

POLITECHNIKA POZNAŃSKA

WYDZIAŁ AUTOMATYKI, ROBOTYKI I ELEKTROTECHNIKI

INSTYTUT ROBOTYKI I INTELIGENCJI MASZYNOWEJ

ZAKŁAD STEROWANIA I ELEKTRONIKI PRZEMYSŁOWEJ



PRACA DYPLOMOWA MAGISTERSKA

MIKROPROCESOROWY INKLINOMETR  
OPRACOWANY Z WYKORZYSTANIEM  
AKCELEROMETRU MEMS

ALIAKSANDR HILEVICH

PROMOTOR:

DR INŻ. DOMINIK ŁUCZAK

DOMINIK.LUCZAK@PUT.POZNAN.PL

POTWIERDZAM PRZYJĘCIE PRACY

.....  
DATA I PODPIS PROMOTORA PRACY

POZNAŃ 2021

## **Karta opisu pracy (otrzymana z Instytutu)**

**Po wydruku, lecz przed bindowaniem podmień bieżącą stronę na kartę otrzymaną z Instytutu.**

**W wersji „do dziekanatu” wstaw oryginał karty tematu. Natomiast w wersjach dla promotora i recenzenta wstaw kopię karty tematu.**

**Proszę dodatkowo przygotować rozszerzone streszczenie promocyjne pracy dyplomowej w celu umieszczenia na stronie ZSEP (format docx/doc). Przykładowe rozszerzone streszczenia znajdują się pod adresem <http://zsep.cie.put.poznan.pl/index.php/dydaktyka/prace-dyplomowe> . W streszczeniu proszę umieścić cel pracy i zakres przeprowadzonych prac. Proszę pamiętać o dodaniu ilustracji uzyskanych wyników.**

## STRESZCZENIE

Praca dotyczy badania opracowania mikroprocesowego inklinometru z czujnikiem MEMS oraz badania algorytmów filtracji surowych danych.

Do badań wybrano następujące algorytmy filtrowania danych:

- Filtr komplementarny
- Filtr Madgwicka
- Filtr Kalmana

W pierwszej części pracy się opowiada o opracowaniu algorytmów w Matlab i następnie weryfikacji ich pracy na danych syntetycznych.

Głównym punktem pracy jest implementacja algorytmów w układzie rzeczywistym opartym na płycie NUCLEO-F756GZ z mikrokontrolerem STM32F7. Dane surowe zbierane za pomocą czujnika MEMS LSM6DSOX.

## ABSTRACT

(j. angielski)

## Spis treści

1	Wstęp.....	6
2	Wyznaczenie kątów nachylenia. Filtry.....	7
2.1	Filtr komplementarny .....	9
2.2	Filtr Madgwicka .....	10
2.3	Filtr Kalmana.....	11
3	Opracowanie algorytmów w Matlab .....	<b>Ошибка! Закладка не определена.</b>
3.1	Weryfikacja pracy algorytmów na danych syntetycznych.....	13
4	Część sprzętowa .....	20
4.1	Mikrokontroler STM32F7 .....	20
4.2	Czujnik MEMS LSM6DOX.....	20
4.3	Oprogramowanie .....	20
4.4	Zebranie surowych danych z czujnika .....	20
4.5	Weryfikacja pracy algorytmów z wykorzystaniem surowych danych.....	20
5	Implementacja algorytmów w układzie rzeczywistym .....	20
6	Wyniki .....	20
	Literatura .....	21

# 1 WSTĘP

Czujniki mikroelektromechaniczne (MEMS, ang. Microelectromechanical systems) prędkości kątowej (żyroskopy) i przyspieszenia (akcelerometry) stanowią jeden z aktywnie rozwijających się obszarów technologii mikrosystemów. Czujniki MEMS to zintegrowane systemy o rozmiarach od kilku mikrometrów do kilku milimetrów, które łączą komponenty mechaniczne i elektroniczne. Zasada działania takich czujników polega na zamianie na sygnał elektryczny pojemności różnicowej utworzonej przez ruchome i nieruchome mikromechaniczne płytki grzebieniowe. Zmiana pojemności pod wpływem przyspieszenia liniowego (w akcelerometrach) lub siły Coriolisa (w żyroskopach) umożliwia oszacowanie wartości amplitudy tych wpływów.

Pomimo niewielkich rozmiarów, wagi i zużycia energii, praktyczne zastosowanie czujników MEMS w systemach orientacji i nawigacji typu strap-down jest ograniczone przez niską czułość, niestabilność współczynnika skali oraz wysoki poziom szumu sygnału wyjściowego w porównaniu z innymi typami żyroskopów i akcelerometrów. Spośród istniejących typów żyroskopów MEMS mają największy dryf (do 300 ... 1000 °), co nie pozwala na ich użycie bez okresowego filtrowania współrzędnych kątowych.

Celem pracy jest porównanie algorytmów filtrowania sygnałów z inercyjnych czujników MEMS w celu wyznaczenia współrzędnych kątowych obiektów obracających się w trzech płaszczyznach.

Do badań wybrano następujące algorytmy filtrowania danych:

- Filtr komplementarny
- Filtr Madgwicka
- Filtr Kalmana

Układ rzeczywisty jest oparty na płytce NUCELO-F746GZ z mikrokontrolerem STM32F7. Do zbierania danych surowych będzie wykorzystany trzyosiowy czujnik MEMS LSM6DSOX.

## 2 WYZNACZENIE KĄTÓW NACHYLENIA. FILTRY

### Żyroskop

Urządzenie zdolne do reagowania na zmiany kątów orientacji obiektu, na którym jest zainstalowane, względem bezwładnościowego układu odniesienia.

W przypadku żyroskopu kąt nachylenia można łatwo obliczyć poprzez dyskretną integrację jego prędkości obrotowej.

$$\alpha(t) = \alpha(t - 1) + G_x * dt \quad (2.1)$$

$$\beta(t) = \beta(t - 1) + G_y * dt \quad (2.2)$$

$$\gamma(t) = \gamma(t - 1) + G_z * dt \quad (2.3)$$

gdzie:

$\alpha(t)$  - kąt nachylenia w osi X

$\beta(t)$  - kąt nachylenia w osi Y

$\gamma(t)$  - kąt nachylenia w osi Z

$\alpha(t - 1)$  - kąt nachylenia w osi X w poprzednim momencie

$\beta(t - 1)$  - kąt nachylenia w osi Y w poprzednim momencie

$\gamma(t - 1)$  - kąt nachylenia w osi Z w poprzednim momencie

$G_x$  - prędkość obrotowa wokół osi X

$G_y$  - prędkość obrotowa wokół osi Y

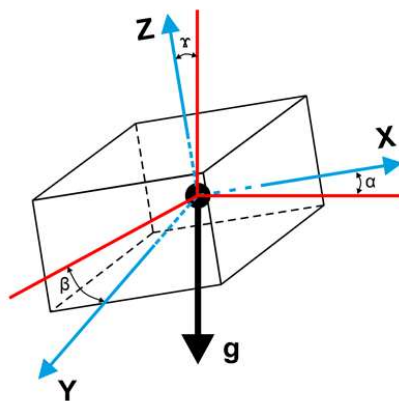
$G_z$  - prędkość obrotowa wokół osi Z

$dt$  – czas próbkowania

Żyroskop MEMS ma jedną wadę, którą nazywa się dryfem zerowym. Istota tej wady sprowadza się do tego, że gdy obrót żyroskopu się zatrzyma, nadal będzie pokazywał wartość inną niż zero. Inną wadą jest stosowanie procedury całkowania dyskretnego, która ze swej natury daje niedokładny wynik, co prowadzi do stopniowej kumulacji błędów obliczeniowych kątów ze względu na ograniczoną dokładność zmiennych mikrokontrolera.

## Akcelerometr

Akcelerometr to urządzenie, które mierzy przyspieszenie pod wpływem sił zewnętrznych. Za pomocą tego czujnika można również obliczyć kąty.



Rys.1 Orientacja obiektu w 3 osiach

$$A_x = g * \sin(\alpha) \rightarrow \alpha = \arcsin\left(\frac{A_x}{g}\right) \quad (2.4)$$

$$A_y = g * \sin(\beta) \rightarrow \beta = \arcsin\left(\frac{A_y}{g}\right) \quad (2.5)$$

$$A_z = g * \sin(\gamma) \rightarrow \gamma = \arcsin\left(\frac{A_z}{g}\right) \quad (2.6)$$

$g$  - przyspieszenie grawitacyjne

$A_x$  - rzut przyspieszenia grawitacyjnego na oś akcelerometru X.

$A_y$  - rzut przyspieszenia grawitacyjnego na oś akcelerometru Y.

$A_z$  - rzut przyspieszenia grawitacyjnego na oś akcelerometru Z.

Przy tej metodzie obliczeń występuje problem z pełnym pokryciem kąta 360, sygnał wyjściowy akcelerometru jest taki sam dla kątów  $\alpha$  i  $\pi - \alpha$ . Dlatego zakres pomiarowy wynosi 180 stopni. Dlatego zaleca się stosowanie stosunku odczytów akcelerometru do i obliczanie kąta pochylenia za pomocą funkcji arctangens.

$$\alpha = \arctan\left(\frac{A_x}{\sqrt{A_y^2 + A_z^2}}\right) \quad (2.7)$$

$$\beta = \arctan\left(\frac{A_y}{\sqrt{A_x^2 + A_z^2}}\right) \quad (2.8)$$



$$\gamma = \arctan\left(\frac{A_z}{\sqrt{A_x^2 + A_y^2}}\right) \quad (2.9)$$

$A_x$ ,  $A_y$ ,  $A_z$  to wartości zwracane przez czujnik.

Jak w przypadku każdego systemu elektromechanicznego, czynniki takie jak hałas, wibracje, przemieszczenie, dryf temperaturowy itp. wpływają na jakość pomiarów akcelerometru. Szum akcelerometru jest spowodowany szumem elektroniki, wahaniami napięcia, błędami ADC. Naturalny biały szum jest reprezentowany przez gęstość szumu w specyfikacji czujnika. Nawet jeśli nie ma własnego szumu, czujnik może doświadczyć zewnętrznych wibracji, które psują sygnał. Dokładność pomiaru można poprawić, uśredniając wynik dla  $N$  próbek, ale zwiększenie liczby próbek zwiększy opóźnienie pomiaru.

## 2.1 FILTR KOMPLEMENTARNY

Filtr ten jest nakładany na dwie wielkości mierzone przez różne czujniki i koryguje jedną z nich tak, aby powoli zbliżała się do drugiej. W cyklu pomiarowym filtr realizowany w następujący sposób.

$$\alpha(t) = K * (\alpha(t-1) + G_x * dt) + (1 - K) * A_x \quad (2.10)$$

$$\beta(t) = K * (\beta(t-1) + G_y * dt) + (1 - K) * A_y \quad (2.11)$$

$$\gamma(t) = K * (\gamma(t-1) + G_z * dt) + (1 - K) * A_z \quad (2.12)$$

Jak widać ze wzorów, całkowity kąt nachylenia jest sumą wartości zintegrowanego żyroskopu i chwilowej wartości akcelerometru. Głównym zadaniem filtra komplementarnego jest zniwelowanie dryftu zerowego żyroskopu oraz dyskretnych błędów całkowania za pomocą wskazań akcelerometru. Na każdym kroku całkowania (kroku cyklu kontrolnego) korygujemy całkę kąta pochylenia za pomocą wskazań akcelerometru. Siła tej korekty jest określona przez współczynnik filtra  $K$ . Wybór współczynnika  $K$  zależy od wielkości dryftu zerowego żyroskopu, od szybkości akumulacji błędów obliczeniowych oraz od warunków użytkowania maszyna. Np. zbyt duża wartość  $K$  spowoduje, że na wynik działania filtra silnie wpłyną zewnętrzne wibracje. Zbyt mała wartość  $K$  może okazać się niewystarczająca do wyeliminowania dryftu zera żyroskopu. Z reguły współczynnik filtra komplementarnego wybiera się za pomocą wzoru 2.13.

$$K = \frac{\tau}{\tau + dt} \quad (2.13)$$

Gdzie:

$\tau$  - stała filtra czasu

## 2.2 FILTR MADGWICKA

Filtr Madgwicka używa kwaternionów. Kwaterniony to jest czterowymiarowa liczba zespolona, której można użyć do przedstawienia współrzędnych w przestrzeni trójwymiarowej.

## 2.3 FILTR KALMANA. DYSKRETNY ALGORYTM.

Filtr Kalmana wykorzystuje dynamiczny model systemu, znane działania kontrolne i pomiary, aby utworzyć optymalne oszacowanie stanu. Algorytm składa się z dwóch powtarzających się faz: przewidywania (ang. Predict) i korekcji (ang. Correct). W pierwszej kolejności obliczana jest prognoza stanu w następnej chwili czasowej (z uwzględnieniem niedokładności ich pomiaru). Z drugiej strony nowa informacja z czujnika koryguje przewidywaną wartość (uwzględniając również niedokładność i szum tej informacji):

Do opisu zarówno procesu jak i systemu pomiarowego stosują się modele matematyczne.

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \quad (2.14)$$

$$z_k = Hx_k + v_k \quad (2.15)$$

Pierwsze równanie to jest model procesu, drugie – model pomiaru. Gdzie:

$x_k$  – wartość stanu.

$x_{k-1}$  – wartość stanu w poprzednim kroku.

$A$  – macierz stanu

$u_{k-1}$  – wartość sterowania w poprzednim kroku.

$B$  – macierz wejścia (sterowania).

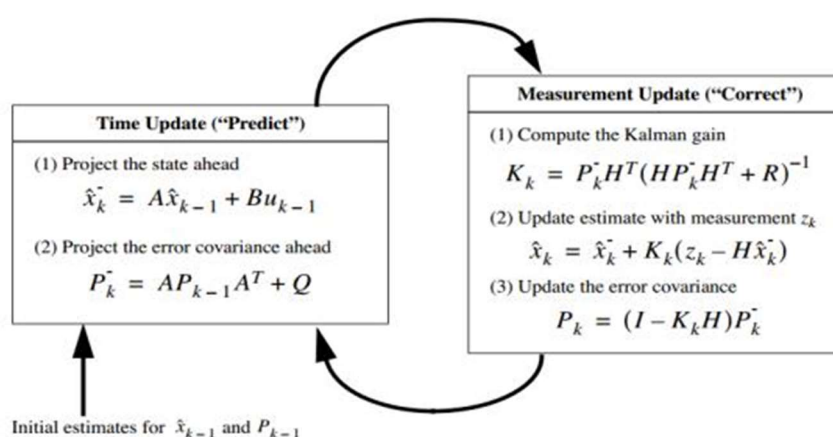
$z_k$  – wartość pomiaru.

$H$  – macierz wyjścia.

$w_{k-1}$  – szum pomiaru.

$v_k$  – zakłócenia.

W trakcie predykcji, bazując na stanie z poprzedniego kroku wyznacza się estymowana wartość  $\hat{x}(k)$  oraz jego kowariancję i są to wartości a priori. Pomiar  $z$  w drugiej fazie jest pewną formą sprzężenia zwrotnego. Na jego podstawie dokonuje się wyznaczenia wartości a posteriori dla stanu i jego kowariancji.



Rys.2 Dyskretny algorytm filtracji Kalmana.

Równania pierwszej fazy:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \quad (2.16)$$

$$P_k^- = AP_{k-1}A^T + Q \quad (2.17)$$

Gdzie  $\hat{x}_k^-$  i  $P_k^-$  - to prognozowane wartości stanu i kowariancji a priori,  $\hat{x}_{k-1}$  i  $P_{k-1}$  - to optymalne szacowane wartości a posteriori wykonane w poprzednim kroku.

Druga faza się zaczyna z wyliczenia wzmocnienia Kalmana, to wzmocnienie pokazując z jaką wagą wpłynie faza korekcji na estymowany czas.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \quad (2.18)$$

Następnie, na podstawie wszystkich dotychczasowych pomiarach, obliczamy optymalne skorygowanie prognozy w czasie  $k$ .

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \quad (2.19)$$

Gdzie  $z_k$  to pomiar, a różnica  $z_k - H\hat{x}_k^-$  - innowacja pomiarowa (ang. measurement innovation).

Na końcu drugiej fazy korygujemy macierz kowariancji.

$$P_k = (I - K_k H)P_k^- \quad (2.20)$$

Gdzie  $I$  to macierz jednostkowa.

### 3 IMPLEMENTACJA ALGORYTMÓW W MATLAB. WERYFIKACJA NA DANYCH SYNTETYCZNYCH.

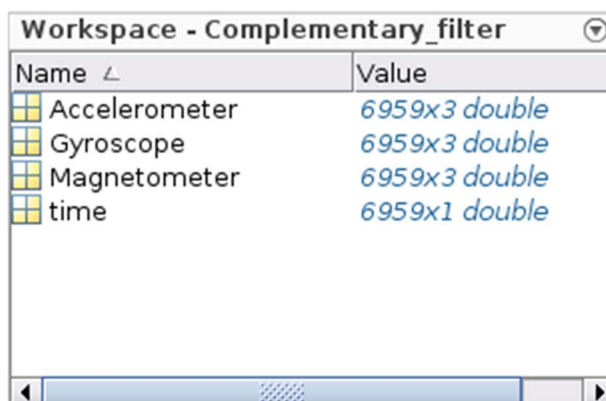
W tym rozdziale przeprowadzona implementacja algorytmów filtracji, które były opisane w poprzednim rozdziale, oraz sprawdzenie ich pracy na danych syntetycznych. Matlab jest używany w wersji R2020b.

Dane syntetyczne zawierają wartości prędkości kątowej w 3 osiach XYZ oraz rzuty przyspieszenia grawitacyjnego na osi XYZ. Dane były pobrane ze źródła [5].

*Listing 1 Ładowanie danych i wyświetlanie ich*

```
load('ExampleData.mat'); % ładowanie danych
figure('Name', 'Dane syntetyczne');
axis(1) = subplot(2,1,1);
hold on;
plot(time, Gyroscope(:,1), 'r'); % prędkość kątowa X
plot(time, Gyroscope(:,2), 'g'); % prędkość kątowa Y
plot(time, Gyroscope(:,3), 'b'); % prędkość kątowa Z
legend('X', 'Y', 'Z');
xlabel('Time (s)');
ylabel('Angular rate (deg/s)');
title('Gyroscope');
hold off;
axis(2) = subplot(2,1,2);
hold on;
plot(time, Accelerometer(:,1), 'r');
plot(time, Accelerometer(:,2), 'g');
plot(time, Accelerometer(:,3), 'b');
legend('X', 'Y', 'Z');
xlabel('Time (s)');
ylabel('Acceleration (g)');
title('Accelerometer');
hold off;
```

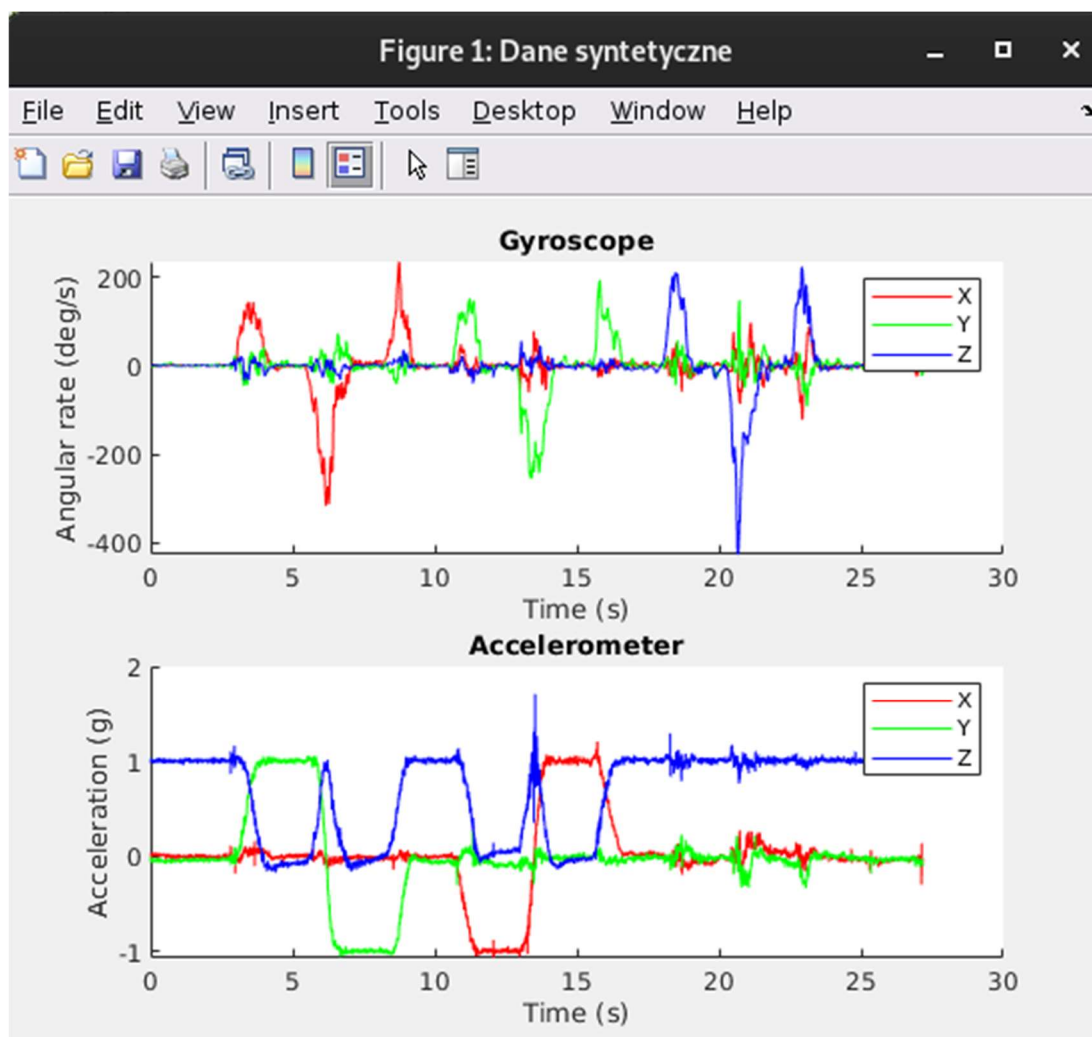
Do ładowania danych wykorzystana funkcja *load*, do wyświetlania *plot*.



Name	Value
Accelerometer	6959x3 double
Gyroscope	6959x3 double
Magnetometer	6959x3 double
time	6959x1 double

*Rys.3 Dane syntetyczne.*

Dane „*Magnetometer*” nie będą wykorzystywane w pracy.



Rys.4 Wykresy danych syntetycznych.

Kąty z danych akcelrometru obliczamy według wzorów 2.7, 2.8 i 2.9, korzystając z funkcji *atan*.

Listing 2 Pusta funkcja główna w języku C

```
Ax = atan(Accelerometer(:,1)./(sqrt(Accelerometer(:,2).^2 + Accelerometer(:,3).^2)));
Ay = atan(Accelerometer(:,2)./(sqrt(Accelerometer(:,1).^2 + Accelerometer(:,3).^2)));
Az = atan(Accelerometer(:,3)./(sqrt(Accelerometer(:,1).^2 + Accelerometer(:,2).^2)));
```

Tworzymy 3 puste matryce *Alpha*, *Beta*, *Gamma* do przechowywania kątów po filtrze.

Listing 3 Pusta funkcja główna w języku C

```
Alpha = zeros(size(Gyroscope(:,1)));
Beta = zeros(size(Gyroscope(:,2)));
Gamma = zeros(size(Gyroscope(:,3)));
```

### 3.1 FILTR KOMPLEMENTARNY

Algorytm napisany według wzorów 2.10, 2.11 i 2.12.

*Listing 4 Filtr Komplementarny*

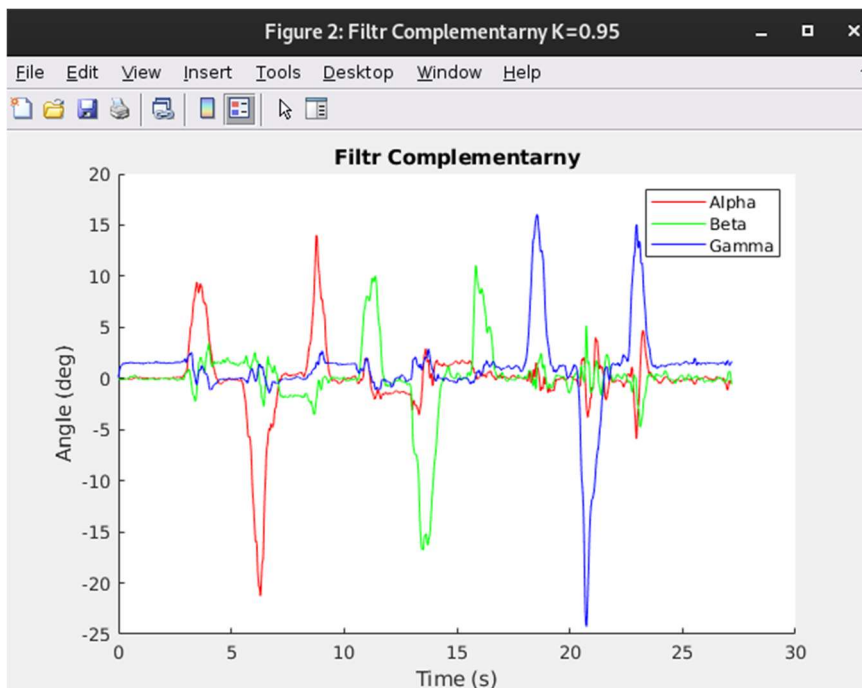
```
dt = 1/256;

for t = 1:length(time)
    if t == 1
        Alpha(t) = K * (Alpha(t) * dt) + (1-K) * Ax(t);
        Beta(t) = K * (Beta(t) * dt) + (1-K) * Ay(t);
        Gamma(t) = K * (Gamma(t) * dt) + (1-K) * Az(t);
    else
        Alpha(t) = K * (Alpha(t-1) + Gyroscope(t,1) * dt) + (1-K) * Ax(t);
        Beta(t) = K * (Beta(t-1) + Gyroscope(t,2) * dt) + (1-K) * Ay(t);
        Gamma(t) = K * (Gamma(t-1) + Gyroscope(t,3) * dt) + (1-K) * Az(t);
    end
end
```

Współczynnik filtra K jest siłą korekcji całkowanego kąta nachylenia z wykorzystaniem danych akcelerometru.

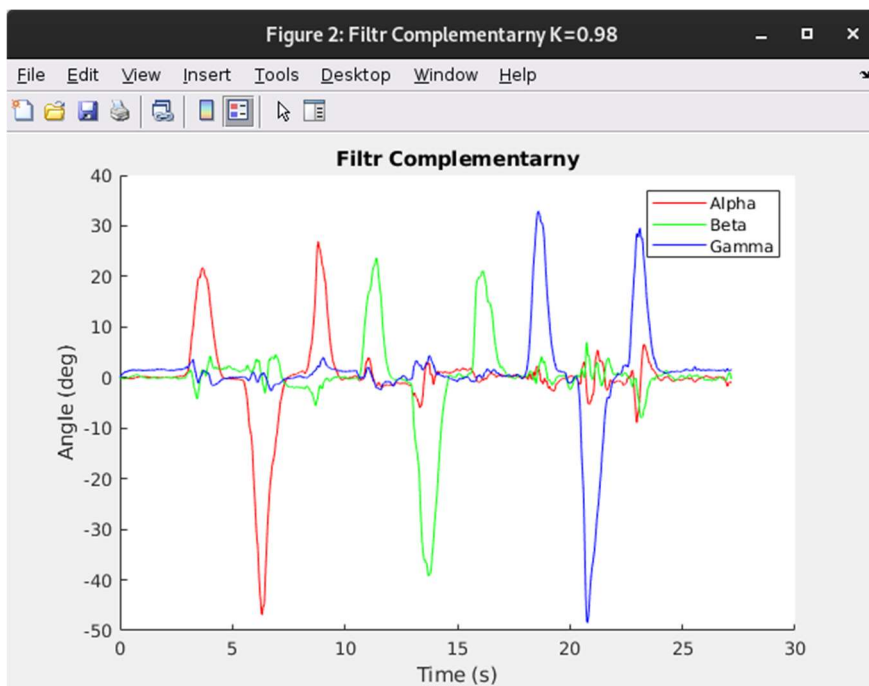
### Wyniki symulacji

Symulacja dla  $K = 0.95$



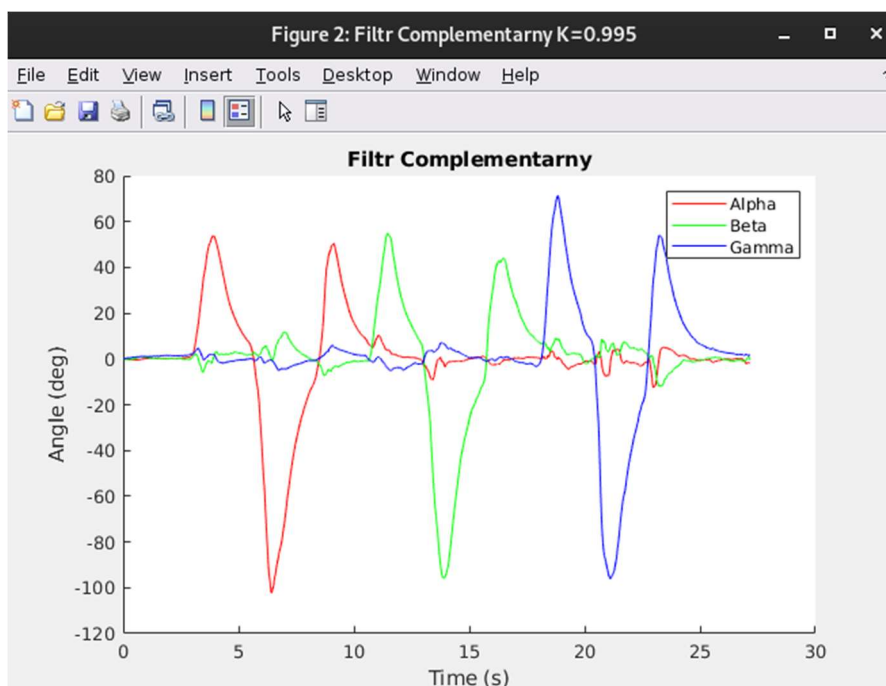
*Rys.4 Wykres danych syntetycznych po komplementarnym filtrze( $K=0.95$ )*

Symulacja dla  $K = 0.98$



Rys.4 Wykres danych syntetycznych po komplementarnym filtrze( $K=0.98$ )

Symulacja dla  $K = 0.995$



Rys.4 Wykres danych syntetycznych po komplementarnym filtrze( $K=0.995$ )



## 3.2 FILTR MADGWICKA

*Listing 5 Filtr Madgwicka. Realizacja*

```
q = obj.Quaternion;

if(norm(Accelerometer) == 0), return; end

Accelerometer = Accelerometer / norm(Accelerometer);

F = [2*(q(2)*q(4) - q(1)*q(3)) - Accelerometer(1)
      2*(q(1)*q(2) + q(3)*q(4)) - Accelerometer(2)
      2*(0.5 - q(2)^2 - q(3)^2) - Accelerometer(3)];
J = [-2*q(3),      2*q(4),      -2*q(1), 2*q(2)
      2*q(2),      2*q(1),      2*q(4), 2*q(3)
      0,          -4*q(2),      -4*q(3), 0      ];
step = (J'*F);
step = step / norm(step);

qDot = 0.5 * quaternProd(q, [0 Gyroscope(1) Gyroscope(2) Gyroscope(3)]) - obj.Beta * step';
q = q + qDot * obj.SamplePeriod;
obj.Quaternion = q / norm(q);
```

### 3.3 FILTR KALMANA

Równanie sytemu ma postać

$$\theta_k = \theta_{k-1} + (w_{k-1} - g_{bias})dt \quad (2.21)$$

Gdzie  $\theta$  – odchylenie katowe,  $w$  – prędkość katowa, a  $g_{bias}$  – to dryft żyroskopu.

W fazie pierwszej będzie uwzględniony pomiar z żyroskopu, w posłuży jako sterowanie  $u$ . Można zapisać równanie systemu w postaci:

$$x_k = Ax_{k-1} + Bu \quad (2.22)$$

Wektor stanu:

$$x_k = [\theta_x \ \theta_y \ \theta_z \ g_{bias,x} \ g_{bias,y} \ g_{bias,z}] \quad (2.23)$$

Macierz stanu:

$$A = \begin{bmatrix} 1 & 0 & 0 & -dt & 0 & 0 \\ 0 & 1 & 0 & 0 & -dt & 0 \\ 0 & 0 & 1 & 0 & 0 & -dt \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.24)$$

Wektor sterowania:

$$u = [w_x \ w_y \ w_z] \quad (2.24)$$

Macierz wejścia:

$$B = \begin{bmatrix} dt & 0 & 0 \\ 0 & dt & 0 \\ 0 & 0 & dt \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad (2.24)$$

Wektor pomiaru:

$$z = [A_x \ A_y \ A_z] \quad (2.24)$$

Macierz wyjścia:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \quad (2.24)$$

Zaprogramowanie algorytmu filtracji Kalmana pokazano w Lisnigu 5,6 i 7.

*Listing 5 Filtr Kalmana. Definiowanie modelu procesu*

```
x = zeros(6,length(time)); %wektor stanu
A = [1 0 0 -dt 0 0;
     0 1 0 0 -dt 0;
     0 0 1 0 0 -dt;
     0 0 0 1 0 0;
     0 0 0 0 1 0;
     0 0 0 0 0 1]; %macierz stanu 6x6
u = [W_alpha W_beta W_gamma]';
B = [dt 0 0;
     0 dt 0;
     0 0 dt;
     0 0 0;
     0 0 0;
     0 0 0]; %macierz wejścia(sterowania) 6x3
H = [1 0 0 0 0 0;
     0 1 0 0 0 0;
     0 0 1 0 0 0]; %macierz wyjścia 3x6
z = [Ax Ay Az]'; %wektor pomiaru
```

*Listing 6 Filtr Kalmana. Definicja macierzy kowariancji*

```
q = 0.00001;
r = 50;
p = 0;
Q = eye(6).* q; %macierz kowariancji modelu
R = eye(3).* r; %macierz kowariancji pomiarów
P = eye(6).* p; %macierz kowariancji stanu
```

*Listing 7 Filtr Kalmana. Realizacja*

```
x_post = zeros(6,1);
P_post = zeros(6,6);

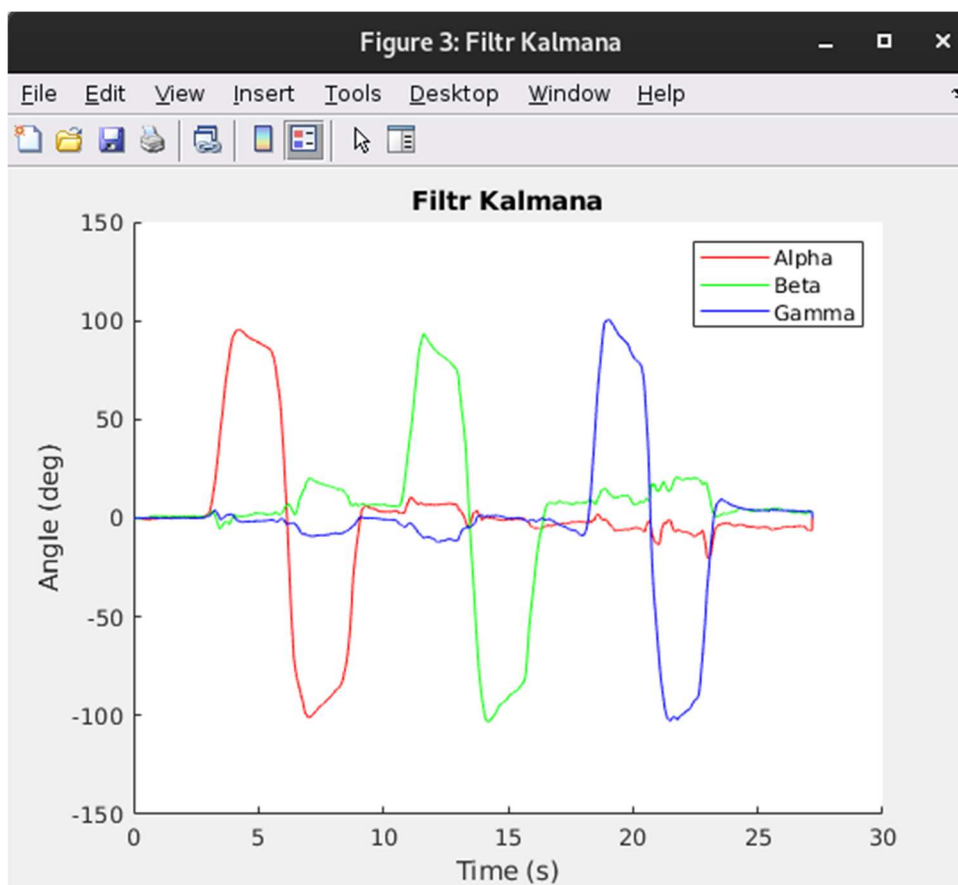
for k = 2:length(time)

    %Predykcja
    x_pri = A * x_post + B * u(:,k-1);
    P_pri = A * P_post .* A' + Q;

    %Korekcja
    K = P_pri * H' * (H * P_pri * H' + R)^(-1);
    x_post = x_pri + K * (z(:,k) - (H * x_pri));
    P_post = (I - K * H) * P_pri;

    %zapis
    Alpha(k) = x_post(1);
    Beta(k) = x_post(2);
    Gamma(k) = x_post(3);

end
```



Rys.4 Wykres danych syntetycznych po filtrze Kalmana

## 4 CZĘŚĆ SPRZĘTOWA

### 4.1 MIKROKONTROLER STM32F7

### 4.2 CZUJNIK MEMS LSM6DOX

### 4.3 OPROGRAMOWANIE

### 4.4 ZEBRANIE SUROWYCH DANYCH Z CZUJNIKA

### 4.5 WERYFIKACJA PRACY ALGORYTMÓW Z WYKORZYSTANIEM SUROWYCH DANYCH

## 5 IMPLEMENTACJA ALGORYTMÓW W UKŁADZIE RZECZYWISTYM

## 6 WYNIKI

## LITERATURA

1. AWERSOWY. *Polski: Rektorat Politechniki Poznańskiej na Wildzie* [online]. 25 maj 2014. s.l.: s.n. [udostępniono 27.11.2014]. Pobrano: [http://commons.wikimedia.org/wiki/File:Rektorat\\_Politechniki\\_Pozna%C5%84skiej\\_04.JPG](http://commons.wikimedia.org/wiki/File:Rektorat_Politechniki_Pozna%C5%84skiej_04.JPG).
2. DOMINIK ŁUCZAK. *Szablon pracy dyplomowej* [online]. 2014. s.l.: s.n. Pobrano: <http://zsep.cie.put.poznan.pl/>.
3. *ISO 690* [online]. s.l.: s.n., brak daty. [udostępniono 27.11.2014]. Pobrano: [http://pl.wikipedia.org/w/index.php?title=ISO\\_690&oldid=38364628](http://pl.wikipedia.org/w/index.php?title=ISO_690&oldid=38364628). Page Version ID: 38364628
4. Zotero | Home [online]. [udostępniono 7.12.2014]. Pobrano: <https://www.zotero.org/>.
5. Open source IMU and AHRS algorithms. [udostępniono 31.07.2012]. Pobrano: <http://x-io.co.uk/open-source-imu-and-ahrs-algorithms/>.
6. Jak zainstalować i skonfigurować Zotero? *Warsztat badacza – Emanuel Kulczycki* [online]. [udostępniono 27.11.2014]. Pobrano: [http://ekulczycki.pl/warsztat\\_badacza/jak-zainstalowac-i-skonfigurowac-zotero/](http://ekulczycki.pl/warsztat_badacza/jak-zainstalowac-i-skonfigurowac-zotero/).
7. MathType - Equation Editor [online]. [udostępniono 27.11.2014]. Pobrano: <http://www.dessci.com/en/products/mathtype/>.
8. LaTeX – A document preparation system [online]. [udostępniono 7.12.2014]. Pobrano: <http://www.latex-project.org/>.
9. *MathType 5.0 Manual - MathType5WinManual.pdf* [online]. s.l.: s.n. [udostępniono 27.11.2014]. Pobrano: <http://www.dessci.com/en/dl/MathType5WinManual.pdf>.
10. *Transformacja falkowa* [online]. s.l.: s.n., brak daty. [udostępniono 27.11.2014]. Pobrano: [http://pl.wikipedia.org/w/index.php?title=Transformacja\\_falkowa&oldid=38481890](http://pl.wikipedia.org/w/index.php?title=Transformacja_falkowa&oldid=38481890). Page Version ID: 38481890
11. Kody pól: pole Ref - Word [online]. [udostępniono 7.12.2014]. Pobrano: <http://office.microsoft.com/pl-pl/word-help/kody-pol-pole-ref-HP005186139.aspx>.
12. Blokowanie lub odblokowywanie pola - Word [online]. [udostępniono 7.12.2014]. Pobrano: <http://office.microsoft.com/pl-pl/word-help/blokowanie-lub-odblokowywanie-pola-HP005189392.aspx>.