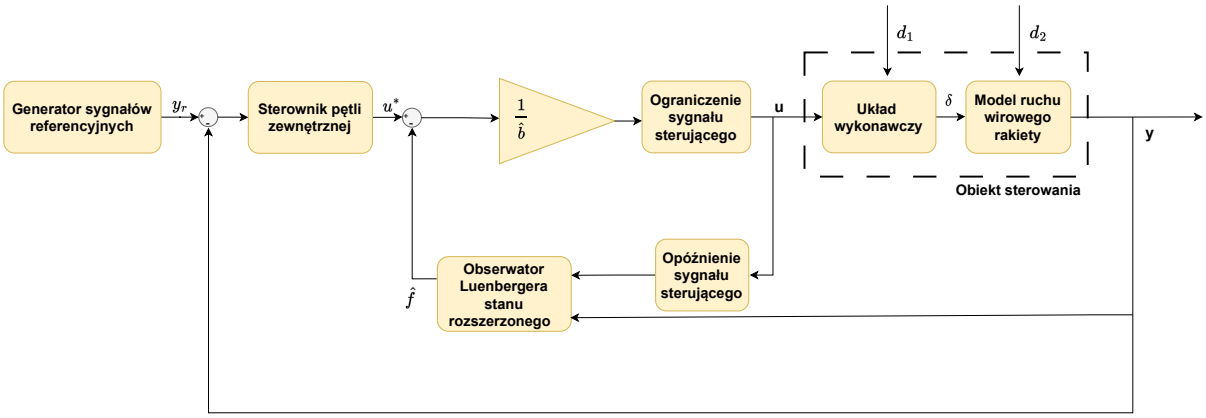
Następnie zakłada się, że w wybranej chwili czasowej estymata $\hat{f}$ jest dostępna. Na tej podstawie sygnał sterujący zdefiniowany jest jako [31] :

$$u \triangleq \frac{u^* - \hat{f}}{\hat{b}}, \tag{3.5}$$

gdzie u^* jest sygnałem sterującym pętli zewnętrznej (rys. 3.1). Podstawiając równanie (3.5) do (3.3) otrzymuje się:

$$y^n + g(\cdot) = u^* + \underbrace{(f(\cdot) - \hat{f})}_{\epsilon_f}, \tag{3.6}$$

gdzie: $f(\cdot)$ jest rzeczywistym całkowitym zaburzeniem, natomiast ϵ_f jest błędem estymacji zaburzenia. Jeżeli estymata $\hat{f}$ jest dobrze znana to $\epsilon_f = 0$. Wówczas kompensacja jest idealna. W rzeczywistości dąży się do jak najmniejszej wartości $|\epsilon_f|$ [31]. Wybrana struktura sterownika ADRC jest przedstawiona na rys. 3.1. Uwzględnia ona opóźnienie czasowe układu wykonawczego poruszającego lotkami sterującymi [35] oraz ograniczenie zakresu sygnału sterującego.



Rysunek 3.1: Schemat blokowy ADRC

Wybrany sterownik składa się z dwóch pętli sterowania. Nadrzędna powiązana jest ze sterownikiem konwencjonalnym (np. P/PI/PD/PID), podrzędna natomiast z pętlą kompensacji całkowitego zaburzenia [46].

Pętla podrzędna składa się przede wszystkim z wybranego liniowego obserwatora stanu rozszerzonego [46] (LESO ang. *Luenberger/Linear Extended State-Observer* — rys. 3.1). LESO pozwala na estymację aktualnie działającego całkowitego zaburzenia ($\hat{f}$ na rys. 3.1) oraz stanu obiektu sterowania (y na rys. 3.1). Zakłócenia działające na obiekt na rys. 3.1, oznaczone przez d_1 oraz d_2 jak i niepewności struktury modelu układu wykonawczego oraz obiektu sterowania potraktowane są jako działające całkowite zaburzenie [46]. Dodatkowo ze względu na charakter układu wykonawczego zostało uwzględnione opóźnienie w sygnale sterującym dla LESO (rys. 3.1) [35]. W obiekcie występuje również ograniczenie sygnału sterującego wynikające z dopuszczalnego kąta wychylenia lotek sterujących. Z racji tego taka informacja przekazywana jest również do obserwatora.

Pętla nadrzędna z wybranym sterownikiem posiada dodatkową opcję wyboru sygnału pomiarowego lub też sygnału estymowanego $\hat{y}$. Dzięki tej możliwości w przypadku pomiarów z dużym szumem pomiarowym istnieje możliwość wykorzystania estymowanej wartości y [23]. W wybranej strukturze zdecydowano wykorzystać sygnał pomiarowy (rys. 3.1).

Następnie zostaną wyprowadzone ogólne wzory umożliwiające projekt obserwatora stanu rozszerzonego Luenbergera (LESO) zgodnie z [24]. Model obiektu opisany jest, więc następująco:

$$\dot{x} = Ax + Bu + W\dot{f}(\cdot), \quad y = Cx, \tag{3.7}$$

gdzie: A jest macierzą stanu, B — macierzą wejść, C — macierzą wyjść, W — macierzą związaną z określeniem wpływu pochodnej całkowitego zaburzenia $\dot{f}(\cdot)$. Korzystając z (3.7), równanie obserwatora można zapisać następująco:

$$\dot{\widehat{x}} = A\widehat{x} + Bu + L(y - \widehat{y}), \tag{3.8}$$

gdzie: L jest wektorem wzmocnień obserwatora, czyli parametrem projektowym, y jest mierzonym wyjściem, a $\hat{y}$ — wartością wyjścia estymowaną przez LESO. Na podstawie (3.7) dostosowując do przypadku obserwatora, można podstawić do równania (3.8) oraz otrzymać:

$$\dot{\widehat{x}} = (A - LC)\widehat{x} + Bu + Ly. \tag{3.9}$$

Wyznaczając macierz L korzysta się z klasycznego wyprowadzenia dla obserwatora Luenbergera. W związku z tym nowa macierz stanu zdefiniowana jest jako:

$$H = A - LC, \tag{3.10}$$

gdzie H ma być macierzą Hurwitza, w której wszystkie wartości własne leżą w lewej półpłaszczyźnie zespolonej. Mając tak zdefiniowaną macierz H, zostaną ulokowane jej wartości własne:

$$det\,(\lambda I - H) = (\lambda + \omega_0)^n, \tag{3.11}$$

gdzie: I oznacza macierz jednostkową, ω_0 jest parametrem projektowym, który określa pasmo przenoszenia LESO, natomiast n jest rzędem obserwatora stanu rozszerzonego Luenbergera. Na tej podstawie wyznaczana jest macierz wzmocnień obserwatora L.

Chcąc zaimplementować sterownik ADRC na mikrokontrolerze, potrzebna jest jego dyskretyzacja. Wykorzystując metodę dyskretyzacji Eulera wprzód [47] równanie (3.8) zapisane zostanie jako:

$$\underline{\widehat{x}}(n) = \underbrace{(I + T_a A)}_{A_d}\widehat{x}(n-1) + \underbrace{T_a B}_{B_d}u(n-1) + \underbrace{T_a LC}_{O_d}(\epsilon(n-1)), \tag{3.12}$$

gdzie T_a oznacza okres próbkowania równań obserwatora, natomiast $\epsilon = y(n-1) - \hat{y}(n-1)$. W celu uzyskania dobrej jakości estymacji to znaczy otrzymania estymowanego całkowitego zaburzenia bliskiego prawdziwemu całkowitemu zaburzeniu, należy zmniejszać T_a oraz zwiększać ω_0.

3.4 Wyprowadzenie równań dla sterownika ADRC

Na początku wykorzystując wyprowadzone ogólne wzory, zostanie zaprojektowany LESO. Układ wykonawczy poruszający lotkami sterującymi został zdefiniowany jako obiekt inercyjny pierwszego rzędu bez opóźnienia czasowego (na podstawie [35]):

$$\frac{\delta}{\delta_z} = \frac{k}{sT + 1}, \tag{3.13}$$

gdzie: $\delta_z = u$ jest pozycją zadaną, a δ rzeczywistym wychyleniem lotek sterujących. Z racji tego można zapisać:

$$\dot{\delta} = \frac{k\delta_z}{T} - \frac{\delta}{T}. \tag{3.14}$$

Następnie równianie przyspieszenia rotacji przy założeniu, że rakieta jest symetryczna, to znaczy zależności skrośne prędkości w osi pionowej oraz poprzecznej, nie mają wpływu na przyspieszenie rotacji, równanie (3.1) rozpisane zostanie następująco:

$$\dot{p}I_x + \underbrace{\left(\frac{\rho v_m^2 A_{ref} d_{ref} C_{l_{d_s}}}{2} + \frac{\rho v_m^2 A_{ref} d_{ref} C_{l_{d_c}}}{2}\right)}_{C(v_m)} p \underbrace{- \frac{\rho v_m^2 A_{ref} d_{ref} C_{l_{f_s}}}{2}\delta_s}_{\tau_d(\delta_s, v_m)} = \underbrace{\frac{\rho v_m^2 A_{ref} d_{ref} C_{l_{f_c}}}{2}\delta}_{\mu(v_m)}, \quad (3.15)$$

co można przepisać jako:

$$\dot{p} = -\frac{C(v_m)}{I_x}p + \frac{\tau_d}{I_x} + \underbrace{\frac{\mu(v_m)}{I_x}}_{b_0}\delta. \quad (3.16)$$

Następnie równianie (3.16) zostanie zapisane według (3.3):

$$\dot{p} = b_0\delta + f_0(p, \tau_d, v_m), \quad (3.17)$$

gdzie $f_0(p, \tau_d, v_m) = \frac{\tau_d}{I_x} - \frac{C(v_m)}{I_x}p$. Następnie równianie (3.17) zostanie zróżniczkowane stronami:

$$\ddot{p} = \dot{b}_0\delta + b_0\dot{\delta} + \dot{f}_0(\cdot). \quad (3.18)$$

Następnie podstawiając równanie (3.14) do (3.18) zostanie otrzymane:

$$\ddot{p} = \underbrace{\frac{b_0 k}{T}}_{b_1}\delta_z + f_1(\dot{b}_0, \delta, b_0, f_0(\cdot)) \quad (3.19)$$

gdzie $f_1(\dot{b}_0, \delta, b_0, f_0(\cdot)) = \dot{b}_0\delta - \frac{b_0\delta}{T} + \dot{f}_0(\cdot)$. Co ostatecznie może być zapisane jako:

$$\ddot{p} = \underbrace{[b_1\delta_z + f_1(\cdot) - \hat{b}u]}_{F(b_1, \delta_z, f_1(\cdot))} + \hat{b}u, \quad (3.20)$$

gdzie całkowite zaburzenie zdefiniowane jest jako $F(b_1, \delta_z, f_1(\cdot)) = b_1\delta_z + f_1(\cdot) - \hat{b}u$. Definiując zmienne stanu jako:

$$x_1 = p \qquad x_2 = \dot{p} \qquad x_3 = F(\cdot),$$

równianie dynamiki stanu rozszerzonego zostanie zapisane w postaci (3.7):

$$\underline{\dot{x}} = \begin{bmatrix} \dot{p} \\ \ddot{p} \\ \dot{F}(\cdot) \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}}_{x} + \underbrace{\begin{bmatrix} 0 \\ \widehat{b} \\ 0 \end{bmatrix}}_{B} u + \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}}_{w} \cdot \dot{F}(\cdot), \quad (3.21)$$

$$y = \underbrace{\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}}_{C} \underline{x}. \quad (3.22)$$

Chcąc uzyskać równanie obserwatora, macierze A, B oraz równanie y podstawione zostanie do równania (3.8). W celu uzyskania pełnego wyprowadzenia równania obserwatora zdefiniowana zostanie macierz L na podstawie równań (3.9), (3.10) oraz (3.11), gdzie n=3:

$$L = \begin{bmatrix} 3\omega_0 \\ 3\omega_0^2 \\ \omega_0^3 \end{bmatrix}. \quad (3.23)$$

Następnie zaprojektowana zostanie pętla zewnętrzna. Ze względu na to, że jest to obiekt drugiego rzędu, postanowiono wykorzystać sterownik PD. Dzięki temu umożliwione zostało przyspieszenie reakcji

układu regulacji oraz modyfikacja tłumienia. W związku z tym obiekt sterowania przy założeniu idealnej kompensacji można zdefiniować jako obiekt drugiego rzędu z członem opóźniającym:

$$Y(s) = G^*(s) \cdot U^*(s) = \frac{e^{-sT_0}U^*(s)}{s^2}, \tag{3.24}$$

gdzie $U(s)^*$ jest transformatą Laplace'a sygnału u^* z rys. 3.1, a T_0 oznacza opóźnienie członu wykonawczego. Zakładając $\hat{f}(\cdot) \equiv f(\cdot)$, co oznacza, że estymowane całkowite zaburzenie jest równe prawdziwemu całkowitemu zaburzeniu, zdefiniowano sygnał sterujący u^* za sterownikiem PD:

$$u^*(t) = k_1(\dot{y}_r - \dot{y}) + k_0(y_r - y), \tag{3.25}$$

gdzie k_1 oznacza nastawę członu różniczkującego, natomiast k_0 nastawę członu proporcjonalnego. Zadaniem sterowania jest zerowanie prędkości kątowej, wobec tego można zdefiniować $y_r = 0$ oraz $\dot{y}_r = 0$. Na tej podstawie równanie (3.25) zostanie zapisane w dziedzinie Laplace'a w postaci:

$$U^*(s) \triangleq -k_1 sY(s) - k_0 Y(s). \tag{3.26}$$

Dalej podstawiając (3.24) do (3.26) po uporządkowaniu otrzymuje się:

$$s^2 Y(s) = [-k_1 sY(s) - k_0 Y(s)]e^{-sT_0}. \tag{3.27}$$

Człon e^{-sT_0} jest aproksymowany wyrażeniem $(1-sT_0)$. Podstawiając wspomnianą aproksymację do (3.27) po uporządkowaniu, otrzymano:

$$s^2 Y(s) + \underbrace{\frac{k_1 - k_0 T_0}{1 - k_1 T_0}}_{k_1^*} sY + \underbrace{\frac{k_0}{1 - k_1 T_0}}_{k_0^*} Y = 0, \tag{3.28}$$

gdzie stosując metodę lokowania biegunów [32] [23]:

$$k_1^* = 2\xi\omega_c, \tag{3.29}$$

oraz

$$k_0^* = \omega_c^2, \tag{3.30}$$

gdzie $\xi = (0; 1]$ oznacza współczynnik tłumienia, natomiast $\omega_c > 0$ pulsację nietłumionych drgań własnych układu sterowania, pulsację pętli zewnętrznej. Przy czym ξ oraz ω_c są parametrami projektowymi. Następnie rozwiązanie układu równań (3.29) oraz (3.30) pozwala wyznaczyć nastawy sterownika PD:

$$k_1 = \frac{2\xi\omega_c + T_0\omega_c^2}{1 + 2T_0\xi\omega_c + T_0^2\omega_c^2}, \tag{3.31}$$

$$k_0 = \frac{\omega_c^2}{1 + 2T_0\xi\omega_c + T_0^2\omega_c^2}. \tag{3.32}$$

Mając zdefiniowaną strukturę sterownika, omówione zostaną jego nastawy. Dla pętli adaptacji dobierane są wartości $\hat{b}$ oraz pasma przepustowego ω_0 obserwatora LESO. Wartość $\hat{b}$ jest zgrubną, oszacowaną wartością wzmocnienia toru sterowania. W związku z tym, że jest to wartość zależna od zmieniających się parametrów, jej wartość musi mieścić się w obliczonym zakresie. W przypadku wyznaczonych równań definiuje się $\hat{b}$, a następnie oblicza najmniejszą i największą możliwą wartość. W przypadku ω_0 im większa wartość, tym lepsza jakość estymacji oraz w konsekwencji bardziej efektywne sterowanie. Ograniczenie dla tego parametru wiąże się z tym, że przy dużych wartościach ω_0 może pojawić się niepożądany efekt nadmiernego wzmacniania szumów pomiarowych [23]. Z racji tego, że obiekt sterowania jest złożeniem układu wykonawczego oraz procesu zmiany prędkości wirowania rakiety, w konsekwencji obiekt

jest drugiego rzędu z opóźnieniem. W związku z tym nałożone są dodatkowe ograniczenia dla nastaw pętli zewnętrznej. Pasmo przenoszenia musi zostać dobrane bardziej konserwatywnie oraz wzrasta czułość systemu na dobór wartości $\hat{b}$ [43]. Pasmo pętli zewnętrznej ω_c musi być mniejsze niż pasmo obserwatora ω_0. Dzięki temu pętla adaptacji jest szybsza niż pętla zewnętrzna.

Dodatkowo na podstawie (3.28) można wyznaczyć następujące ograniczenia dla nastaw regulatora PD:

$$1 - k_1 T_0 > 0 \Rightarrow \frac{1}{T_0} > k_1 \tag{3.33}$$

drugie założenie, przy spełnieniu (3.33):

$$\frac{k_0}{1 - k_1 T_0} > 0 \Rightarrow k_o > 0 \tag{3.34}$$

oraz trzecie:

$$k_1 - k_0 T_0 > 0 \Rightarrow k_1 > k_0 T_0. \tag{3.35}$$

Wyznaczone zależności nastaw dla LESO, oraz regulatora PD stanowią podstawę strojenia ADRC zapewniającą stabilność układu.

Rozdział 4

Implementacja układu sterowania i modelu rakiety w środowisku MATLAB/Simulink

4.1 Implementacja modelu rakiety

Poniżej opisano sposób implementacji modelu rakiety w środowisku MATLAB/Simulink na podstawie równań przedstawionych w rozdziale 2 oraz parametrów fizycznych wykorzystanej rakiety.

Do wykonania modelu, poza parametrami geometrycznymi rakiety, wykorzystano charakterystyki oraz przebiegi niektórych parametrów wygenerowane przez program OpenRocket [7]. Jest to jeden z najczęściej używanych programów symulujących lot rakiety, wykorzystywanych przez zespoły zajmujące się budową rakiet sportowych. Pozwala on m.in. na wyznaczanie zapasu stabilności rakiety, wysokości apogeum, maksymalnej prędkości rakiety podczas lotu. Umożliwia także wyznaczenie przebiegów niektórych zmiennych w czasie parametrów rakiety.

Do parametrów wygenerowanych przy użyciu programu OpenRocket należą:

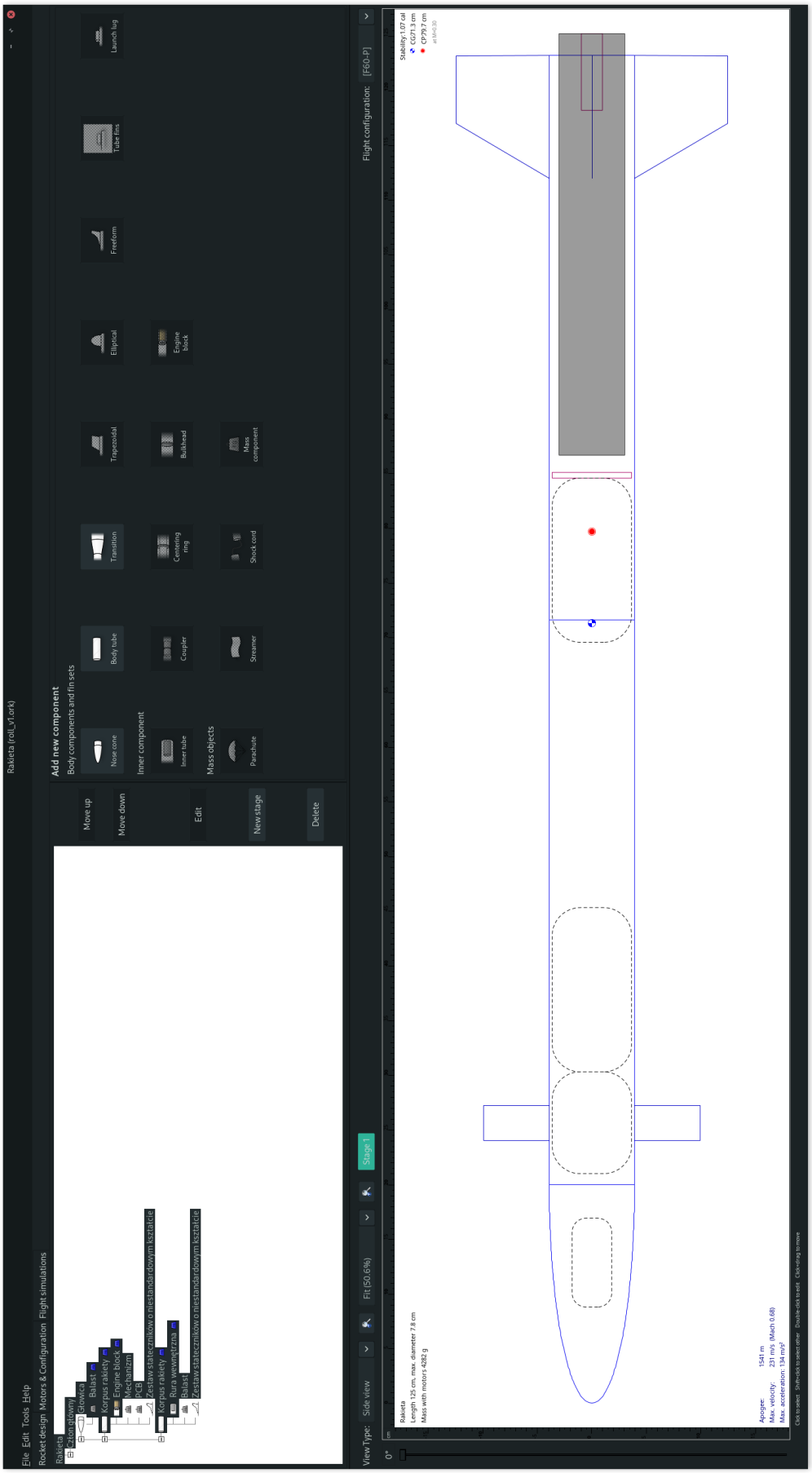
- charakterystyka silnika w postaci zależności wartości siły ciągu od czasu,
- masa rakiety, położenie środka ciężkości oraz momenty bezwładności, które są zmienne w czasie ze względu na wypalanie się paliwa,
- współczynnik siły oporu aerodynamicznego.

4.1.1 Parametry fizyczne wykorzystanej rakiety

Kadłub wykorzystanej w symulacji rakiety składa się z głowicy oraz dwie części kadłuba, związanych z lotkami sterującymi i stabilizującymi. Jego średnica jest równa $7,8 \cdot 10^{-2}$ m. Głowica ma kształt połowy elipsoidy, a jej długość wynosi $2 \cdot 10^{-1}$ m. Długości każdej z sekcji kadłuba jest równa $5,16 \cdot 10^{-1}$ m.

Zamodelowana rakieta wyposażona jest w cztery lotki stabilizujące o kształcie trapezu, umiejscowione w tylnej części kadłuba rakiety. Kształt każdego z nich można opisać korzystając z parametrów przedstawionych na rysunku 2.3. Rozpiętość (s_s) lotek stabilizujących wynosi $8,5 \cdot 10^{-2}$ m. Podstawy mają długość $1,12 \cdot 10^{-1}$ m (C_{r_s}) oraz $6,2 \cdot 10^{-2}$ m (C_{t_s}). Odległość pomiędzy początkami podstaw lotek stabilizujących (x_{t_s}) jest równa $5 \cdot 10^{-2}$ m. W celu zamodelowania niedokładności montażu lotek stabilizujących każdy z nich został wychylony o 0,5° od pionu.

Rakieta posiada także dwie prostokątne lotki sterujące, umiejscowione bliżej głowicy rakiety. Każdy z nich ma rozpiętość (s_c) równą $6 \cdot 10^{-2}$ m oraz długość podstawy $C_{r_c} = C_{t_c} = 3,2 \cdot 10^{-2}$ m. Ze względu na prostokątny kształt lotek sterujących parametr x_{t_c} jest równy 0 m.

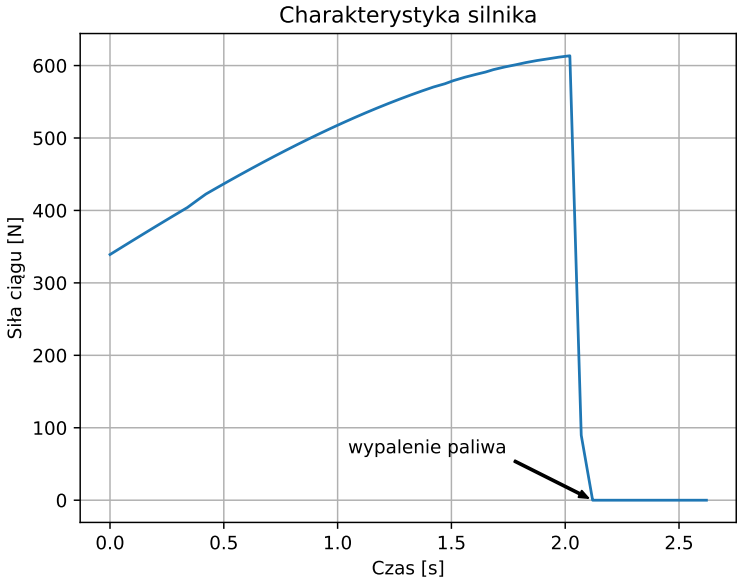


Rysunek 4.1: Zrzut ekranu przedstawiający główne okno programu OpenRocket [7]

W symulowanej rakiecie wykorzystano silnik na paliwo stałe. Jest to typ silnika, który generuje siłę ciągu, wykorzystując spalanie mieszanki paliwowej (paliwo z utleniaczem) w formie stałej. Impuls całkowity zastosowanego silnika wynosi 1049 Ns, co można wyznaczyć jako całkę z siły po czasie:

$$I = \int_0^{T_b} F_p(t)\, dt, \tag{4.1}$$

gdzie T_b oznacza czas wypalania się silnika. Siła ciągu generowana przez silnik została użyta w symulacji jako przebieg jej wartości w czasie. Wykres przedstawiający przebieg siły ciągu został przedstawiony na rys. 4.2. Można zauważyć, że zastosowany silnik wypala się po upływie ok. 2,12 s, osiągając maksymalną



Rysunek 4.2: Wykres siły ciągu w czasie

wartość siły ciągu na poziomie 613,53 N.

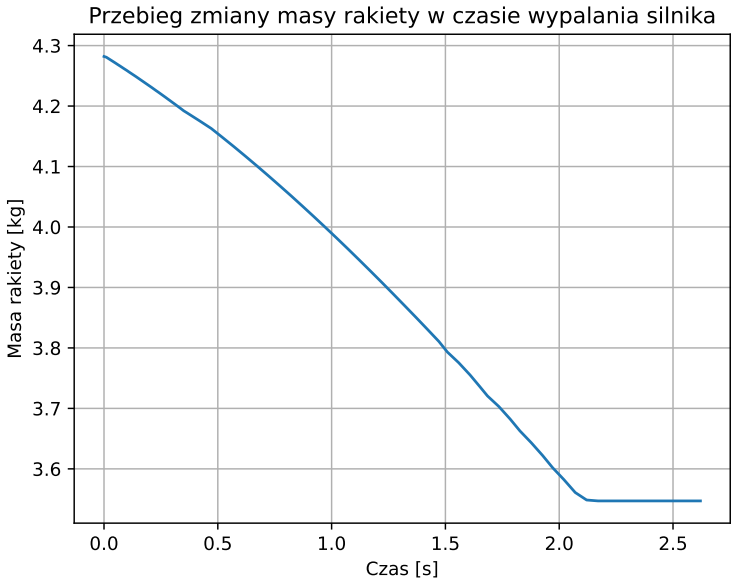
Ze względu na wypalanie się paliwa w silniku, masa rakiety ulega zmianie, dlatego potrzebne było wyznaczenie jej przebiegu w czasie. Wykres przedstawiający zmianę masy rakiety w czasie wypalania silnika przedstawiono na rys. 4.3.

Z powodu malejącej masy rakiety podczas wypalania paliwa, zmianie ulega także położenie środka ciężkości rakiety oraz wartości momentów bezwładności. Przebiegi przedstawiające zmianę wartości tych parametrów przedstawiono na rys. 4.4, 4.5 i 4.6.

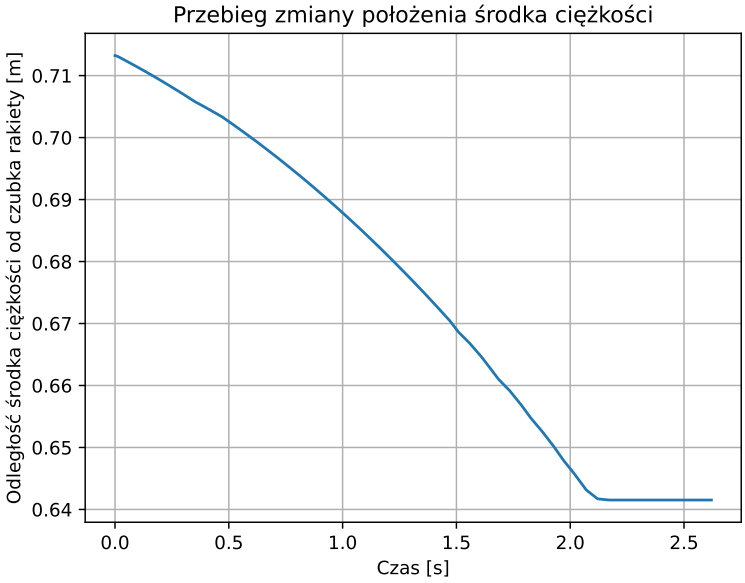
4.1.2 Sposób implementacji modelu rakiety w środowisku MATLAB/Simulink

Implementacja modelu rakiety została podzielona na 5 części, z których każda odpowiedzialna jest za obliczanie innych wielkości podczas lotu tj.:

- wyznaczanie parametrów takich jak:
  - gęstość powietrza,
  - prędkość dźwięku,
  - prędkość rakiety oraz jej liczba Macha,
  - ciśnienie dynamiczne,



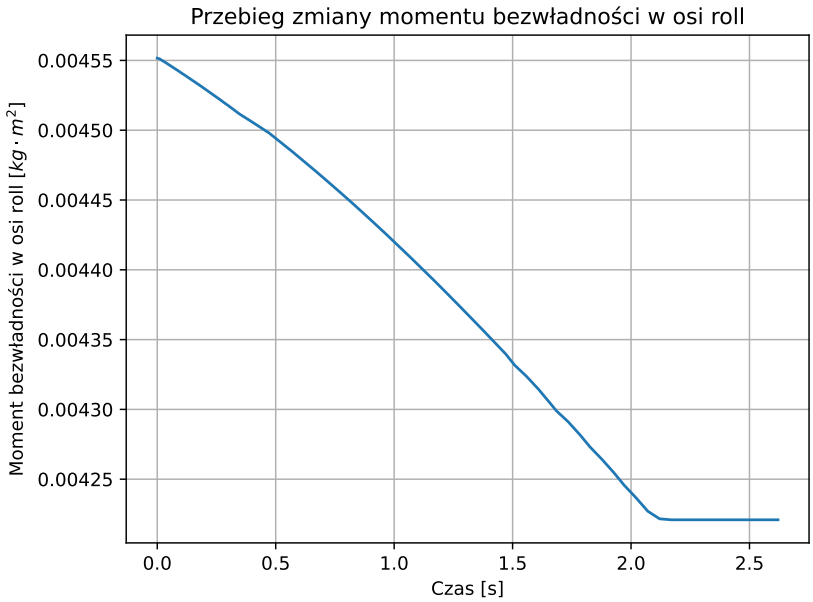
Rysunek 4.3: Wykres masy rakiety w czasie



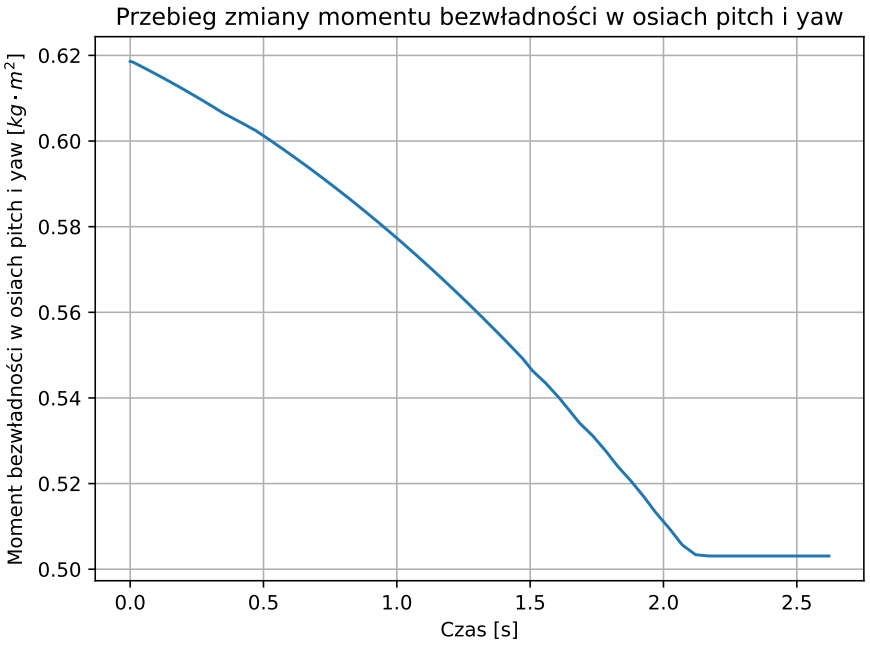
Rysunek 4.4: Wykres zmiany położenia środka ciężkości

  - wartości siły normalnej oraz momentów działających w osiach poprzecznej i pionowej,
  - pochodnych współczynników siły normalnej dla lotek sterujących i stateczników,

- prędkości w ruchu postępowym,
- prędkości w ruchu obrotowym,
- wartości kątów Eulera,
- przekształcenia pomiędzy lokalnym i globalnym układem współrzędnych.



Rysunek 4.5: Wykres zmiany momentu bezwładności w osi podłużnej

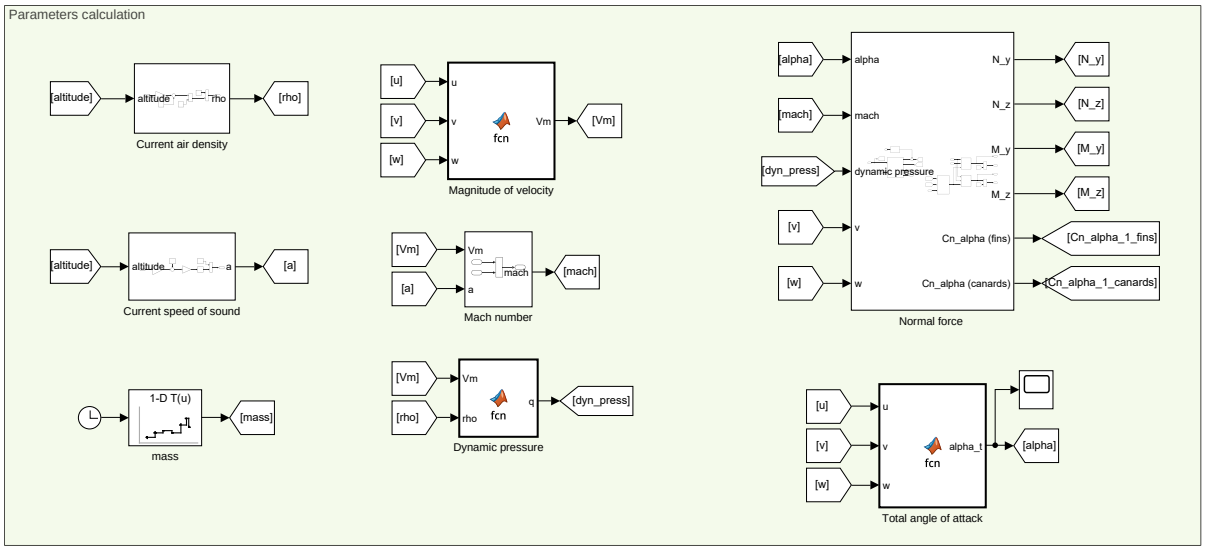


Rysunek 4.6: Wykres zmiany momentu bezwładności w osiach poprzecznej i pionowej

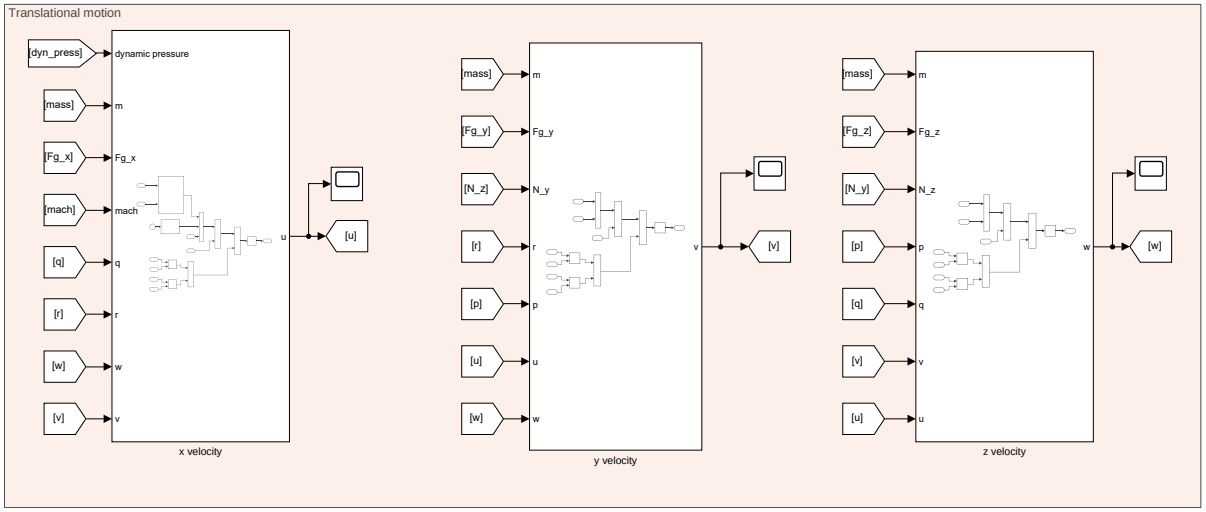
Schematy wykonanych układów w programie Simulink zostały przedstawione na rys. 4.7, 4.8, 4.9, 4.10 oraz 4.11.

Fragment modelu służący do wyznaczania niezbędnych wielkości fizycznych podczas lotu przedstawiony na rys. 4.7 składa się m.in. z podsystemu (ang. *subsystem*) obliczającego prędkość dźwięku na postawie aktualnej wysokości lotu (rys. 4.12). Prędkość dźwięku wyznaczana była za pomocą wzoru (2.13).

Kolejnym z wyznaczanych parametrów jest gęstość powietrza w zależności od aktualnej wysokości na podstawie wzoru (2.15). Implementacja podsystemu, który jest za to odpowiedzialny, widoczna jest



Rysunek 4.7: Fragment modelu obliczający parametry



Rysunek 4.8: Fragment modelu obliczający odpowiedź obiektu w ruchu postępowym

na rys. 4.13.

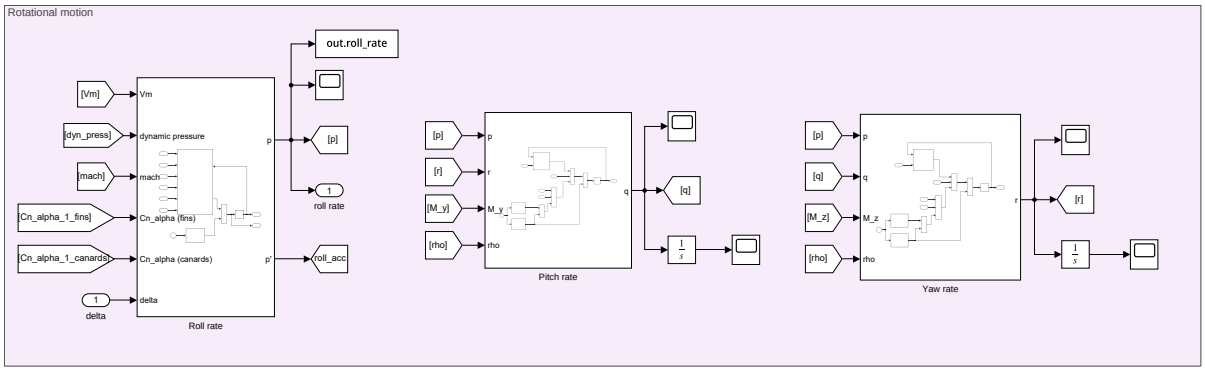
Zmiana masy rakiety (tak samo jako pozostałe parametry rakiety wygenerowane przy użyciu programu OpenRocket) została przedstawiona za pomocą bloku z tablicą przeglądową (ang. *lookup table*). Zawiera ona dane wygenerowane przez program OpenRocket w postaci krzywej dopasowanej do tych danych za pomocą funkcji *fit*.

Wartość wypadkowej prędkości postępowej rakiety została wyznaczona jako norma wektora prędkości, którego elementami są składowe prędkości w osiach x, y oraz z, czyli odpowiednio u, v, w. W modelu rakiety jest ona wyznaczana za pomocą bloku *MATLAB function* wykorzystującego funkcję:

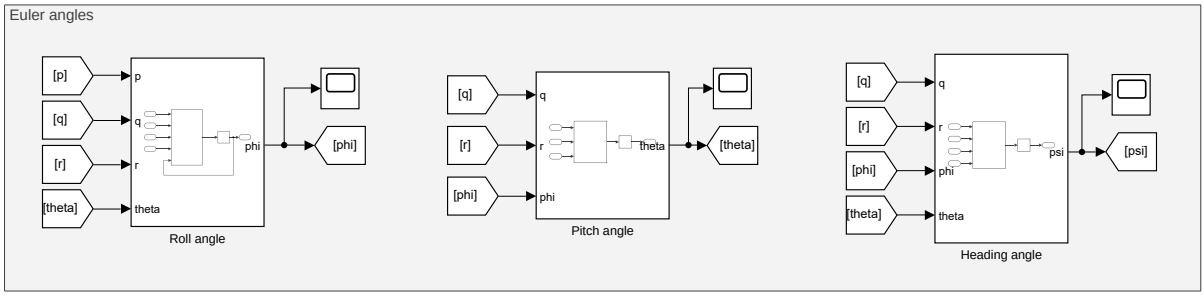
```
1 function Vm = fcn(u, v, w)
2 Vm = norm([u, v, w]);
```

Listing 4.1: Funkcja obliczająca wartość wypadkowej prędkości postępowej

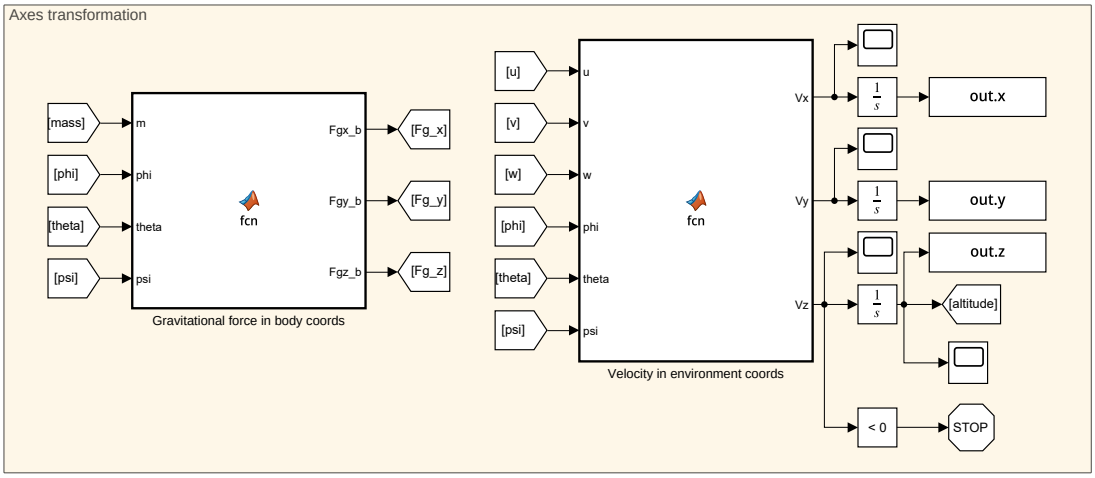
Na podstawie prędkości wyznaczonej przez funkcje z listingu 4.1 oraz aktualnej prędkości dźwięku wyznaczana jest wartość liczby Macha. Jest ona obliczana na podstawie wzoru (2.12) za pomocą podsystemu



Rysunek 4.9: Fragment modelu obliczający odpowiedź obiektu w ruchu obrotowym



Rysunek 4.10: Fragment modelu obliczający wartości kątów Eulera



Rysunek 4.11: Fragment modelu odpowiadający za przekształcenia pomiędzy lokalnym i globalnym układem wsp.

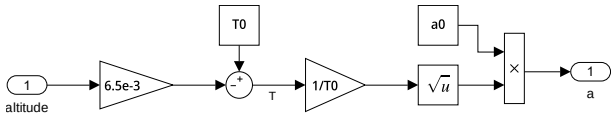
przedstawionego na rys. 4.14.

Wartość ciśnienia dynamicznego obliczana jest na podstawie wzoru (2.1) za pomocą bloku *MATLAB function* o nazwie *Dynamic pressure*. W strukturze bloku została zaimplementowana funkcja przedstawiona na listingu 4.2.

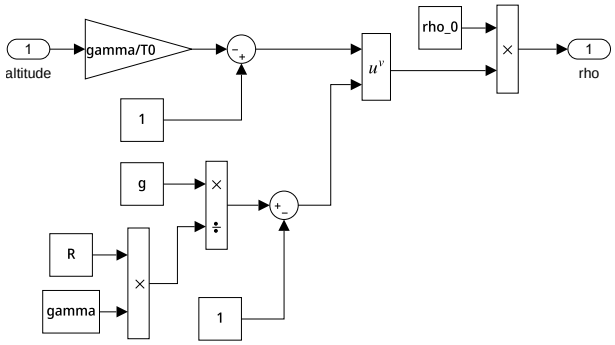
```
1 function q = fcn(Vm, rho)
2 q = 0.5 * Vm^2 * rho;
```

Listing 4.2: Funkcja obliczająca wartość ciśnienia dynamicznego

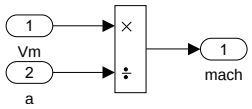
W bloku *MATLAB function* o nazwie *Total angle of attack* obliczana jest także wartość kąta natarcia. Wyznaczana jest ona na podstawie zależności wartości składowych wypadkowej prędkości postępowej



Rysunek 4.12: Podsystem wyznaczający aktualną prędkość dźwięku



Rysunek 4.13: Podsystem wyznaczający aktualną gęstość powietrza



Rysunek 4.14: Podsystem wyznaczający aktualną wartość liczby Macha

rakiety, korzystając ze wzoru:

$$\alpha = \operatorname{arctg}\frac{\sqrt{v^2 + w^2}}{u}. \tag{4.2}$$

Funkcja zaimplementowana we wspomnianym bloku została przedstawiona na listingu 4.3.

```
1 function alpha_t = fcn(u, v, w)
2 alpha_t = atan2(sqrt(v^2 + w^2), u);
```

Listing 4.3: Funkcja obliczająca wartość kąta natarcia

We fragmencie modelu odpowiadającym za wyznaczanie niektórych parametrów i wielkości fizycznych obliczane są także wartości składowych siły normalnej oraz momentów siły, które są przez nie generowane. Na rys. 4.15 przedstawiono podsystem, który za to odpowiada. Wyznacza on także pochodne współczynników siły normalnej ($C_{n_{\alpha_1}}$) generowane przez pojedyncze powierzchnie sterowe.

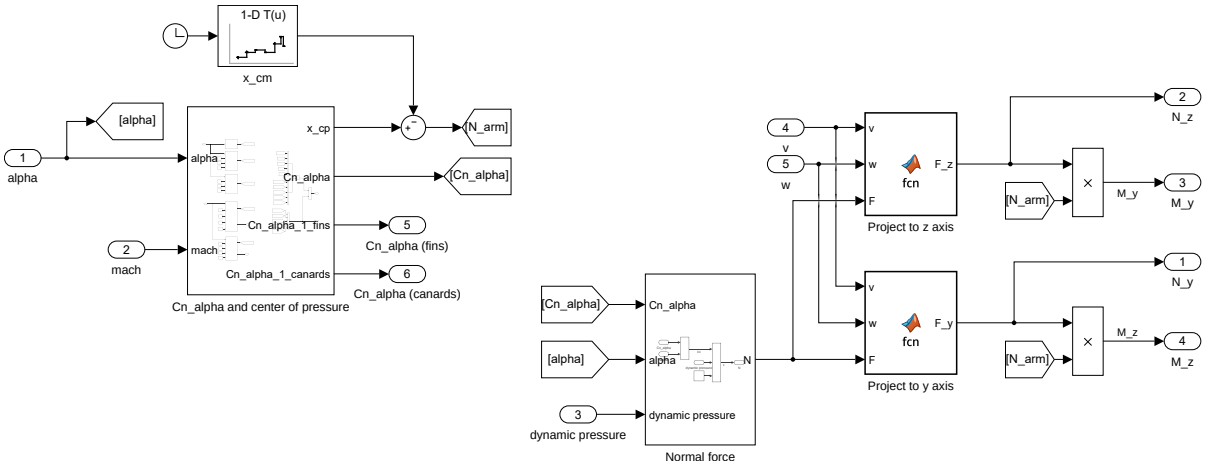
Wartość siły normalnej wyznaczana jest w bloku *Normal force* na podstawie wzoru (2.5), który został przedstawiony na rysunku 4.16. Następnie jest ona rzutowana na osie y i z układu współrzędnych rakiety za pomocą bloków *Project to y axis* oraz *Project to z axis*. Rzutowanie odbywa się za pomocą wzorów (2.35), które zostały zaimplementowane jako funkcje przedstawie na listingach 4.4 i 4.5.

```
1 function F_y = fcn(v, w, F)
2 F_y = -F * v / sqrt(v^2 + w^2);
```

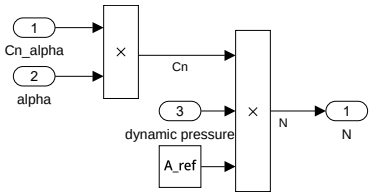
Listing 4.4: Funkcja obliczająca wartość siły normalnej w osi y

```
1 function F_z = fcn(v, w, F)
2 F_z = F * w / sqrt(v^2 + w^2);
```

Listing 4.5: Funkcja obliczająca wartość siły normalnej w osi z



Rysunek 4.15: Podsystem wyznaczający wartość siły normalnej, momentów przez nią generowanych oraz pochodne wsp. siły normalnej



Rysunek 4.16: Podsystem wyznaczający wartość siły normalnej

W celu obliczenia momentów sił generowanych przez składowe siły normalnej potrzebna jest długość ramienia, na jakiej te siły działają tj. różnica odległości środka parcia i środka ciężkości od czubka rakiety. Środek ciężkości wyznaczany jest, wykorzystując tablicę przeglądową (analogicznie do masy rakiety), natomiast środek parcia obliczany jest wewnątrz bloku *Cn_alpha and center of pressure*. Jego budowa wewnętrzna została przedstawiona na rys. 4.17.

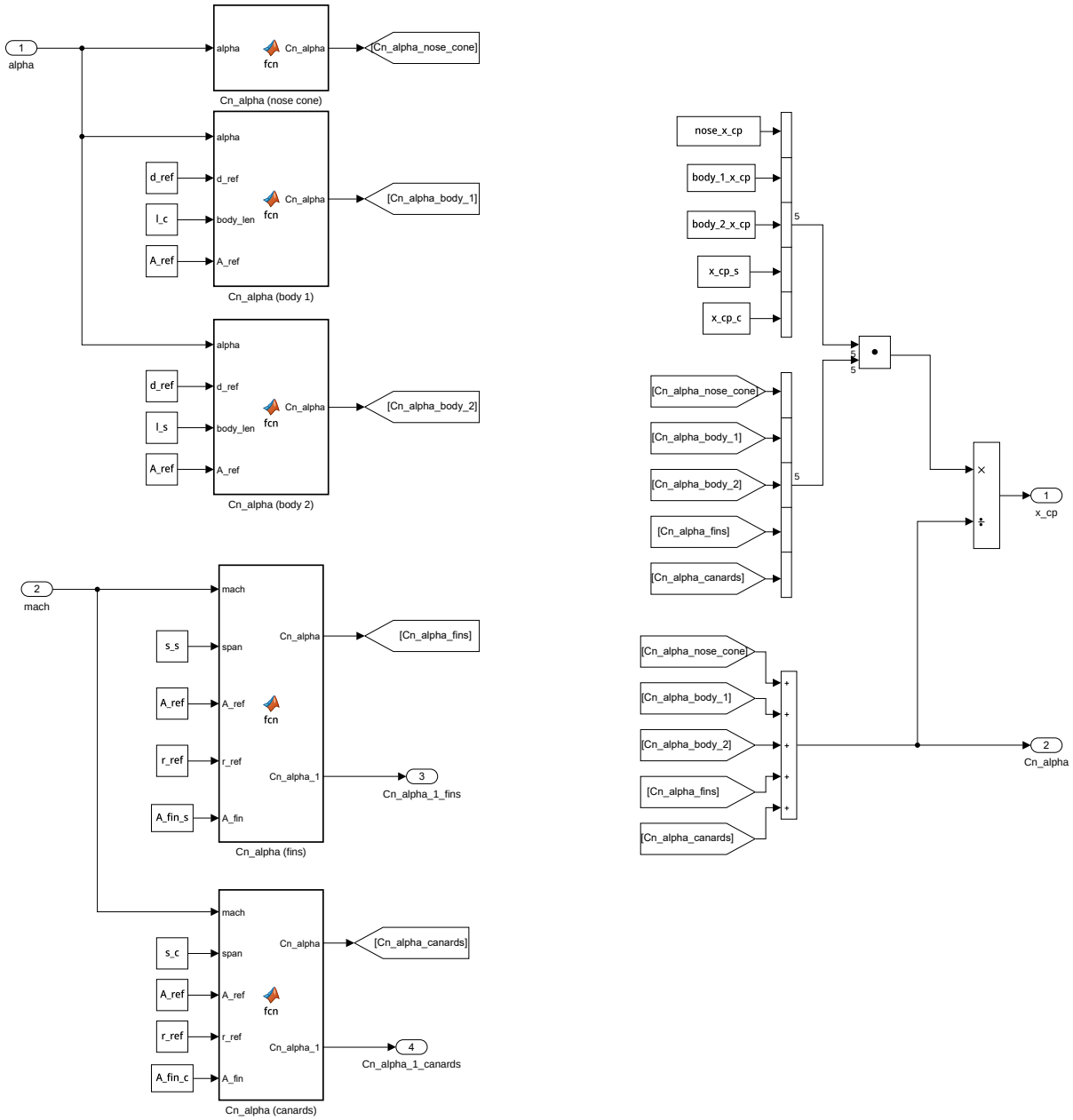
Wartość odległości środka parcia od czubka rakiety wyznaczana jest na podstawie wzoru (2.16) tj. jako średnia ważona odległości środków parcia komponentów rakiety, gdzie wagami są ich pochodne współczynników siły normalnej. Odległości środka parcia od czubka rakiety dla komponentów rakiety mają stałe wartości i zostały wyznaczone, wykorzystując wzory (2.17), (2.19), (2.20), (2.21), (2.22). Poniżej przedstawiono listingi z funkcjami pomocniczymi służącymi do obliczania położenia środków parcia głowicy, lotek sterujących oraz stateczników.

```
1 function x_cp = nose_cone_x_cp(nose_len, r_ref)
2 V = 2 / 3 * pi * nose_len * r_ref^2;
3 x_cp = nose_len - V / (pi * r_ref^2);
4 end
```

Listing 4.6: Funkcja obliczająca położenie środka parcia głowicy

```
1 function x_cp = canards_x_cp(nose_len, body_1_len, Cr, Ct)
2 x_cp = nose_len + (body_1_len - Cr - 2.5e-2) + 1 / 6 * (Cr^2 + Ct^2 + Cr * Ct) / (Cr + Ct);
3 end
```

Listing 4.7: Funkcja obliczająca położenie środka parcia lotek sterujących



Rysunek 4.17: Podsystem wyznaczający położenie środka parcia oraz wartość pochodnej współczynnika siły normalnej

```
1 function x_cp = fin_x_cp(nose_len, body_1_len, body_2_len, Cr, Ct, xt)
2 x_cp = nose_len + body_1_len + (body_2_len - Cr - 1e-2) + xt / 3 * (Cr + 2 * Ct) / (Cr + Ct) +
      ↪ 1 / 6 * (Cr^2 + Ct^2 + Cr * Ct) / (Cr + Ct);
3 end
```

Listing 4.8: Funkcja obliczająca położenie środka parcia stateczników

Wartości pochodnych współczynników siły normalnej wyznaczane są za pomocą wzorów (2.8), (2.9) i (2.10). Wzór (2.8) został zaimplementowany w bloku *Cn_alpha (nose cone)* w postaci funkcji przedstawionej na listingu 4.9.

```
1 function Cn_alpha = fcn(alpha)
2
```

```
3 if alpha ~= 0
4     Cn_alpha = 2 * sin(alpha) / alpha;
5 else
6     Cn_alpha = 2;
7 end
```

Listing 4.9: Funkcja obliczająca pochodną współczynnika siły normalnej dla głowicy rakiety

Warunek logiczny we wspomnianej funkcji służy jako zabezpieczenie przed dzieleniem przez 0 w sytuacji, kiedy kąt natarcia jest równy 0. Pochodne współczynników siły normalnej dla sekcji kadłuba wyznaczane są za pomocą bloków *Cn_alpha (body 1)* i *Cn_alpha (body 2)*, w których została zaimplementowana funkcja:

```
1 function Cn_alpha = fcn(alpha, d_ref, body_len, A_ref)
2 Cn_alpha = 1.1 * d_ref * body_len / A_ref * sin(alpha)^2;
```

Listing 4.10: Funkcja obliczająca pochodną współczynnika siły normalnej dla sekcji kadłuba

Wartości C_{n_α} jak i $C_{n_{\alpha_1}}$ dla lotek sterujących i stateczników obliczane są w blokach *Cn_alpha (canards)* oraz *Cn_alpha (fins)*. Funkcje, które są w nich zawarte przedstawiono odpowiednio na listingu 4.11 oraz 4.12.

```
1 function [Cn_alpha, Cn_alpha_1] = fcn(mach, span, A_ref, r_ref, A_fin)
2 Cn_alpha_1 = 2 * pi * span^2 / A_ref / (1 + sqrt(1 + (sqrt(abs(mach^2 - 1)) * span^2 /
      ↪ A_fin)^2));
3 Cn_alpha = (1 + r_ref / (span + r_ref)) * (sin(pi / 2)^2 * Cn_alpha_1 + sin(pi / 2 + pi)^2 *
      ↪ Cn_alpha_1);
```

Listing 4.11: Funkcja obliczająca pochodną współczynnika siły normalnej dla lotek sterujących

```
1 function [Cn_alpha, Cn_alpha_1] = fcn(mach, span, A_ref, r_ref, A_fin)
2 Cn_alpha_1 = 2 * pi * span^2 / A_ref / (1 + sqrt(1 + (sqrt(abs(mach^2 - 1)) * span^2 /
      ↪ A_fin)^2));
3 Cn_alpha = (1 + r_ref / (span + r_ref)) * 2 * Cn_alpha_1 * (1 - 0.06 * sin(2 * pi / 2));
```

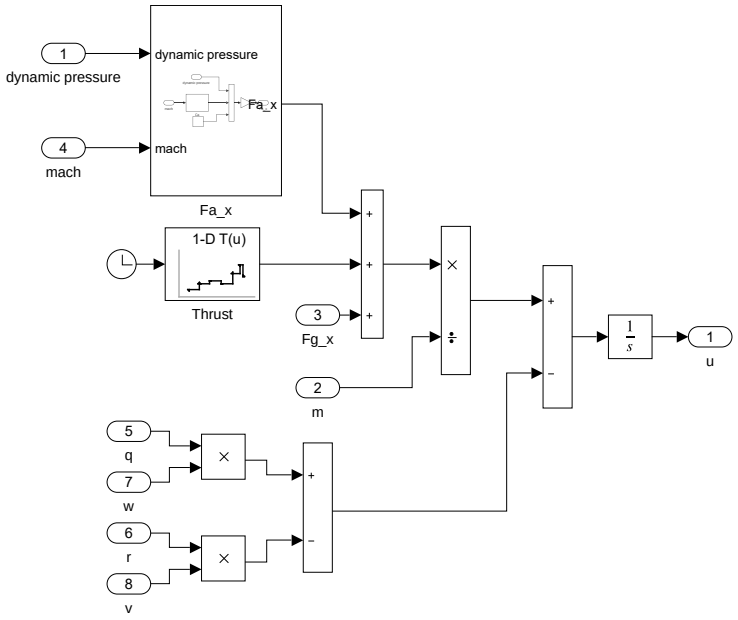
Listing 4.12: Funkcja obliczająca pochodną współczynnika siły normalnej dla stateczników

Całkowita pochodna współczynnika siły normalnej obliczana jest jako suma współczynników każdego z komponentów rakiety zgodnie z (2.7).

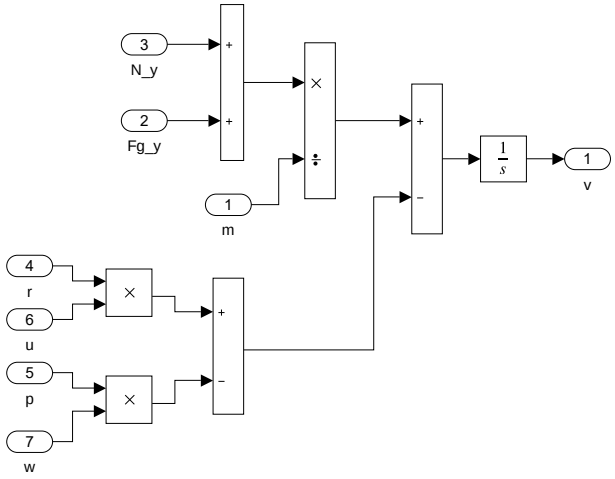
Kolejny fragment modelu odpowiedzialny jest za obliczanie prędkości w ruchu postępowym rakiety (rys. 4.8). Składa się on z 3 podsystemów *x velocity*, *y velocity* oraz *z velocity*, z których każdy służy do obliczania jednej ze składowych prędkości postępowej rakiety odpowiednio wzdłuż osi x, y i z całkując równanie (2.33). Ich implementacje zostały przedstawione na rysunkach 4.18, 4.19 i 4.20. Przyspieszenia obliczone za pomocą równania (2.33) są całkowane za pomocą integratorów w celu uzyskania składowych prędkości postępowej. Warunki początkowe każdej składowej są równe 0, ponieważ symulowano lot rakiety, zakładając, że uruchomienie silnika następuje w chwili $t = 0$.

Siła oporu aerodynamicznego w bloku *x velocity* wyznaczana jest z wykorzystaniem wzoru (2.4) w podsystemie *Fa_x* (rys. 4.21). Wykorzystuje on współczynnik siły oporu aerodynamicznego wygenerowany przy użyciu programu OpenRocket w formie tablicy przeglądowej, który został uzależniony od wartości liczby Macha.

Na wejścia *N_y* oraz *N_z* w blokach na rys. 4.19 i 4.20 podawane są wartości składowych w osiach y i z siły normalnej obliczanej w podsystemie z rys. 4.15. Wartości *Fg_x*, *Fg_y* i *Fg_z* oznaczają składowe



Rysunek 4.18: Podsystem wyznaczający prędkość rakiety w osi x



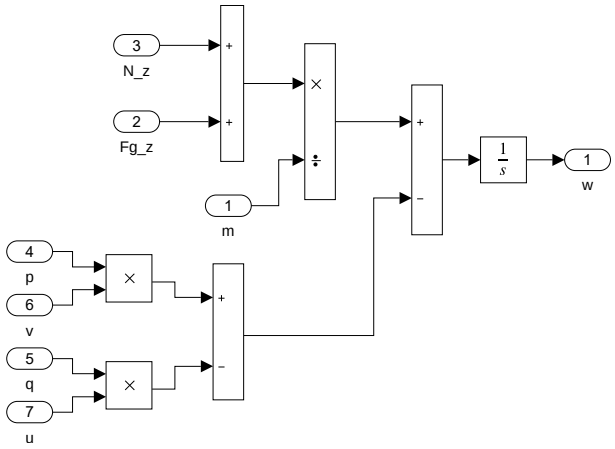
Rysunek 4.19: Podsystem wyznaczający prędkość rakiety w osi y

siły grawitacji w układzie lokalnym rakiety. Wejścia u, v, w oznaczają odpowiednio wartości prędkości postępowych rakiety, a p, q, r wartości prędkości kątowych. Na wejście m podawana jest aktualna masa rakiety wyznaczana z tablicy przeglądowej w sekcji na rys. 4.7.

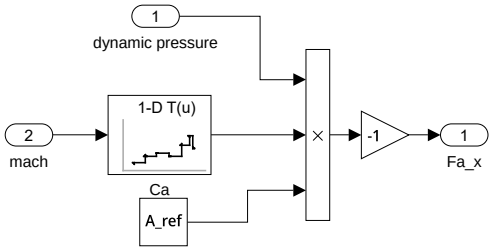
W sekcji przedstawionej na rys. 4.9 znajdują się podsystemy wyznaczające prędkości kątowe rakiety w osiach podłużnej, pionowej i poprzecznej całkując równania dynamiki w ruchu obrotowym (2.44). Ich implementacje przedstawione są na rysunkach 4.22, 4.23 i 4.24.

Blok *L_a* w podsystemie na rys. 4.22 służy do obliczania aerodynamicznego momentu siły generowanego przez powierzchnie sterowe. Jego wartość obliczana jest na podstawie wzoru (2.46). Sposób jego implementacji przedstawiono na rys. 4.25.

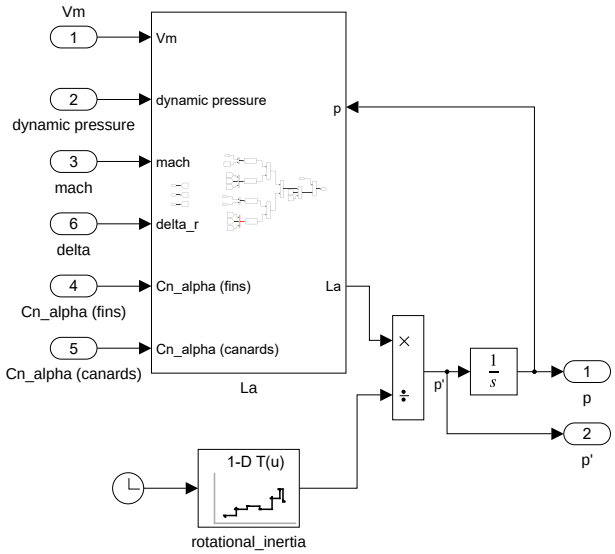
Wartości współczynników momentów sił forsujących oraz tłumiących ruch obrotowy w osi podłużnej, obliczane są za pomocą bloków *Interpreted MATLAB Fcn*. Umieszczono wewnątrz nich uchwyty funkcji (ang. *function handles*) wykorzystujących równania (2.47) i (2.49). Dzięki takiemu rozwiązaniu uniknięto potrzeby podawania stałych parametrów jako argumentów wykorzystywanych bloków, co zwiększa jego czytelność.



Rysunek 4.20: Podsystem wyznaczający prędkość rakiety w osi z



Rysunek 4.21: Podsystem wyznaczający wartość siły oporu aerodynamicznego

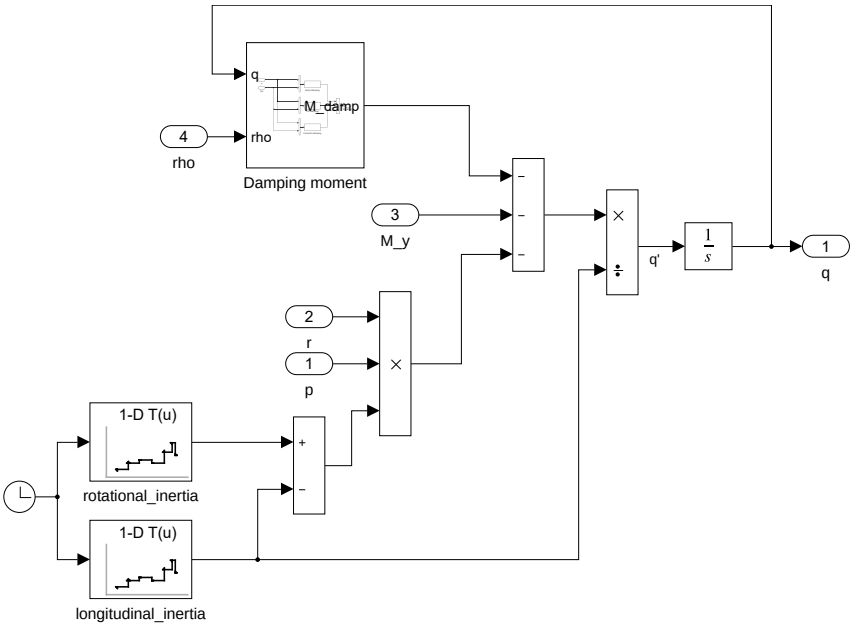


Rysunek 4.22: Podsystem wyznaczający wartość prędkości rotacji w osi podłużnej

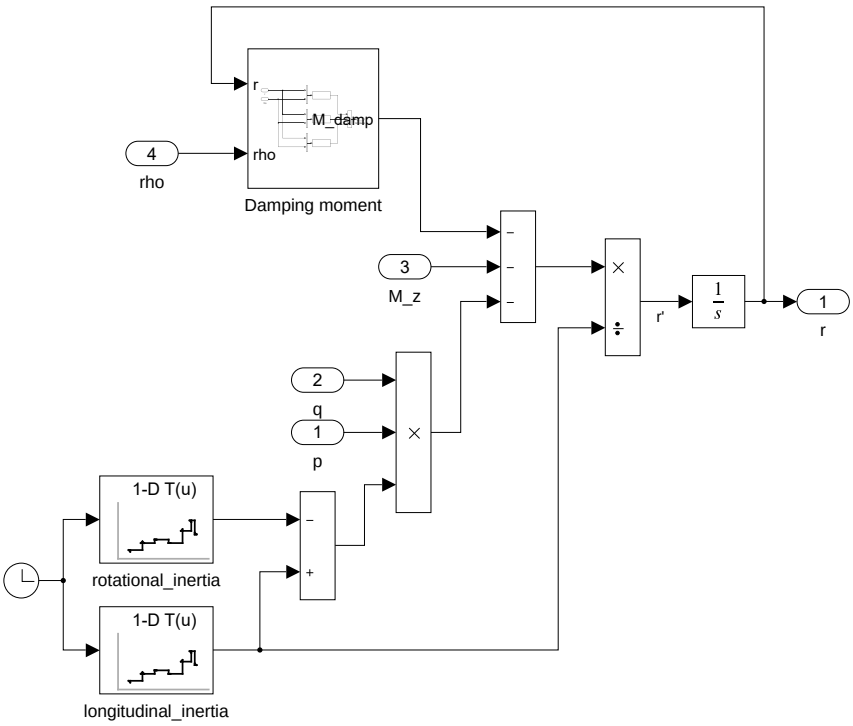
```
Clf_fins_s = @(Cnalfa_1, delta) Clf_fins(s_s, r_ref, Cr_s, Ct_s, d_ref, delta, N_s, Cnalfa_1);
Cld_fins_s = @(v0, omega, M) Cld_fins(d_ref, v0, N_s, omega, r_ref, Cr_s, s_s, Ct_s, M, A_ref);

Clf_fins_c = @(Cnalfa_1, delta) Clf_fins(s_c, r_ref, Cr_c, Ct_c, d_ref, delta, N_c, Cnalfa_1);
Cld_fins_c = @(v0, omega, M) Cld_fins(d_ref, v0, N_c, omega, r_ref, Cr_c, s_c, Ct_c, M, A_ref);
```

Listing 4.13: Uchwyty funkcji obliczających współczynniki momentów w osi podłużnej



Rysunek 4.23: Podsystem wyznaczający wartość prędkości rotacji w osi poprzecznej

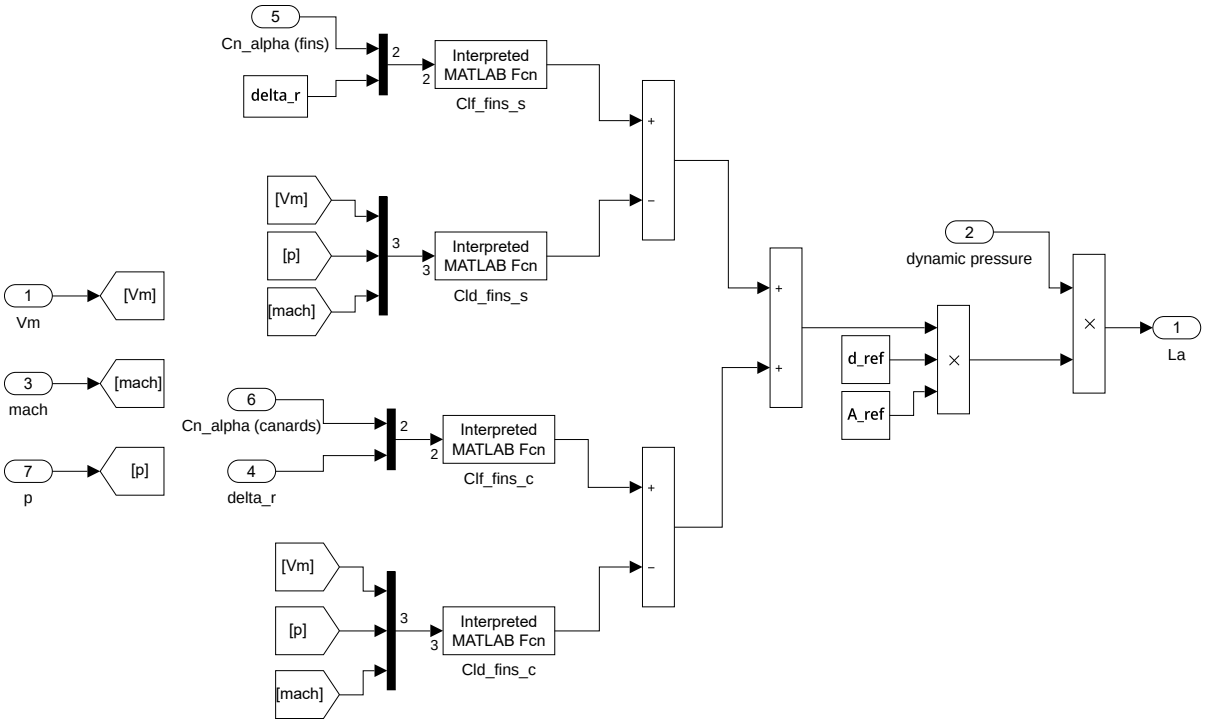


Rysunek 4.24: Podsystem wyznaczający wartość prędkości rotacji w osi pionowej

Implementacje funkcji `Clf_fins` i `Cld_fins`, które wyznaczają wartości współczynników przedstawione, zostały na poniższych listingach.

```
1 function Clf_fins = Clf_fins(s, r_ref, Cr, Ct, d_ref, delta, N, Cnalfa1)
2 yMAC = s * (Cr + 2*Ct) / (3 * (Cr + Ct));
3 Clf_fins = N * (yMAC + r_ref) * Cnalfa1 * delta / d_ref;
4 end
```

Listing 4.14: Funkcja obliczająca wartość współczynnika forsującego momentu siły



Rysunek 4.25: Podsystem wyznaczający wartość momentu siły generowanego przez powierzchnie sterowe

```
function Cl_d = Cld_fins(d_ref, v0, N, omega, r_ref, Cr, s, Ct, M, A_ref)
betha = sqrt(abs(1 - M^2));
CNa0 = 2 * pi / betha;
Cl_d = N * CNa0 * omega / (A_ref * d_ref * v0) * ((Cr + Ct) / 2 * r_ref^2 * s + (Cr + 2 * Ct) /
    ↪ 3 * r_ref * s^2 + (Cr + 3 * Ct) / 12 * s^3);
end
```

Listing 4.15: Funkcja obliczająca wartość współczynnika tłumiącego momentu siły

W przypadku podsystemów wyznaczających prędkość rotacji w osiach podłużnej i pionowej (rys. 4.23 i 4.24) wejścia *M_y* oraz *M_z* oznaczają wartości momentów sił generowanych przez działanie siły normalnej. Obliczane są one w podsystemie z rys. 4.15. Natomiast wartości momentów sił tłumiących ruch obrotowy w osiach podłużnej i pionowej obliczane są za pomocą bloków *Damping moment*. Oba bloki zostały zaimplementowane w analogiczny sposób. Blok dotyczący osi podłużnej został przedstawiony na rys. 4.26. Wartość tłumiącego momentu siły obliczana jest jako suma momentów generowanych przez korpus rakiety oraz oba zestawy powierzchni sterowych według równań (2.51) i (2.52). Równanie (2.51) zostało zaimplementowane za pomocą funkcji przedstawionej na listingu 4.16, a równanie (2.52) za pomocą funkcji 4.17.

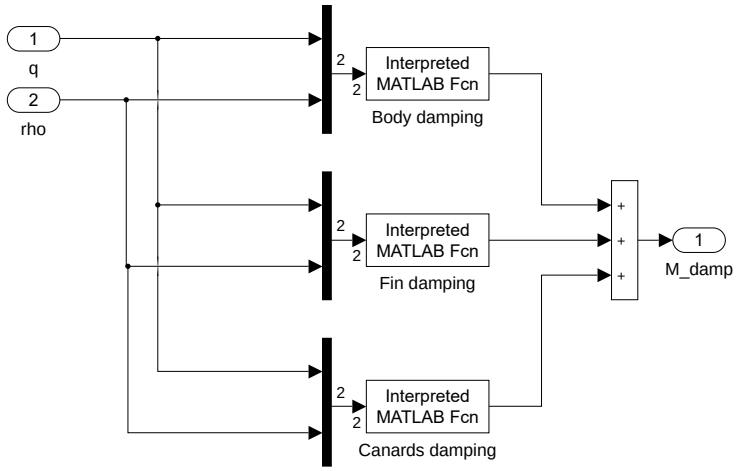
```
function M_damp = body_tube_damping(angular_rate, air_density, body_len, tube_radius)
M_damp = sign(angular_rate) * 0.275 * air_density * body_len^4 * tube_radius * angular_rate^2;
end
```

Listing 4.16: Funkcja obliczająca wartość momentu tłumiącego w osi poprzecznej i pionowej dla korpusu rakiety

```
function M_damp = fin_damping(angular_rate, air_density, num_of_fins, fin_area, kappa)
M_damp = sign(angular_rate) * 0.3 * air_density * num_of_fins * fin_area * kappa^3 *
```

```
    ↪ angular_rate^2;
3 end
```

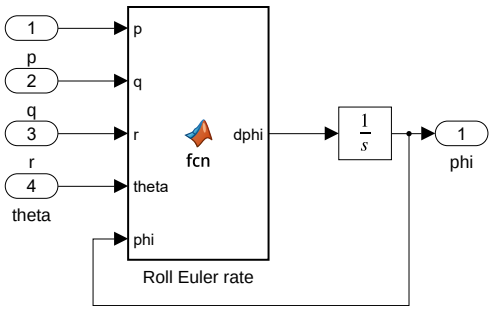
Listing 4.17: Funkcja obliczająca wartość momentu tłumiącego w osi poprzecznej i pionowej dla zestawu lotek



Rysunek 4.26: Podsystem wyznaczający wartość momentu siły tłumiącego ruch w osi podłużnej

Momenty bezwładności rakiety wykorzystywane przez podsystemy obliczające prędkości rotacji wyznaczane są za pomocą tablic przeglądowych wygenerowanych na podstawie danych z programu OpenRocket. Na potrzeby symulacji przyjęto zerowe warunki początkowe w integratorach obliczających prędkości rotacji, ponieważ założono, że rakieta nie porusza się przed startem.

Podsystem służący do wyznaczania wartości kątów Eulera (rys. 4.10), wykorzystywanych przy opisie przekształceń pomiędzy globalnym układem współrzędnych a układem współrzędnych rakiety, składa się z 3 podsystemów. Każdy z nich oblicza wartość jednego z 3 kątów φ, θ i ψ na podstawie równania (2.40). Na rys. 4.27 widoczna jest implementacja podsystemu odpowiadającego za obliczanie wartości kąta φ. Zaimplementowana w tym podsystemie funkcja w bloku *MATLAB function* została przedstawiona na li-



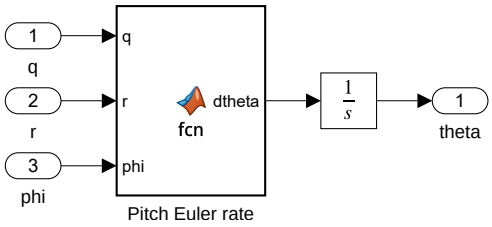
Rysunek 4.27: Podsystem wyznaczający wartość kąta φ

stingu 4.18. Przyjęto zerową wartość początkową kąta φ.

```
1 function dphi = fcn(p, q, r, theta, phi)
2 dphi = p + (q * sin(phi) + r * cos(phi)) * tan(theta);
```

Listing 4.18: Funkcja obliczająca prędkość zmiany wartości kąta φ

Podsystem obliczający wartość kąta θ został przedstawiony na rys. 4.28. Listing 4.19 przedstawia funkcję zaimplementowaną wewnątrz bloku *MATLAB function*. Przyjęto, że początkowa wartość kąta θ jest równa $\frac{\pi}{2}$ co oznacza, że rakieta przed startem skierowana jest pionowo do góry.

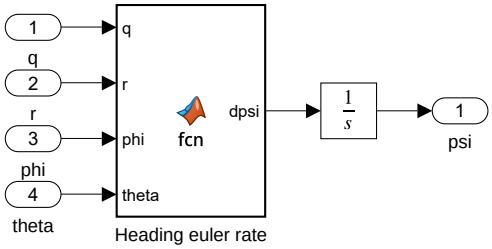


Rysunek 4.28: Podsystem wyznaczający wartość kąta θ

```
function dtheta = fcn(q, r, phi)
dtheta = q * cos(phi) - r * sin(phi);
```

Listing 4.19: Funkcja obliczająca prędkość zmiany wartości kąta θ

Ostatni z podsystemów odpowiadający za wyznaczanie wartości kąta ψ został przedstawiony na rys. 4.29. Funkcja zaimplementowana w bloku *MATLAB function* została przedstawiona na listingu 4.20.



Rysunek 4.29: Podsystem wyznaczający wartość kąta ψ

```
function dpsi = fcn(q, r, phi, theta)
dpsi = (q * sin(phi) + r * cos(phi)) / (cos(theta) + (cos(theta) == 0) * eps);
```

Listing 4.20: Funkcja obliczająca prędkość zmiany wartości kąta ψ

Fragment służący do przekształceń pomiędzy globalnym układem współrzędnych i układem współrzędnych rakiety (rys. 4.11) składa się z dwóch bloków *MATLAB function*. Pierwszy z nich (*Gravitational force in body coords*) jest odpowiedzialny za wyznaczenie wartości składowych siły grawitacji działającej na rakietę w jej lokalnym układzie współrzędnych. Funkcja (listing 4.21) zaimplementowana wewnątrz tego bloku wykorzystuje macierz transformacji z równania (2.38).

```
function [Fgx_b, Fgy_b, Fgz_b] = fcn(m, phi, theta, psi)

Fgz_e = m * 9.81;

Te_b = [cos(theta) * cos(psi), sin(phi) * sin(theta) * cos(psi) - cos(phi) * sin(psi), cos(phi)
    ↪ * sin(theta) * cos(psi) + sin(phi) * sin(psi);
       cos(theta) * sin(psi), sin(phi) * sin(theta) * sin(psi) + cos(phi) * cos(psi), cos(phi)
           ↪ * sin(theta) * sin(psi) - sin(phi) * cos(psi);
       -sin(theta), sin(phi) * cos(theta), cos(phi) * cos(theta)];

b = Te_b' * [0; 0; Fgz_e];
Fgx_b = b(1);
Fgy_b = b(2);
```

```
12 Fgz_b = b(3);
```

Listing 4.21: Funkcja obliczająca wartości składowych siły grawitacji w układzie wsp. rakiety

Równanie (2.38) jest także wykorzystywane w bloku *Velocity in environment coords* w celu wyznaczenia składowych prędkości w ruchu postępowym rakiety w układzie globalnym. Funkcja, która wykorzystywana jest do tego celu została przedstawiona na listingu 4.22.

```
1  function [Vx, Vy, Vz] = fcn(u, v, w, phi, theta, psi)
2
3  Te_b = [cos(theta) * cos(psi), sin(phi) * sin(theta) * cos(psi) - cos(phi) * sin(psi), cos(phi)
       ↪ * sin(theta) * cos(psi) + sin(phi) * sin(psi);
4          cos(theta) * sin(psi), sin(phi) * sin(theta) * sin(psi) + cos(phi) * cos(psi), cos(phi)
               ↪ * sin(theta) * sin(psi) - sin(phi) * cos(psi);
5          -sin(theta), sin(phi) * cos(theta), cos(phi) * cos(theta)];
6
7  b = Te_b * [u; v; w];
8  Vx = b(1);
9  Vy = b(2);
10 Vz = -b(3);
```

Listing 4.22: Funkcja obliczająca wartości składowych prędkości w ruchu postepowym w układzie globalnym

Zmiana znaku prędkości w osi z układu globalnego wynika z tego, że podczas symulacji obserwowana jest prędkość wznoszenia rakiety, a oś z układu globalnego skierowana jest w dół. Obliczana jest także wysokość, na jakiej aktualnie znajduje się rakieta (przyjęto zerową wysokość początkową). Dodatkowo symulacja jest przerywana (za pomocą bloku *Stop Simulation*) w momencie wykrycia apogeum wysokość tj. zmiany znaku prędkości wznoszenia rakiety.

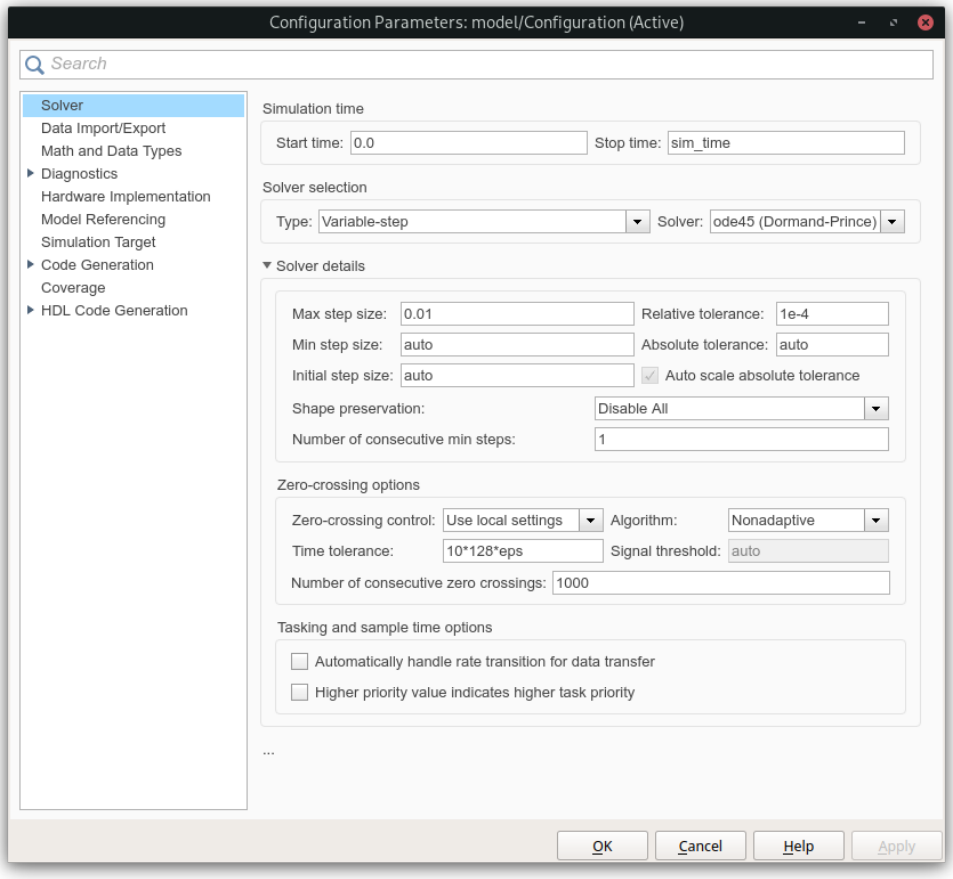
Na rys. 4.30 przedstawiono ustawienia solvera wykorzystywanego do symulacji w programie MATLAB. Wybrany został solver *ode45* o zmiennym kroku czasowym. Maksymalna wartość kroku czasowego była równa 0,01 s.

4.1.3 Implementacja układu wykonawczego w środowisku MATLAB/Simulink

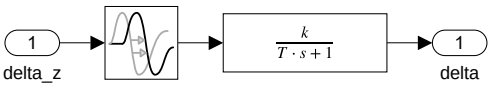
W celu symulowania działania zaprojektowanego układu regulacji potrzebne było także zamodelowanie układu wykonawczego służącego do poruszania lotkami sterującymi. Przyjęto, że serwomechanizm realizujący ruch lotek sterujących można zamodelować za pomocą inercji pierwszego rzędu z opóźnieniem czasowym. Na podstawie identyfikacji modelu serwomechanizmu stwierdzono, że wzmocnienie statyczne k wykorzystanego modelu jest równe 1, stała czasowa została przyjęta na poziomie $T = 0,0197$, a opóźnienie czasowe jako $T_o = 0,025$. Podsystem implementujący model układu wykonawczego przedstawiono na rys. 4.31.

4.2 Porównanie działania modelu rakiety z programem OpenRocket

Skuteczność działania modelu rakiety zaimplementowanego w programie MATLAB została porównana z wynikami uzyskanymi z programu OpenRocket. Na rys. 4.32 przedstawiono porównanie przebiegów prędkości rotacji rakiety podczas lotu uzyskane za pomocą zaimplementowanego modelu rakiety oraz programu OpenRocket. Można zauważyć, że pojawia się różnica w skali wykresów. Prędkość rotacji uzyskiwana w symulacji w programie OpenRocket jest ok. 1,28 razy większa niż uzyskana przez model



Rysunek 4.30: Ustawienia solvera wykorzystanego w modelu



Rysunek 4.31: Podsystem zawierający model mechanizmu

rakiety w programie MATLAB. Wynika to ze zmian wprowadzonych przez autora programu OpenRocket w sposobie wyznaczania prędkości rotacji [7].

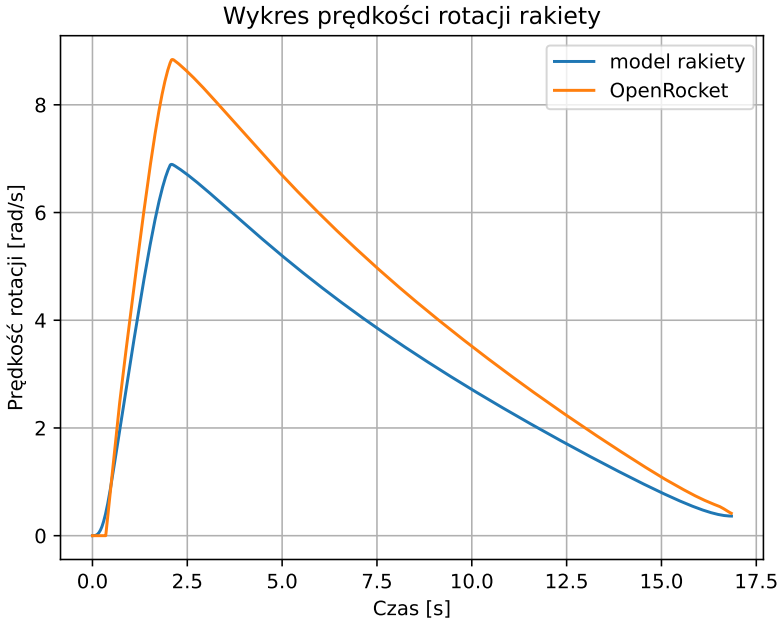
Porównano także przebiegi prędkości wznoszenia rakiety (rys. 4.33) oraz wysokości lotu (rys. 4.34). Można zauważyć, że zachowanie modelu rakiety oraz symulacja w programie OpenRocket są bardzo do siebie zbliżone.

Maksymalna prędkość rakiety podczas lotu wynosi ok. $228,47\,\frac{\mathrm{m}}{\mathrm{s}}$. Jest ona osiągana w momencie wypalenia silnika. Po osiągnięciu maksymalnej prędkości jej wartość spada do zera w momencie osiągnięcia apogeum wysokości.

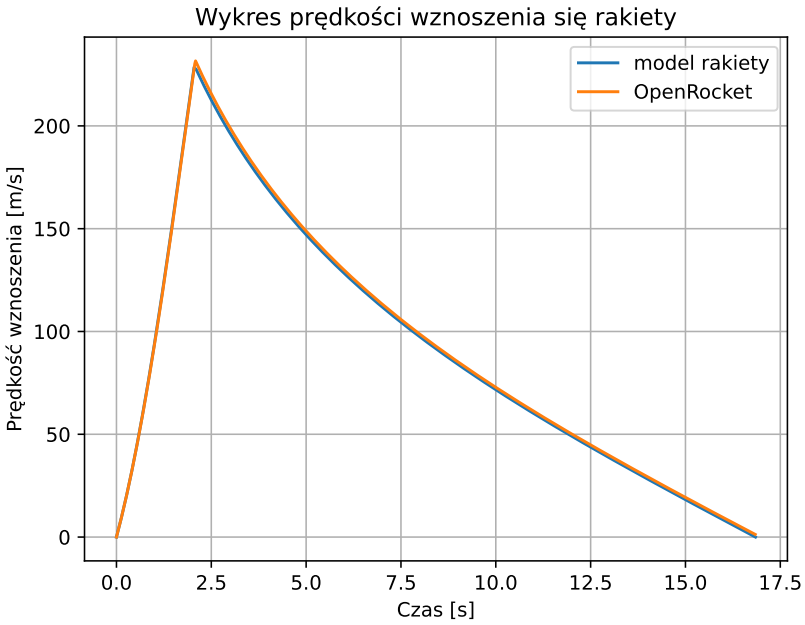
Przebiegi wysokości lotu także są zbliżone do siebie. Można z nich odczytać, że apogeum wysokości podczas lotu jest równe ok. 1520,23 m.

4.3 Implementacja układu sterowania ADRC

W celu symulacji działania systemu sterowania potrzebne było zaimplementowanie zaprojektowanego sterownika ADRC w programie MATLAB. Dzięki niemu możliwe było sprawdzenie skuteczności układu sterowania przed eksperymentalną weryfikacją z wykorzystaniem rzeczywistego obiektu.



Rysunek 4.32: Porównanie wykresów prędkości rotacji rakiety

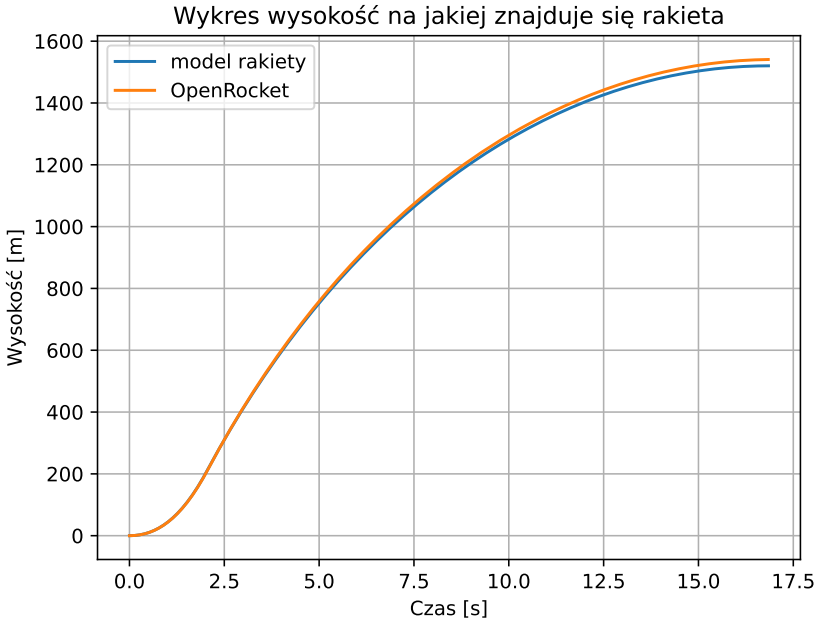


Rysunek 4.33: Porównanie wykresów prędkości wznoszenia rakiety

4.3.1 Implementacja w środowisku MATLAB/Simulink

Podsystem, w którym został zaimplementowany sterownik ADRC został przedstawiony na rys. 4.35. Wejściami bloku sterownika są wartość referencyjna prędkości rotacji *p_ref* (na potrzeby tej pracy jest ona równa 0) oraz aktualna prędkość rotacji p, a wyjściem u jest sygnał sterujący wyznaczony przez sterownik. Wartości sygnałów wejściowych są próbkowane z okresem 0,001 s.

Na podstawie wartości referencyjnej prędkości rotacji oraz jej aktualnej wartości, obliczany jest uchyb,



Rysunek 4.34: Porównanie wykresów wysokości lotu rakiety

którego wartość podawana jest na wejście bloku implementującego sterownik pętli zewnętrznej. Sterownika pętli zewnętrznej został zaimplementowany na podstawie wzoru (3.25). Funkcja wykorzystywana wewnątrz bloku *Sterownik zewn.* została przedstawiona na listingu 4.23.

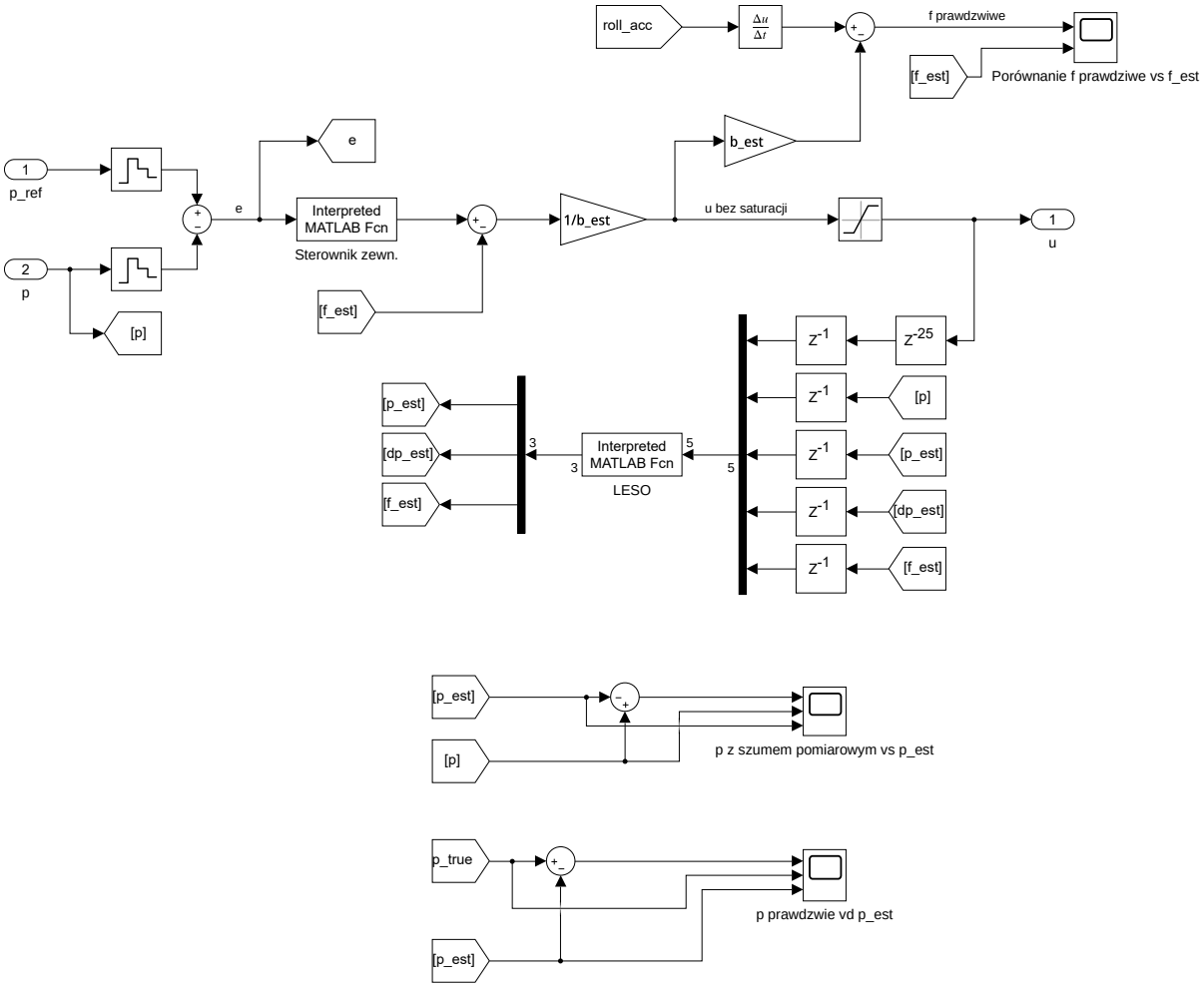
```
1 function u_g = sterownik_zew(e)
2 global kp kd prev_e Tc;
3 u_g = kp * e + kd * (e - prev_e) / Tc;
4 prev_e = e;
5 end
```

Listing 4.23: Funkcja implementująca sterownik zewnętrzny

Człon różniczkujący sterownika został zaimplementowany, korzystając z metody Eulera wstecz. Zmienne `kp` oraz `kd` oznaczają nastawy sterownika (odpowiednio k_0 oraz k_1). Wartość `prev_e` przechowuje poprzednią wartość uchybu, a `Tc` przechowuje wartość okresu próbkowania pętli zewnętrznej sterownika.

Obserwator LESO także został zaimplementowany za pomocą bloku *Interpreted MATLAB Fcn*. Funkcja wykorzystana w środku tego bloku została przedstawiona na listingu 4.24. Obliczanie wartości próbek estymowanego całkowitego zaburzenia oraz estymowanych wartości prędkości i przyspieszenia rotacji zostały zapisane na podstawie równania (3.12). Na wejście LESO podawane są poprzednie próbki sygnału sterującego (z uwzględnieniem opóźnienia czasowego układu wykonawczego wynoszącego 25 próbek tj. 0,025 s), prędkości rotacji, estymowanej prędkości rotacji, estymowanego przyspieszenia rotacji oraz estymowanego całkowitego zaburzenia.

```
1 function xp_est = LESO(u)
2 us = u(1);
3 p = u(2);
4 p_est = u(3);
5 dp_est = u(4);
6 f_est = u(5);
```



Rysunek 4.35: Podsystem zawierający implementację sterownika ADRC

```
7
8  global A_d B_d O_d;
9  x_est = [p_est; dp_est; f_est];
10
11 xp_est = A_d * x_est + B_d * us + O_d * (p - p_est);
12 end
```

Listing 4.24: Funkcja implementująca obserwator LESO

Wyznaczenie wartości macierzy dyskretnych równań stanu obserwatora wykonywane jest w głównym skrypcie programu w sposób przedstawiony na listingu 4.25.

```
1 L = [3 * omega_0; 3 * omega_0^2; omega_0^3];
2
3 % macierze do rownania czasu ciaglego LESO
4 A = [0 1 0; 0 0 1; 0 0 0];
5 B = [0; b_est; 0];
6 C = [1 0 0];
7
8 % macierze do rownania czasu dyskretnego LESO
9 A_d = eye(3)+ A * Ta;
```

```
10 B_d = Ta * B;
11 O_d = Ta * L;
```

Listing 4.25: Wyznaczenie macierzy dyskretnych równań stanu LESO

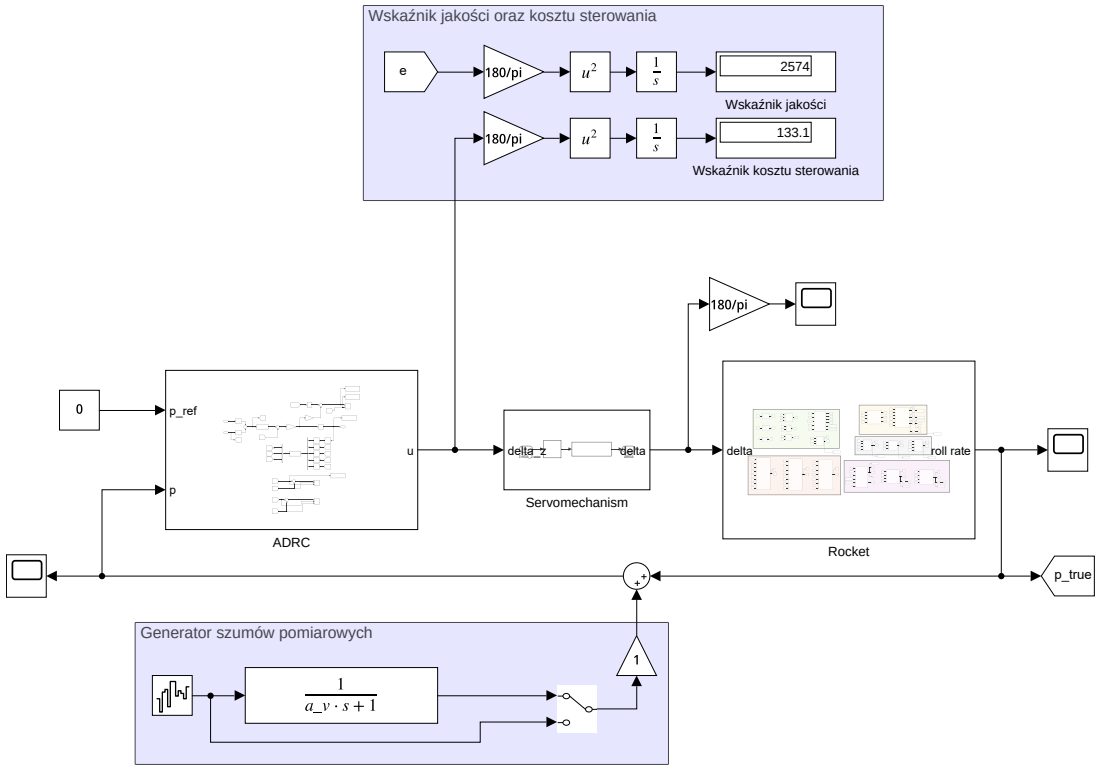
Zmienna `omega_O` $= 10\ \frac{\text{rad}}{\text{s}}$ zawiera wartość pasma przenoszenia obserwatora, a `Ta` $= 0{,}001$ s oznacza wartość okresu próbkowania obserwatora. Wyjściem obserwatora LESO są estymowane wartości prędkości rotacji *p_est*, przyspieszenia rotacji *dp_est* oraz całkowitego zaburzenia *f_est*.

W implementacji sterownika w programie MATLAB zostały także dodane bloki *Scope* umożliwiające obserwację porównania przebiegów estymowanego całkowitego zaburzenia z prawdziwym całkowitym zaburzeniem oraz porównania estymowanej wartości prędkości rotacji z wartością prawdziwą, oraz prędkością z dodanym szumem pomiarowym. Wartość prawdziwego całkowitego zaburzenia jest obliczana na podstawie równania (3.20).

Na rys. 4.36 przedstawiono układ sterowania po połączeniu wszystkich podsystemów. Poza podsystemami zawierającymi sterownik, model mechanizmu oraz model rakiety, została także zaimplementowana sekcja odpowiadająca za dodawanie szumu do wartości prędkości rotacji. Parametry szumu zostały dobrane w taki sposób, aby był on w podobnej skali co szum pomiarowy wykorzystywanego czujnika. Przyjęto, że jest to szum kolorowy powstający w wyniku przefiltrowania szumu białego o odchyleniu standardowym równym 0,01 oraz wartości średniej równej 0 za pomocą filtra o transmitancji równej

$$G_n(s) = \frac{1}{a_v\, s + 1},$$

gdzie $a_v = 0{,}002$.



Rysunek 4.36: Układ sterowania zaimplementowany w programie MATLAB

W modelu układu sterowania zamieszczono także bloki *Display* wyświetlające wartości wykorzystanych całkowych wskaźników jakości sterownia i kosztu sterowania. Wykorzystanym wskaźnikiem jakości

sterowania jest całka z kwadratu uchybu

$$J_e = \int_0^{T_n} e^2(t)\,dt, \tag{4.3}$$

a wskaźnikiem kosztu sterowania jest całka z kwadratu sygnału sterującego

$$J_u = \int_0^{T_n} u^2(t)\,dt, \tag{4.4}$$

gdzie T_n jest czasem trwania symulacji.

4.3.2 Dobór parametrów sterownika

Początkowo parametry sterownika pętli zewnętrznej zostały wyznaczone, korzystając z równań (3.31) i (3.32). Przyjęto, że pasmo przenoszenia obserwatora jest równe $\omega_0 = 15\,\frac{rad}{s}$. Wartość pasma przenoszenia sterownika została dobrana w taki sposób, aby jego wartość była mniejsza niż pasma przenoszenia obserwatora przyjmując wartość $\omega_c = 0.5 \cdot \omega_0 = 7.5\,\frac{rad}{s}$. Wartość współczynnika tłumienia wynosiła $\xi = 1$. Uzyskano w ten sposób wartości nastaw sterownika pętli zewnętrznej

$$\begin{aligned} k_0 &= 41,2398, \\ k_1 &= 11,8127. \end{aligned} \tag{4.5}$$

Tak wyznaczone nastawy spełniają ograniczenia podane w równaniach (3.33), (3.34) i (3.35).

Konieczne było także dobranie wartości wzmocnienia $\hat{b}$, które powinno być z zakresu $(0; 176112\rangle$. Wykorzystano wartość $\hat{b} = 5000$.

4.4 Wyniki działania układu sterowania w symulacji

Na rys. 4.37 przedstawiono prędkość rotacji rakiety z włączonym sterownikiem wykorzystującym wyznaczone wartości nastaw (4.5). Można zauważyć, że obiekt sterowania zachowuje stabilność kosztem znacznych oscylacji. Widać także bardzo długi czas ustalania odpowiedzi. Prędkość rotacji zawiera się w zakładanym tunelu $\pm 20\frac{^\circ}{s}$ dopiero po upływie czasu ok. 10,33 s. Całkowy wskaźnik jakości osiągnął wartość równą ok. $J_e = 7,753 \cdot 10^5$ a wskaźnik kosztu sterowania był równy ok. $J_u = 846,3$, dla horyzontu czasowego $T_n \approx 16{,}85$ s. Ze względu na duże oscylacje oraz długi czas ustalania taka jakość sterowania jest nieakceptowalna.

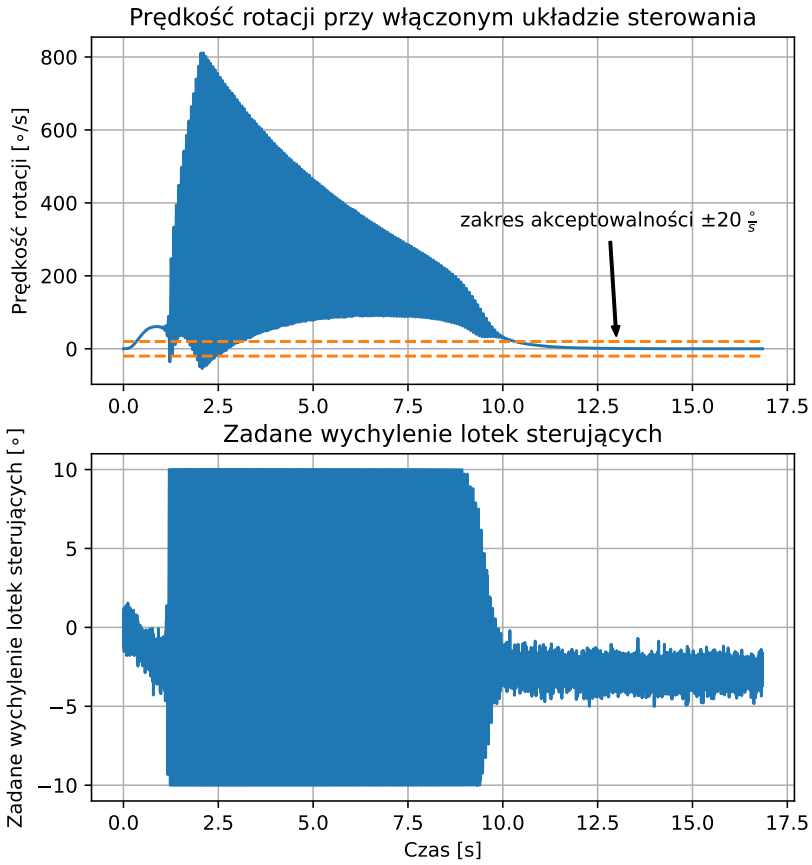
Na rys. 4.37 przedstawiono także wykres przebiegu sygnału sterującego. Na nim także widać bardzo duże oscylacje powodujące m.in. wzrost kosztu sterowania.

Porównano także przebiegi prędkości rotacji rakiety z przebiegiem prędkości rotacji estymowanej przez obserwator (rys. 4.38). Można zauważyć, że przez bardzo mocno ograniczone pasmo przenoszenia obserwatora, estymowana prędkość rotacji znacznie różni się od prawdziwej wartości w przedziale czasu, w którym występują duże oscylacje.

Wykreślono także przebiegi prawdziwego oraz estymowanego całkowitego zaburzenia. Wyniki przedstawiono na rys. 4.39. Można zauważyć, że w momencie występowania dużych oscylacji prędkości rotacji, wartość estymowanego całkowitego zaburzenia znacznie odbiega od jego prawdziwej wartości. Po ustaniu oscylacji, w końcowej fazie lotu, wartość estymowanego całkowitego zaburzenia zbiega w pobliże jego prawdziwej wartości tj. ok. 255.

Przy wykorzystaniu nastaw wyznaczonych za pomocą równań (3.31) i (3.32) pojawiają się znaczne oscylacje prędkości rotacji. Do tego czas ustalania znacznie przekracza zakładaną wartość $2\,s$. Obliczono znormalizowany czas opóźnienia układu wykonawczego, wykorzystując wzór [30]

$$\mu = \frac{T_o}{T_o + T}, \tag{4.6}$$



Rysunek 4.37: przebieg prędkości rotacji oraz zadanego wychylenie lotek sterujących podczas symulacji

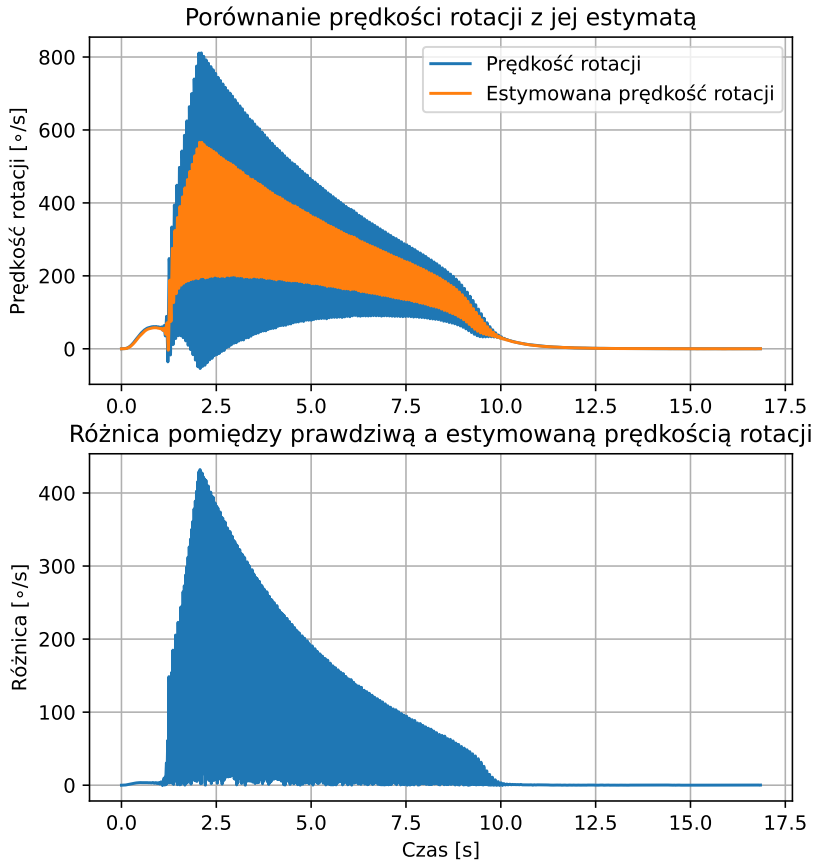
gdzie T_o jest opóźnieniem transportowym, a T jest stałą czasową inercji w modelu serwomechanizmu poruszającego lotkami sterującymi. Otrzymano $\mu \approx 0,56$. Oznacza to, że opóźnienie czasowe jest porównywalnej z T i tym samym obiekt jest trudny w sterowaniu.

Z tych powodów ponownie dobrano nastawy sterownika pętli zewnętrznej, wykorzystując metodę heurystyczną. Do tego celu ograniczono wartość współczynnika tłumienia do wartości $\xi = 0.08$. Nastawy sterownika zostały wyznaczone za pomocą równań (3.29) oraz (3.30) nieuwzględniających opóźnienia transportowego, otrzymując

$$\begin{aligned} k_0 &= 58,5225, \\ k_1 &= 1,224. \end{aligned} \tag{4.7}$$

Na wykresie widocznym na rys. 4.40 przedstawiono przebieg prędkości rotacji po zmianie wartości nastaw sterownika. Można zauważyć, że w tym przypadku nie pojawiają się już tak znaczne oscylacje, jak w przypadku symulacji wykorzystującej poprzednie nastawy sterownika. Na początku lotu rakiety widoczne jest przeregulowanie do wartości ok. $64,47\frac{^\circ}{s}$. Czas, po jakim prędkość rotacji osiąga zakładaną wartość $\pm 20\frac{^\circ}{s}$ wynosi ok. 1,39 s, a co za tym idzie, jest mniejszy niż zakładany czas 2 s. Po zmianie nastaw sterownika uzyskano wartość wskaźnika jakości na poziomie $J_e = 2574$ oraz wskaźnika kosztu sterowania na poziomie $J_u = 133,1$, dla horyzontu czasowego $T_n \approx 16{,}85$ s.

Na rys. 4.40 w przedstawiono także przebieg sygnału sterującego. Widać na nim, że zadany kąt wychylenia lotek sterujących oscyluje wokół wartości ok. $-2.9°$. Utrzymanie stosunkowo niskiej wartości sygnału



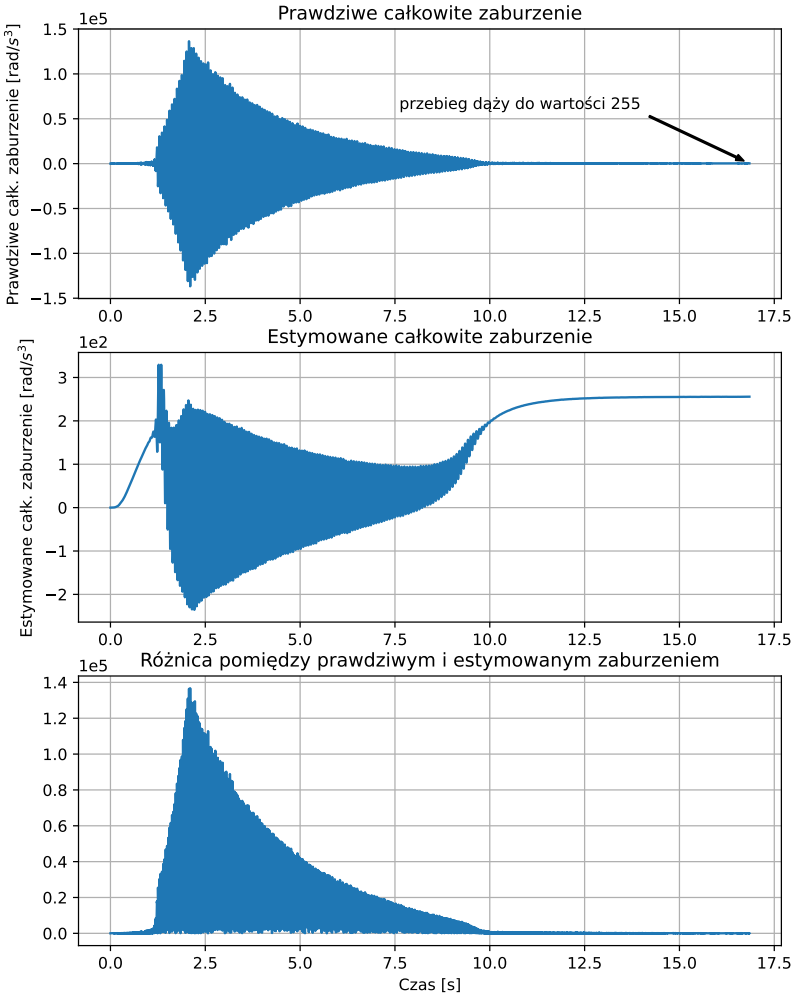
Rysunek 4.38: Porównanie prawdziwej oraz estymowanej prędkości rotacji

sterującego bez oscylacji o znacznej amplitudzie pozwala na znaczne zmniejszenie kosztu sterowania.

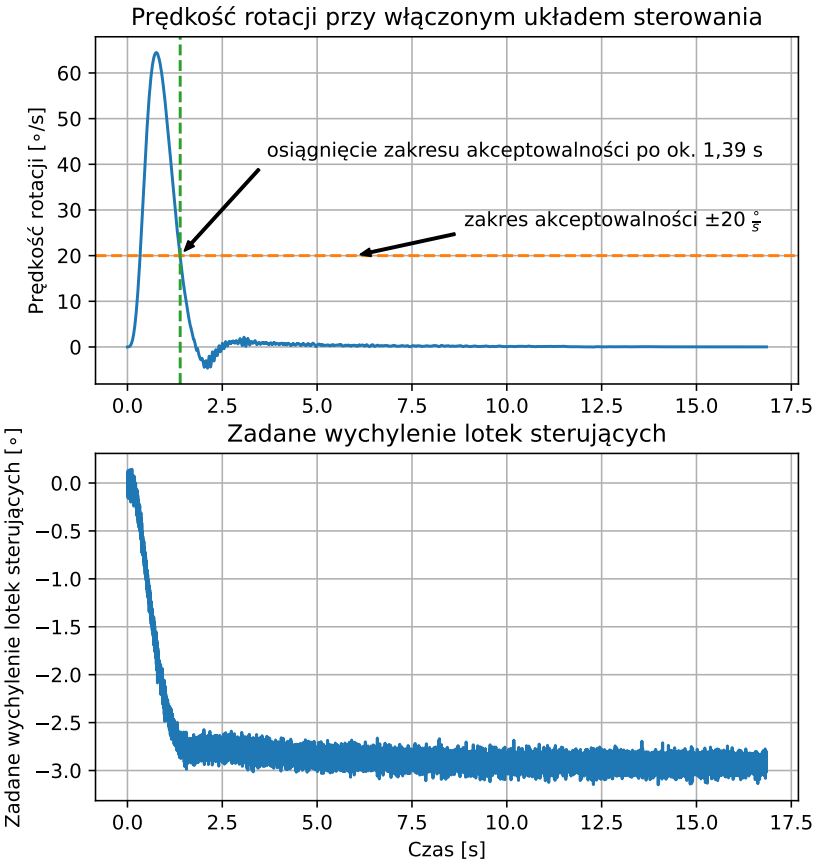
Porównano także przebiegi prawdziwej oraz estymowanej wartości prędkości rotacji rakiety (rys. 4.41). Można zauważyć, że po dobraniu nowego zestawu nastaw (4.7), poprzez wyeliminowanie znacznych oscylacji, różnica pomiędzy przebiegiem prawdziwej oraz estymowanej prędkości rotacji utrzymuje się w zakresie nieprzekraczającym ok. $4,47\frac{^\circ}{s}$.

Na rys. 4.42 porównano prawdziwe oraz estymowane całkowite zaburzenie. Estymowane całkowite zaburzenie podąża przebiegiem średniej kroczącej prawdziwego całkowitego zaburzenia. Widać jednak bardzo duże zaszumienie sygnału prawdziwego zaburzenia wynikające z braku dostępu do prawdziwej wartości zrywu rotacji.

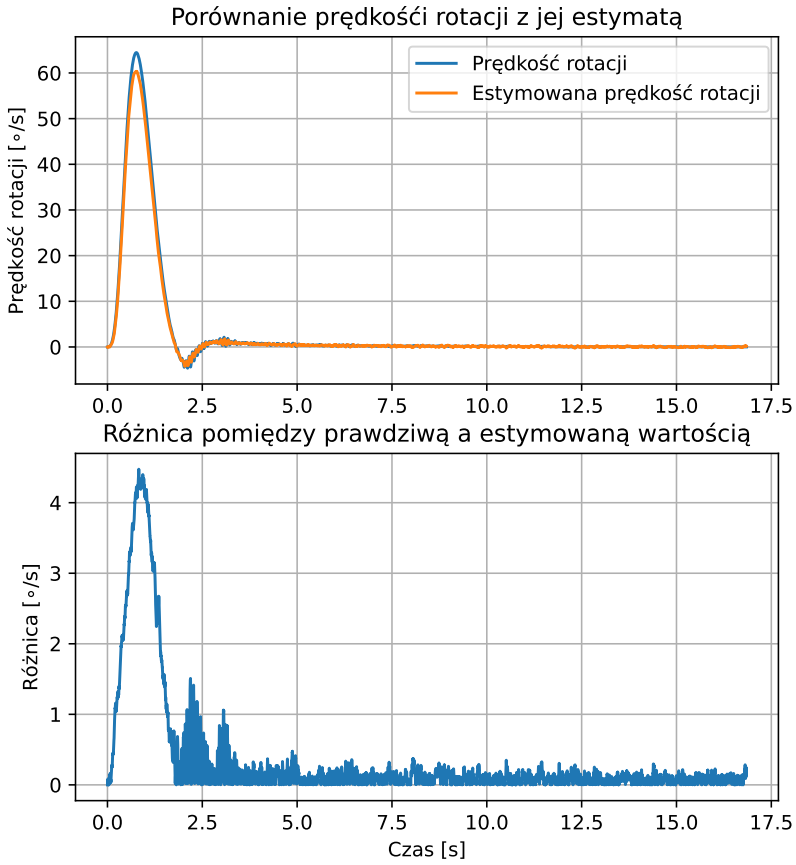
Po włączeniu sterownika, mimo redukcji prędkości rotacji, lot rakiety odbywa się w sposób stabilny. Można to zauważyć na rys. 4.43 przedstawiającym trajektorię lotu rakiety. Widać na nim, że rakieta przez cały czas lotu porusza się prawie pionowo do góry (przesunięcie w osiach x i y układu globalnego są nie większe niż $1\,m$). Na rys. 4.44 oraz 4.45 przedstawiono prędkości kątowe rakiety wokół osi poprzecznej i pionowej. Ich przebiegi pokazują, że rakieta delikatnie drga w tych osiach przez większość lotu, a zauważalny wzrost prędkości kątowych następuje dopiero w pobliżu apogeum wysokości.



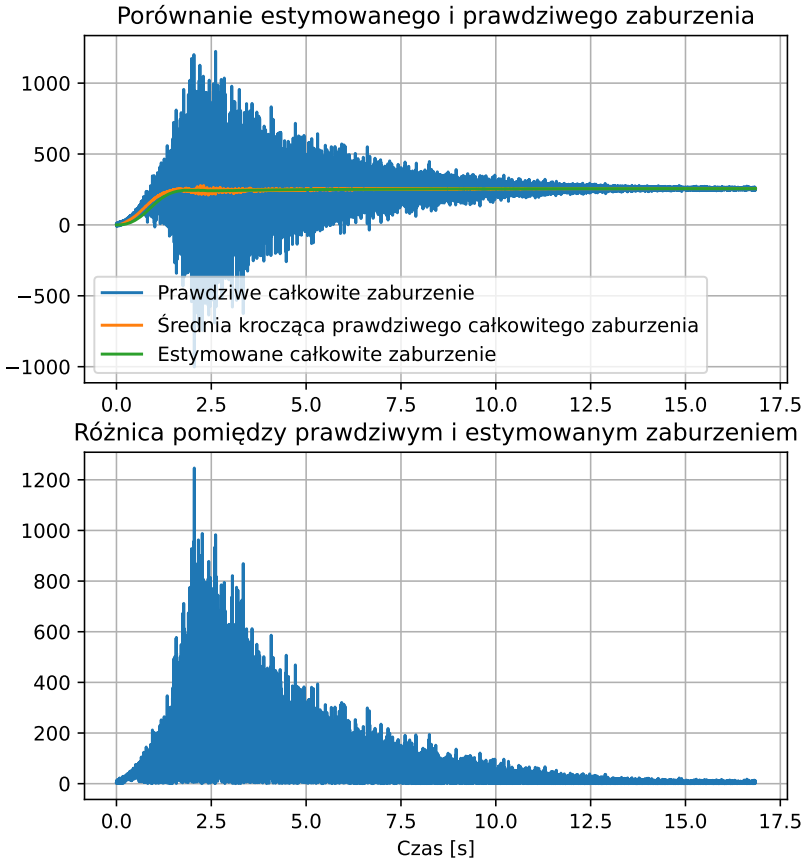
Rysunek 4.39: Porównanie prawdziwego oraz estymowanego przebiegu całkowitego zaburzenia



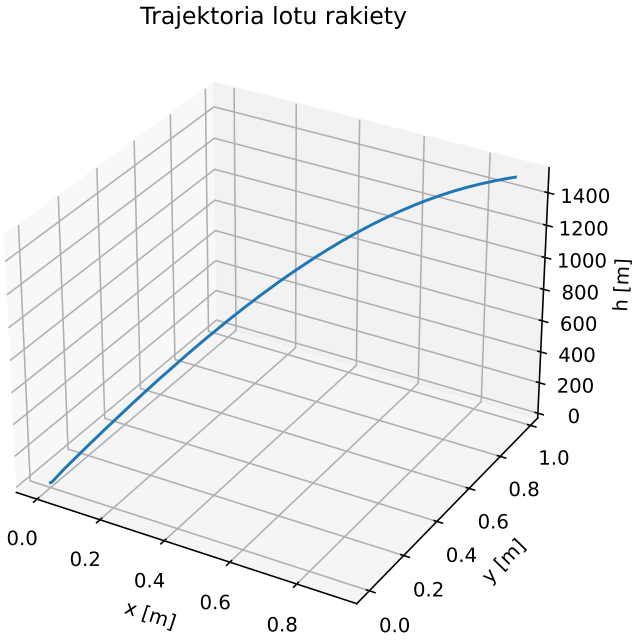
Rysunek 4.40: Przebieg prędkości rotacji oraz zadane wychylenie lotek sterujących podczas symulacji (dla nastaw (4.7))



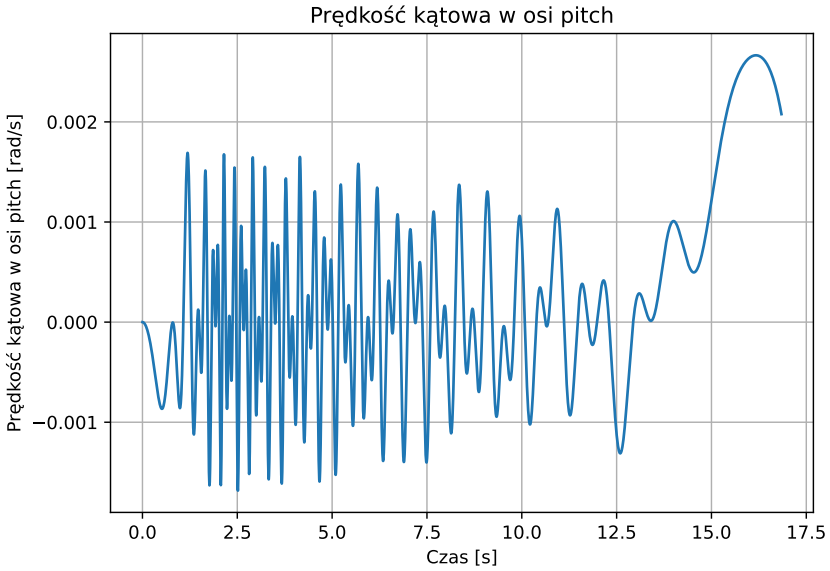
Rysunek 4.41: Porównanie prawdziwej oraz estymowanej wartości prędkości rotacji (dla nastaw (4.7))



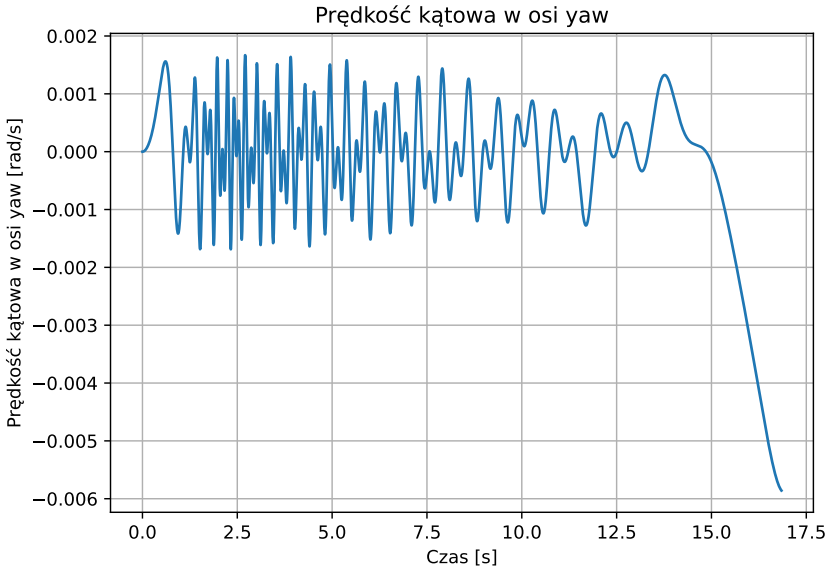
Rysunek 4.42: Porównanie prawdziwej oraz estymowanej wartości całkowitego zaburzenia (dla nastaw (4.7))



Rysunek 4.43: Przebieg trajektorii rakiety w układzie ustalonym



Rysunek 4.44: Przebieg prędkości kątowej rakiety w osi poprzecznej



Rysunek 4.45: Przebieg prędkości kątowej rakiety w osi pionowej

Rozdział 5

Implementacja układu sterowania na pokładzie rakiety sportowej

5.1 Projekt płytki drukowanej komputera pokładowego

W celu weryfikacji działania rzeczywistego układu sterowania zaprojektowano oraz wykonano komputer pokładowy umożliwiający pomiar prędkości rotacji rakiety wokół osi podłużnej. Wymagane były także wyjścia sterujące serwomechanizmami modelarskimi, poruszającymi lotkami sterującymi oraz możliwość zapisu danych podczas działania układu sterowania.

Projekt płytki drukowanej (PCB, ang. *printed circuit board*) został wykonany przy użyciu oprogramowania o otwartych źródłach (ang. *opensource*) KiCAD [6]. Pozwala ono na rysowanie ideowych schematów elektronicznych, jak i projektu rozmieszczenia elementów wraz z łączącymi je ścieżkami.

5.1.1 Dobór elementów elektronicznych komputera pokładowego

Jako główną jednostkę obliczeniową wybrano mikrokontroler STM32F412RET6 [39]. Jest to 32-bitowy mikrokontroler wyposażony w FPU (ang. *Floating-Point Unit*), czyli jednostkę wspomagającą mikrokontroler w obliczeniach wykonywanych na liczbach zmiennoprzecinkowych. Do wykorzystanego mikrokontrolera dobrano rezonator kwarcowy o częstotliwości 25 MHz, który jest źródłem bazowej częstotliwości mikrokontrolera. Dzięki zawartej w mikrokontrolerze pętli PLL (ang. *Phase Locked Loop*) możliwe jest zwiększenie częstotliwości taktowania rdzenia do wartości 96 MHz. Wybrany mikrokontroler współpracuje z wieloma interfejsami komunikacyjnymi takimi jak UART, I^2C, SPI, CAN czy SDIO. Pozwala także na generowanie sygnałów PWM (ang. *Pulse-Width Modulation*).

Jako komponent służący do pomiaru prędkości rotacji rakiety został wybrany tzw. moduł IMU (ang. *Inertial Measurement Unit*) typu BMI088 [15]. Jest to układ zawierający w sobie żyroskop oraz akcelerometr. Moduł ten pozwala na komunikację z wykorzystaniem magistrali I^2C oraz SPI, za pomocą których można pobierać wartości wykonanych pomiarów. Pomiar i akwizycja prędkości rotacji, jak i przyspieszeń liniowych wykonywany jest wewnątrz układu za pomocą 16-bitowych przetworników ADC, co pozwala na uzyskanie przy zakresie pomiaru do $2000\,\frac{^\circ}{s}$ rozdzielczości na poziomie ok. $0,061\,\frac{^\circ}{s}$. Próbkowanie może odbywać się z maksymalną szybkością 2000 próbek na sekundę.

Zaprojektowany komputer pokładowy wyposażony jest także w gniazdo na kartę microSD oraz dodatkowy moduł pamięci flash (ang. *flash memory*), które służą do zapisu danych podczas działania układu sterowania. Jako główny sposób zapisu danych wybrano wykorzystanie karty microSD ze względu na możliwość bezpośredniego przenoszenia zebranych danych na komputer. Komunikacja z kartą microSD odbywa się za pomocą interfejsu SDIO. Dodatkowa pamięć flash jest zapasowym podzespołem, do którego mogą zostać zapisane dane np. w razie awarii karty microSD. Wybranym modułem pamięci flash

jest W25Q64 [45]. Jest to pamięć flash typu NOR o pojemności 64 Mb. Komunikacja z pamięcią odbywa się za pomocą magistrali SPI od prędkości maksymalnej do 100 MHz.

Do zasilania komputera pokładowego została wykorzystana bateria 2S złożona z 2 ogniw Li-Ion. Napięcie wyjściowe takiego pakietu ogniw to maksymalnie 8,4 V co pozwala na bezpośrednie zasilanie serwomechanizmów modelarskich wykorzystanych do wychylania lotek sterujących. Wykorzystany mikrokontroler, jak i IMU oraz pamięci wykorzystywane do zapisu danych zasilane są z napięcia stałego o wartości 3,3 V. Z tego powodu potrzebne było dobranie stabilizatora napięcia, który pozwoli zamienić napięcie baterii na napięcie robocze podzespołów. Wybrano stabilizator TLV76633QWDRBRQ1 [40], który pozwala na zamianę napięć z przedziału 2,5 V — 16 V na napięcie 3,3 V. Stabilizator ten cechuje się tolerancją napięcia wyjściowego na poziomie $\pm 0,5\%$.

W celu ułatwienia obsługi komputera pokładowego dodano możliwość ładowania baterii za pomocą ładowarki o napięciu wyjściowym $5\,V$ poprzez złącze USB micro typu B. Układ scalony zarządzający procesem ładowania baterii, jaki został wybrany podczas projektowania płytki PCB to MP2639AGR-Z [33]. Pozwala on na ładowanie pakietów ogniw 2S łącznie z balansowaniem serii ogniw (ang. *cell balancing*) wchodzących w skład baterii tj. wyrównywaniem napięć na składowych ogniwach. Jest to konieczne, ponieważ ze względu na różnice we właściwościach użytych ogniw mogą one ładować się w różnym tempie. Mogłoby to doprowadzić do nadmiernego naładowania jednego z ogniw w momencie, kiedy drugie z nich nie byłoby jeszcze w pełni naładowane, co może prowadzić do szybszej degradacji baterii. Wykorzystany układ posiada także zabezpieczenia nadprądowe, jak i możliwość podłączenia diod LED pełniących funkcję wskaźnika naładowania baterii.

5.1.2 Projekt PCB

W pierwszym etapie projektowania płytki drukowanej komputera pokładowego przygotowano schemat elektroniczny przedstawiający sposób połączenia wszystkich użytych elementów.

Na rys. 5.1 znajduje się fragment schematu przedstawiający sposób podłączenia użytego mikrokontrolera. Zasilany jest on napięciem 3,3 V z wykorzystaniem kondensatorów filtrujących *C19* i *C22-C26*. Dodatkowo zasilanie sekcji analogowej mikrokontrolera zostało podłączone poprzez koralik ferrytowy *FB1* oraz kondensatory filtrujące *C7* i *C8*. Ma to przeciwdziałać zakłóceniom na linii zasilającej ze strony innych układów, które mogłyby zaburzyć poprawną pracę mikrokontrolera.

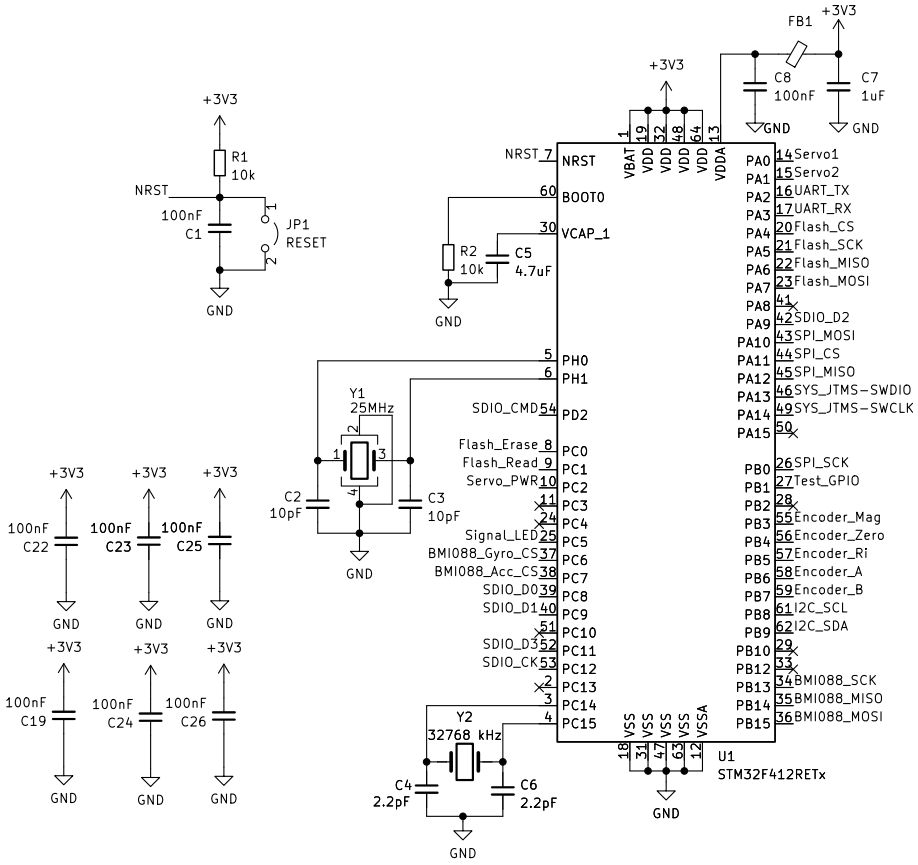
Podłączone zostały także rezonatory kwarcowe *Y1* oraz *Y2* razem z kondensatorami obciążającymi, które są wykorzystywane przez mikrokontroler jako źródła podstawowych częstotliwości pracy procesora oraz zegara czasu rzeczywistego RTC (ang. *real-time clock*).

Na rys. 5.1 widoczna jest także zworka *JP1*, która służy do resetowania układu. Układ resetowany jest po podaniu stanu niskiego na wejście *NRST*, dlatego do zworki resetującej został dołączony rezystor podciągający *R1* wymuszający stan wysoki jako domyślny. Poza tym dołączony został kondensator *C1* służący do filtracji zakłóceń podczas przełączania stanu wejścia *NRST* co pozwala na uniknięcie wielokrotnego resetowania mikrokontrolera.

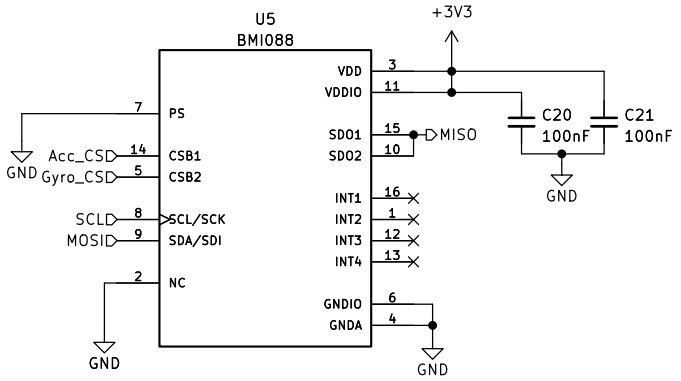
Na rys. 5.2 przedstawiono fragment schematu, pokazujący sposób podłączeniu układu BMI088. Układ zasilany jest napięciem 3,3 V poprzez kondensatory filtrujące *C20* oraz *C21*. Został on podłączony do mikrokontrolera w taki sposób, aby możliwa była komunikacja z modułem żyroskopu oraz akcelerometru za pomocą interfejsu SPI (wyjścia/wejścia *MISO*, *MOSI*, *SCL*, *Gyro_CS* i *Acc_CS*).

Na rys. 5.3 widoczny jest sposób podłączenia gniazda na kartę microSD. Karta pamięci zasilana jest napięciem 3,3 V poprzez kondensator filtrujący *C9*. Można także zauważyć wymagane przez interfejs SDIO rezystory podciągające *R3* - *R8* podłączone do linii komunikacyjnych.

Sposób podłączenia pamięci flash został przedstawiony na rys. 5.4. Na linii zasilającej pamięć z napięcia 3,3 V znajduje się kondensator *C10* filtrujący zakłócenia. Pamięć wykorzystuje interfejs SPI do komu-



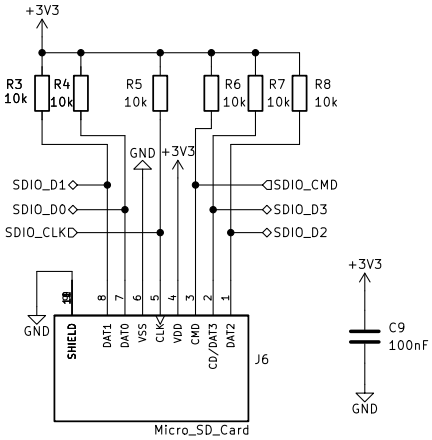
Rysunek 5.1: Schemat elektroniczny podłączenia mikrokontrolera



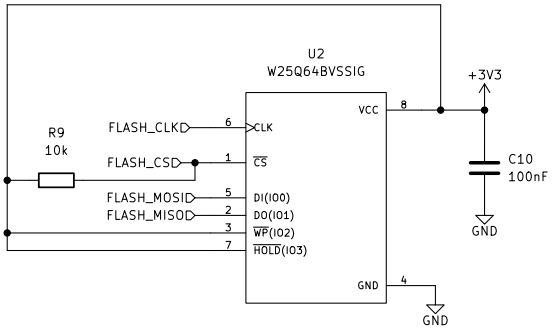
Rysunek 5.2: Schemat elektroniczny podłączenia układu BMI088

nikacji z mikrokontrolerem (wejścia/wyjścia *FLASH_MISO*, *FLASH_MOSI*, *FLASH_CLK*, *FLASH_CS*). Dodatkowo do wejścia *FLASH_CS* podpięty został rezystor podciągający *R9* wymuszający domyślny stan wysoki co w przypadku interfejsu SPI oznacza brak zezwolenia na komunikację. Na wejścia $\overline{WP}$ oraz $\overline{HOLD}$ został podany stan wysoki, aby wyłączyć opcję ochrony przed zapisem oraz trybu ignorowania komunikatów przychodzących po interfejsie SPI.

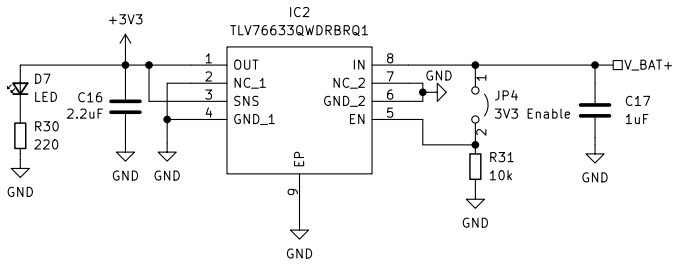
Sposób, w jaki został podłączony stabilizator napięcia 3,3 V, został pokazany na rys. 5.5. Układ zasilany jest bezpośrednio z baterii, a na jego wyjściu generowane jest napięcie 3,3 V. Na wyjściu, jak i wejściu układu znajdują się kondensatory filtrujące zakłócenia na liniach zasilających. Dodatkowo do wyjścia stabilizatora została dołączona dioda LED *D7* sygnalizująca włączenie zasilania wraz z rezystorem *R30* ograniczającym przepływ prądu. Zworka *JP4* wraz z rezystorem *R31* pozwala na włączanie oraz wyłą-



Rysunek 5.3: Schemat elektroniczny podłączenia gniazda na kartę microSD



Rysunek 5.4: Schemat elektroniczny podłączenia pamięci flash

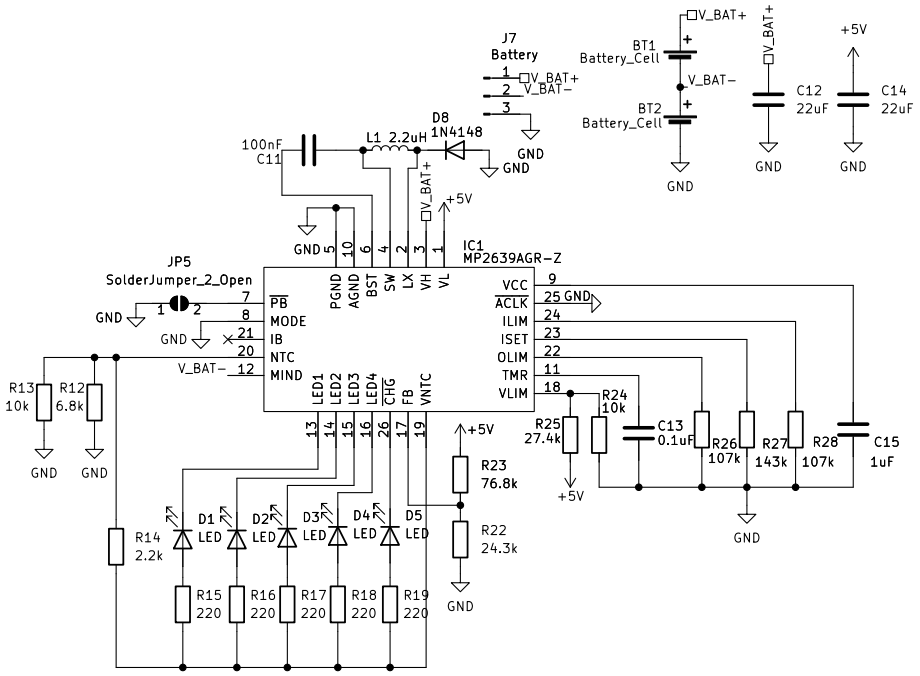


Rysunek 5.5: Schemat elektroniczny podłączenia stabilizatora napięcia

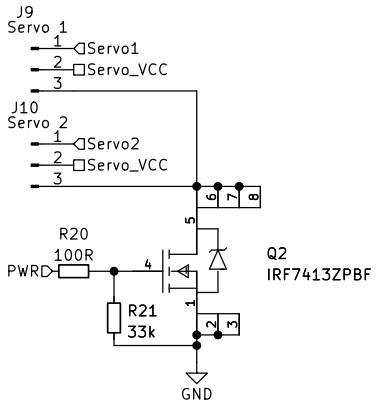
czanie stabilizatora poprzez zmianę stanu na wejściu *EN*.

Schemat podłączenia układu zarządzającego ładowanie baterii przedstawiono na rys. 5.6. Do wejścia układu podłączone jest napięcie 5 V pochodzące ze złącza USB micro B, a do wyjścia podłączona jest bateria zasilająca komputer pokładowy. Na wejściu, jak i wyjściu układu znajdują się kondensatory *C12* i *C14* filtrujące zakłócenia na liniach zasilających. Widoczne są także diody LED *D1* - *D5*, które sygnalizują tryb działania układu, jak i stan naładowania baterii. Wartości pozostałych elementów pasywnych są, według dokumentacji użytego układu [33], wymagane do poprawnego działania. Na schemacie na rys. 5.6 widoczny jest także sposób podłączenia ogniw *BT1*, *BT2* baterii zasilającej komputer.

Rys. 5.7 przedstawia sposób podłączenia wyjść sterujących serwomechanizmami modelarskimi. Serwomechanizmy zasilane są bezpośrednio z baterii zasilającej komputer pokładowy poprzez tranzystor *Q2* MOSFET z kanałem N, który pełni rolę klucza załączającego zasilanie. Rezystor *R20* pełni rolę ogranicznika prądu bramki tranzystora, a rezystor *R21* służy do podciągnięcia bramki tranzystora do masy w celu uniknięcia stanów nieustalonych.



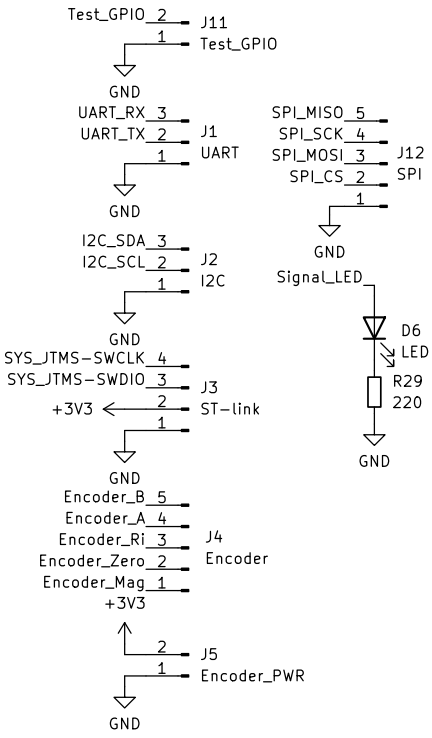
Rysunek 5.6: Schemat elektroniczny podłączenia układu ładującego baterię



Rysunek 5.7: Schemat elektroniczny wyprowadzeń sterujących serwomechanizmów

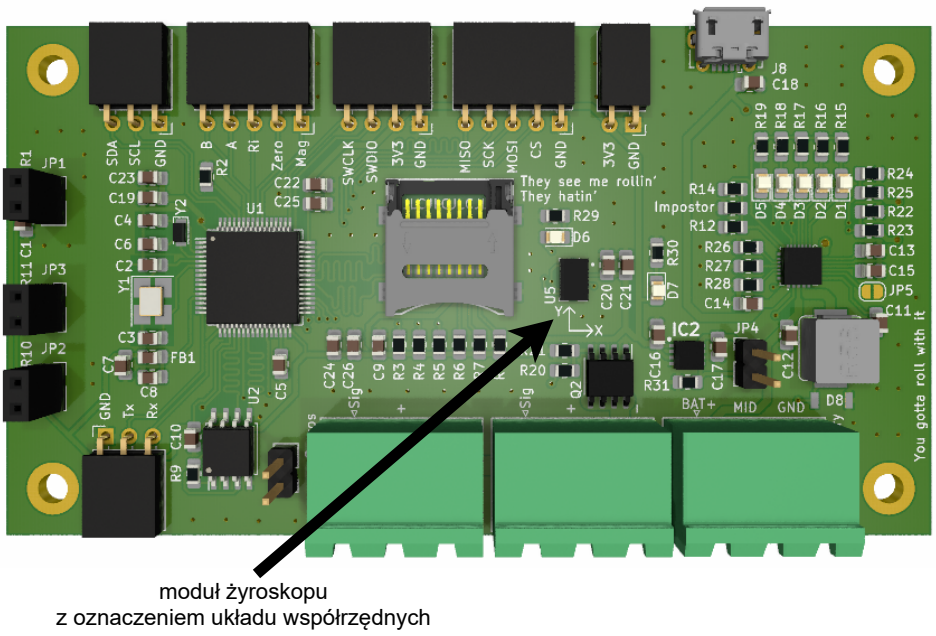
Na rys. 5.8 przedstawiono schemat wyprowadzeń komputera pokładowego. Został on wyposażony w złącza *J4* i *J5*, służące do podłączenia dodatkowego modułu enkodera magnetycznego. Złącze *J3* służy do podłączenia programatora mikrokontrolera. Złącza *J1*, *J2* i *J12* służą jako dodatkowe wyprowadzenia interfejsów komunikacyjnych UART, I^2C oraz SPI ułatwiające rozbudowę funkcjonalności komputera. Komputer został także wyposażony w złącze *J11*, które zawiera jedno z wyjść cyfrowych mikrokontrolera. Może ono służyć do celów weryfikacji działania komputera. Dioda LED *D6* została dodana jako dioda sygnalizująca uruchomienie oprogramowania komputera pokładowego. Dzięki naprzemiennemu przełączaniu zasilania diody za pomocą wyjścia cyfrowego możliwa jest sygnalizacja zawieszenia się systemu operacyjnego komputera (tzw. *heartbeat*).

Rysunek 5.9 przedstawia wizualizację płytki drukowanej komputera pokładowego, na której widać rozmieszczenie wszystkich zastosowanych elementów. Widać na nim także ścieżki obwodu drukowanego łączące podzespoły komputera. Szerokość ścieżek została dobrana w zależności od obciążenia prądem. Dodatkowo ścieżki linii komunikacyjnych pamięci flash oraz kart microSD zostały wyrównane pod względem długości, aby zapewnić zbliżony czas propagacji sygnałów. Taki zabieg jest wymagany w przypadku



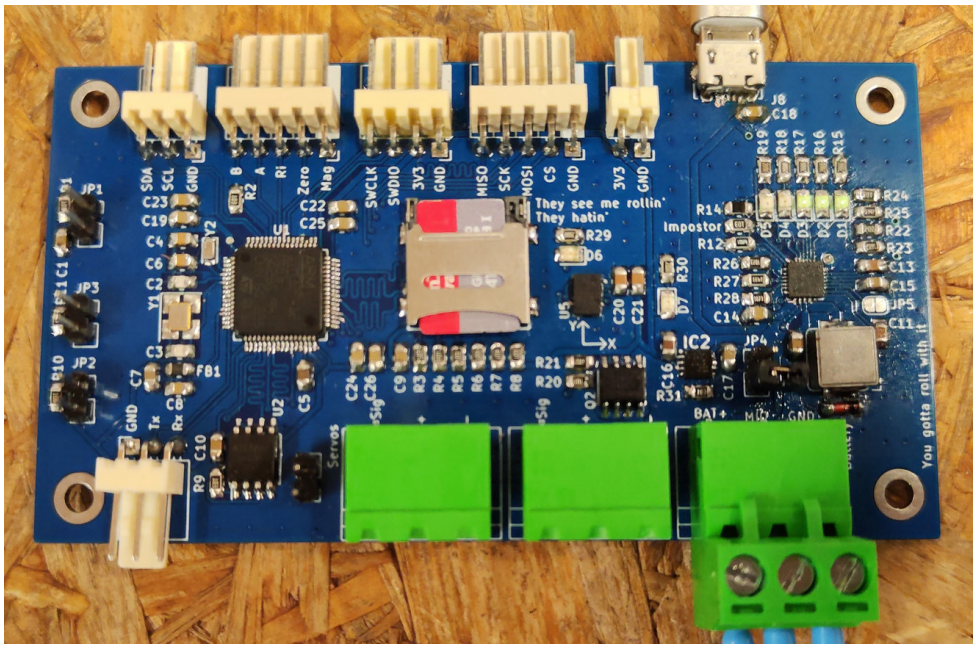
Rysunek 5.8: Schemat elektroniczny wyprowadzeń komputera pokładowego

karty microSD do jej poprawnego i powtarzalnego funkcjonowania. W przypadku pamięci flash pozwala to na bezproblemowe przeprowadzenie komunikacji z wykorzystaniem interfejsu SPI przy szybkości transmisji rzędu 100 MHz. Układ IMU BMI088 został umiejscowiony w równych odległościach od górnej i dolnej krawędzi płytki, a na samej płytce drukowanej został oznaczony układ współrzędnych żyroskopu.



Rysunek 5.9: Wizualizacja płytki drukowanej komputera pokładowego

Wykonany komputer pokładowy został przedstawiony na rys. 5.10. Poza komputerem widoczna jest także bateria zasilająca komputer podczas ładowania za pomocą ładowarki 5 V USB.



Rysunek 5.10: Zdjęcie wykonanego komputera pokładowego

5.2 Oprogramowanie komputera pokładowego

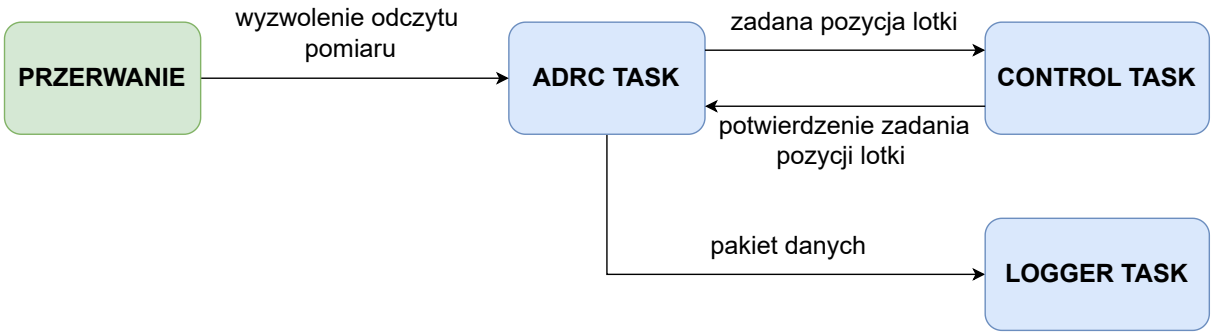
W związku z tym, że wybraną jednostką obliczeniową jest mikrokontroler z rodziny STM32, zdecydowano się wykorzystać dedykowane środowisko programistyczne STM32CubeIDE [38]. Pozwala ono na szczegółowe zdefiniowanie pracy peryferiów oraz na zaawansowaną konfigurację. Wspomniane środowisko wspiera również implementację systemu czasu rzeczywistego. Kod został napisany za pośrednictwem oprogramowania umożliwiającego obsługę warstwy abstrakcji sprzętowej HAL ang. *hardware abstraction layer*. Wykorzystanymi językami programowania są C oraz C++.

Oprogramowanie komputera pokładowego wymagało napisania własnych interfejsów oraz bibliotek. Interfejs obsługujący zbieranie danych z żyroskopu został napisany na podstawie bibliotek udostępnionych przez producenta [14]. Biblioteka do obsługi serwomechanizmów została wykonana z wykorzystaniem interfejsu HAL. Zapisywanie danych na karcie SD realizowane jest za pomocą systemu plików FatFS [4], który również jest wspierany przez środowisko STM32CubeIDE. Dodatkowo układ pamięci flash obsługiwany jest poprzez zrewidowaną bibliotekę wykorzystywaną w ramach działalności koła naukowego PUT Rocketlab.

Kod został napisany z wykorzystaniem interfejsu CMSIS-RTOS API v2 [2] opartego na systemie operacyjnym czasu rzeczywistego FreeRTOS [5]. Wykorzystanie tego podejścia programistycznego umożliwiło efektywne wykonywanie zadań przez mikrokontroler.

5.2.1 Opis architektury oprogramowania

Z racji tego, że wykorzystany został system czasu rzeczywistego, zadania zostały rozdzielone na wątki. Na rys. 5.11 przedstawiona jest poglądowa architektura oprogramowania, gdzie kolorem niebieskim oznaczono wątki, a kolorem zielonym sprzętową obsługę przerwań.



Rysunek 5.11: Poglądowy schemat architektury oprogramowania komputera pokładowego

Wątek opisany jako *ADRC task*, początkowo oczekuje na zwolnienie semafora poprzez funkcję `OsSemaphoreAcquire`. Widoczne jest to w pierwszej linii na listingu 5.1 fragmentu kodu pochodzącego z nieskończonej pętli wątku.

```
1  osSemaphoreAcquire(SensorSampleSemHandle,osWaitForever);
2  auto gyroData = bmi088.readGyroData();
3
4  taskENTER_CRITICAL();
5  auto test = gyroData.x * (float)M_PI / 180.f;
6  adrc.calculateLESO(gyroData.x * (float)M_PI / 180.f);
7  taskEXIT_CRITICAL();
8
9  if (osSemaphoreAcquire(ControlCheckSemHandle,0) == osOK) {
10     taskENTER_CRITICAL();
11     position = adrc.calculatePosition(0, gyroData.x * (float)M_PI / 180.f);
12     xTaskNotify((TaskHandle_t)ControlTaskHandle, *reinterpret_cast<uint32_t*>(&position),
           ↪ eSetBits);
13     taskEXIT_CRITICAL();
14 }
```

Listing 5.1: Fragment kodu z wątku ADRC task

Semafor zostaje zwolniony przez przerwanie pochodzące od sprzętowego licznika (ang. *timer*) tak jak jest to wskazane na rys. 5.11. Licznik ustawiony na odmierzanie 1 ms, po upływie tego czasu zwalnia semafor. Dzięki temu kod jest wykonywany dalej, a następnie funkcja `readGyroData()` z listingu 5.1 dokonuje pomiaru aktualnej prędkości rotacji. Następnie obliczenia pętli adaptacji zostają zamknięte w sekcji krytycznej, która zapewnia obliczenie estymat bez zmiany kontekstu. Kolejnym krokiem jest obliczenie zadanej pozycji lotki oraz wysłanie jej za pomocą struktury *task notification* (co widoczne jest w liniach 10-13 na listingu 5.1) do wątku sterującego układem wykonawczym. Wspomniany fragment kodu zostaje zamknięty również w sekcji krytycznej, co zapewnia nieprzerwanie wykonywanej operacji przez planistę (ang. *scheduler*). *Task notification* jest to struktura pozwalająca na wysłanie informacji bezpośrednio do wskazanego wątku, co zapewnia szybsze przekazanie wiadomości. Następne obliczenie oraz wysłanie kolejnej pozycji lotki uzależnione jest od potwierdzenia zadania pozycji przez wątek *Control task*. Takie zabezpieczenie pozwala na zadanie pozycji mimo ewentualnych opóźnień. Jednocześnie pozwala to obserwatorowi LESO nieprzerwanie wyliczać estymaty. Przy dynamice charakterystycznej dla wytwarzanych rakiet badawczych, LESO dalej dostaje informacje i pozwala przy opóźnieniu wykonywania zadań przez wątek *Control task* na obliczanie aktualnych estymat, co bezpośrednio skutkuje obliczeniem odpowied-

niej pozycji lotki. Do innych zadań wątku *ADRC task* należy wysłanie pakietu danych w zdefiniowanej strukturze za pośrednictwem kolejki do wątku *Logger task*.

Control task jest wątkiem, który obsługuje sterowanie serwomechanizmów będących częścią układu wykonawczego. Początkowo wspomniany wątek oczekuje na pojawienie się informacji, następnie obliczona pozycja podawana jest po przeliczeniu przez funkcję przełożenia zadanej pozycji lotki sterującej na położenie kątowe serwomechanizmu na układ wykonawczy. Na końcu ustawiane jest potwierdzenie zadania pozycji poprzez zwolnienie semafora. Sterowanie serwomechanizmami jest realizowane przez jeden licznik, co zapewnia synchronizację.

Do zadań wątku *Logger task* należy odbieranie pakietu danych z kolejki wysyłanej z wątku *ADRC task* oraz zapis na kartę SD.

5.2.2 Implementacja sterownika ADRC w oprogramowaniu komputera

Sterownik ADRC został rozdzielony na obliczenia dotyczące obserwatora oraz zadanej pozycji lotek sterujących. Obserwator LESO został zaimplementowany za pomocą wzoru (3.12). Fragment kodu przedstawiający wspomnianą implementacje znajduje się na listingu 5.2.

```
1  Vector LESO::estimate(float u, float rollRateSensor) {
2      x_est_[0] = p_est_;
3      x_est_[1] = dp_est_;
4      x_est_[2] = f_est_;
5
6      for (uint8_t i = uBuffer_.size() - 1; i > 0; i--) {
7          uBuffer_[i] = uBuffer_[i - 1];
8      }
9
10     uBuffer_[0] = u;
11
12     // estymacja predkosci roll
13     xp_est_[0] = (Ad_[0][0] * x_est_[0] + Ad_[0][1] * x_est_[1] + Ad_[0][2] * x_est_[2]) +
           ↪ Bd_[0] * uBuffer_[24] + Od_[0] * (rollRateSensor - p_est_);
14     // estymacja przyspieszenia roll
15     xp_est_[1] = (Ad_[1][0] * x_est_[0] + Ad_[1][1] * x_est_[1] + Ad_[1][2] * x_est_[2]) +
           ↪ Bd_[1] * uBuffer_[24] + Od_[1] * (rollRateSensor - p_est_);
16     // estymacja calkowitego zaburzenia
17     xp_est_[2] = (Ad_[2][0] * x_est_[0] + Ad_[2][1] * x_est_[1] + Ad_[2][2] * x_est_[2]) +
           ↪ Bd_[2] * uBuffer_[24] + Od_[2] * (rollRateSensor - p_est_);
18
19     p_est_ = xp_est_[0];
20     dp_est_ = xp_est_[1];
21     f_est_ = xp_est_[2];
22
23     return xp_est_;
24 }
```

Listing 5.2: Fragment kodu z biblioteki LESO

Początkowo do wektora zmiennych stanu przypisywane są wartości z poprzedniej chwili czasowej. Następnie dodawane jest opóźnienie sygnału sterującego. Po tym obliczane są kolejne estymaty na podstawie wcześniej zdefiniowanych macierzy oraz aktualnych danych. Na końcu zapisywane są wartości z aktualnej chwili czasowej do tymczasowej pamięci mikrokontrolera.

Zadana pozycja lotek sterujących obliczana jest w bibliotece ADRC. Fragment kodu przedstawiający wspomnianą implementacje widoczny jest na listingu 5.3.

```
1  float ADRC::calculatePosition(float targetPostion, float rollRateSensor){
2      float error, u_kpd, u;
3
4      //uchyb
5      error = targetPostion - rollRateSensor;
6      // sterownik proporcjonalno - roniczkujcy
7      u_kpd = kp_ * error + kd_ * (error - prev_error_) / Tc_;
8      prev_error_ = error;
9      // sygnal sterujacy
10     u = (u_kpd - f_est_) / b_est_;
11
12     // saturacja
13     if (u < -saturation_) {
14         u = -saturation_;
15     } else if (u > saturation_) {
16         u = saturation_;
17     }
18
19     u_ = u; // przypisanie wartosci sygnau sterujcego dla LESO
20     return u;
21 }
```

Listing 5.3: Fragment kodu z biblioteki ADRC

Początkowo obliczany jest uchyb, który następnie zostaje wykorzystany przez sterownik PD. Następnie 10. linia kodu z listingu 5.3 przedstawia obliczenie sygnału sterującego na podstawie wzoru (3.5). Ostatecznie sygnał sterujący sprawdzany jest poprzez warunki ograniczenia.

5.2.3 Sposób zapisu danych pomiarowych

Dane z wątku *ADRC task* wysyłane są za pośrednictwem kolejki struktur do wątku *Logger task* (rys. 5.11). Zdefiniowana struktura pakietu danych przedstawiona jest na listingu 5.4.

```
1  struct LoggerDataADRC{
2      uint32_t timeStamp{};
3      float setPosition{};
4      float rollRate{};
5      float p_est{};
6      float dp_est{};
7      float f_est{};
8  };
```

Listing 5.4: Fragment kodu przedstawiający strukture danych

Pakiet danych zawiera informacje o aktualnej próbce czasowej [ms], zadanej pozycji lotek sterujących [rad], prędkości rotacji z żyroskopu $[\frac{deg}{s}]$, estymowanej prędkości rotacji $[\frac{rad}{s}]$, estymowanym przyspieszeniu $[\frac{rad}{s^2}]$ oraz całkowitym zaburzeniu $[\frac{rad}{s^3}]$.

Zapis danych na kartę SD odbywa się co 100 struktur zdefiniowanych na listingu 5.4. Taka praktyka pozwala na oszczędzeniu czasu na zamykaniu i otwieraniu pliku. Dane zapisywane są na kartę SD poprzez

4-bitowy interfejs SDIO z taktowaniem 48 MHz. Taki dobór parametrów komunikacji pozwala na zapis wszystkich danych z taktowaniem 1000 Hz.

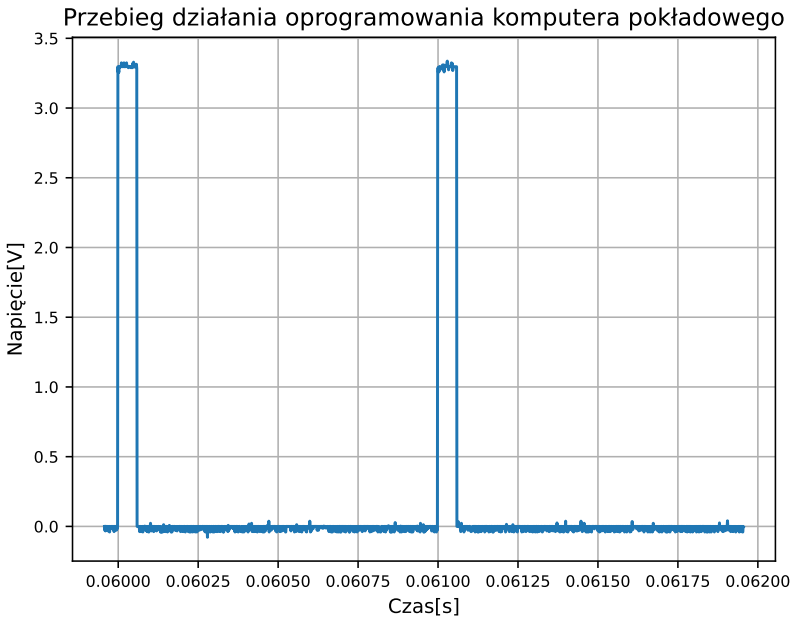
5.3 Weryfikacja działania oprogramowania komputera

W związku z napisaniem oprogramowania oraz projektem płytki drukowanej wykonano wstępne testy poprawności działania. Zweryfikowano komputer pokładowy pod względem działania oprogramowania oraz efektywności żyroskopu.

5.3.1 Test oprogramowania komputera pokładowego

Do weryfikacji działania postanowiono zmierzyć czas obliczeń sterownika ADRC. Kryterium akceptacji testu jest następujące: czas obliczeń sterownika powinien zajmować do $\frac{1}{10}T_p$ (10% pętli taktowania), gdzie T_p oznacza okres próbkowania układu [12]. Kryterium to jest praktyczną zasadą czasu obliczeń związanych ze sterownikiem dla implementacji na jednostkach obliczeniowych.

Do wykonania wspomnianego testu został wykorzystany oscyloskop. Test polegał na wystawianiu stanu wysokiego na pinie GPIO przy starcie obliczeń ADRC, a następnie wystawianie stanu niskiego po ich zakończeniu. Dzięki temu wygenerowany został sygnał prostokątny o wypełnieniu równym czasu obliczeń sterownika, który został przedstawiony na rys. 5.12.



Rysunek 5.12: Wyniki weryfikacji działania oprogramowania komputera pokładowego

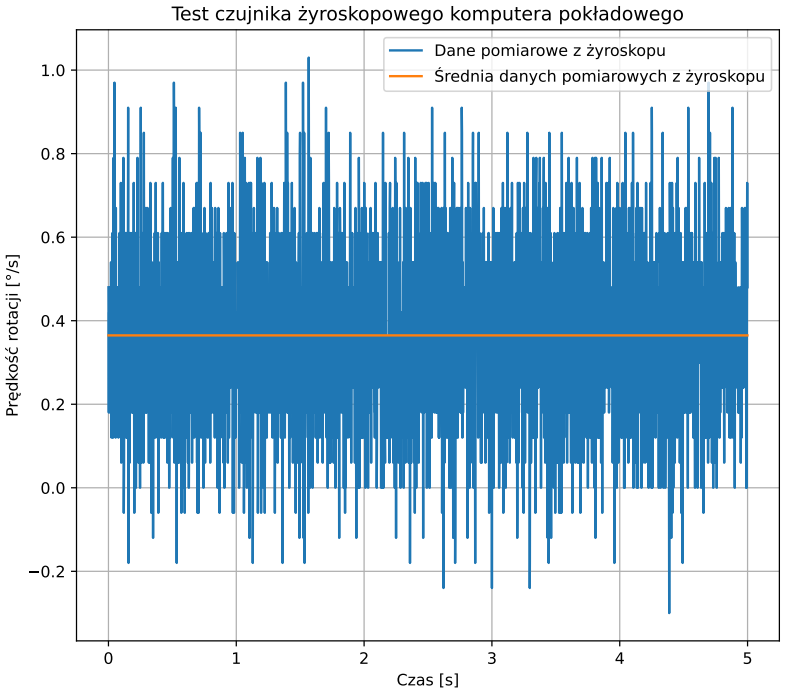
Pomiary częstotliwości taktowania oraz wypełnienia zostały wykonane przez oscyloskop i wynoszą:

- częstotliwość taktowania: 1000 Hz (1 ms),
- wypełnienie 5.98 %.

W efekcie czas obliczeń sterownika ADRC jest równy procentowo 5.98% okresu próbkowania, co spełnia kryterium akceptacji oprogramowania komputera pokładowego.

5.3.2 Test czujnika żyroskopowego komputera pokładowego

Test czujnika żyroskopowego komputera pokładowego sprawdza poprawność działania napisanej biblioteki i implementacji sprzętowej. Polega on na pozostawieniu komputera w pozycji nieruchomej i zebraniu danych z żyroskopu. Zebrane wyniki pozwolą zweryfikować poprawność działania napisanej biblioteki, implementacji sprzętowej oraz efektywność montażu. Kryterium akceptacji jest osiągnięcie $\pm 1\frac{^\circ}{s}$ szumu wokół zerowej prędkości rotacji. Wskazany zakres prędkości rotacji pokrywa niepewności podane przez producenta [15] oraz akceptowalny wpływ błędu wynikający z montażu. Na rys. 5.13 widoczny jest przebieg wykonanego testu, dla którego okres próbkowania danych wynosi $T_p = 0,001s$.



Rysunek 5.13: Wynik testu czujnika żyroskopowego komputera pokładowego

Na przebiegu 5.13 widoczne jest, że stała składowa przesunięta jest do wartości około $0.36\frac{^\circ}{s}$. Wynika to z charakterystyki zastosowanego czujnika. Mimo tego maksymalne bezwzględne wartości prędkości rotacji mieszczą się w zakresie $\pm 1\frac{^\circ}{s}$. Kryterium akceptacji zostało, więc spełnione.

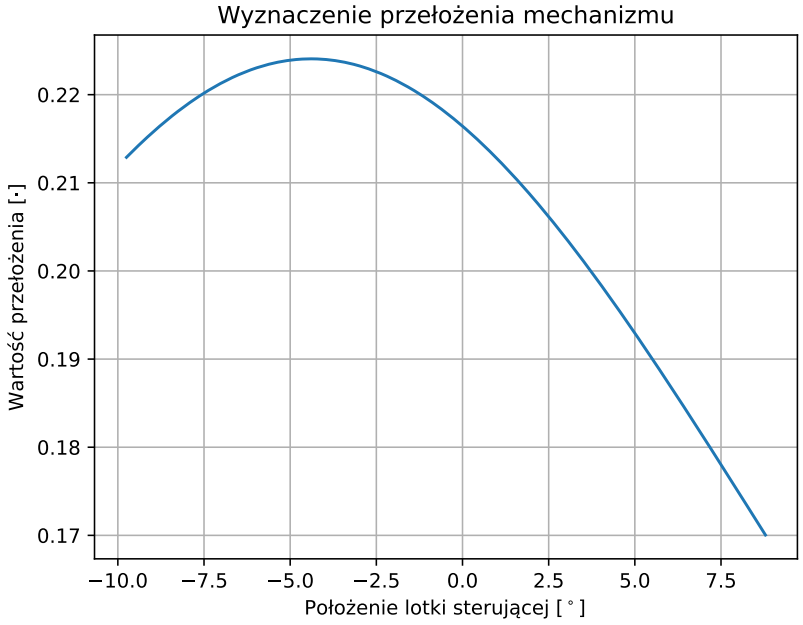
5.4 Budowa i opis działania mechanizmu wykonawczego poruszającego lotkami sterującymi

W związku tym, że zaproponowany układ regulacji został zaprojektowany dla lotów poddźwiękowych, układ wykonawczy musi spełniać określone wymagania. Założeniami projektowymi układu wykonawczego była wytrzymałość na przewidywane obciążenia oraz wymagana rozdzielczość sterowania. Za projekt mechanizmu pozwalającego na zadanie odpowiednich kątów wychylenia lotek sterujących odpowiedzialni byli: Bartosz Buda i dr inż. Bartosz Ziegler z koła naukowego PUT Rocketlab.

Jest kilka możliwości realizacji wspomnianego układu wykonawczego spełniającego wyżej opisane wymogi. Zdecydowano się wykorzystać mechanizm orczykowy, który spełnia założenia projektowe. Do jego

potencjalnych wad należą zwykle duże luzy występujące na połączeniach, które postanowiono zmniejszyć odpowiednio ciasnymi połączeniami mechanicznymi.

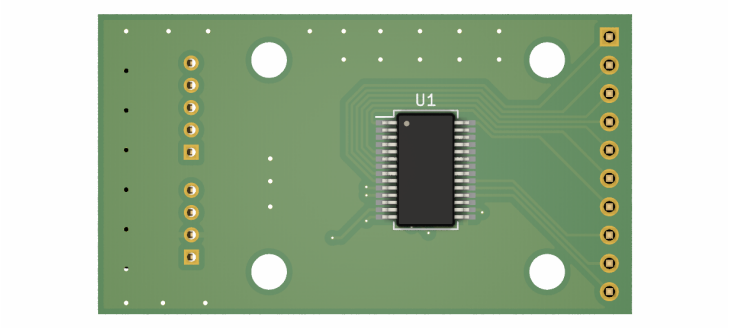
Zakładane, maksymalne kąty wychyleń powierzchni sterowych wynoszą $\pm 10°$. W tak niewielkim zakresie kątów przy prędkościach rakiety układ jest w stanie generować znaczne momenty siły (rzędu kilku niutonometrów). Z tego powodu konieczne było zaprojektowanie odpowiedniego przełożenia pozwalające na dokładniejsze sterowanie. Wykres wartości przełożenia od lotki sterującej dla zaprojektowanego układu wykonawczego widoczny jest na rys. 5.14. Można zauważyć, że przełożenie znajduje się w zakresie około $[0.17; 0.225]$ w zależności od wychylenia lotki sterującej.



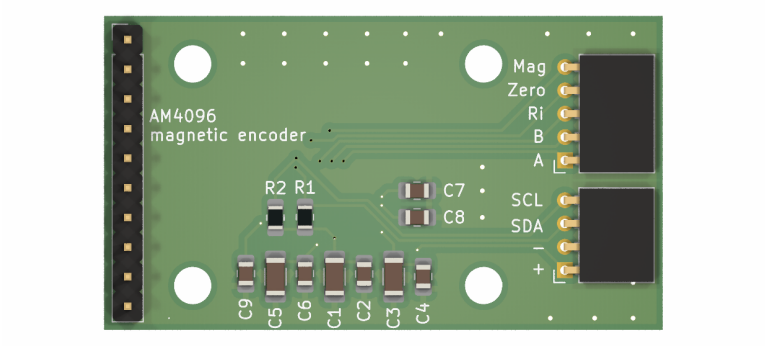
Rysunek 5.14: Wykres przełożenia dla układu wykonawczego

W związku z przewidywanymi obciążeniami oraz rygorem szybkiej reakcji układu mechanicznego koniecznością było dobranie odpowiedniego serwomechanizmu. Wybrano model Power HD R12-S [9]. Wspomniany serwomechanizm pracuje przy napięciu roboczym od 6 do 8,4 V, co pozwala na zasilanie bezpośrednio z baterii 2S. Generuje moment wynoszący około 0,9 – 1,2 Nm. Innym ważnym parametrem jest szybkość sterowania, która dla tego serwomechanizmu wynosi $857.14\frac{^\circ}{s}$ przy napięciu zasilania 6 V oraz $1000\frac{^\circ}{s}$ przy napięciu 7,4 V. Układ wykonawczy został zaprojektowany tak, że każda powierzchnia sterowa jest połączona z serwomechanizmem przez dwa orczyki. Ich parametry geometryczne determinują przełożenie oraz możliwe aplikowane obciążenie. Lotki sterujące sterowane są z osobnych serw. Dodatkowo układ wykonawczy posiada mocowanie dostosowane do dwóch modułów enkoderów magnetycznych. Głównym układem na wspomnianej płytce drukowanej jest enkoder inkrementalny, magnetyczny AM4096 [3]. Pozostałymi komponentami są elementy pasywne, wykorzystywane głównie do filtracji zasilania. Na rys. 5.16 oraz rys. 5.15 znajdują się widoki górny i dolny wspomnianej płytki. Enkoder działa na zasadzie czujnika Halla, wykrywając pole magnetyczne. Magnes umieszczany jest typowo 0,5-1,5 mm od układu scalonego enkodera magnetycznego. Dodatkowo wykorzystywany magnes, musi być namagnesowany radialnie. Na rys. 5.17 przedstawiony jest widok mechanizmu wykonawczego bez górnej podstawy. Owalny magnes znajduje się na końcu wałka podtrzymującego powierzchnie sterowe. Widoczne są również moduły enkodera magnetycznego umieszczone w odpowiedniej odległości od magnesów. Gdy wałek się obraca, a co za tym idzie również magnes, enkoder generuje przebieg prostokątny, który jest wykorzystywany przez

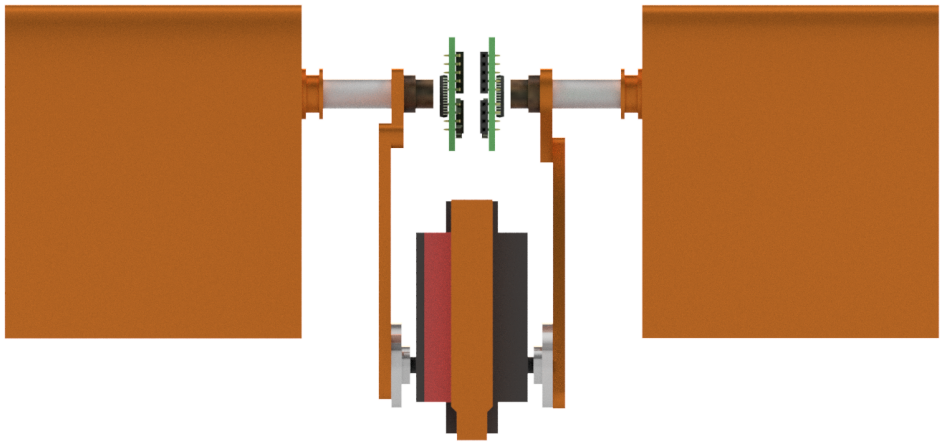
mikrokontroler do wyznaczenia położenia kątowego. Układ ten pozwala na ustawianie pozycji zera oraz mierzy obrót z rozdzielczością 12 bitów.



Rysunek 5.15: Widok modułu enkodera magnetycznego z góry

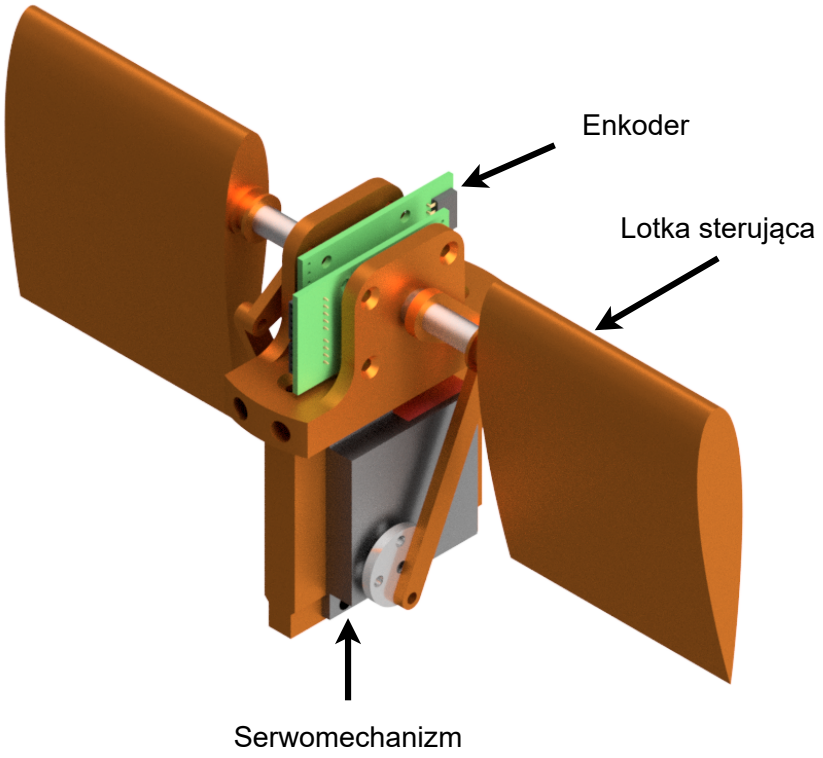


Rysunek 5.16: Widok modułu enkodera magnetycznego z dołu

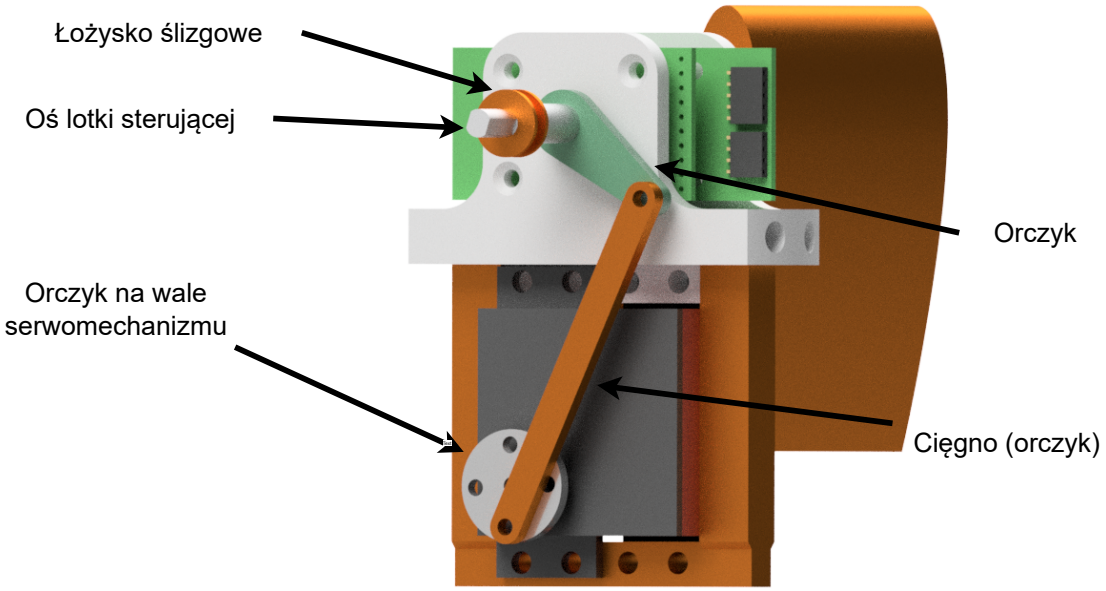


Rysunek 5.17: Widok modułu enkodera magnetycznego zamontowanego do mechanizmu. Projekt: Bartosz Buda, dr inż. Bartosz Ziegler, PUT Rocketlab. Wykonanie: PUT Rocketlab

Całościowo widok zaprojektowanego mechanizmu został przedstawiony na rys 5.18. Na rys. 5.19 przedstawiony jest również szczegółowy widok mechanizmu z wyodrębnionymi kolorystycznie elementami.



Rysunek 5.18: Widok mechanizmu wykonawczego. Projekt: Bartosz Buda, dr inż. Bartosz Ziegler, PUT Rocketlab. Wykonanie: PUT Rocketlab



Rysunek 5.19: Szczegółowy widok mechanizmu wykonawczego. Projekt: Bartosz Buda, dr inż. Bartosz Ziegler, PUT Rocketlab. Wykonanie: PUT Rocketlab

Lotki, podstawa górna, montowanie serwomechanizmów oraz łożyska ślizgowe wykonane zostały poprzez druk 3D. W tym celu wykorzystano materiał PETG. Wałki podtrzymujące powierzchnie sterowe zostały wytoczone z aluminium. Materiałem okrągłych orczyków (orczyki na wale serwomechanizmów) jest również aluminium. Rzeczywiste widoki wykonanego układu wykonawczego znajdują się na rys. 5.22 oraz 5.23.

5.5 Identyfikacja modelu układu wykonawczego poruszającego lotkami sterującymi

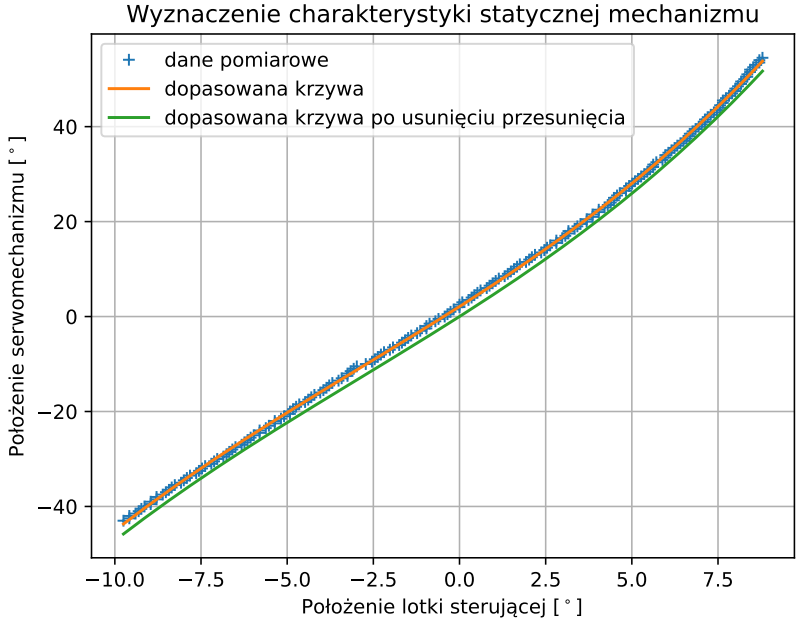
5.5.1 Wyznaczenie charakterystyki statycznej

W związku z zastosowaniem układu wykonawczego ze zmiennym przełożeniem (rys. 5.14) konieczne jest wyprowadzenie charakterystyki opisującej położenie lotki sterującej w zależności od położenia wału serwomechanizmu. Do wykonania wspomnianej identyfikacji wykorzystano układ wykonawczy, komputer pokładowy oraz moduły enkoderów magnetycznych (rys. 5.15, 5.16) do zebrania pomiarów. Widok wykorzystanego mechanizmu wraz z zamontowanymi enkoderami znajduje się na rys. 5.18.

W celu wyznaczenia charakterystyki podawano na serwomechanizm pozycję z zakresu $[-65°; 65°]$ ze skokiem $0.5°$. Wartość przesunięcia kątowego zmierzona została poprzez enkoder magnetyczny. Następnie wielomian trzeciego stopnia został dopasowany do danych pomiarowych:

$$f(\delta) = 0.00816074 \cdot \delta^3 + 0.07168273 \cdot \delta^2 + 4.62038378 \cdot \delta,$$

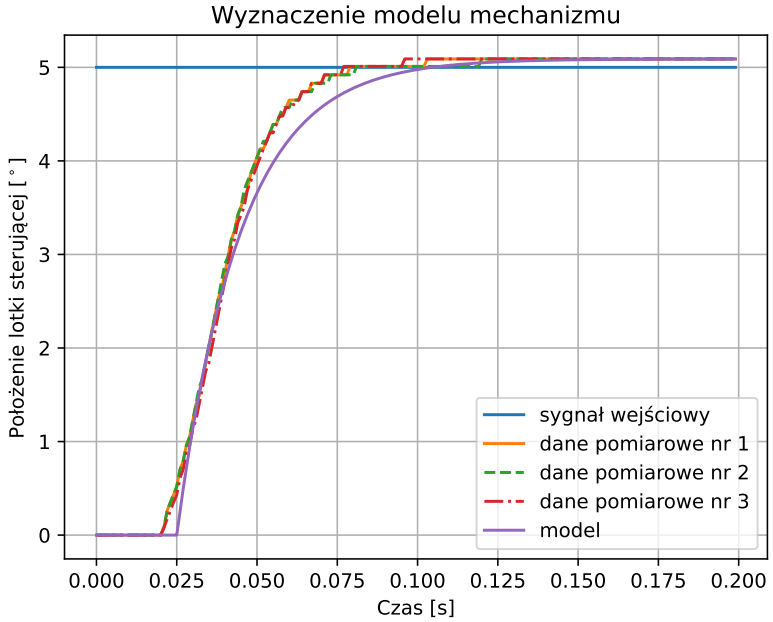
gdzie: δ oznacza pozycję lotki sterującej. Po podstawieniu wartości δ uzyskiwana jest wymagana pozycja podawana na serwomechanizm. Wynikowa krzywa była przesunięta od punktu $(0, 0)$, co bezpośrednio wynika z luzów w układzie wykonawczym. Postanowiono usunąć wspomniane przesunięcie (przesunąć krzywą do zera w osi y), z tego względu, że błąd sterowania wynikający z powstawania luzów w układzie wykonawczym jest akceptowalny oraz sterownik bierze pod uwagę niepewności strukturalne modelu. Dane pomiarowe, dopasowany wielomian oraz zaaplikowane przesunięcie przedstawione są na rys. 5.20.



Rysunek 5.20: Charakterystyka statyczna układu wykonawczego

5.5.2 Wyznaczenie modelu dynamiki układu wykonawczego poruszającego lotkami sterującymi

Układ wykonawczy został zidentyfikowany również pod względem dynamicznym. Wykonano serię pomiarów, do których następnie zostało dopasowane równanie modelu. Do przeprowadzenia badania wykorzystano układ wykonawczy wraz z enkoderami oraz komputer pokładowy. Identyfikację modelu przeprowadzono na podstawie odpowiedzi skokowej. Okres próbkowania, z jakim zbierano dane, wynosił 1 ms. Wykorzystując wyznaczoną charakterystykę statyczną podano wymuszenie w postaci zadanego kąta wychylenia lotki sterującej na wejście serwomechanizmu wynoszące 5° w chwili zerowej. Wyniki dla trzech serii pomiarów, zadanego sygnału wejściowego oraz wyznaczonego modelu znajdują się na rys. 5.21.



Rysunek 5.21: Wyniki wyznaczania charakterystyki dynamicznej

Na podstawie danych pomiarowych przyjęto strukturę modelu w postaci:

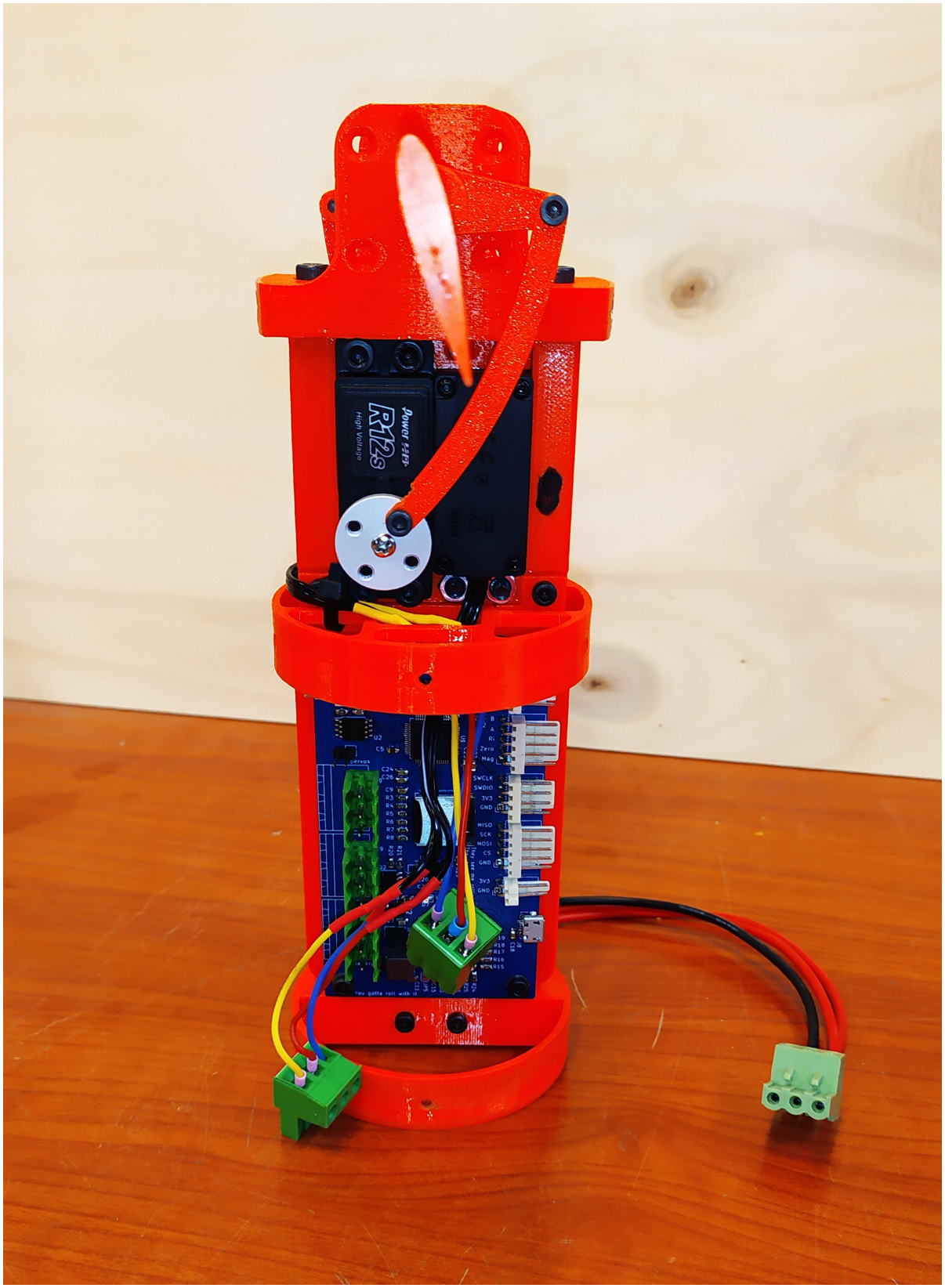
$$G_s(s) \triangleq \frac{ke^{-sT_0}}{1+sT},$$

gdzie:

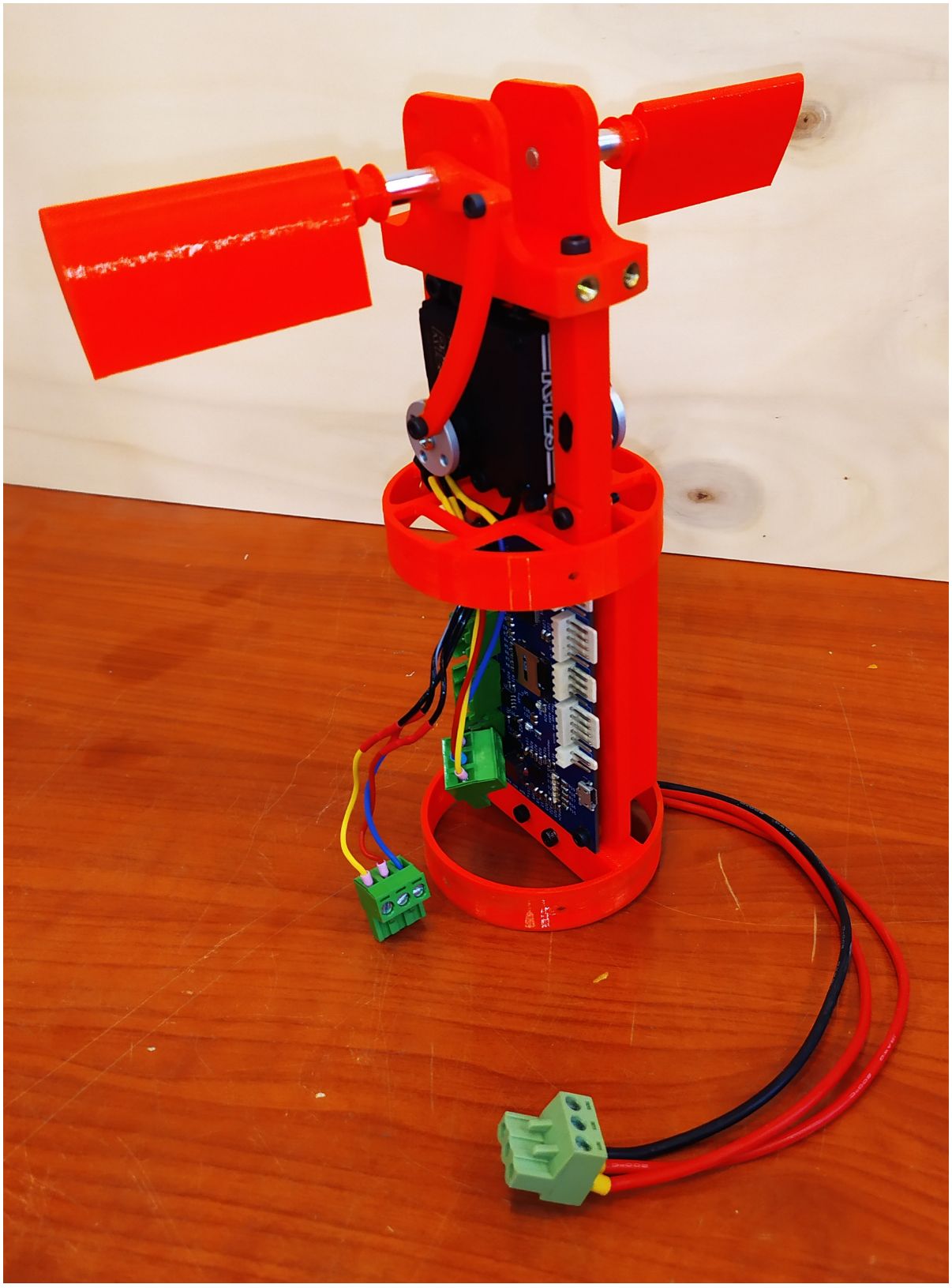
$$G_s(s) \triangleq \frac{\Delta(s)}{\Delta_z(s)},$$

gdzie: $\Delta(s)$ jest sygnałem δ (pozycja lotki sterującej) zapisanym w dziedzinie Laplace'a oraz $\Delta_z(s)$ jest sygnałem δ_z (pozycja zadana lotki sterującej) zapisanym w dziedzinie Laplace'a. Zidentyfikowany model jest, więc inercją pierwszego rzędu z opóźnieniem. Wykorzystując metodę najmniejszych kwadratów, zidentyfikowano parametry przyjętej struktury modelu jako: $T_0 = 0.025$, stała czasowa: $T = 0.0197$ oraz wzmocnienie w przybliżeniu wynoszące $k = 1$. Model w postaci transmitancji przedstawia się następująco:

$$\widehat{G}(s) = \frac{1}{1+0.0197s} \cdot e^{-0.025s}.$$



Rysunek 5.22: Widok boczny wykonanego mechanizmu wykonawczego. Projekt: Bartosz Buda, dr inż. Bartosz Ziegler, PUT Rocketlab. Wykonanie: PUT Rocketlab



Rysunek 5.23: Widok wykonanego mechanizmu wykonawczego. Projekt: Bartosz Buda, dr inż. Bartosz Ziegler, PUT Rocketlab. Wykonanie: PUT Rocketlab

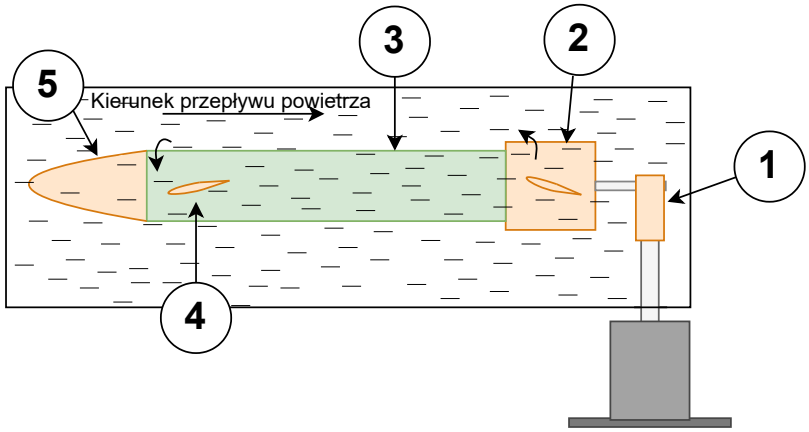
Rozdział 6

Stanowisko testowe i wyniki eksperymentów

6.1 Opis stanowiska testowego

Do wstępnego sprawdzenia systemu sterowania (komputer pokładowy, układ wykonawczy, sterownik) postanowiono wykorzystać tunel aerodynamiczny. Na jego podstawie zaprojektowano eksperyment. Projekt stanowiska testowego został wykonany we współpracy z PUT Rocketlab.

Schematyczna zasada działania stanowiska testowego została przedstawiona na rys. 6.1.

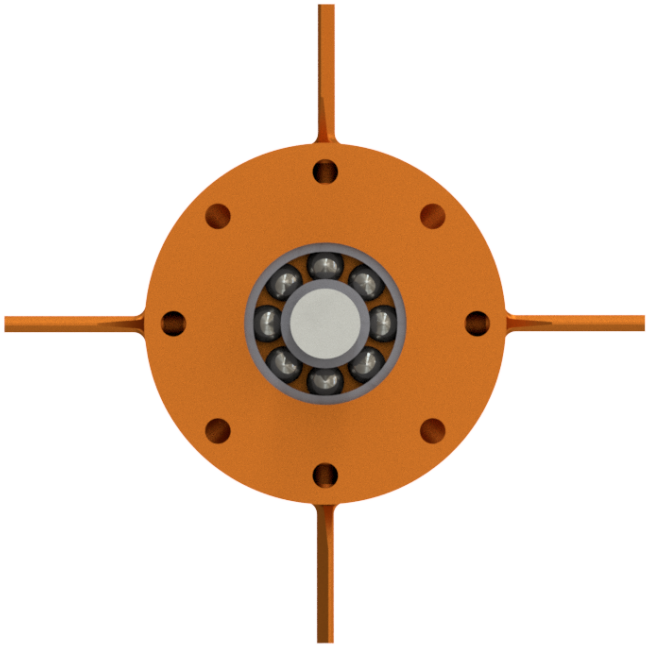


Rysunek 6.1: Schematyczny widok zasady działania stanowiska doświadczalnego w tunelu aerodynamicznym

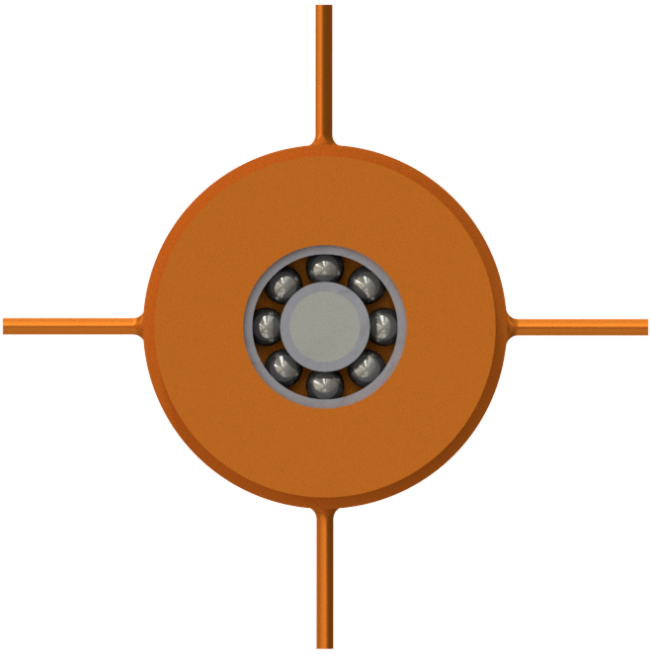
Na rys. 6.1 widoczny jest obszar tunelu aerodynamicznego oraz przepływ powietrza. Odniesienie oznaczone numerem 1 na rys. 6.1 jest mocowaniem części korpusu rakiety. Następnie 2 wskazuje na tylne mocowanie wraz z lotkami, które bezpośrednio wprowadzają asymetrię (przez przekrzywienie względem osi podłużnej), co skutkuje powstawaniem niezerowej prędkości rotacji. Dalej 3 wskazuje korpus rakiety, w którym umieszczony jest układ wykonawczy wraz z komputerem pokładowym. Ruchome powierzchnie sterowe kompensujące prędkość rotacji oznaczone są numerem 4. W celu jak najbardziej rzeczywistego odzwierciedlenia warunków lotu rakiety oraz niezakłócania przepływu, na przedniej części korpusu rakiety umieszczona jest głowica (5 na rys. 6.1).

Zasada generowania prędkości rotacji schematycznie przedstawiona na rys. 6.1 polega na umieszczeniu tylnych lotek pod kątem, na które napiera przepływ powietrza. W ten sposób generowany jest moment, którego schematycznie zwrot oznaczony jest poprzez strzałkę. Układ wykonawczy wychyla lotki sterujące i generuje moment kompensujący ruch wirowy rakiety.

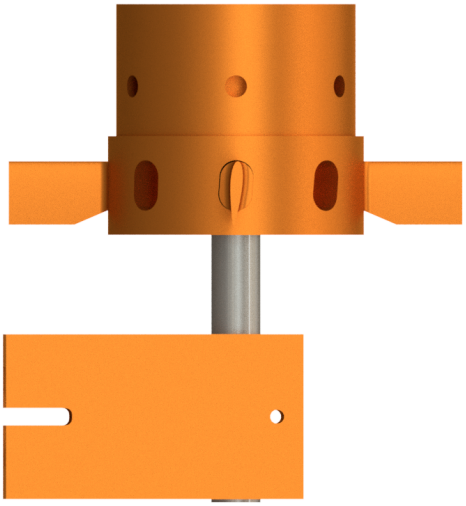
Zaprojektowana część generująca niezerowe prędkości rotacji przedstawiona jest na rys. 6.2 oraz 6.3. Wymiary mocowania wynoszą w najszerszej średnicy 77,5 mm, natomiast średnica części wchodzącej do korpusu wynosi 72,5 mm. Wysokość mocowania lotek generujących prędkość rotacji wynosi 70 mm. We wspomnianym elemencie znajdują się dwa wgłębienia na łożyska kulkowe 6202ZZ. Przez środek łożysk przechodzi pręt stalowy o średnicy 15mm. Złożenie wspomnianych elementów wraz z mocowaniem do profilu widoczne jest na rys. 6.4.



Rysunek 6.2: Mocowanie dolne korpusu rakiety widok z dołu. Projekt oraz wykonanie: PUT Rocketlab

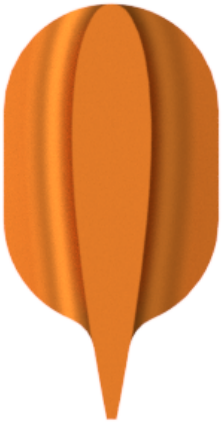


Rysunek 6.3: Mocowanie dolne korpusu rakiety widok z dołu. Projekt oraz wykonanie: PUT Rocketlab



Rysunek 6.4: Mocowanie korpusu rakiety w tunelu aerodynamicznym, widok boczny. Projekt oraz wykonanie: PUT Rocketlab

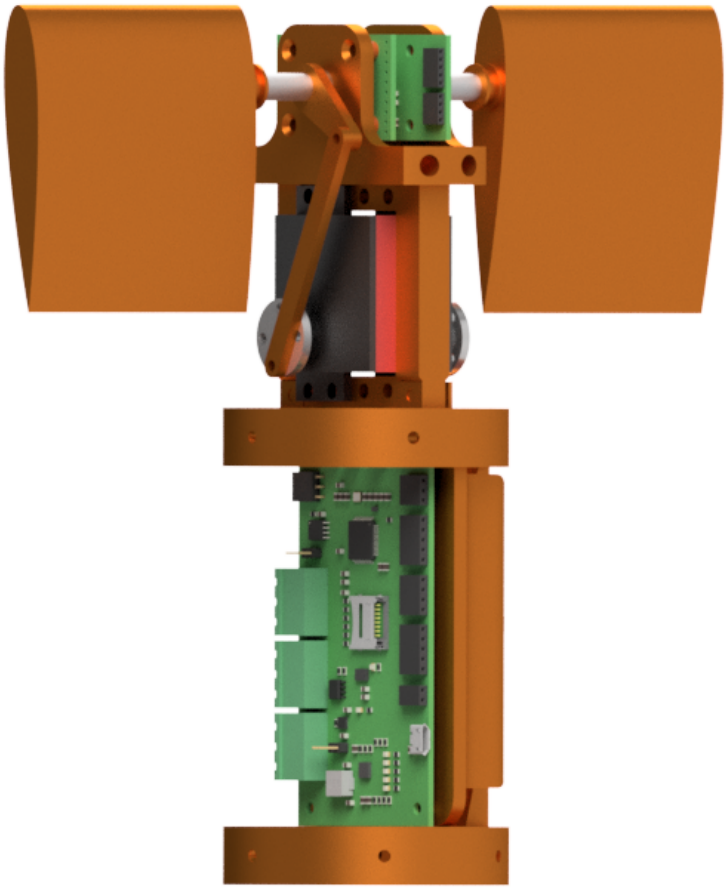
Lotki widoczne na rys 6.4 mają przekrój NACA0018 [16]. Został on ustandaryzowany przez komitet NACA (ang. *The National Advisory Committee for Aeronautics*). Jest to przekrój, który ma zdefiniowane zależności geometryczne oraz parametry aerodynamiczne. Dzięki temu stanowisko testowe jest łatwiejsze do odtworzenia. Na rys. 6.5 oraz 6.6 znajduje się widok wykorzystanej lotki. Ten sam przekrój został wykorzystany do projektu lotek sterujących. Mocowanie elektroniki wraz z mechanizmem przedstawione jest na rys. 6.7.



Rysunek 6.5: Widok przekroju lotki. Projekt oraz wykonanie: PUT Rocketlab

Rysunek 6.6: Widok boczny lotki. Projekt oraz wykonanie: PUT Rocketlab

Złożenie stanowiska testowego bez podstawy przedstawione jest na rys. 6.8. Widoczny korpus rakiety został wykonany z tekturowej tuby, na którą zostało nawinięte włókno szklane pokryte żywicą epoksy-



Rysunek 6.7: Mocowanie elektroniki wraz z mechanizmem poruszającym lotkami sterującymi. Projekt oraz wykonanie: PUT Rocketlab.

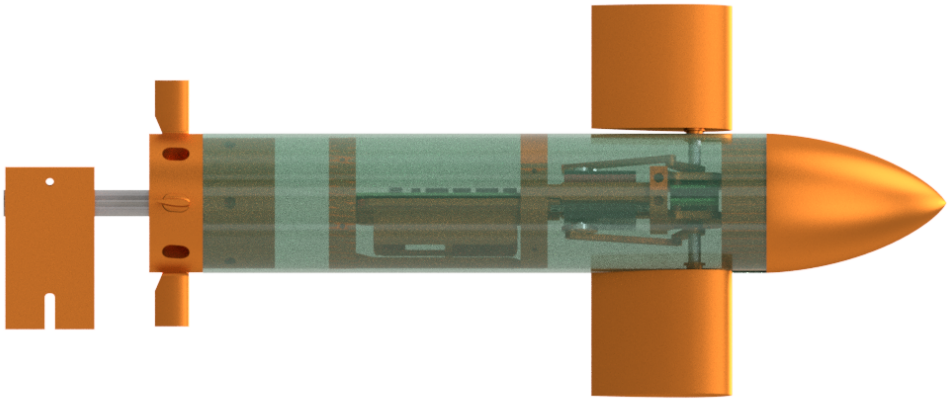
dową. Wewnętrzna średnica korpusu wynosi 73 mm, natomiast zewnętrzna 78 mm. Profil głowicy został wyznaczony według kształtów głowicy z serii Haacka. Wzory, według których została zaprojektowana głowica, przedstawiają się następująco [42]:

$$\theta(x) = arccos(1 - \frac{2x}{L}), \tag{6.1}$$

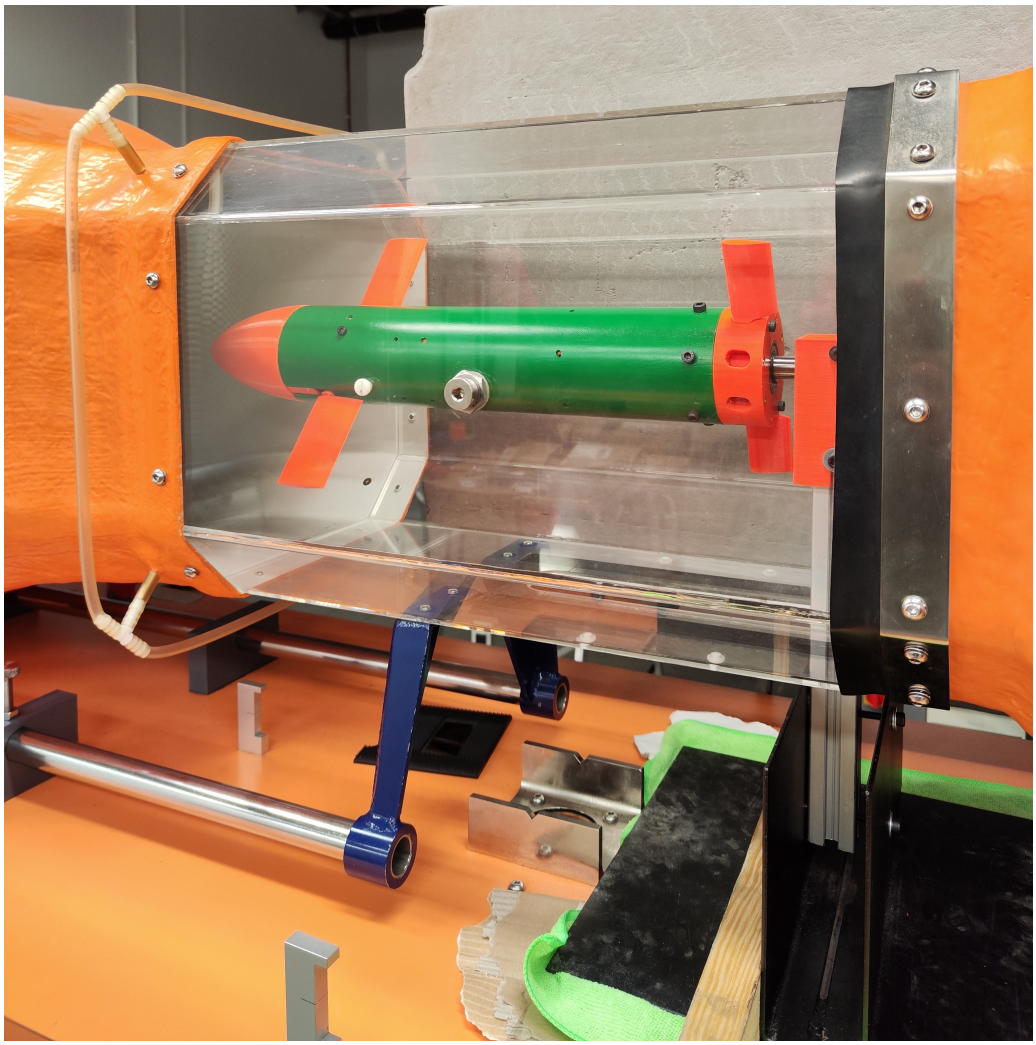
$$y(\theta, C) = \frac{R}{2\sqrt{\pi}} \cdot \sqrt{\theta - \frac{sin(2\theta)}{2} + Csin^3(\theta)}, \tag{6.2}$$

gdzie: R oznacza promień głowicy, L długość głowicy, natomiast C to stała determinująca wypukłość głowicy.

Elementy oznaczone kolorem pomarańczowym na rys. 6.8, 6.7, 6.6, 6.5, 6.4, 6.3 oraz 6.2 zostały wykonane za pomocą druku 3D. W tym celu został wykorzystany materiał PETG. Na rys. 6.9 znajduje się zdjęcie wykonanego stanowiska testowego.



Rysunek 6.8: Złożenie stanowiska testowego bez mocowania stołowego. Projekt oraz wykonanie: PUT Rocketlab.



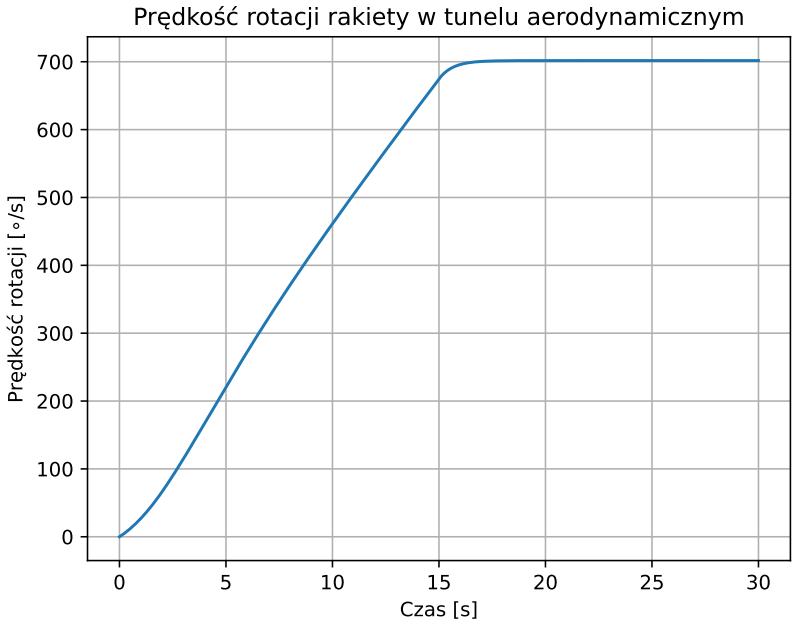
Rysunek 6.9: Zdjęcie wykonanego stanowiska testowego. Projekt oraz wykonanie: PUT Rocketlab.

6.2 Dobór parametrów geometrycznych stanowiska

Parametry geometryczne lotek generujących prędkość kątową zostały dobrane w taki sposób, by wprowadzać porównywalne wartości prędkości rotacji do rzeczywistego lotu rakiety. W symulacji dla wybranego modelu rakiety maksymalna prędkość rotacji wynosi około $400\frac{^\circ}{s}$. Z rzeczywistych lotów wynika, że prędkość rotacji jest często większa [34] niż w symulacji. Postanowiono, więc dobrać tak parametry by wynikowa prędkość obrotowa w tunelu była większa niż ta dla symulowanego modelu rakiety. Wynikowe parametry geometryczne lotek powodujących rotację przedstawiają się następująco (zgodnie z rys. 2.3 s – rozpiętość C_r – podstawa lotki):

- Zestaw 1 – rozpiętość: 55 mm, podstawa: 30 mm
- Zestaw 2 – rozpiętość 42 mm, podstawa: 20 mm

Symulowano prędkość przepływu w tunelu w uproszczony sposób, tzn. jako rampę o czasie narastania 15 sekund. Dla wybranych lotek wprowadzających ruch wirowy uzyskano przebieg prędkości rotacji widoczny na rys. 6.10.



Rysunek 6.10: Symulacja prędkości rotacji w tunelu dla wybranego zestawu lotek

Maksymalna prędkość rotacji wynosi około $700\frac{^\circ}{s}$. Symulacja została wykonana dla 4 lotek o powierzchni z pierwszego zestawu. Przekrzywienia wszystkich zestawów lotek powodujących rotację wynoszą 8°. Na tej podstawie dobrano odpowiednie parametry dla lotek sterujących, które wynoszą (zgodnie z rys. 2.3 s – rozpiętość C_r – cięciwa):

- rozpiętość: 75 mm
- cięciwa: 42 mm

Profil głowicy został wyznaczony za pomocą wzorów (6.1) oraz (6.2), używając wartości C = 0,33 (typ LV-Haack), R = 39 mm, L = 100 mm.

6.3 Wyniki eksperymentu

Eksperyment polegał na realizacji dwóch scenariuszy testowych. Pierwszy scenariusz związany był z włączeniem sterowania dopiero po osiągnięciu maksymalnej prędkości przepływu powietrza. Drugi scenariusz natomiast polegał na włączeniu sterowania od początku działania tunelu. Za każdym razem prędkość przepływu powietrza w tunelu aerodynamicznym wynosiła $26\frac{m}{s}$.

Test wykonano dla następujących nastaw sterownika (3.29), (3.30):

$$\omega_0 = 15,$$

$$\omega_c = 0,5 \cdot \omega_0,$$

$$\xi = 0,08,$$

$$k_0 = 58,5225,$$

$$k_1 = 1,224,$$

$$\hat{b} = 5000.$$

Te same nastawy zostały wykorzystane w symulacji w rozdziale 4.

Wyniki pierwszego scenariusza testowego zostaną przedstawione za pomocą przebiegów. Tunel został włączony w zerowej chwili. Na początku widoczny jest wzrost prędkości rotacji. Następnie po osiągnięciu maksymalnej prędkości przepływu, zostaje włączony sterownik. Przebiegi prędkości rotacji z żyroskopu, estymowanej prędkości rotacji, estymowanego całkowitego zaburzenia, zadanej pozycji lotek sterujących oraz przyspieszenia znajdują się na rys. 6.11, rys. 6.12 oraz rys. 6.13.

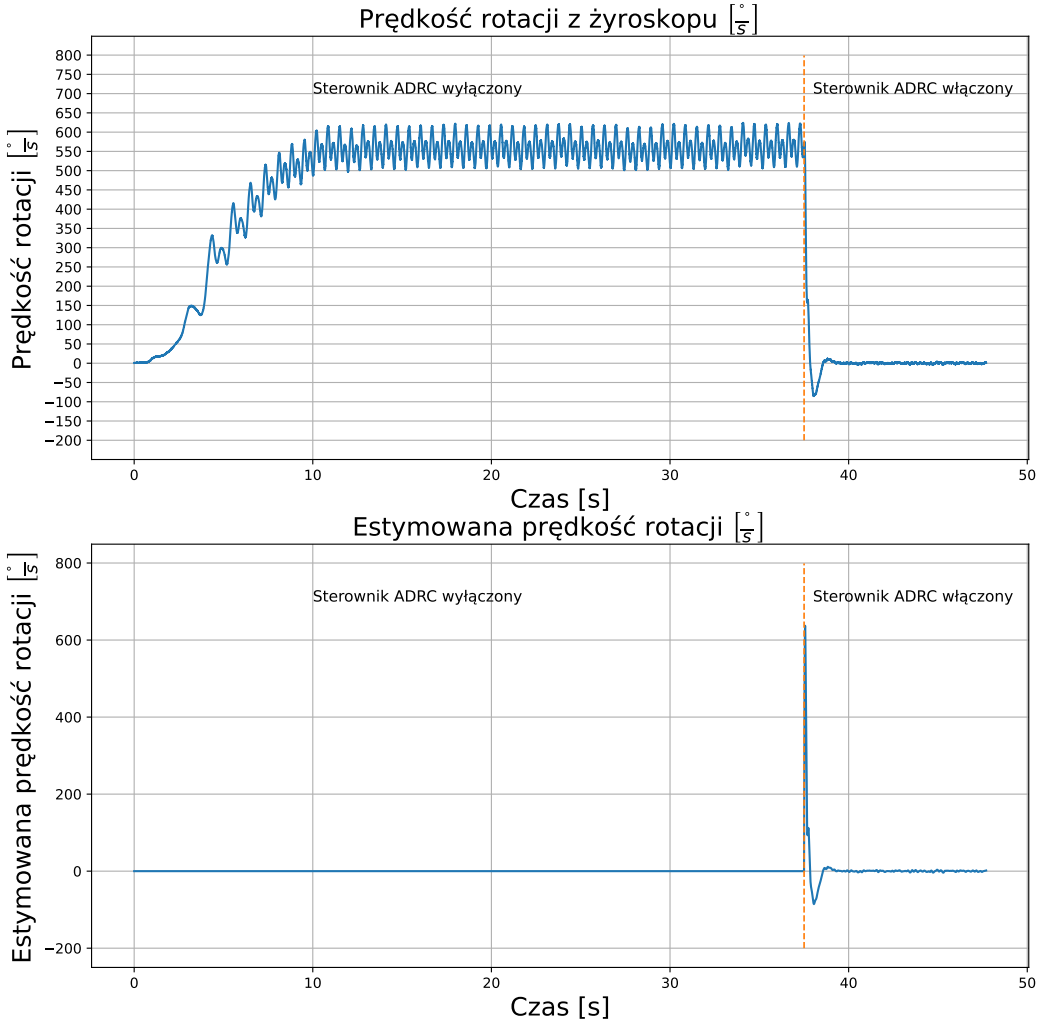
Na przebiegu prędkości rotacji z rys. 6.11 od 0 s do około 37 s widoczne są mody powodujące nagłe wzrosty oraz zmniejszenia prędkości obrotowej. Ich powstawanie związane jest głównie z niesymetrycznie rozłożoną masą i zaburzeniami w przepływie dla lotek napędzających rotację. Średnia prędkość rotacji stabilizuje się na około $550\frac{^\circ}{s}$. W chwili 37,506 s następuje włączenie sterownika ADRC. Ten fakt przejrzyście pokazany jest na przebiegu estymowanej prędkości rotacji. Widoczny pik estymowanej wartości prędkości obrotowej oznacza moment włączenia sterowania. Prędkość rotacji zmniejsza się do tunelu akceptacji wynoszącego $\pm 20\frac{^\circ}{s}$ w 38,465 sekundzie. Daje to czas ustalania równy 0,959 s. Na rys. 6.12 w momencie włączenia sterowania widoczny jest pik estymowanego całkowitego zaburzenia, po czym około 39 sekundy następuje ustabilizowanie na w przybliżeniu stałej wartości. Bezpośrednio przejawia się to w tym, że zadana pozycja lotki sterującej jest odpowiednia do utrzymywania prędkości rotacji w zdefiniowanym zakresie akceptowalności. Zatem zadanie sterowania zostało zrealizowane z wymaganą jakością sterowania. Na przebiegu estymowanego przyspieszenia rotacji na rys. 6.13 widoczny znaczny pik około 37 s wynoszący około $6000\frac{^\circ}{s^2}$ wynika z włączenia sterowania. Następnie pik $-4000\frac{^\circ}{s}$ pomiędzy 37-38 sekundą oznacza hamowanie rakiety dla ruchu w osi podłużnej.

Uzyskana maksymalna prędkość rotacji jest mniejsza, niż zasymulowano w podrozdziale 6.2. Wynika to głównie z tego, że zastosowano dwa różne zestawy lotek oraz oporów łożysk.

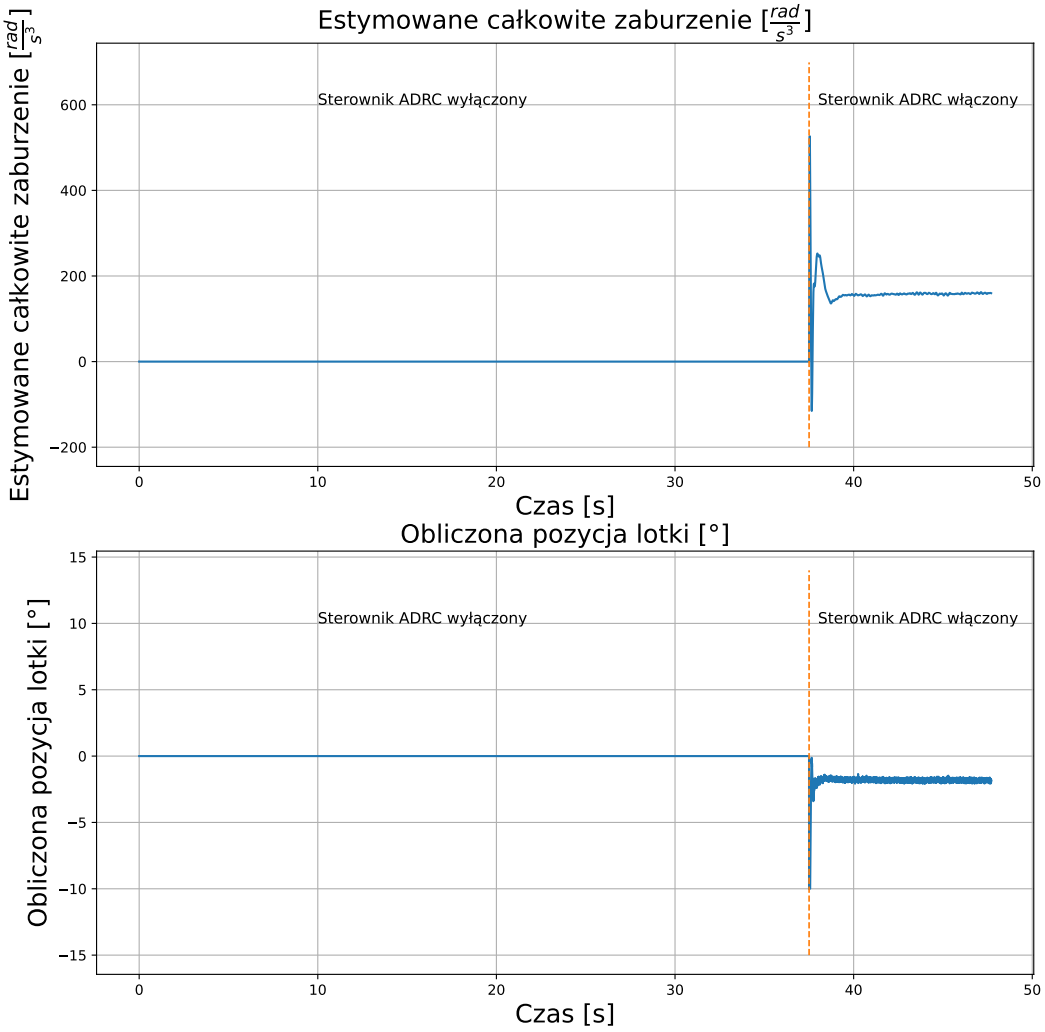
Następnie przedstawione zostaną wyniki dla drugiego scenariusza testowego. W tym przypadku sterownik ADRC jest włączony od początku działania tunelu. W chwili 0 s następuje włączenie tunelu.

Początkowo prędkość przepływu powietrza narasta, w chwili około 2.7 sekundy na przebiegu prędkości rotacji z żyroskopu na rys. 6.14 widoczny jest pik prędkości wynoszący około $-60\frac{^\circ}{s}$. Wynika on z tego, że przy montażu stanowiska, komputer pokładowy był włączany, a następnie mocowany w korpusie. Podczas dalszego montażu, przy przypadkowym niekontrolowanym obrocie pojawiła się początkowa pozycja lotek sterujących wynosząca około 6.8° widoczne na rys. 6.15. Przy narastającej prędkości przepływu powietrza początkowa pozycja lotek sterujących spowodowała pik prędkości rotacji, który następnie został zmniejszony. Na rys. 6.15 widoczne jest, jak przy narastaniu prędkości zmienia się estymowane całkowite

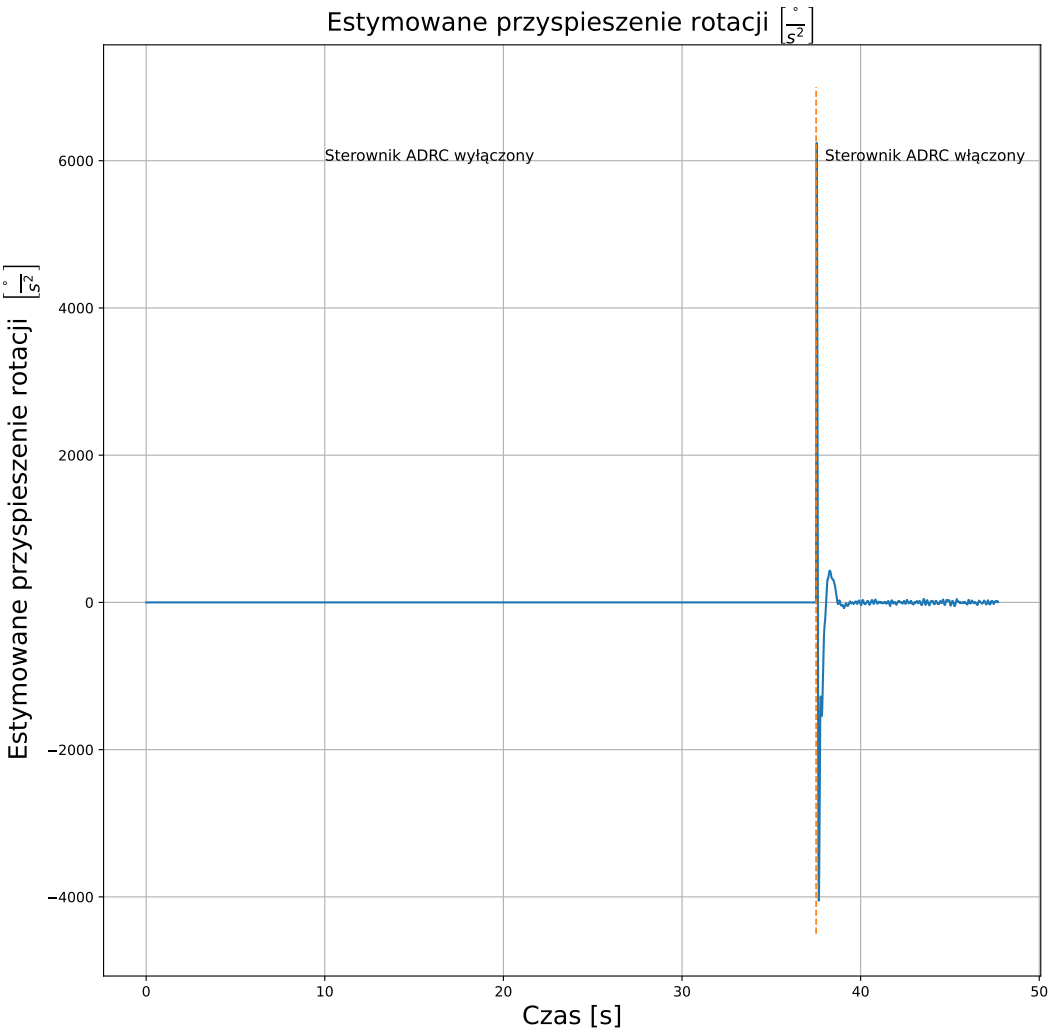
zaburzenie. Po około 12 sekundach następuje osiągnięcie maksymalnej prędkości przepływu i stabilizacja wartości całkowitego zaburzenia. Przyspieszenie i hamowanie rakiety w osi podłużnej, dla tego scenariusza testowego widoczne jest na rys. 6.16. Zadanie sterowania zostało zrealizowane z wymaganą jakością sterowania.



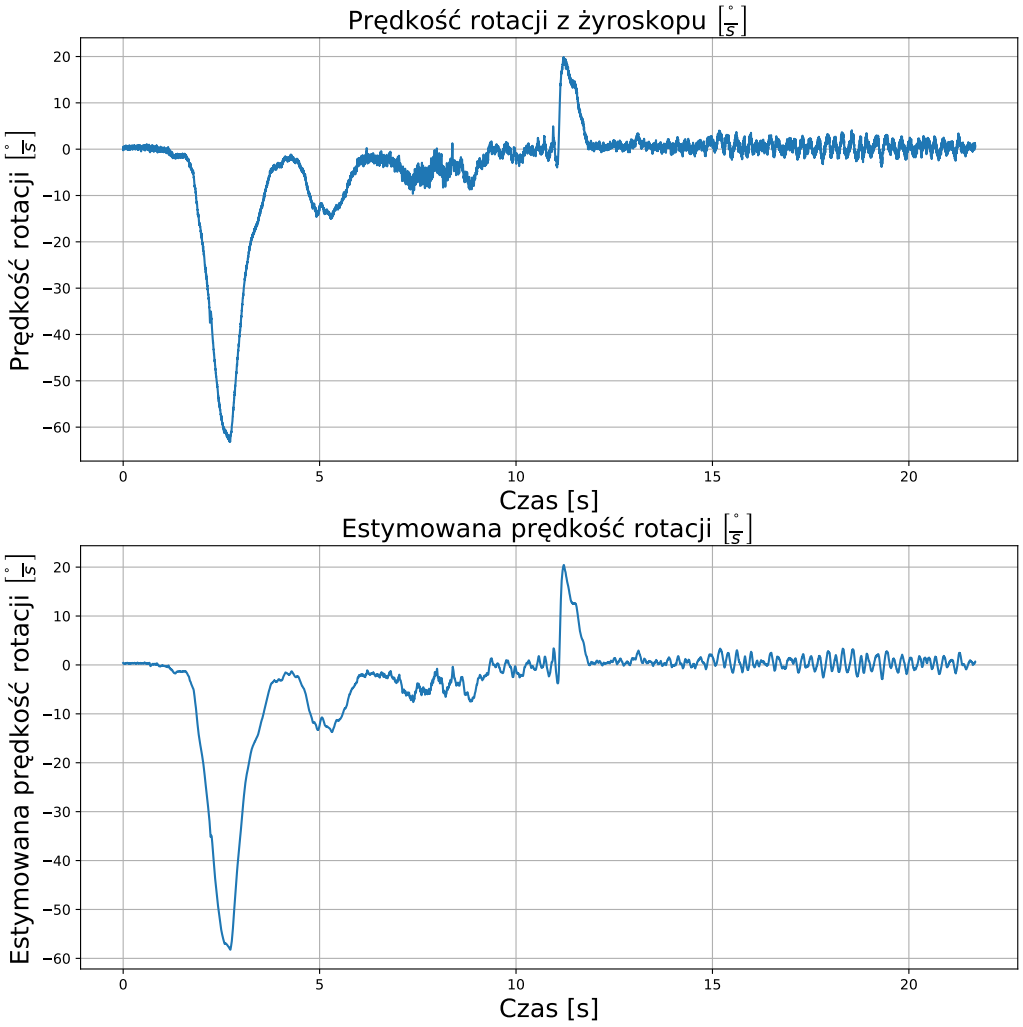
Rysunek 6.11: Scenariusz testowy 1. Przebiegi prędkości rotacji z żyroskopu oraz estymowanej prędkości rotacji.



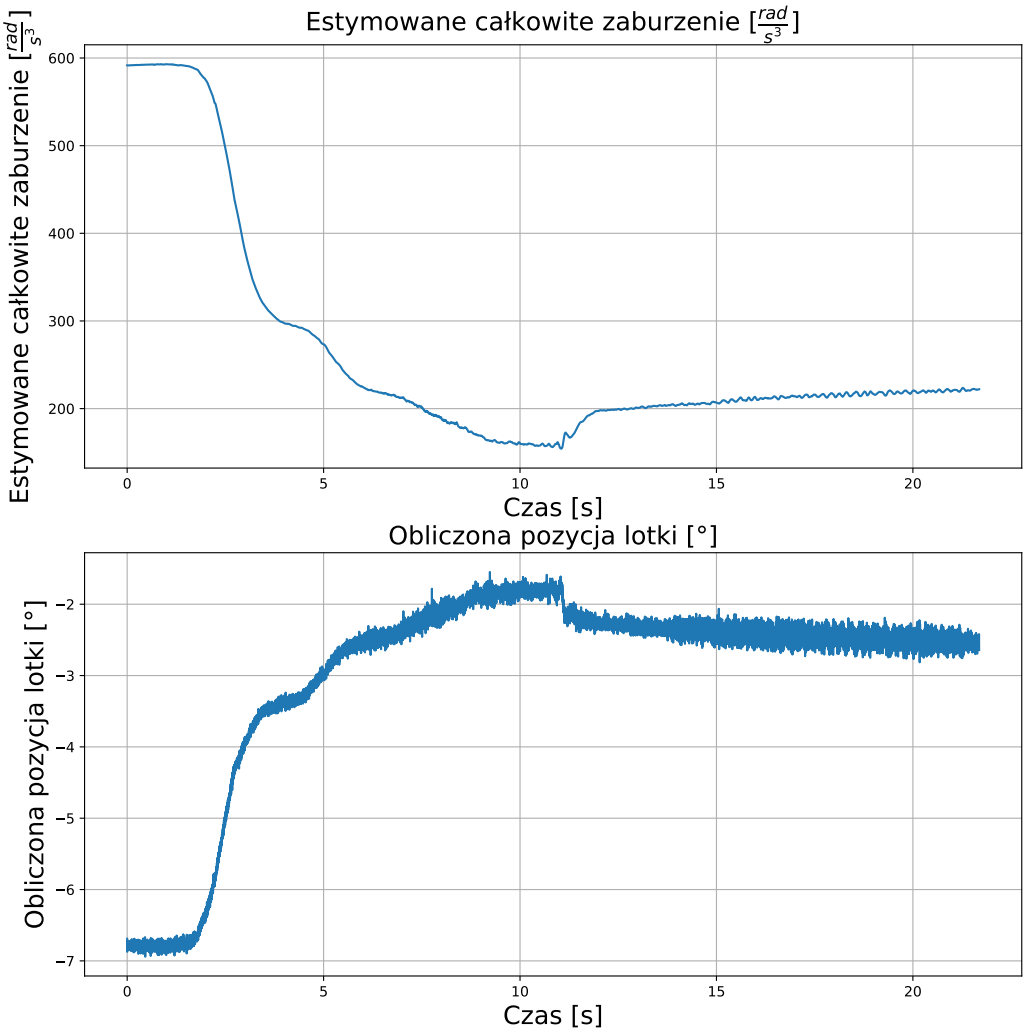
Rysunek 6.12: Scenariusz testowy 1. Przebiegi estymowanego całkowitego zaburzenia oraz zadanej pozycji na lotki sterujące.



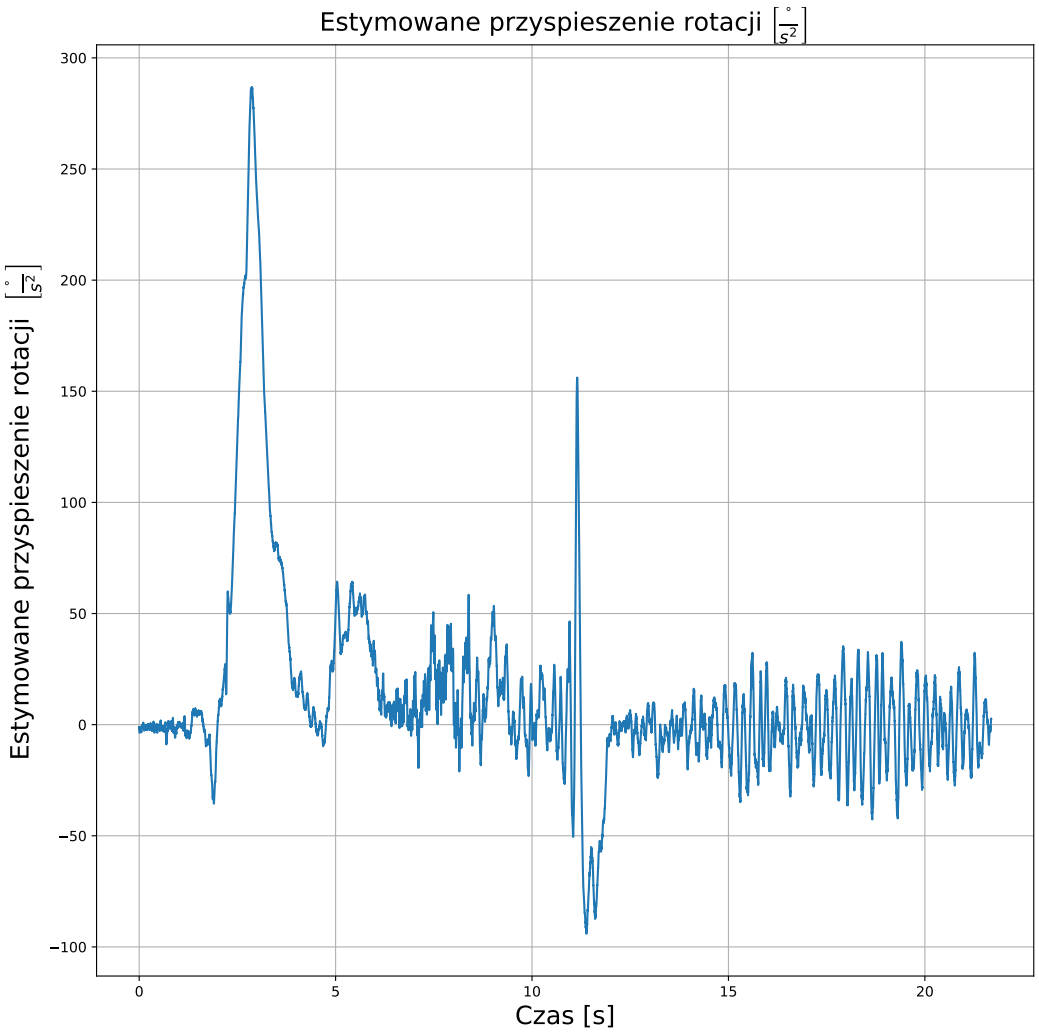
Rysunek 6.13: Scenariusz testowy 1. Przebieg estymowanego przyspieszenia rotacji.



Rysunek 6.14: Scenariusz testowy 2. Przebiegi prędkości rotacji żyroskopu oraz estymowanej prędkości rotacji



Rysunek 6.15: Scenariusz testowy 2. Przebiegi estymowanego całkowitego zaburzenia oraz pozycji lotki sterującej.



Rysunek 6.16: Scenariusz testowy 2. Przebieg estymowanego przyspieszenia rotacji.

Rozdział 7

Podsumowanie

Wykonana praca dyplomowa stanowiła połączenie aerodynamiki, automatyki, elektroniki oraz programowania. Spojrzenie z różnych perspektyw pozwoliło wysunąć spójne wnioski i wybrać drogi rozwoju projektu. Wzięto pod uwagę ograniczenia wynikające z poszczególnych wymienionych sektorów.

Przeprowadzone testy w tunelu aerodynamicznym potwierdziły skuteczność wykonanego systemu sterowania. Wykonanie dwóch scenariuszy testowych pozwoliło na sprawdzenie systemu w różnych warunkach. Pierwsze dwa testy skupiały się na sprawdzeniu skuteczności układu sterowania przy opóźnionym załączeniu sterownika. Pierwszy scenariusz testowy wykazał, że cel sterowania został spełniony przy opisanych zaburzeniach. Po włączeniu sterowania mody występujące przy narastaniu prędkości są prawie niewidoczne. W drugim scenariuszu testowym skupiono się na przetestowaniu układu sterowania w stanie zmieniającej się prędkości przepływu powietrza. Przy narastaniu prędkości przepływu w tunelu, początkowa pozycja lotki spowodowała rotacje, która została następnie skompensowana. Do obszaru ustabilizowania się prędkości przepływu, widoczne były fluktuacje znajdujące się w zdefiniowanym zakresie akceptacji $\pm 20\frac{^\circ}{s}$. System więc zachowuje się adaptacyjnie względem zmieniających się warunków, co potwierdzają wykonane testy.

Warto podkreślić, że zmiana prędkości w tunelu aerodynamicznym w porównaniu ze zmianą prędkości rakiety podczas lotu jest mała. W przypadku tunelu prędkość przepływu zmienia się od 0 do 0,07 liczby Macha w około 15 sekund. W przypadku rakiety w fazie napędzanej silnikiem jest to zmiana od 0 do około 0,8 Macha w około 2 sekundy. W fazie lotu bezwładnościowego od około 0,8 do 0 liczby Macha w ciągu około 15 sekund. Jest to znaczna różnica pomiędzy warunkami eksperymentalnymi a rzeczywistym lotem. Układ wykonawczy sterujący lotkami ma opóźnienie liczące 0,025 s, natomiast jego stała czasowa wynosi 0,0197 sekundy. Co daje 0,0447 sekundy zwłoki w ustaleniu zadanej pozycji. W przypadku tunelu aerodynamicznego znaczenie tego było niewielkie. Natomiast w locie rakiety, kiedy prędkość znacząco zmienia się w krótkim czasie, możliwe jest niespełnienie zadania sterowania z akceptowalną jakością (większa wartość czasu ustalania, nieosiągnięcie zakresu akceptowalnej prędkości rotacji). Dodatkowo w symulacji zauważono, że z powodu znacznego opóźnienia układu wykonawczego istnieje wąski zakres nastaw, dla którego zadanie sterowania zostaje zrealizowane z akceptowalną jakością.

Zostało potwierdzone, że zaimplementowany sterownik ADRC z uwzględnieniem opóźnienia realizuje zadanie sterowania z akceptowalną jakością w warunkach eksperymentalnych i symulacyjnych. Istnieją jednak wątpliwości dotyczące efektywności wybranego sterownika w przypadku tak znaczącego opóźnienia. W automatyce jednym ze sposobów radzenia sobie ze znacznymi opóźnieniami jest zastosowanie predyktora Smitha [19][37]. Predyktor Smitha pozwala między innymi na zwiększenie pasma pętli zamkniętej. W praktyce poprawia jakość sterowania dla obiektów ze znacznymi opóźnieniami. Jednym z założeń zastosowania wspomnianego predyktora jest dokładna znajomość modelu w celu uzyskania dobrej jakości sterowania. W przypadku zaprezentowanego obiektu to założenie nie jest spełnione. Jednak ist-

nieje rozszerzenie sterownika, które pozwala połączyć zalety predyktora Smitha oraz klasycznego ADRC. Zaproponowany w [20] sterownik FFC-ADRC redukuje wpływ opóźnienia. Poprawia jakość estymaty całkowitego zaburzenia poprzez umożliwienie zwiększenia pasma przenoszenia obserwatora. Warto jednak zauważyć, że zastosowanie tego rozszerzenia skomplikuje kod oprogramowania komputera pokładowego i wydłuży czas obliczeń sterownika. Może się to wiązać z przekroczeniem maksymalnego wymaganego czasu obliczeń.

Model rakiety zaimplementowany w programie MATLAB pozwolił na wstępną weryfikację działania układu sterowania przed wykonaniem eksperymentu w świecie rzeczywistym. Dzięki temu, że zaimplementowano model rakiety 6DoF, możliwe było nie tylko sprawdzenie skuteczności algorytmu sterowania, ale także weryfikację stabilności lotu rakiety. Symulator nie daje jednak możliwości symulacji zachowania rakiety podczas lotu przy zmiennym wietrze. W związku z tym w przyszłości należałoby zaimplementować model wiatru pozwalający na symulację warunków pogodowych zbliżonych do rzeczywistych.

Podczas testów od strony oprogramowania nie zauważono błędów powodujących przerwanie testu. Zaimplementowana architektura pozwoliła mikrokontrolerowi na wykonywanie obliczeń związanych ze sterownikiem w ramach $5,98\%$ okresu próbkowania. W przypadku wykorzystanych parametrów sterownika jest to satysfakcjonujący wynik, jednakże jest to część, którą można poddać rewizji. Do tego można wykorzystać bibliotekę CMSIS-DSP [1], której funkcje wykonujące operacje na macierzach, wykorzystują instrukcje SIMD (ang. *Single Instruction Multiple Data*). Jest to rodzaj instrukcji pozwalający na wykonywanie serii obliczeń w jednym cyklu. Jest to jedna z metod zrównoleglania przetwarzania danych. Zastosowanie tej biblioteki pozwoliłoby na skrócenie czasu obliczeń. Zauważono również, że wybrana zastępcza pamięć nieulotna flash pozwala na zapis pakietów danych przez około 5 minut. Jest to czas, który jest wystarczający do zapisu danych z lotu, jednakże powoduje to komplikację kodu. W związku z tym korzystniejszą opcją jest zastosowanie zapasowej pamięci nieulotnej pozwalającej na zapis większej ilości danych.

Wykorzystany mikrokontroler w połączeniu z oprogramowaniem pozwalał na sprawne wykonywanie stosownych obliczeń. Zastosowany żyroskop pozwolił na wykonywanie akceptowalnej jakości pomiarów prędkości rotacji przy wykorzystanej szybkości próbkowania na poziomie 1000 Hz. Zastosowane złącze na kartę microSD pozwoliło na ułatwiony dostęp do danych pomiarowych uzyskanych podczas wykonywania eksperymentów. Dzięki wbudowanej ładowarce do ogniw litowo-jonowych możliwe było bezproblemowe utrzymywanie odpowiedniego stanu naładowania baterii. Możliwe jest jednak dalsze usprawnianie komputera pokładowego. Mimo możliwości sterowania serwomechanizmami modelarskimi wykorzystując sygnał PWM generowany przez mikrokontroler, na płytce drukowanej brakuje zabezpieczenia nadprądowego, które zmniejszyłoby ryzyko uszkodzenia komputera w przypadku zwarcia. Dodatkowym elementem zmniejszającym szansę na wywołanie awarii byłaby implementacja zabezpieczenia przed odwrotną polaryzacją napięcia zasilającego. Zauważono także potrzebę dodania dodatkowego złącza na kartę microSD w celu zwiększenia redundancji metod zapisu danych pomiarowych. Spowodowane jest to częstymi nagłymi awariami kart SD. Zwiększenie liczby komponentów zapisujących dane może wymagać także dodanie dodatkowej jednostki obliczeniowej (np. w postaci drugiego mikrokontrolera) w celu zmniejszenia obciążenia mikrokontrolera odpowiadającego za sterowanie.

Wykorzystany układ wykonawczy jest efektywny przy określonych nastawach sterownika oraz warunkach eksperymentalnych. Serwomechanizmy modelarskie wprowadzają znaczne opóźnienie. Są one bezpośrednią przyczyną ograniczenia wartości nastaw, co skutkuje w pogorszeniu jakości sterowania. Jednym z rozwiązań tego problemu jest więc rewizja układu wykonawczego.

Istnieją wątpliwości co do powtarzalności wykonanego układu sterowania podczas lotu rakiety w rzeczywistych warunkach. Bezpośrednią przyczyną powstałych wątpliwości są konsekwencje wyboru serwomechanizmów modelarskich powodujących duże opóźnienie czasowe układu wykonawczego. W związku z tym istnieje ograniczenie pasma przenoszenia obserwatora LESO, jak i wąski zakres doboru nastaw dla

sterownika pętli zewnętrznej zapewniających akceptowalną jakość sterowania. Może skutkować to małą skutecznością układu sterowania w przypadku szybkozmiennych zaburzeń. Zaproponowano sposoby rewizji systemu sterowania. Po pierwsze należałoby przeprojektować układ wykonawczy poruszający lotkami pod kątem zastosowania podzespołów generujących mniejsze opóźnienia czasowe. Innym ze sposobów zmniejszenia wpływu opóźnienia na jakość sterowania byłoby zastosowanie sterownika FFC-ADRC.

Końcowo planowane są testy układu sterowania podczas lotu rakiety w rzeczywistych warunkach. Po analizie danych, sprawdzeniu powtarzalności, finalnie byłaby możliwość prowadzenia pierwszych misji. Warto wspomnieć, że system sterowania prędkością rotacji zaprojektowany został dla lotów poddźwiękowych. W przyszłości można rozszerzyć projekt dla lotów przekraczający obszar poddźwiękowy.

Literatura

[1] CMISIS-DSP. https://www.keil.com/pack/doc/CMSIS/DSP/html/group__MatrixVectMult.html.

[2] CMSIS-RTOS API V2. https://www.keil.com/pack/doc/CMSIS/RTOS2/html/group__CMSIS__RTOS.html.

[3] Enkoder magnetyczny AM4096. https://www.rls.si/eng/fileuploader/download/download/?d=1&file=custom%2Fupload%2FAM4096D02_09_bookmark.pdf.

[4] FatFS. https://www.st.com/resource/en/user_manual/um1721-developing-applications-on-stm32cube-with-fatfs-stmicroelectronics.pdf.

[5] FreeRTOS. https://www.freertos.org/a00106.html.

[6] KiCAD. https://www.kicad.org/.

[7] OpenRocket — kod źródłowy. https://github.com/openrocket/openrocket.

[8] PUT Rocketlab. https://rocketlab.put.poznan.pl.

[9] Serwomechanizm PowerHD R12-S. http://www.chd.hk/UploadFiles/Att/2022030216415071.pdf.

[10] A. Brown, Jr., Martin L. Nason. Flight Investigation to evaluate the roll-rate stabilization system of the naval ordnance test station Slidewinder missle at mach numbers from 0.9 to 2.3. *Langley Aeronautical Laboratory, National Advisory Committee for Aeronautics*, 1954.

[11] James S. Barrowman. The practical calculation of the aerodynamic characteristics of slender finned vehicles. Master's thesis, The Catholic University of America, Waszyngton, USA, 1967.

[12] Karl J. Åström Björn Wittenmark. *Adaptive Control. Second edition.* Addison-Wesley, USA, 1995.

[13] John H. Blakelock. *Automatic control of aircraft and missiles. Second edition.* John Wiley & Sons Inc., Air Force Institute of Technology, USA, 1991.

[14] Bosch Sensortec GmbH. BMI08X Sensor API. https://github.com/BoschSensortec/BMI08x-Sensor-API.

[15] Bosch Sensortec GmbH. *BMI088: Datasheet*, styczeń 2022.

[16] Christopher L. Rumsey Christopher A. Eggert. CFD Study of NACA 0018 Airfoil with Flow Control. *National Aeronautics and Space Administration*, 2017.

[17] Michael V. Cook. *Flight dynamics principles.* Butterworth-Heinemann, MA, USA, 2007.

[18] Mark Davies. *The Standard Handbook for Aeronautical and Astronautical Engineers.* McGraw-Hill, Nowy Jork, USA, 2003.

[19] Stephen J. Dodds. *Feedback Control. Linear, nonlinear and robust techniques and design with industrial applications.* Springer, Londyn, UK, 2015.

[20] Dongyang Zhang, Xiaolan Yao, Qinghe Wu, and Zhuoyue Song. ADRC based control for a class of input time delay systems. *Journal of Systems Engineering and Electronics*, 28(6):1210–1220, 2017.

[21] Fen Wu, Andy Packard, Gary Balas. Systematic gain-scheduling control design: a missile autopilot example. *Asian Journal of Control*, 4(3):341–347, 2002.

[22] E.L. Fleeman. *Tactical missile design (Vol. 468).* American Institute of Aeronautics and Astronautics, Reston, USA, 2006.

[23] Zhiqiang Gao. Scaling and bandwidth-parameterization based controller tuning. *American Control Conference, Denver, USA*, pages 4989–4996, 2003.

[24] Zhiqiang Gao. Active disturbance rejection control: A paradigm shift in feedback control system design. *American Control Conference, Minneapolis, MN, USA*, 41(2):2399–2405, 2006.

[25] Jingqing Han. From PID to Active Disturbance Rejection Control. *IEEE Transactions on Industrial Electronics*, 56(3):900–906, 2009.

[26] Jinho Jang, Anuradha M. Annaswamy, and Eugene Lavretsky. Adaptive control of time-varying systems with gain-scheduling. *American Control Conference*, pages 3416–3421, 2008.

[27] Justin R. Smith, Jeff Cullina. CFD Assessment of Fin Manufacturing Defect to Set Fin Cant Angle and Achieve Nominal Roll Rate. *AIAA Applied Aerodynamics Conference, Honolulu, Hawaii*, 2011.

[28] James C. Manning. Some flight-induced environmental data obtained from two Sidewinder-Arcas sounding-rocket launches. *Langley Research Center, National Aeronautics and Space Administration*, 1971.

[29] Martin L. Nason, Clarence A. Brown, Jr., Rupert S. Rock. An evaluation of the roll-rate stabilization system of the Sidewinder missile at Mach numbers from 0.9 to 2.3. *Langley Aeronautical Laboratory, National Advisory Committee for Aeronautics*, 1955.

[30] Maciej Marcin Michałek. Modele obiektów sterowania do celów PUR - materiały wykładowe z przedmiotu Projektowanie układów regulacji, Politechnika Poznańska.

[31] Maciej Marcin Michałek. Sterowanie Active/Adaptive Disturbance Rejection Control (ADRC) - materiały wykładowe z przedmiotu Sterowanie adaptacyjne, Politechnika Poznańska.

[32] Maciej Marcin Michałek. Robust trajectory following without availability of the reference time-derivatives in the control scheme with active disturbance rejection. pages 1536–1541, 2016.

[33] Monolithic Power Systems Inc. *MP2639A 2-Cell Li-Ion or Li-Polymer Switching Charger Compatible with 5V Input and Integrated, Bidirectional Charge/Discharge with Cell Balance*, grudzień 2019.

[34] Sampo Niskanen. Development of an Open Source model rocket simulation software. Master's thesis, Helsinki University of Technology, Helsinki, Finlandia, 2009.

[35] Shen Zhao, Zhiqiang Gao. Modified active disturbance rejection control for time-delay systems. *ISA Transactions*, 53(4):882–888, 2013.

[36] George M. Siouris. *Missile guidance and control systems.* Springer, OH, USA, 2004.

[37] O.J.M. Smith. Closed control of loops with dead time. *Chemical Engineering Progress*, 53:217–219, 1957.

[38] STMicroelectronics N.V. STM32CubeIDE. `https://www.st.com/en/development-tools/stm32cubeide.html`.

[39] STMicroelectronics N.V. *STM32F412xE STM32F412xG Datasheet*, grudzień 2017.

[40] Texas Instruments Inc. *TLV766-Q1 500-mA, 16-V Linear Voltage Regulator*, grudzień 2020.

[41] Natalia Wiśniewska Tomasz Krakowski. Wstępna analiza aktywnego aerodynamicznego sterowania prędkości rotacji dla rakiet poddźwiękowych. *Nauka dla obronności. Bezpieczeństwo infrastruktury krytycznej.*, 2(2):305–318, 2022.

[42] Departament Obrony USA. *Design of aerodynamically stabilized free rockets.* 1990.

[43] Wenchao Xue, Yi Huang. On Frequency-domain Analysis of ADRC for Uncertain System. *American Control Conference, Waszyngton, DC, USA*, pages 6637–6642, 2013.

[44] Jeff S. Shamma Wilson J. Rugh. Research on gain scheduling. *Automatica*, 36:1401–1425, 2000.

[45] Winbond Electronics Corp. *3V 64M-bit serial flash memory with dual, quad SPI*, marzec 2018.

[46] Jingqing Han Zhiqiang Gao, Yi Huang. An alternative paradigm for control system design. *IEEE Conference on Decision and Control*, 2001.

[47] Zhiqiang Gao, Aaron Radke, Robert Miklosovic. Discrete Implementation and Generalization of the Extended State Observer. *American Control Conference (ACC)*, 2006.

Instytut Automatyki i Robotyki, Wydział Automatyki, Robotyki i Elektrotechniki
Politechnika Poznańska