Projekt z Teorii Sterowania

AiR studia II stopnia

Temat: Lewo-niezmienniczy rozszerzony filtr Kalmana

Urszula Mayer-Gawron, Piotr Majorczyk

22 września 2016

Katedra Sterowania i Inżynierii Systemów, Politechnika Poznańska

Streszczenie

Projekt traktuje o lewo-niezmienniczym filtrze Kalmana. Zrealizowany został na podstawie pracy naukowej a do jego implemenmacji użyto środowiska numerycznego Mtalab/Simulink. W efekcie realizacji projektu uzyskano opis ruchu obrotowego ciała sztywnego za pomocą kwaternionów oraz udało się odszumić sygnał zaszumiony.

1 Wprowadzenie i cel projektu

Celem projektu było odtworzenie tytularnego rozszerzonego filtru Kalmana na podstawie pracy naukowej Silvere Bonnabel opublikowanej w 2007 roku. Projekt był wykonywany na zajęciach laboratoryjnych z przedmiotu Teoria Sterowania w ramach których udostępnione zostało środowisko Matlab i Simulink. W celu realizacji zadania należało zapoznać się z pojęciem kwaternionów oraz działań na nich.

1.1 Kwaterniony

Kwaterniony to obiekty matematyczne opisane czterema liczbami. Znajdują one zastosowanie w reprezentowaniu rotacji i orientacji obiektów w przestrzeni trójwymiarowej. Na potrzeby projektu przyjęto reprezentację kwaternionu jako złożenia liczby skalarnej p oraz wektora $p \in \mathbf{R}^3$ .Reprezentacja ta przedstawiona jest na równaniu [1]

$$p = \begin{pmatrix} p_0 \\ \vec{p} \end{pmatrix} \tag{1}$$

2 Przedstawienie obiektu

Obiektem naszych badań do którego stworzono filtr Kalmana było „latające ciało sztywne„ (które można uznać za dron) opisane równaniem kinematycznym [2]

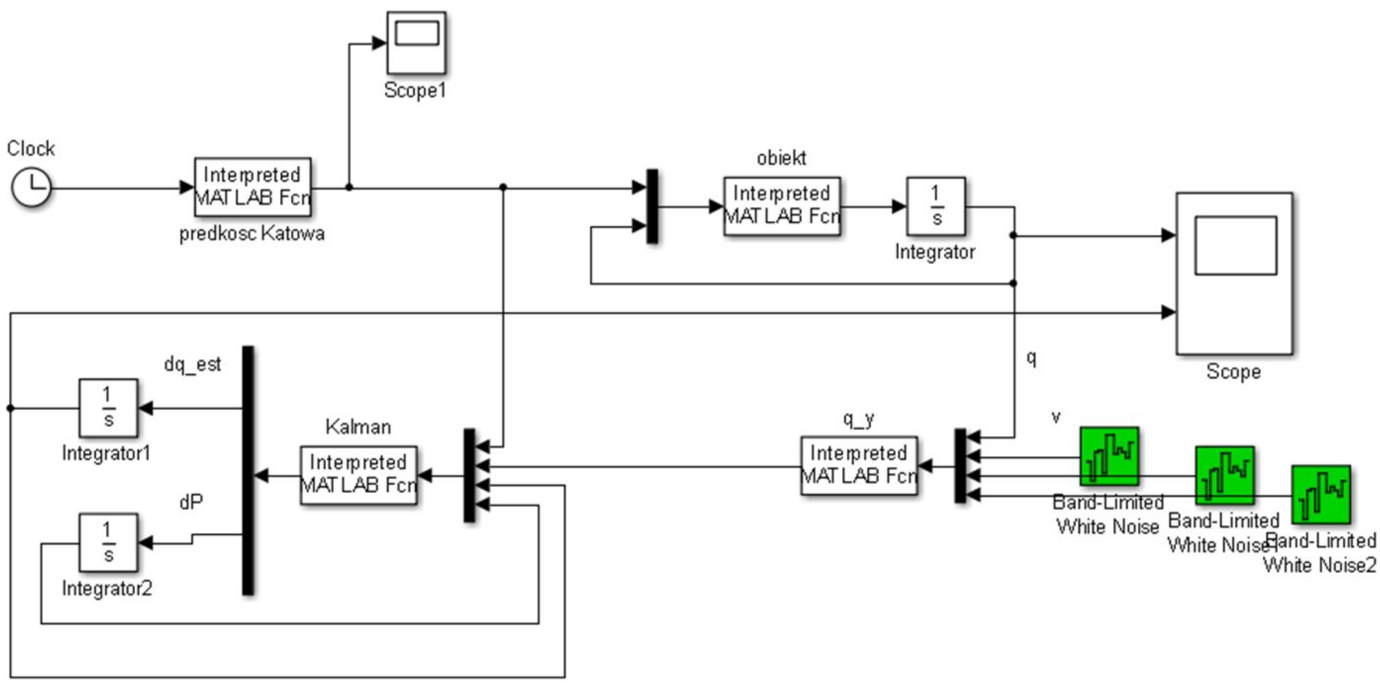
$$\dot{q} = \frac{1}{2} q * \omega + q * M w \tag{2}$$

Gdzie:
q – kwaternion przedstawiający rotację obiektu w przestrzeni 3D
ω – chwilowa prędkość kątowa mierzona przez żyroskopy
M – macierz stałych rozmiaru 3x3
w – Gaussowski szum biały

Człon drugi został pominięty ze względu na przyjęcie założenia o braku szumu na obiekcie.Implementacja wzoru wykonywana jest przez funkcję obiekt.m

3 Algorytm sterowania

Celem opisanego poniżej algorytmu było odszumianie sygnału określającego prędkość kątową obiektu. Zostało to zrealizowane przy pomocy obserwatora jakim jest rozszerzony filtr Kalmana pobierający na wejście zaszumione sygnały i podający na wyjściu estymatę sygnału pozbawionego szumów. W pierwszej kolejności trzy składowe prędkości kątowej zaszumiane były trzema losowymi i niejednakowymi szumami białymi. Tak przygotowane dane były podawane na wejście filtru Kalmana działającego w pętli sprzężenia zwrotnego z członem całkującym. Efektem działania filtru było podanie estymowanej wartości pochodnej kwaternionu przedstawiającego rotację obiektu w przestrzeni 3D. Po scałkowaniu uzyskiwano szukaną wartość kwaternionu. Poszczególne etapy tego algorytmu zostały opisane w niniejszym rozdziale.



Rysunek 1 Schemat programu w Simulinku

3.1 Inicjalizacja stałych i zmiennych

Przed uruchomieniem symulacji należy zdeklarować początkowe wartości całkowań, estymowanego kwaternionu, macierze stałych kowariancji szumów (M,N) oraz macierz wzmocnienia w filtrze Kalmana P(t). Czynności te wykonywane są poprzez uruchomienie pliku init.m. Przyjęto następujące wartości początkowe:

$$q(0) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\hat{q}(0) = \begin{bmatrix} \cos(\pi/3) \\ \frac{\sin(\pi/3)}{\sqrt{3}} \\ \frac{-\sin(\pi/3)}{\sqrt{3}} \\ \frac{\sin(\pi/3)}{\sqrt{3}} \end{bmatrix}$$

$$M = 0{,}5 I_3$$

$$N = 0{,}2 I_3$$

$$P(0) = 0{,}1 I_3$$

3.2 Prędkość kątowa

Ruch obiektu reprezentowana jest za pomocą 3 wartości prędkości obrotowych w 3 osiach. Dla uproszczenia przyjęto, że obiekt obraca się tylko wokół jednej osi. W bloku opis prędkości dokonywany jest przy pomocy wzoru [3]

$$\omega_1 = A_1 \sin(t + \phi_1) \tag{3}$$

3.3 Działania na kwaternionach

3.3.1 Mnożenie kwaternionów

Pomimo podobieństwa kwaterniony nie mogą być traktowane jak wektory czteroelementowe, gdyż obowiązują je inne zasady wykonywania działań. W niniejszym projekcie wykorzystywano mnożenie kwaternionów (wzór [4]) które zostało zaimplementowane jako funkcja multiply.

$$p * q = \begin{pmatrix} p_0 q_0 - \vec{p} \cdot \vec{q} \\ p_0 \vec{q} + q_0 \vec{p} + \vec{p} \times \vec{q} \end{pmatrix} \tag{4}$$

3.3.2 Odwrotność kwaternionów

Na potrzeby projektu konieczne było zdefiniowanie działania obliczającego odwrotność danego kwaternionu. Wychodzimy z założenia, że:

$$p * p^{-1} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Jeśli:

$$p = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

$$p^{-1} = q = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

to powstaje nam równanie:

$$p * q = \begin{pmatrix} p_0 q_0 - \vec{p} \cdot \vec{q} \\ p_0 \vec{q} + q_0 \vec{p} + \vec{p} \times \vec{q} \end{pmatrix} = \begin{bmatrix} p_0 \\ p_1 \\ p_2 \\ p_3 \end{bmatrix}$$

Wykorzystując równania na iloczyn skalarny i wektorowy otrzymujemy następujące równania:

$$\begin{bmatrix} p_0 q_0 - p_1 q_1 - p_2 q_2 - p_3 q_3 \\ p_0 q_1 + p_1 q_0 + p_2 q_3 - p_3 q_2 \\ p_0 q_2 + p_2 q_0 + p_3 q_1 - p_1 q_3 \\ p_0 q_3 + p_3 q_0 + p_1 q_2 - p_2 q_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Otrzymaliśmy następującą postać:

$$AX = B$$

Macierz X jest poszukiwanym przez nas kwaternionem odwrotnym. W związku z tym:

$$X = A^{-1} B$$

Rozwiązując powyższe równanie w Matlabie otrzymujemy:

$$X = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} = \begin{bmatrix} \frac{p_0}{{p_0}^2 + {p_1}^2 + {p_2}^2 + {p_3}^2} \\ \frac{-p_1}{{p_0}^2 + {p_1}^2 + {p_2}^2 + {p_3}^2} \\ \frac{-p_2}{{p_0}^2 + {p_1}^2 + {p_2}^2 + {p_3}^2} \\ \frac{-p_3}{{p_0}^2 + {p_1}^2 + {p_2}^2 + {p_3}^2} \end{bmatrix}$$

W przypadku gdy kwaternion jest zerowy jest przypisywana mu zerowa wartość gdyż nie istnieje odwrotność z zerowego kwaterniona. Algorytm ten zrealizowany jest w funkcji quaternionInverse.m

3.4 Rozszerzony filtr Kalmana

Równania na rozszerzony filtr Kalmana zostały wyprowadzone poprzez przekształcenie klasycznych wzorów na rozszerzony filtr Kalmana z uwzględnieniem specyfiki kwaternionów.

$$K(t) = P(t)(NN^T)^{-1}$$
$$\dot{P}(t) = A(t)P(t) + P(t)A^T(t) + MM^T - P(t)(NN^T)^{-1}P(t)$$
$$\frac{d}{dt}\hat{q} = \frac{1}{2}\hat{q} * w + \hat{q} * K(t)(\hat{q}^{-1} * q_y - 1)$$

gdzie: $A(t) * \eta = \frac{1}{2}(\eta * \omega - \omega * \eta)$

$$\omega = \begin{pmatrix} 0 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} \eta = \begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \eta_4 \end{pmatrix}$$

$$\eta * \omega = \begin{pmatrix} 0 * \eta_1 - \omega_1\eta_2 - \omega_2\eta_3 - \omega_3\eta_4 \\ \eta_1 \cdot \overline{\omega} + \omega_2\eta_4 - \eta_3\omega_3 \\ \eta_1 \cdot \overline{\omega} + \eta_2\omega_3 - \omega_1\eta_4 \\ \eta_1 \cdot \overline{\omega} + \omega_1\eta_3 - \omega_2\eta_2 \end{pmatrix}$$

$$\omega * \eta = \begin{pmatrix} 0 * \eta_1 - \omega_1\eta_2 - \omega_2\eta_3 - \omega_3\eta_4 \\ \eta_1 \cdot \overline{\omega} - \omega_2\eta_4 + \eta_3\omega_3 \\ \eta_1 \cdot \overline{\omega} - \eta_2\omega_3 + \omega_1\eta_4 \\ \eta_1 \cdot \overline{\omega} - \omega_1\eta_3 + \omega_2\eta_2 \end{pmatrix}$$

Więc:

$$\frac{1}{2}(\eta * \omega - \omega * \eta) = \begin{pmatrix} 0 \\ \omega_2\eta_4 - \eta_3\omega_3 \\ \eta_2\omega_3 - \omega_1\eta_4 \\ \omega_1\eta_3 - \omega_2\eta_2 \end{pmatrix} = A\eta = A\eta = \begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \eta_4 \end{pmatrix}$$

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & \omega_3 & -\omega_2 \\ 0 & -\omega_3 & 0 & \omega_1 \\ 0 & \omega_2 & -\omega_1 & 0 \end{bmatrix}$$

Lecz w obliczeniach wykorzystujemy:

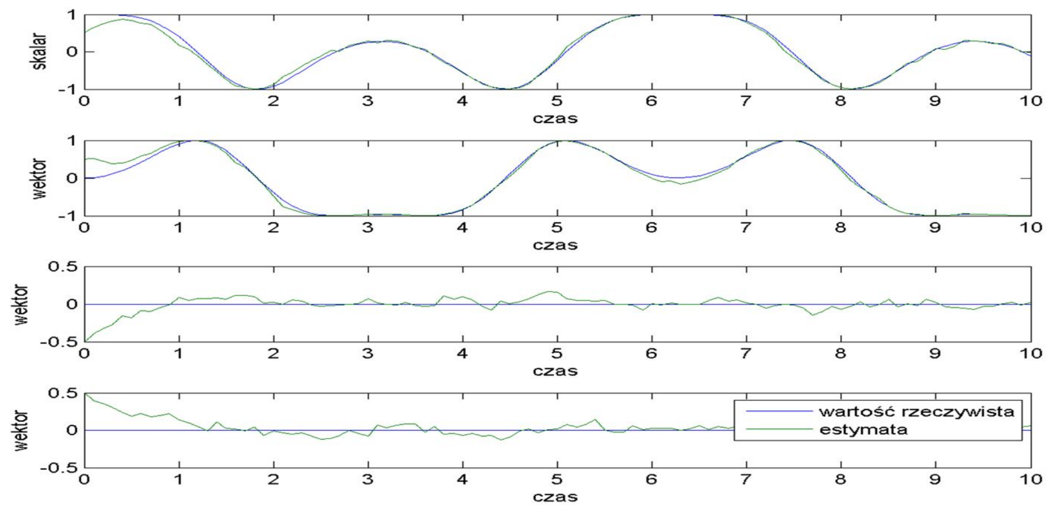
$$A = \begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix}$$

Wyżej opisane działania realizowane są przez funkcję Kalman.m Dodanie szumu do sygnału wejściowego dla filtru Kalmana odbyło się zgodnie ze wzorem [5]

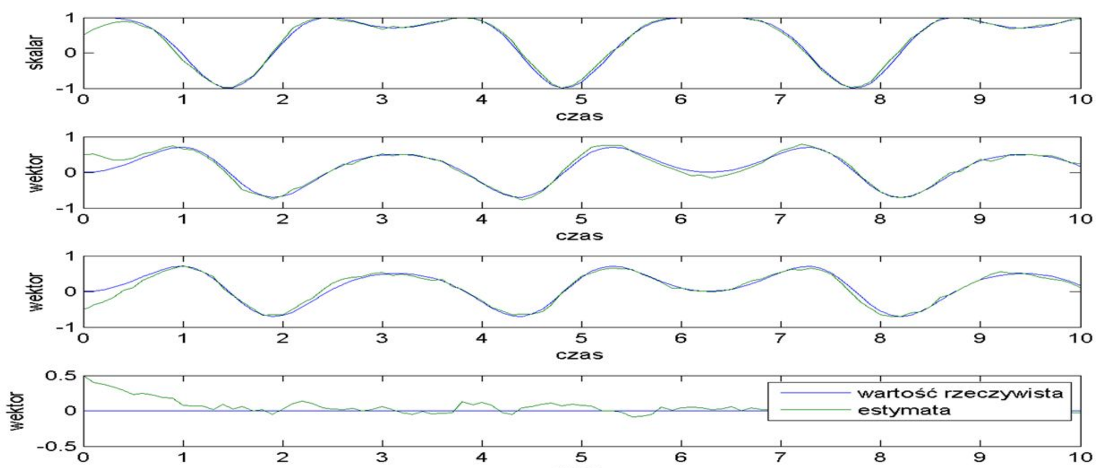
$$q_y = q + q * Nv \tag{5}$$

4 Przedstawienie i dyskusja wyników

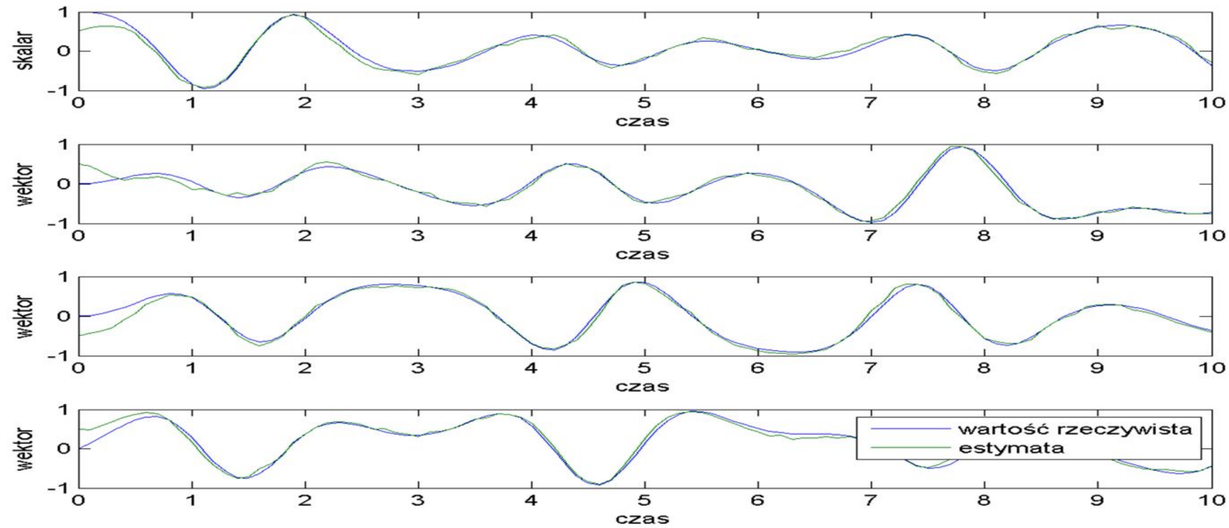
W efekcie uruchomienia wyżej opisanego programu uzyskano wyniki zaprezentowane na rysunku [2]. Gdzie kolorem żółtym oznaczono przebieg rzeczywisty a kolorem magenta przebieg estymowany. Patrząc od góry wykresy przedstawiają poszczególne składowe kwaternionu tj. wartość skalarna oraz trzy składowe wektora. Jak widać estymata jest zbliżona do wartości rzeczywistych a spora początkowa różnica w wartościach wynika z przyjętych za opracowaniem wartości początkowych dla członów całkujących. Zgodnie z przewidywaniami zmienia się tylko pierwsza składowa wektora obrotu gdyż obrót ten występuje tylko wokół jednej osi. Wartość skalarna jest „dopełnieniem„ wartości kwaternionu aby zachowana była właściwość kwaternionu zgodnie z którą jego norma jest zawsze równa 1.Na rysunkach 3 i 4 przedstawiono działanie programu dla obrotu wokół 2 oraz 3 osi, dla różnych przesunięć między fazami. Jak widać również w tych przypadkach estymata jest zbliżona do wartości rzeczywistej co pozwala wnioskować o prawidłowym działaniu programu.



Rysunek 2 Porównanie wartości rzeczywistej z jej estymatą dla obrotu wokół 1 osi



Rysunek 3 Porównanie wartości rzeczywistej z jej estymatą dla obrotu wokół 2 osi



Rysunek 4 Porównanie wartości rzeczywistej z jej estymatą dla obrotu wokół 3 osi

5 Podsumowanie

Zdaniem autorów udało się osiągnąć opisany we wstępie cel. Przygotowano i opracowano model rozszerzonego filtru Kalmana wykorzystującego kwaterniony w celu estymowania orientacji obiektu.

Głównymi problemami jakie pojawimy się podczas realizacji projektu były:

- Niekonsekwencja w stosowaniu oznaczeń w materiale źródłowym
- Wybrakowany opis jednej z wielkości
- Konieczność stosowania nieznanych wcześniej pojęć matematycznych (kwaterniony i działania na nich)
- Wysokiego stopnia abstrakcji
- Konieczność wyprowadzania niektórych działań i ręczne sprawdzanie ich poprawności

Zdaniem autorów projektu temat był ciekawy i rozwijający lecz trudności w zrozumieniu działania algorytmu mogłyby uniemożliwić zaimplementowanie algorytmu w praktycznym zastosowaniu.

Bibliografia

1. Silvere Bonnabel, Left-invariant extended Kalman filter and attitude estimation, IEEE CDC, New Orleans, USA, 2007
2. http://www.mathworks.com/help/

Listingi

init.m

```
clear all

q1=[1; 0; 0; 0];
%q0_est=q1;
q0_est=[cos(pi/3);sin(pi/3)/(3)^0.5;-sin(pi/3)/(3)^0.5;sin(pi/3)/(3)^0.5];

global N
global M

% Macierze kowariancji szumow
M=0.5.*eye(3,3);
N=0.2.*eye(3,3);
P_0=0.1.*eye(3,3);
P_0=reshape(P_0,9,1);
```

Kalman.m

```
function [wynik]=Kalman(vals)
% dane
omega=vals(1:4);
q_y=vals(5:8);
q_est=vals(9:12);
P=vals(13:21);
P=reshape(P,3,3);

global N
global M

A=[0 -omega(3) omega(2); omega(3) 0 -omega(1);-omega(2) omega(1) 0];

K=P*inv(N*N');

q_inv=quaternionInverse(q_est);
q1=multiply(q_inv,q_y) - [1;0;0;0];
q2=K*q1(2:4);

dq_est=0.5.*(multiply(q_est,omega))+multiply(q_est,[0;q2]);
dP=A*P+P*A'+M*M'-P*inv(N*N')*P;

wynik=[dq_est;reshape(dP,9,1)];
end
```

kwaterniony.m

```
function [wynik]=Kalman(vals)
```

```
% dane
omega=vals(1:4);
q_y=vals(5:8);
q_est=vals(9:12);
P=vals(13:21);
P=reshape(P,3,3);

global N
global M

A=[0 -omega(3) omega(2); omega(3) 0 -omega(1);-omega(2) omega(1) 0];

K=P*inv(N*N');

q_inv=quaternionInverse(q_est);
q1=multiply(q_inv,q_y) -[1;0;0;0];
q2=K*q1(2:4);

dq_est=0.5.*(multiply(q_est,omega))+multiply(q_est,[0;q2]);
dP=A*P+P*A'+M*M'-P*inv(N*N')*P;

wynik=[dq_est;reshape(dP,9,1)];
end
```

multiply.m

```
function [result]=multiply(p,q)
%% MNOZENIE QWATERNIONOW

%% Sciaganie danych

% Kwaterion p
p_skalar=p(1);
p_wektor=p(2:4);

% Kwaterion q
q_skalar=q(1);
q_wektor=q(2:4);

%% Obliczanie
wynik_Skalar=p_skalar*q_skalar-dot(p_wektor,q_wektor);
wynik_Wektor=p_skalar.*q_wektor+q_skalar.*p_wektor+cross(p_wektor,q_wektor);

%% Wynik
result=[wynik_Skalar; wynik_Wektor];
end
```

obiekt.m

```
function [dq]=obiekt(vals)

omega=vals(1:4);
q1=vals(5:8);

dq = 1/2.*multiply(q1,omega);

end
```

quaternionInverse.m

```
function [p_inv]=quaternionInverse(p)

    if (p(1)==0 && p(2)==0 && p(3)==0 && p(4)==0)

        p_inv=[0;0;0;0];

    else
        mianownik=(p(1)^2+p(2)^2+p(3)^2+p(4)^2);
        p_inv=[p(1)/mianownik; -p(2)/mianownik;...
            -p(3)/mianownik;-p(4)/mianownik];
    end
    p_inv;
end
```