

SZAKMAI DOLGOZAT - 36. OTDK

ÁLLAPOTBECSLŐ ALGORITMUS
AUTONÓM JÁRMŰVEKHEZ

Hallgató

CSANAK VIKTOR

Mentor

DR. PLETL SZILVESZTER

Feladatkiírás

Napjainkban egyre nagyobb szerepet kapnak a különböző mobilis, hordozható eszközök, legyenek azok hétköznapi technikai eszközök, mint okostelefonok vagy fitness karszalagok, hordozható orvosi készülékek, vagy különböző felépítésű és célú mobil robotok. Azonban egy közös tulajdonságon minden hasonló eszköz osztozik. Az alapvető funkcióikhoz vagy akár azok magasabb szintű képességekkel való kiegészítéséhez szükséges meghatározni az eszköz térbeli helyzetét.

Erre a feladatra természetes megoldást nyújtanak az úgynevezett inercia szenzorok(IMU) illetve azok magnetométerrel kiegészített formája(MARG), ugyanis ezek az eszközre szerelhetők, a számítások lokálisan, külső szenzorok felhasználása nélkül elvégezhetők.

A szakdolgozatom témáját olyan algoritmusok és azok elvi alapjaik képezik, amelyek a fent említett szenzorok felhasználásával képesek meghatározni a mobil eszközök orientációját egy lokális referencia rendszerhez viszonyítva.

Köszönetnyilvánítás

Ez úton szeretném megköszönni témavezetőmnek, Dr. Pletl Szilveszternek a jelen dolgozat alapját képező kutatómunka végzése során nyújtott felbecsülhetetlen értékű segítségét, útmutatását és szakmai tanácsait, amelyekkel többször is tovább lendítette a munkám, amikor az megtorpant.

Továbbá szeretném megköszönni Dr. Šarčević Péternek az AHRS algoritmusok validálásához nyújtott tanácsát, amellyel a dolgozat harmadik fejezetének megszületéséhez járult hozzá.

Absztrakt

A szakdolgozat keretein belül mozgó rendszerekre illeszthető IMU és MARG szenzorok felhasználásával olyan állapotbecslő algoritmusokat kutattam fel és mutatok be, amelyek képesek a rendszer orientációjának meghatározására.

A fenti algoritmusok megértéséhez szükséges matematikai és fizikai elveket részletesen ismertetem, az orientáció reprezentációjának legelterjedtebb formáit bemutatom és olyan szűrőket elemzek, amelyek ezen ábrázolási módszerek közül több felhasználására gyakorlati példát szolgáltatnak.

Az általam kielemezett algoritmusok mindegyike az angol szakirodalomban Complementary-Filterként ismert algoritmusra alapszik, amely a hagyományos Kálmán-szűrő működőképes és bizonyos szempontokból előnyösebb alternatívájaként szolgál.

A munkám során az Euler-szög reprezentációt felhasználó algoritmust MATLAB szoftvercsomagban reprodukáltam, majd a szakdolgozatban ismertetett módszerrel generált adatokon annak teljesítményét kivizsgáltam.

Tartalomjegyzék

Feladatkiírás	ii
Köszönetnyilvánítás	iii
Absztrakt	iv
Tartalomjegyzék.....	v
Szótár.....	vii
Bevezetés.....	1
1. Térbeli fizikai mennyiségek ábrázolása és manipulálása.....	2
1.1. Transzláció	2
1.2. Orientáció és forgató mátrixok.....	3
1.3. Euler rotációs elmélete és elemi rotációk.....	6
1.3.1. Euler rotációs elmélete	6
1.3.2. Elemi rotációk	7
1.3.3. Összetett rotációk	8
1.4. Orientáció reprezentációja Euler-szögekkel.....	9
1.5. Orientáció reprezentációja egység kvaterniókkal	12
1.5.1. Egység kvaternió ekvivalenciája a tengely/szög paraméterekkel	13
2. MEMS szenzorok és AHRS alkalmazásaik	14
2.1. Gyorsulásmérő	14
2.2. Giroszkóp	14
2.3. Magnetométer.....	15
2.4. AHRS rendszerek	15
3. Szenzoradatok gyűjtése és kalibrációja.....	16
3.1. Mérési modell.....	16
3.2. Kalibrációs modell	19
4. Orientáció-szűrő.....	21
4.1. Információ források.....	21

4.1.1. Giroszkóp	21
4.1.2. Gyorsulásmérő	22
4.1.3. Magnetométer.....	23
4.2. Információ fúzió	24
4.2.1 Klasszikus komplement szűrő	25
4.2.2. Kaszkád komplement szűrő	28
4.2.3. Mahony szűrő	29
4.2.4. Madgwick szűrő	31
5. Kísérleti eredmények.....	34
5.1. Adatgeneráció paraméterei.....	34
5.2. Eredmények.....	35
Konklúzió.....	38
Melléklet A.....	39
Melléklet B	41
Melléklet C.....	43
Irodalom	45

Szótár

Inertial Measurement Unit (IMU) – inercia szenzor.

Magnetic Angular Rate and Gravity (MARG) – inercia szenzor magnetométerrel ellátva.

Attitude Heading Reference System (AHRS) – Egy dinamikus rendszer pozíciójának és orientációjának meghatározására használt rendszer.

Microelectromechanical Systems (MEMS) – Mikro-elektromechanikus rendszerek

NED(North-East-Down) reference frame – (Észak-Kelet-Lefelé) koordináta-rendszer

ENU(East-North-UP) reference frame – (Kelet-Észak-Felfelé) koordináta-rendszer

Gimbal-lock – Rotációs mátrixok szingularitási problémája

4D – Négy dimenziós

GNNS -Globan Navigation Satelite System – Globális Navigációs Műhold Rendszer

GPS – Global Positioning System – Globális Pozícionáló Rendszer

KF – Kalman Filter – Kálmán-szűrő

CF – Complementary Filter – Komplementszűrő

LCF – Linear Complementary Filter - Lineáris

NCF - Non-linear Complementary Filter – Nem-lineáris Komplementszűrő

CCF – Cascade Complementary Filter – Kaszkád Komplementszűrő

CPO – Cross Product Operator – Vektor szorzat operátor

RMSE – Root Mean Square Error – Négyzetes közép.

Bevezetés

A MEMS technológia elterjedésével a felhasználót az analóg fizikai mennyiségekkel összekötő szenzorok ára, mérete és fogyasztása nagyságrendekkel csökkent[1], lehetővé téve azok nagymértékű elterjedését, amelynek következtében ma már szinte minden okostelefonban, hordható eszközben és játékokban is megtalálhatóak[7].

Ezen szenzorok eredetileg olyan iparágakban jelentek meg, mint repülés- és űrtechnika vagy tengerészet, azok mérete és költségei miatt, és elsősorban navigációra szolgáltak. Azonban a MEMS szenzorok elterjedésével ma már hordható eszközökön is megtalálhatóak és újmódú alkalmazások egész tárháza előtt nyitották meg a kaput. Ilyen alkalmazások magukba foglalják a hordható orvosi rehabilitációs eszközöket, a robotok működéséhez szükséges adatokat szolgáltató rendszereket és navigációs berendezéseket[2], mint az első sorban IMU és MARG szenzor alapú AHRS rendszerek.

A szakdolgozatomban részletesen tárgyalom az IMU és MARG szenzorok méréseit korrumpáló karakterisztikáikat, és a MATLAB szoftvercsomag által biztosított hiba modellel szenzor adatokat szintetizálok, amelyek felhasználhatók a szenzor adatokra épülő algoritmusok pontosságának meghatározására.

Ugyan ezen karakterisztikák felhasználásával, a mérési hibák kompenzálásának céljából a szenzor adatok kalibrációját leíró módszert is bemutatok.

A szakdolgozatomban továbbá a szakirodalomban fellelhető AHRS algoritmusokat is bemutatok, illetve kiértékelem azok teljesítményét szintetizált adatokon.

A szakdolgozat keretein belül általánosan a következő jelölések használtak:

1. Skalárok: nem félkövér kis latin vagy görög betűk.
2. Vektorok: félkövér kis latin vagy görög betűk.
3. Mátrixok: félkövér nagy latin vagy görög betűk.

1. Térbeli fizikai mennyiségek ábrázolása és manipulálása

1.1. Transzláció

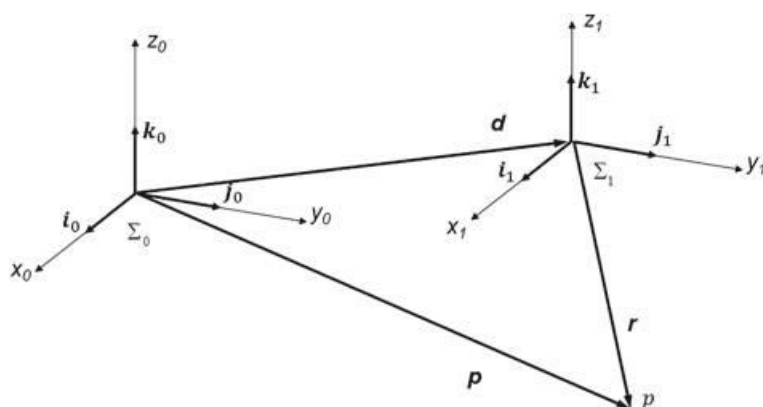
Vegyünk két Descartes koordináta-rendszert, Σ_0 és Σ_1 rendre (x_0, y_0, z_0) és (x_1, y_1, z_1) kardinális tengelyekkel, az $(\mathbf{i}, \mathbf{j}, \mathbf{k})$ egységvektorok mentén és tételezzük fel, hogy Σ_1 origója (d_x, d_y, d_z) távolságra helyezkedik el, amelyet $\mathbf{d}$ vektor jelöl.

Vegyünk egy p pontot, amelyet a két koordináta-rendszerben a következőképpen határozunk meg:

$$\mathbf{p}^{(0)} = p_x^{(0)}\mathbf{i} + p_y^{(0)}\mathbf{j} + p_z^{(0)}\mathbf{k} = \begin{bmatrix} p_x^{(0)} \\ p_y^{(0)} \\ p_z^{(0)} \end{bmatrix} \quad (1.1)$$

$$\mathbf{p}^{(1)} = p_x^{(1)}\mathbf{i} + p_y^{(1)}\mathbf{j} + p_z^{(1)}\mathbf{k} = \begin{bmatrix} p_x^{(1)} \\ p_y^{(1)} \\ p_z^{(1)} \end{bmatrix} \quad (1.2)$$

Ahol $\mathbf{p}^{(0)}$ és $\mathbf{p}^{(1)}$ p pontot jelölik Σ_0 és Σ_1 koordináta-rendszerekben.



ábra 1

Az 1-es ábrából a lineáris algebra szabályait felhasználva látható, hogy $\mathbf{p}^{(0)}$ meghatározható a következő egyenlettel:

$$\mathbf{p}^{(0)} = \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} + \begin{bmatrix} p_x^{(1)} \\ p_y^{(1)} \\ p_z^{(1)} \end{bmatrix} \quad (1.3)$$

Amely egyenlet a Galilei-transzformációként ismert[3].

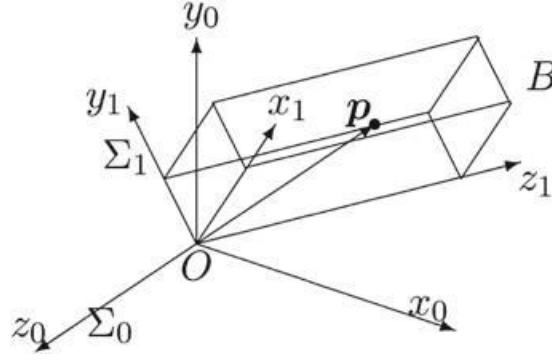
A Galilei-transzformáció az egymás közt relatív mozgást végző viszonyítási rendszerek közti kapcsolatot írja le. A fenti példában a két viszonyítási rendszer ugyan az az ortonormált bázis szerint volt meghatározva, azonban amennyiben ez a feltétel nem teljesül, az (1.3) egyenlet módosításra szorul. Ezt az esetet az 1.2-es fejezetben ismertetem.

Fontos továbbá megjegyezni, hogy mérnöki gyakorlatban a viszonylagos mozgást végző rendszerekre világ- és test koordináta-rendszerekként hivatkozunk. A test koordináta-rendszer, egy mozgó tárgy, például egy IMU vagy MARG szenzor középpontjához csatolt koordináta-rendszer, míg a világ koordináta-rendszer egy statikus, különböző konvenciók szerint meghatározott viszonyítási rendszer. Ezek ún. NED és ENU koordináta-rendszerek, amelyek közül tengerészeti és légiforgalmi célú navigációhoz a NED viszonyítási rendszer használatos.

A NED koordinátarendszer gyakran a vizsgált rendszer kezdeti állapotának középpontjában van meghatározva oly módon, hogy annak (x, y, z) tengelyei rendre észak, kelet és a Föld középpontjának irányába mutatnak. Azonban, ez a világ koordináta-rendszer meghatározható úgy is, hogy (x, y, z) tengelyei a rendszer kezdeti helyzetében előre, jobbra és lefelé irányuljanak a test középpontjához viszonyítva.

1.2. Orientáció és forgató mátrixok

Vegyük az előző fejezet $\sum_0$ és $\sum_1$ viszonyítási rendszereit, azzal a változtatással, hogy azok középpontjai egybeesnek és (x_0, y_0, z_0) és (x_1, y_1, z_1) tengelyei $(\mathbf{i}_0, \mathbf{j}_0, \mathbf{k}_0)$ és $(\mathbf{i}_1, \mathbf{j}_1, \mathbf{k}_1)$ egységvektorok mentén meghatározottak és $\sum_1$ B testhez kötött. Ebben a felállásban egy p pontot az (1.1) és (1.2) egyenletek módosításával határozhatunk meg.



ábra 2

$$p^{(0)} = p_x^{(0)} i_0 + p_y^{(0)} j_0 + p_z^{(0)} k_0 = \begin{bmatrix} p_x^{(0)} \\ p_y^{(0)} \\ p_z^{(0)} \end{bmatrix} \quad (1.4)$$

$$p^{(1)} = p_x^{(1)} i_1 + p_y^{(1)} j_1 + p_z^{(1)} k_1 = \begin{bmatrix} p_x^{(1)} \\ p_y^{(1)} \\ p_z^{(1)} \end{bmatrix} \quad (1.5)$$

Ilyen esetben a kapcsolatot a két viszonyítási rendszer között orientációnak nevezzük, ugyanis azok bázisai egymáshoz viszonyítva el vannak forgatva. Ezt az összefüggést úgy tudjuk meghatározni, hogy p pont adott rendszerben vett reprezentációjának meghatározzuk a vetületét a másik viszonyítási rendszerben. Ezt a kiinduló koordináta-rendszerben vett reprezentáció lineáris kombinációjának a cél bázis egység vektorainak egyenként vett skalárszorzatával határozhatjuk meg. Ezeket a $\mathbf{p}^{(1)}$ vektor Σ_0 rendszerben vett vetületének, $\mathbf{p}^{(0)}$ meghatározásával szemléltetve az alábbi egyenletekkel tudjuk felírni[3]:

$$p_x^{(0)} = p^{(1)} \cdot i_0 \quad (1.6)$$

$$p_y^{(0)} = p^{(1)} \cdot j_0 \quad (1.7)$$

$$p_z^{(0)} = p^{(1)} \cdot k_0 \quad (1.8)$$

$$p_x^{(0)} = p_x^{(1)} i_1 \cdot i_0 + p_y^{(1)} j_1 \cdot i_0 + p_z^{(1)} k_1 \cdot i_0 \quad (1.9)$$

$$p_y^{(0)} = p_x^{(1)} i_1 \cdot j_0 + p_y^{(1)} j_1 \cdot j_0 + p_z^{(1)} k_1 \cdot j_0 \quad (1.10)$$

$$p_z^{(0)} = p_x^{(1)} i_1 \cdot k_0 + p_y^{(1)} j_1 \cdot k_0 + p_z^{(1)} k_1 \cdot k_0 \quad (1.11)$$

A fenti (1.6)-(1.11) kifejezések a következőképp is felírhatók:

$$p^{(0)} = R_0^1 p^{(1)} \quad (1.12)$$

ahol R_0^1 egy ún. **rotációs mátrix**, amely egy lineáris transzformáció és a $\sum_1$ rendszerben meghatározott vektorok $\sum_0$ rendszeren belüli ekvivalens vektorait határozza meg.

A (1.9)-(1.11) kifejezésekből a következőképpen definiálhatjuk:

$$R_0^1 \triangleq \begin{bmatrix} i_1 \cdot i_0 & j_1 \cdot i_0 & k_1 \cdot i_0 \\ i_1 \cdot j_0 & j_1 \cdot j_0 & k_1 \cdot j_0 \\ i_1 \cdot k_0 & j_1 \cdot k_0 & k_1 \cdot k_0 \end{bmatrix} \quad (1.13)$$

Ugyan ezt a módszert alkalmazva meghatározható a $\sum_0$ rendszerből $\sum_1$ rendszerbe transzformáló rotációs mátrix R_1^0 is.

$$R_1^0 \triangleq \begin{bmatrix} i_0 \cdot i_1 & j_0 \cdot i_1 & k_0 \cdot i_1 \\ i_0 \cdot j_1 & j_0 \cdot j_1 & k_0 \cdot j_1 \\ i_0 \cdot k_1 & j_0 \cdot k_1 & k_0 \cdot k_1 \end{bmatrix} \quad (1.14)$$

A (1.13) és (1.14) egyenletben definiált mátrixok egymás inverzei továbbá, a skalár szorzat kommutatív tulajdonságából adódóan a fenti ortonormált mátrixok inverze megegyezik azok transzponáltjával [3]. Ebből adódóan $R_1^0 = (R_0^1)^T$ és $R_0^1 = (R_1^0)^T$. Továbbá, jobbkezes koordináta-rendszerekben a fenti mátrixok determinánsa egységnyi nagyságú: $|R| = 1$.

A fenti mátrixok meghatározhatók az alábbi egyenlet szerint, ahol a mátrix oszlop vektorai - $r_i \in R^3$ - és sor vektorai - $n_i \in R^3$ - egység magnitúdójúak és kölcsönösen ortogonálisak:

$$R = [r_1, \quad r_2, \quad r_3] = \begin{bmatrix} n_1^T \\ n_2^T \\ n_3^T \end{bmatrix} \quad (1.15)$$

mivel a fenti vektorok eleget tesznek az ortonormalitás feltételének:

$$r_i \cdot r_j = \begin{cases} 0 & \text{ha } i \neq j \\ 1 & \text{ha } i = j \end{cases} \quad (1.16)$$

$$n_i \cdot n_j = \begin{cases} 0 & \text{ha } i \neq j \\ 1 & \text{ha } i = j \end{cases} \quad (1.17)$$

Ezen tulajdonságaikból kifolyólag az $n = 3$ dimenziójú négyzetes forgatómátrixok az ún. harmad rendű speciális ortogonális csoport elemei - $R \in SO(3)$ -.

Az ilyen ortogonális mátrixok tulajdonsága, hogy lineáris transzformációt végeznek és amint az a (1.13) és (1.14) egyenletekből látható, a forgatómátrixokban szereplő skalárszorzatok két ortonormált bázis egymáshoz viszonyított vetületeit határozzák meg és amely skalárszorzatokat vektorok elemeivel skálázva – a forgatómátrixot jobbról szorozva – a bázisok között végezhetünk transzformációt.

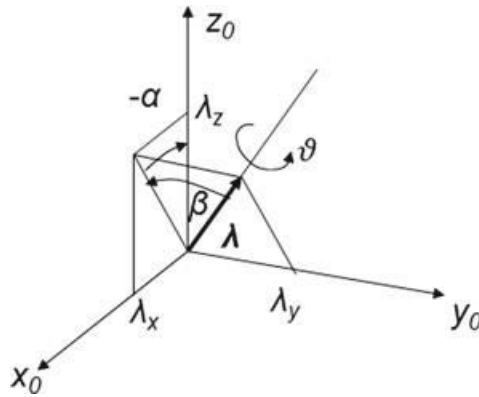
Ezek ismeretében az 1.1 fejezetben ismertetett Galileo-transzformáció a következőképp módosítható:

$$p^{(0)} = \begin{bmatrix} d_x \\ d_y \\ d_z \end{bmatrix} + R_0^1 \begin{bmatrix} p_x^{(1)} \\ p_y^{(1)} \\ p_z^{(1)} \end{bmatrix} \quad (1.18)$$

1.3. Euler rotációs elmélete és elemi rotációk

1.3.1. Euler rotációs elmélete

Euler rotációs elmélete szerint bármilyen orientáció leírható egy tetszőleges tengely menti elforgatással. Ez a tengely meghatározható, mint egy koordináta-rendszerben meghatározott λ irányvektor.



ábra 3

Amennyiben a tengely körül vett elfordulást ϑ jelöli, a $\begin{bmatrix} \vartheta \\ \lambda \end{bmatrix}$ vektor egy ún. tengely-szög rotációs párt alkot[3].

Figyelembe véve, hogy egy rotációs mátrix szabadságfoka – a legkevesebb független változók száma, amelyekkel meghatározható a mátrix – három, így a tengely/szög rotációs pár egy redundáns ábrázolási mód. Ezért annak érdekében, hogy a három szabadságfok függetlenségét megőrizzük, a λ irányvektornak eleget kell tennie az egység magnitúdó felételnek[3].

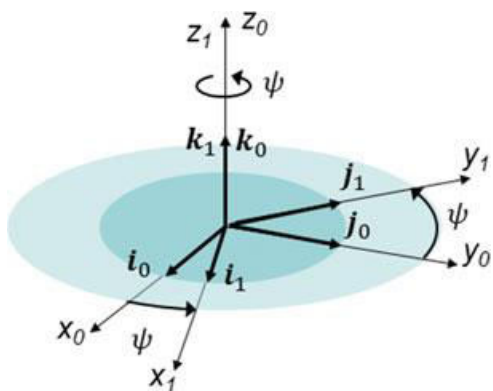
Az $SO(3)$ rotációs csoport tulajdonsága, hogy a létezik egy egység nagyságú sajátértéke - $\lambda_1 = 1$ – amelyhez tartozó sajátvektora a λ irányvektor lesz, amely körül a forgás végbemegy[3].

Abban a speciális esetben, amikor a λ irányvektor egybeesik egy viszonyítási rendszer valamely egységvektorával, elemi rotációról beszélhetünk.

1.3.2. Elemi rotációk

Az elemi rotációk egy világ vagy test koordináta-rendszer egy kardán-tengelye körüli forgást leíró forgatómátrixok. Vegyük a már korábban definiált Σ_1 és Σ_0 viszonyítási rendszereket.

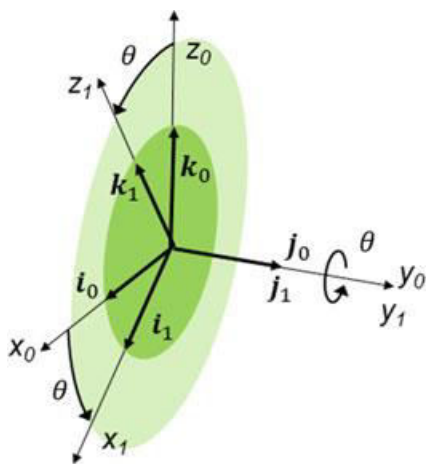
Ebben az esetben a z_1 kardántengely körüli rotációt ψ jelöli és az alábbi matrix írja le,



$$R_{z,\psi} = \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (1.19)$$

ábra 4

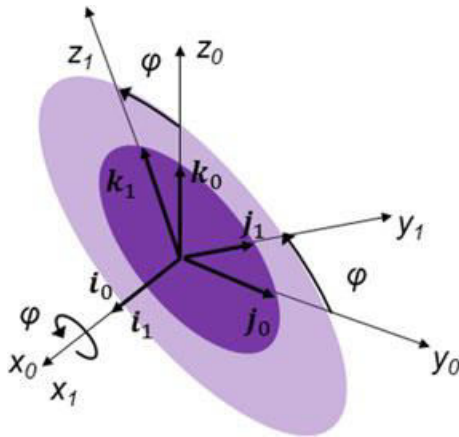
míg y_1 körül θ elfordulást a következőképp határozhatunk meg



$$R_{y,\theta} = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \quad (1.20)$$

ábra 5

és x_1 tengely körül ϕ elfordulást az alábbi egyenlettel fejezünk ki,



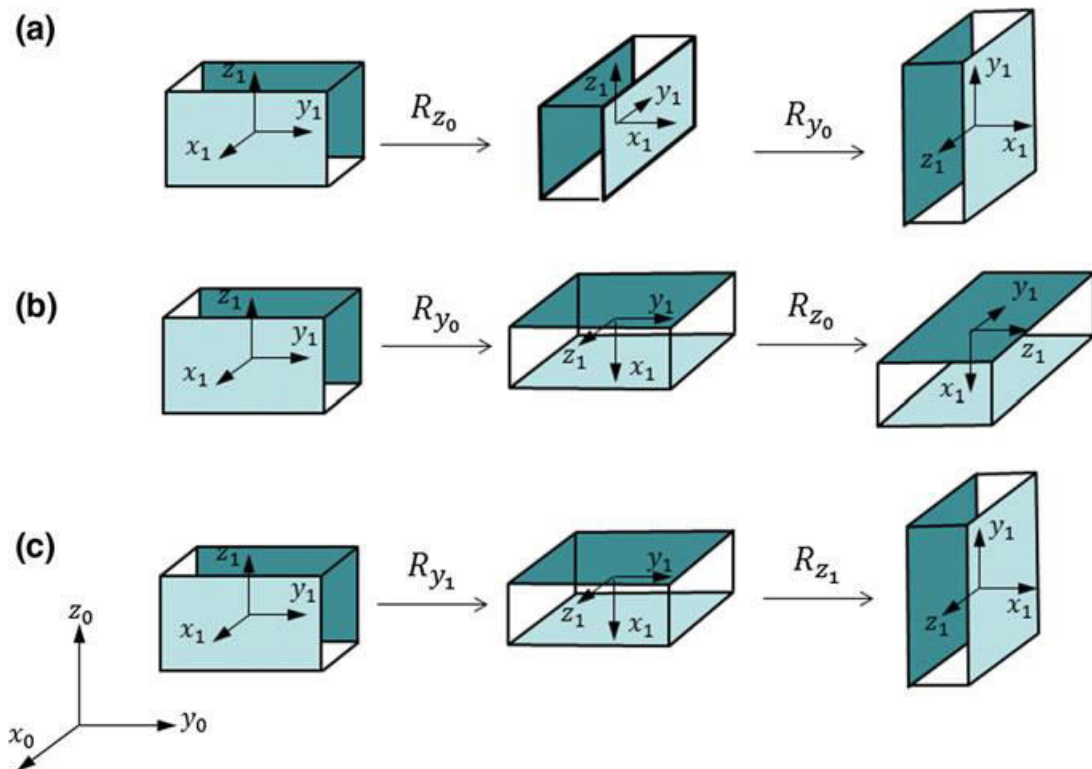
$$R_{x,\phi} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\phi & -s\phi \\ 0 & s\phi & c\phi \end{bmatrix} \quad (1.21)$$

ábra 6

amely egyenletekben $c(x) = \cos(x)$ és $s(x) = \sin(x)$.

1.3.3. Összetett rotációk

Egy test orientációját meghatározhatjuk úgy is, mint annak elemi rotációi a kardántengelyek mentén. Ezt az alábbi ábra szemlélteti:



ábra 7

Mint ahogy azt a 7-es ábrán is láthatjuk, a test sorozatos – összetett – rotációk esetén különböző orientációt vesz fel, attól függően, hogy a forgatásokat a tengelyek körül milyen sorrendben végezzük, sőt az is kiemelt fontossággal bír, hogy a forgatásokat melyik viszonyítási rendszer tengelyei mentén végezzük. Ezek lehetnek a világ koordináta-rendszer tengelyei mentén végzett rotációk – 7-es ábra (a) és (b) esetei – vagy a világ koordináta-rendszerhez viszonyított orientációját minden elforgatással változtató test koordináta-rendszer tengelyei mentén – 7-es ábra (c) esete -.

Fontos megjegyezni, hogy amennyiben a koordináta rendszert, amely tengelyei körül a rotációkat végezzük, és a forgatások sorrendjét is megváltoztatjuk, a végső orientáció ugyan az lesz[3], ahogy az a 7-es ábrán is látható.

1.4. Orientáció reprezentációja Euler-szögekkel

Ahogy azt már az 1.3.1. fejezetben említettem, egy rotációs mátrixnak – avagy egy referencia rendszer orientációja egy másik viszonyítási rendszerhez képest – három szabadságfoka van. Ennek értelmében természetesnek mondható, hogy egy test orientációját három elemi rotáció segítségével határozzuk meg. Ilyenkor a három független változó ún. pózvektort képez - (1.22) egyenlet-, amely vektor elemei a három elemi rotáció mértéke, az (1.19) – (1.21) egyenletekben megjelenő szögek, amelyeket Euler-szögeknek nevezünk.

$$\theta = \begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} \quad (1.22)$$

A fenti vektort tekintve, nem létezik olyan kombinációja az Euler-szögeknek, amely egyedi módon határozná meg egy test orientációját, ugyanis az elemi rotációk alapját képző koordináta-rendszerek és azok sorrendjének megváltoztatásával különböző pózvektorok ugyan azt az orientációt írhatják le. Azonban bebizonyítható, hogy összesen tizenkét különböző forgatási sorrend létezik. Ezek alább láthatók.

- | | | |
|----------------|-----------------|-----------------|
| 1. $x - y - z$ | 3. $y - x - z$ | 5. $z - y - x$ |
| 2. $x - z - y$ | 4. $y - z - x$ | 6. $z - x - y$ |
| 7. $x - y - x$ | 9. $y - x - y$ | 11. $z - x - z$ |
| 8. $x - z - x$ | 10. $y - z - y$ | 12. $z - y - z$ |

ábra 8

Amennyiben figyelembe vesszük, hogy a fenti rotációk elvégezhetők a világ- és test koordináta-rendszerek tengelyei mentén is, összesen huszonnég különböző rotációt kapunk.

A dolgozat keretein belül bemutatott algoritmushoz az ún. Tait-Bryant Roll, Pitch és Yaw (RPY) konvenciót használtam fel, ahol az RPY rendre a forgatások alapjául vett (x, y, z) tengelyek menti Euler-szögek.

Amennyiben a rotációk alapjául a világ koordináta-rendszer tengelyeit vesszük, a konvenció szerint az $x - y - z$ rotációs sorrendet követjük. Viszont, amikor test koordináta-rendszer tengelyei mentén definiáljuk a rotációkat, azokat $z - y - x$ sorrendben végezzük. Tekintve, hogy a későbbiekben a számításokhoz használt szenzor adatok az IMU/MARG szenzor test koordináta-rendszerében definiáltak, így ajánlatos az utóbbi rotációs sorrendet alkalmazni. Ebben az esetben a test koordináta-rendszerben meghatározott vektor világ koordináta-rendszerbeli vetületét a következő módon határozhatjuk meg:

$$v^{(i)} = R_\phi R_\theta R_\psi v^{(j)} \quad (1.23)$$

Amennyiben az így kapott rotációs mátrix elemeit $r_{ij}, i, j \in [1, 2, 3]$ jelöli, az RPY Euler-szögek az alábbi egyenletekkel meghatározhatók[3]:

$$\phi = \arctan2(r_{32}, r_{33}) \in [-\pi, \pi] \quad (1.24)$$

$$\theta = \arctan\left(\frac{-r_{31}}{\sqrt{r_{11}^2 + r_{21}^2}}\right) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \quad (1.25)$$

$$\psi = \arctan2(r_{21}, r_{11}) \in [-\pi, \pi] \quad (1.26)$$

Az Euler-szögekkel meghatározott rotációs mátrixok különböző szingularitási problémáktól szenvednek, amelyek közül a legismertebb az ún. gimbal-lock, amely akkor lép fel, ha a forgatási sorrendben második mátrixba $\pm \frac{\pi}{2}$ szöget viszünk be. Azonban, ennek ellenére Euler-szögek intuitív mivoltuk miatt gyakran használatosak a gyakorlatban, első sorban az orientáció emberi megfigyelő által könnyen értelmezhető reprezentációjára, illetve számításokban is[4].

Az (1.23) egyenletben látott forgatómátrix kifejtve a következőképp határozható meg[5]:

$$R_i^j = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c_\theta c_\psi & c_\theta s_\psi & -s_\theta \\ c_\psi s_\theta s_\phi - s_\psi c_\phi & s_\psi s_\theta s_\phi + c_\psi c_\phi & c_\theta s_\phi \\ c_\psi s_\theta c_\phi + s_\psi s_\phi & s_\psi s_\theta c_\phi - c_\psi s_\phi & c_\theta c_\phi \end{bmatrix} \quad (1.27)$$

Az (1.27)-es egyenletbe $\theta = \pm \frac{\pi}{2}$ -t helyettesítve az alábbi trigonometriai azonosságokat felhasználva, a következő forgatómátrixot kapjuk:

$$\sin(\alpha \pm \beta) = \sin(\alpha)\cos(\beta) \pm \sin(\beta)\cos(\alpha) \quad (1.28)$$

$$\cos(\alpha \pm \beta) = \cos(\alpha)\cos(\beta) \mp \sin(\alpha)\sin(\beta) \quad (1.29)$$

$$R_i^j = \begin{bmatrix} 0 & 0 & \mp 1 \\ -\sin(x) & \cos(x) & 0 \\ \pm \cos(x) & \pm \sin(x) & 0 \end{bmatrix} \quad (1.30)$$

ahol $x = \psi \mp \phi$.

Mivel a fenti mátrixban csak egy független változó - x - található, ezért szükséges vagy ψ vagy ϕ definiálása annak érdekében, hogy meg tudjuk határozni a másik szöget. Ebben az esetben az Euler-szög paraméterek elveszítik egy szabadságfokukat, esetünkben θ -t[3].

1.5. Orientáció reprezentációja egység kvaterniókkal

A kvaterniók egy 4D komplex számrendszer elemei, amelyeket egy valós és három komplex változó alkot. Ezeket az alábbi formában jelöljük:

$$q = [q_0 \quad q_1 \quad q_2 \quad q_3] = [q_0 \quad q_v] \in H \quad (1.31)$$

Illetve annak komplex konjugáltját:

$$q^* = [q_0 \quad -q_v] \quad (1.32)$$

A kvaterniók a forgatómátrixokhoz hasonlóan olyan matematikai objektumok, amelyek egy R^3 térben meghatározott vektor – $\mathbf{v}$ - rotációs lineáris transzformálását alkalmasak elvégezni, amennyiben azt a vektort R^4 térben a következőképp definiáljuk[5]:

$$p = [0 \quad v] \quad (1.33)$$

Ebben az esetben a $\mathbf{p}$ vektor forgatását az alábbi egyenlettel végezhetjük el[3]:

$$p^{(0)} = q \otimes p^{(1)} \otimes q \quad (1.34)$$

ahol a fenti műveletek az ún. Hamilton-szorzatot jelölik.

1.5.1. Egység kvaternió ekvivalenciája a tengely/szög paraméterekkel

Ahogy az a korábbi fejezetekben láthattuk, különböző orientáció reprezentációk közt, mint a forgatómátrixok és a tengely/szög rotációs pár, egyenlőség határozható meg. Ehhez hasonló módon a kvaterniók inverz exponenciális leképezésén keresztül összefüggés határozható meg azok és a rotációs pár között[3]:

$$q = e^{\boldsymbol{\vartheta}\alpha} = (\cos(\alpha), \boldsymbol{\vartheta}\sin(\alpha)) \quad (1.35)$$

ahol $\boldsymbol{\vartheta}$ egy egységvektor és $\mathbf{q}$ kvaternió vektoriális része megegyezik λ irányvektorral, a rotációs pár vektoriális elemével, illetve $\alpha = \frac{\vartheta}{2}$, a rotációs pár skaláris elemének felével.

Igy egy intuitív módon értelmezhető definíciót kapunk a kvaterniókra. Az így végzett számítások előnye, hogy kevesebb paraméterrel végezhetők el ugyan azok a műveletek, mint forgatómátrixokkal, továbbá mentesek az orientáció más reprezentációiban megjelenő szingularitások minden formájától.

2. MEMS szenzorok és AHRS alkalmazásaik

A MEMS szenzorok olyan mikrométeres nagyságban gyártott, elektromechanikus rendszerek, amelyek elektromos jelet szolgáltatnak egy mért fizikai mennyiségről. Az inercia vagy IMU szenzorok olyan MEMS mérőeszközök, amik egy három tengelyes gyorsulásmérőt és egy három tengelyes giroszkópot[7], míg a MARG szenzorok emellett egy három tengelyes magnetométert is tartalmaznak egy csomagolásban.

2.1. Gyorsulásmérő

A MEMS gyorsulásmérők egy test adott tengelye menti gyorsulásáról szolgáltatnak információt, azt legtöbbször [g] egységekben kifejezve - $g = 9.81\text{m/s}^2$ -. Ennek legegyszerűbb módja Newton második törvényének felhasználása és egy mikrométer nagyságú tömeg-rugó rendszer implementálása a mozgás tengelye mentén. A test gyorsulását előidéző erő hatására egy etalon tömeg a rendszerben elmozdul, ily módon a tömeghez csatolt szigetelőkkel egy kondenzátor kapacitását befolyásolva. Ezekben a rendszerekben a tömeg elmozdulását a test mozgása és a Föld gravitációs tere is előidézheti. Emiatt ezeket a szenzorokat felhasználgják okostelefonokban a képernyő orientációjának meghatározására vagy a magnetométerek elfordulásának kompenzálására a gravitáció mérésének útján[7].

2.2. Giroszkóp

A MEMS giroszkópok egy test szögsebességének mérésére alkalmas berendezések, amely általában [szög/másodpercben] ([dps]) van kifejezve. Ezek a szenzorok általában vibrációs giroszkópok, amelyek két egymásra ortogonális tengely mentén elrendezett rugóból és azokhoz csatolt tömegből álló rendszerek. Az egyik tengely az ún. meghajtó tengely, amely irányában az eszköz egy ismert amplitúdójú sebességgel oszcillációban tartja az etalon tömeget. Ebben a konfigurációban a két tömeg-rugó által alkotott síkra merőleges tengely mentén fellépő elfordulás a meghajtó tengely

mentén oszcilláló súlyra Coriolis-erőt fejt ki, így a másik - mérési - tengely mentén is mozgást végez, amely egy kondenzátor kapacitásának változásával mérhető[7].

2.3. Magnetométer

A magnetométerek a lokális mágneses terek mérésére alkalmasak. Ideális esetben ez a Föld mágneses terét jelenti, azonban ezek a szenzorok magas mértékben érzékenyek minden közeli ferromágneses anyag zavaró hatására. Fontos továbbá megjegyezni, hogy a mágneses téren végzett mérések során figyelembe kell venni a szenzor elfordulásának mértékét is.

2.4. AHRS rendszerek

Az AHRS algoritmusok ún. szenzorfüziós módszerek, amelyek egy test pozícióját és orientációját határozzák meg több szenzor adatainak felhasználásával. Ezek az adatok elengedhetetlenek többek között robotok működéséhez és lehetővé teszik hordozható eszközök építését és azok kiegészítését olyan funkciókkal, mint mozgás érzékelés.

Ezek az adatok általában IMU vagy MARG szenzorok által biztosított mérések, amelyeket kiegészíthetünk GNSS/GPS és egyéb szenzorokkal is, azonban a dolgozat keretein belül elsősorban IMU és MARG szenzor alapú algoritmusok kerülnek bemutatásra, amelyek a test orientációjának meghatározására alkalmasak.

A szenzorfüzió szükségessége abban rejlik, hogy a felsorolt szenzorok ugyan mind képesek bizonyos mértékben információt szolgáltatni a test orientációjáról, egyik sem ad teljes képet arról.

A giroszkópok által mért szögsebességek integrálása valóban elegendő volna a test elfordulásának mérésére, azonban a gyakorlatban ezek a szenzorok nem ideális méréseket végeznek, azok hibákkal terhelték – ezeket a hibákat a harmadik fejezetben ismertetem-. Az integrálás során ezek a hibák összeadódnak és rövid idő alatt felhalmozódnak, így a számítások egyre távolabb kerülnek a valós értékektől[6]. Az angol szakirodalomban erre a jelenségre a drift kifejezést használják.

A gravitációs vektoron végzett mérésekkel a gyorsulásmérő adataiból meghatározható a test orientációja, azonban mérési hibák mellett ezek a számítások érzékenyek a test translációja során fellépő gyorsulásokra, illetve a gravitáció tengelye mentén nem tudjuk meghatározni a test szögelfordulását.

A Föld mágneses terén végzett mérések alapján meghatározható a test orientációja a mágneses északhoz viszonyítva, azonban a magnetométer mérései érzékenyek a közeli mágneses anyagokra, illetve a test elfordulását valamilyen módon kompenzálni kell, hogy az orientációt valóban a Föld teréhez képest tudjuk meghatározni.

3. Szenzoradatok gyűjtése és kalibrációja

Az AHRS algoritmusok validációjához vagyis azok pontosságának meghatározásához már korábban valósnak ítélt ún. „ground-truth” adatokra van szükségünk, amelyhez az algoritmus által visszaállított értékeket viszonyítani tudjuk. Az ilyen adatsorok előállításához vagy költséges cél eszközökre vagy különböző speciális tesztkörnyezetekre és további számításokra van szükség[6]. A munkám során a körülmények egyik módszert sem tettek lehetővé, ezért validációhoz ún. szintetizált adatokat hozok létre. Ennek módja, hogy modell alapú számítások segítségével a meghatározott ground-truth adatokból szintetikus szenzor adatok hozhatók létre. Ehhez a MATLAB szoftvercsomag navigációs könyvtárából az IMU szimulációs modellt használtam fel.

3.1. Mérési modell

Az alábbi egyenlet – amely a már említett könyvtárban definiált modellt írja le[8] – szerint határozható meg az adatok, amelyek egy mozgó koordináta-rendszerben elhelyezkedő szenzor méréseit tükrözik, amennyiben ismert a szenzor koordináta-rendszerének orientációja a világ koordináta-rendszerhez viszonyítva, illetve az utóbbiban meghatározva a mérendő fizikai mennyiségek.

$$s_v = \begin{bmatrix} s_{vx} \\ s_{vy} \\ s_{vz} \end{bmatrix} \quad (3.1)$$

$$s_m^{(s)} = MR_{(s)}^{(w)} s_r^{(w)} + c_{bias} \quad (3.2)$$

Ahol $R_{(s)}^{(NED)}$ egy (a saját implementációmban) Euler-szögekből álló pózvektorból meghatározott forgatómátrix a két viszonyítási rendszer közti transzformálásra.

Az $\mathbf{M}$ mátrix pedig a szenzor tengelyei közt jelenlévő ortogonalitási hibákból származó tengelyek közti érzékenységet leíró mátrix[9]:

$$M = \begin{bmatrix} 1 & m_{xy} & m_{xz} \\ m_{yx} & 1 & m_{yz} \\ m_{zx} & m_{zy} & 1 \end{bmatrix} \quad (3.3)$$

Továbbá c_{bias} az ún. bias, amely egy szenzor nem zérus kimenetét jelenti, zérus bemenet mellett. Ez a mennyiség egy gyűjtő fogalom, amely magába foglalja a szenzor egyenáramú komponensét, vagyis konstans biast és a szenzor kimenetének hőfüggőségét, amelyek determinisztikus mennyiségek, így az adatok kalibrálása során kompenzálhatók[9]. Ezek a mennyiségek a mérési modellben az alábbi módon definiáltak:

$$c_{stat} = c_{offset} + \alpha(T[^\circ\text{C}] - 25) \quad (3.4)$$

ahol α az ún. hő-együttható.

Ezentúl a bias tartalmaz még egy változó, sztochasztikus jellegű összetevőt is, amely a szimulációs modellben az alábbi módon adott:

$$c_{drift} = w * \left(\sqrt{\frac{T}{2}} \left(\text{NoiseDensity} \left[\frac{1}{\sqrt{\text{Hz}}} \right] \right) + \frac{1}{1 - \frac{1}{2}z^{-1}} (\text{BiasInstability}) + \frac{1}{1 - z^{-1}} \left(\frac{\text{RandomWalk}}{\sqrt{\frac{T}{2}}} \right) \right) \quad (3.5)$$

Ahol (3.4) és (3.5) egyenletekben c_{offset} az egyenáramú komponens, w egy zérus átlagú és normál eloszlású változó vagyis fehér zaj, illetve α , NoiseDensity, BiasInstability és a RandomWalk paraméter vektorok.

Újra kiemelném, hogy a drift komponensben leírt folyamatok sztochasztikus jellegűek, így azok az adatok kalibrációja során nem kompenzálhatók.

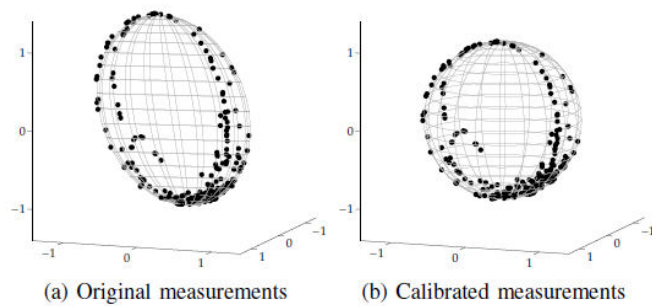
Fontos továbbá újból megemlíteni, hogy a magnetométerek magas mértékben érzékenyek a közeli ferromágneses anyagok jelenlétére, aminek következtében a mérni kívánt mágneses tér torzul.

Amennyiben torzulásoktól mentesen a Föld mágneses terén végeznénk méréseket, az eredmények egy gömb felületén helyezkednének el, melynek középpontja az origóban helyezkedik el.

Azonban torzítások jelenlétében a szférikus tér középpontja elmozdul, illetve eldeformálódik, így egy ellipszoid felületén helyezkednek el a mérési adatok.

A középpont elmozdulása az ún. hard-iron hatásoknak tudhatók be, amelyek olyan ferromágneses anyagok, amelyek a szenzorral együtt mozognak, így azok hatása a mérésekre stacionáriusnak tekinthető és offsetként ábrázoljuk a mérési modellben.

A mágneses zavarok másik formája az ún. soft-iron hatás, amely olyan mágneses anyagok közelsége okoz, amelyek nem mozognak együtt a szenzorral. Ezek hatása a gömbszerű tér ellipszoiddá deformálásával vizualizálható[10].



ábra 9

Az említett mágneses zavarok jelenlétében a (3.2)-es egyenletben leírt mérési modell a következőképp módosítandó:

$$s_m^{(s)} = C_{si} M R_{(s)}^{(w)} (s | r^{(w)} + c_{hi}) + c_{bias} \quad (3.6)$$

A mágneses zavarok gyakran a legnagyobb limitációt jelentik a magnetométerek alkalmazásában. Emiatt azok minél hatékonyabb kompenzációja ma is egy aktuális probléma, azonban az kívül esik a dolgozat keretein.

Továbbá, a MATLAB csomag által szolgáltatott mérési modell ezeket a hatásokat nem szimulálja, így a munkám során a méréseket mágneses zavartól mentesnek, illetve már kalibráltak tekinthetjük.

3.2. Kalibrációs modell

A (3.2)-(3.5) egyenletekben ismertettem egy mérési modellt, amely a legfőbb, valós szenzorokban megjelenő mérési hibákat foglalja magába. Ezen mérési hibák közül több sztochasztikus folyamatnak tekinthető, amelyek kalibráció során nem becsülhetők meg, így azok hatása nem kompenzálható. Azonban tartalmaz olyan determinisztikus folyamatokat is, amelyek kalibráció útján felbecsülhetők és az algoritmus futása során kompenzálhatók. Ezek a hatások első sorban az egyenáramú komponens és az ortogonalitási hibákból származó tengelyek közti érzékenység[11].

$$s_c^{(s)} = M^{-1} \left(s_m^{(s)} - c_{offset} \right) \quad (3.7)$$

A kalibrációs paraméterek meghatározására több módszer is létezik[9]. Ezek közül az offset felbecslésére használható módszereket ismertetem, míg az ortogonalitási hibákat, mint a szimulációs modell paraméterei, meghatározottnak tekintem.

Az egyenáramú komponens meghatározására két módszert mutatok be. Az első módszert az alábbi egyenlet írja le:

$$c_{offset} = \frac{1}{N} \sum_{i=1}^N s_i \quad (3.8)$$

Ahogy az látható, ebben az esetben az egyenáramú komponenst a szenzoradatok átlagaként határozhatjuk meg, amikor a szenzor bemenetei zérus nagyságúak[6]. Ily módon elvégezhető a giroszkóp adatok kalibrációja, azonban abban az esetben, amennyiben zérus bemenet nem érhető el a szenzor minden tengelye mentén, mint a gyorsulásmérő esetében, ahol a gravitációs gyorsulás ezt nem teszi lehetővé, a második módszert alkalmazzuk.

Az ilyen jellegű kalibráció során a szenzor adott tengelyeire – gyorsulásmérő esetében horizontális helyzetben a gravitáció tengelye – pozitív és negatív referenciát alkalmazunk adott ideig, amelyek átlagát (3.8) szerint határozzuk meg. Ekkor az adott tengelyhez tartozó pozitív és negatív referenciák átlagaként határozható meg az egyenáramú összetevő[6].

$$c_{i_{offset}} = \frac{s_{+ref} + s_{-ref}}{2} \quad (3.9)$$

ahol i a szenzor adott tengelye.

4. Orientáció-szűrő

A fejezet célja egy olyan, az irodalomban fellelhető források alapján készített AHRS algoritmus, amely képes egy főképp rotációs mozgást végző tárgy orientációjának meghatározására egy NED világ koordináta-rendszer és a mozgó test referencia-rendszere között.

Az orientáció reprezentációjára az (1.22) egyenletben leírt pózvektort használjuk, amely paraméterei egy a test tengelyei mentén végzett x-y-z rotációs sorrendet leíró Euler-szögek.

Ahogy azt már említettem, a különböző szenzorok alkalmasak az orientáció felbecslésére, azonban önmagukban nem elegendők. Ezért a fejezetben a különböző forrásokból való információ kinyerésének és azok fúziójának módját mutatom be.

4.1. Információ források

4.1.1. Giroszkóp

Az előzőleg definiált pózvektor változásának idő szerinti integrálásával meghatározható a test pillanatnyi orientációja:

$$\theta = \int (\dot{\theta} * dt) \quad (4.1)$$

A giroszkóp adatok alapján meghatározható a test szögsebessége, mint $\omega = [\omega_x \omega_y \omega_z]^T$. A szögsebesség vektor egyedi módon határozza meg a test rotációs mozgásának sebességét. Azonban tudjuk, hogy az Euler-szögekkel definiált pózvektor attól függően, hogy a stacionárius vagy mozgó koordináta-rendszer tengelyei mentén, illetve, hogy milyen sorrendben határozza meg az elfordulásokat, ugyan az a pózvektor több orientációt is leír, illetve egy orientációt több pózvektor is megadhat. Emiatt a pózvektor nem egyedi reprezentációja az orientációnak. Ebből következik,

hogy a pózvektor deriváltja nem egyezik meg a test szögsebességével bármelyik viszonyítási rendszerben is definiáljuk azt[3]:

$$\dot{\theta} \neq \omega \quad (4.2)$$

Ezért ilyen formában a giroszkóp adatok nem alkalmasak a pózvektor meghatározására, azonban léteznie kell egy összefüggésnek a szögsebesség és az Euler-szögek változása között. Ezt az összefüggést az alábbi egyenlet írja le[3]:

$$\dot{\theta} = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & \frac{s_\phi}{c_\theta} & \frac{c_\phi}{c_\theta} \end{bmatrix} \omega^{(b)} \quad (4.3)$$

amely átalakítás után immár a (4.1) egyenlet szerint a teljes pózvektor meghatározható, azonban a még a harmadik fejezetben leírt kalibráció után is sztochasztikus zajokkal terhelt szenzoradatok integrálása során azok is hozzáadódnak a becsült értékhez, így a pózvektor becsült értéke folyamatosan távolodik a valós értéktől.

4.1.2. Gyorsulásmérő

Az elsősorban rotációs mozgást végző testre szerelt gyorsulásmérő segítségével képesek vagyunk a test koordináta-rendszerben a gravitáció mérésére. Tekintve, hogy ismerjük a normalizált gravitációs vektort a világ koordináta-rendszerben - $\hat{G}^{(w)} = [001]^T$ - egy z-y-x rotációs sorrendre felírt rotációs mátrix segítségével meghatározható az összefüggés a mért gravitációs vektor és a világ koordináta-rendszerben definiált vektor között a következő módon[12]:

$$\hat{G}^{(b)} = \begin{bmatrix} \hat{g}_x \\ \hat{g}_y \\ \hat{g}_z \end{bmatrix} = R_{xyz} \hat{G}^{(w)} = \begin{bmatrix} -s_\theta \\ c_\theta s_\phi \\ c_\theta c_\phi \end{bmatrix} \quad (4.4)$$

amely egyenletből felismerhető, hogy a ϕ és θ meghatározhatók, (1.24) - (1.25) egyenletekhez hasonlóan, a gyorsulásmérőből származó adatok alapján[12].

$$\phi = \arctan2(\dot{g}_y, \dot{g}_z) \quad (4.5)$$

$$\theta = \arctan2(-\dot{g}_x, \sqrt{\dot{g}_y^2 + \dot{g}_z^2}) \quad (4.6)$$

Ahogy az látható a (4.5) egyenlet instabillá válik, amikor a gyorsulásmérő méréseinek y és z komponense egyaránt nullával egyenlőek. Ez akkor fordul elő, amikor a gyorsulásmérő x-tengelye párhuzamossá válik a Föld gravitációjának irányával. Ilyenkor az y és z összetevőket pusztán zaj teszi ki és a ϕ szög értéke kiszámíthatatlan módon változni fog. Ez egyúttal magyarázatot ad arra, hogy a gyorsulásmérő segítségével nem lehetséges ψ szög meghatározása. Viszont a (4.5) egyenlet módosítható úgy, hogy a fent leírt szingularitást kiküszöböljük[12]:

$$\phi = \arctan2(\dot{g}_y, \text{sign}(\dot{g}_z)\sqrt{\dot{g}_z^2 + \mu\dot{g}_x^2}) \quad (4.7)$$

Ily módon az inverz négy negyedes tangens függvény szerint a pózvektor első két eleme meghatározható a $(-\pi, \pi]$ tartományban.

A módszer felhasználásával az integráció során fellépő halmozódó hibáktól mentesen részben meghatározható a pózvektor, azonban a számítások csak akkor végezhetők el, ha a test translációs szempontból kvázi stacionárius, továbbá a pózvektor teljes egészében való meghatározásához szükség van még egy magnetométerre is.

4.1.3. Magnetométer

A már korábban meghatározott mágneses zavarok kompenzálása után egy magnetométer segítségével képesek vagyunk méréseket végezni a Föld mágneses terén. Amennyiben ismerjük a

Föld mágneses terének normalizált alakját a világ koordináta-rendszerben - $\hat{B}^{(w)} = [B * \cos(\delta), 0, B * \sin(\delta)]^T$ – egy z-y-x forgatási sorrendre szintén meghatározható a kapcsolat a két viszonyítási rendszerben értelmezett mágneses vektorok között. Azonban tekintve, hogy a Föld mágneses terének vektora a horizonttal adott szöget bezárva a bolygó belseje és észak felé mutat – ebből adódik a fenti normalizált vektor alakja – ezért amennyiben a pózvektor első két eleme nem nulla, a mágneses vektor vertikális eleme hozzáadódik a magnetométer méréseinek x és y komponenseihez, így azok alapján nem határozható meg a pózvektor ψ szöge. Ezért a magnetométer elfordulását a pózvektor másik elemei szerint kompenzálnunk kell. Ehhez a mért vektort az eredeti forgatási sorrenddel fordítottan el kell forgatni. Esetünkben ez az x majd az y tengelyek menti ϕ és θ elforgatást jelent. A z-tengely mentén nem végzünk rotációt, ugyanis az a világ koordináta-rendszerben meghatározott vektort adná, azonban számunkra a z rotáció utáni vektor bír jelentőséggel a ψ szög meghatározásában[13]. Ezek ismeretében az alábbi egyenletek írhatók fel:

$$R_{-x-y}B^{(b)} = R_{-x-y} \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix} = \begin{bmatrix} c_\psi B c_\delta \\ -s_\psi B c_\delta \\ B s_\delta \end{bmatrix} = \begin{bmatrix} b_{fx} \\ b_{fy} \\ b_{fz} \end{bmatrix} \quad (4.8)$$

$$\psi = \arctan2(-b_{fy}, b_{fx}) \quad (4.9)$$

4.2. Információ fúzió

Mint látható, a fenti információ források külön is alkalmasak bizonyos mértékben az orientáció meghatározására, azonban mind tartalmazznak hibaforrásokat is. Egy információfúziós vagyis esetünkben orientáció-szűrő algoritmus célja, hogy ezeket a hibákat meghatározza és kompenzálja. A klasszikus megoldás, amelyet a legtöbb kereskedelmi AHRS algoritmus is alkalmaz, a Kálmán-szűrő és annak továbbfejlesztett változatai. A Kálmán-szűrő egy kétlépcsős modell és mérés alapú becslés alapján képes meghatározni a számításokban fellépő hibákat és azokat kompenzálni, így konvergálva a valós értékek felé. Azonban ezek az algoritmusok nagy számításigénnyel rendelkeznek, illetve azok megfelelő működéséhez szükség van egy dinamikai modell és több paraméter meghatározására is. Ez okokból amellet, hogy a Kálmán-szűrő továbbra is a domináns

megközelítés a területen, az utóbbi évtizedekben több kutatás is eredményes AHRS algoritmusokat produkált. Ezek közül az egyik legismertebb az ún. komplementens-szűrő(CF) illetve annak változatai.

Az általam legkorábban fellelt utalások az algoritmusra a Mahony és munkatársai által 2005-ben[15] és 2008-ban[16] publikált cikkekben szerepel, amely algoritmusok speciális ortogonális mátrixokon alapuló, hiba visszacsatolásos számításokkal határozzák meg egy test orientációját.

Továbbá Sung Kyung Hong 2003-ban publikált kutatása[14], amelyben egy PI kontrollert tartalmazó visszacsatolás segítségével kompenzálható a giroszkóp mérések konstans egyenáramú komponense és amely architektúra egy klasszikus másodrendű CF felépítését tükrözi.

Ezt a felépítést Mahony et al. 2008-as cikkjének [16] melléklete alapján mutatom be.

4.2.1 Klasszikus komplementens szűrő

Tegyük fel, hogy létezik két mérésünk:

$$y_1 = x + \mu_1 \text{ és } y_2 = x + \mu_2 \quad (4.1) \quad (4.2)$$

Ahol x a mérni kívánt jel μ_1 és μ_2 pedig rendre magas és alacsony frekvenciás zajok, illetve zavarok. Ebben az esetben ugyan azon jelnek két mérése egymást kiegészítő spektrális karakterisztikákkal rendelkezik. Két egymást kiegészítő átviteli függvény segítségével, ahol $F_1(s)$ egy alul- és $F_2(s)$ pedig felül-áteresztő szűrők, a mért $x(t)$ jel becslésének átviteli függvény az alábbi módon határozható meg:

$$\hat{X}(s) = F_1(s)Y_1(s) + F_2(s)Y_2(s) \quad (4.3)$$

ahol $F_1(s) + F_2(s) = 1$. Ekkor:

$$\hat{X}(x) = X(s) + F_1(s)\mu_1 + F_2(s)\mu_2 \quad (4.4)$$

Ily módon egy torzítás mentes szűrőt kapunk, mivel mérni kívánt jelet a szűrő nem módosítja, továbbá a magas és alacsony frekvenciás zavarokat rendre $F_1(s)$ és $F_2(s)$ szűrik.

Tegyük fel, hogy képesek vagyunk mérni egy jelet alacsony sáv szélességű szenzorokkal

$$y_x = L(s)x + \mu_x \quad (4.5)$$

ahol $L(s)$ a szenzor aluláteresztő karakterisztikája, μ_x pedig magas frekvenciás zaj.

Továbbá mérni tudjuk ugyan annak a jelnek az első rendű kinematikáját:

$$y_u = u + \mu_u + b(t) \quad (4.6)$$

ahol $u = \dot{x}$, μ_u alacsony frekvenciás zaj és $b(t)$ determinisztikus zavar. A kinematikai mérést, y_u -t integráljuk annak érdekében, hogy meghatározzuk a mérni kívánt jelet.

Ennek a feltételnek eleget tesznek a giroszkóp adatok alapján meghatározott pózvektor változás és a magnetométer, illetve gyorsulásmérő adatokból számolt pózvektor. Ekkor a két szűrőt a következőképp határozzuk meg:

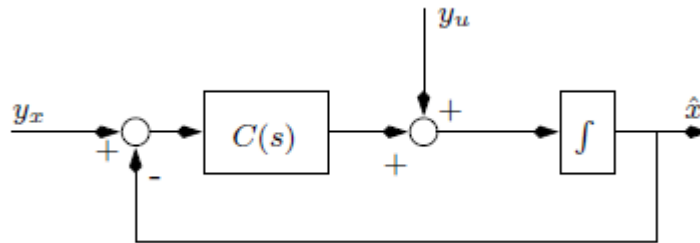
$$F_1(s) = \frac{C(s)}{C(s)+s} \quad (4.7)$$

$$F_2(s) = 1 - F_1(s) = \frac{s}{s+C(s)} \quad (4.8)$$

Ahol $C(s)$ olyan minden-áteresztő szűrő, hogy $L(s)F_1(s) \approx 1$, $L(s)$ sávszélességén. Ekkor a becslés Laplace-transzformáltja:

$$\hat{X}(s) = X(s) + F_1(s)\mu_x(s) + \frac{\mu_u(s)+b(s)}{C(s)+s} \quad (4.9)$$

Az egyenlet alapján a zárt hurkú rendszer az alábbi blokkdiagram szerint készíthető el:



ábra 10

amely segítségével a kimenet felírható mint:

$$\hat{x}(s) = \frac{C(s)}{s+C(s)} y_x(s) + \frac{s}{C(s)+s} \frac{y_u(s)}{s} \quad (4.10)$$

A fenti architektúra lehetővé teszi a klasszikus irányítástechnikai elvek alkalmazását.

Amennyiben $C(s) = kp$ egy proporcionális szabályzó, a rendszer zárt hurkú dinamikája alább adott:

$$\dot{\hat{x}} = y_u + kp(y_x - \hat{x}) \quad (4.11)$$

Ilyenkor a komplement karakterisztikájú szűrők, amelyek a fenti rendszert határozzák meg:

$F_1(s) = \frac{kp}{kp+s}$ és $F_2(s) = \frac{s}{s+kp}$. A kp paraméter tipikusan az y_x jel alacsony sáv szélességű karakterisztikái és az y_u jel alacsony frekvenciájú zaj komponense alapján határozható meg.

Az ilyen jellegű szűrőt (CF) elsőrendű vagy **lineáris komplement szűrőnek** (LCF) nevezik [17][18]. A rendszer segítségével elvégezhető két egymástól független információforrás (szenzor) fúziója, azonban a mérési hibák nem kompenzálhatók ily módon.

Amennyiben a szabályzóhoz hozzáadunk egy integrális tagot, $C(s) = kp + \frac{ki}{s}$ egy PI kontrollert és egy I-es típusú rendszert kapunk, amely képes az y_u jellel bevitt konstans zavarok kompenzálására. Ilyenkor a zárt hurkú dinamika:

$$\dot{\hat{x}} = y_u - \hat{b} + kp(y_x - \hat{x}) \quad (4.12)$$

$$\dot{\hat{b}} = -ki(y_x - \hat{x}) \quad (4.13)$$

Lyapunov módszereinek segítségével és LaSalle invariancia elvével rendre bebizonyítható az állapot becslés konvergenciája a valós értékhez, illetve a konstans zavarok becslésének konvergenciája a valós zavarokhoz [16], stacionárius vagy kvázi stacionárius zavarokra, amennyiben a rendszer konvergenciája gyorsabb, mint a zavarok változása.

A PI paraméterek hangolása klasszikus frekvencia dizájn módszereken alapszik, vagyis:

$$k_i = \omega^2, \quad k_p = 2\zeta\omega \quad (4.14) \quad (4.15)$$

Ezt a rendszert a szakirodalomban másodrendű vagy **nem lineáris komplement szűrőnek** (NCF) nevezik [17][18].

Erre az esetre a komplement karakterisztikájú szűrők átviteli függvényei:

$$F_1(s) = \frac{k_p s + k_i}{s^2 + k_p s + k_i}, \quad F_2(s) = \frac{s^2}{s^2 + k_p s + k_i} \quad (4.16) \quad (4.17)$$

A (4.15) egyenletben ezért célszerű a csillapítási tényezőt mint, $\zeta = \frac{\sqrt{2}}{2}$ meghatározni, így biztosítva a (4.16) és (4.17) egyenletek stabilitását.

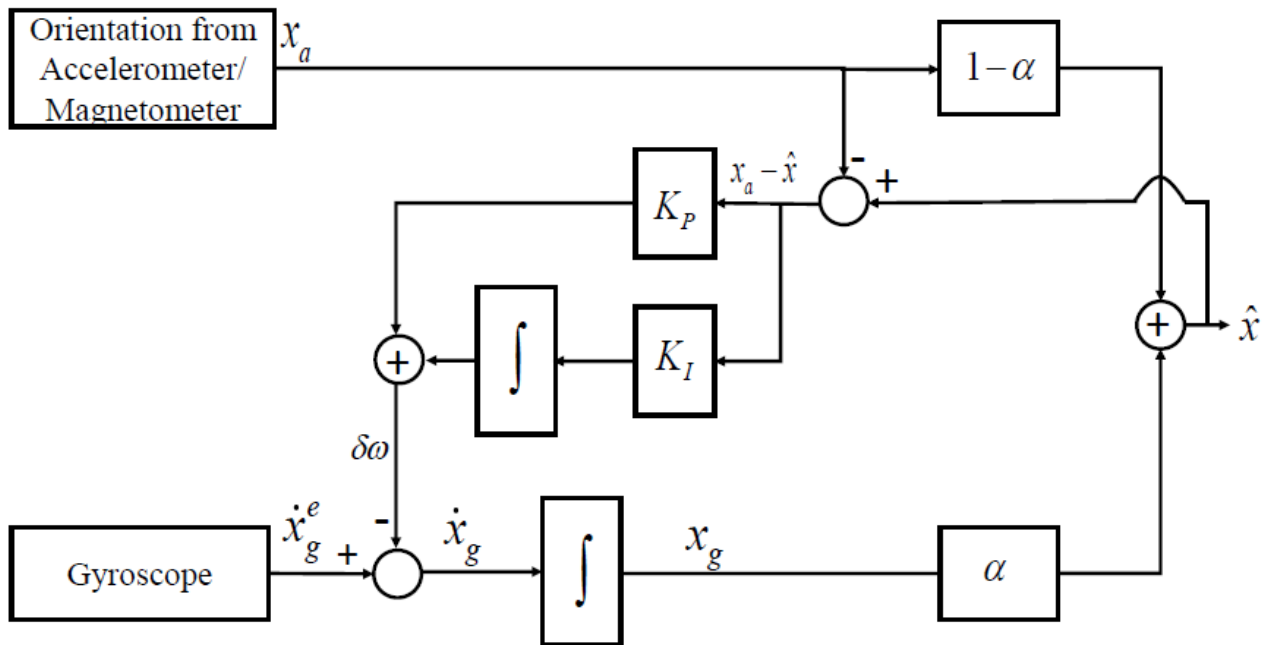
Egy klasszikus NCF alkalmas két szenzor fúziójára, miközben kompenzálja a mért jel első idő szerinti deriváltjában jelenlévő kvázi stacionárius zavarokat, amelyek kompenzáció nélkül driftelést idéznének elő a becslésben. Azonban a szűrő paraméterei a konvergálás sebessége mellett nagy mértékben befolyásolják a becslés pontosságát is [14]. Emiatt különböző kutatások során változatos módszereket alkalmaznak, hogy a legnagyobb pontosságot tudjuk elérni. Ilyen

módszerek például a paraméterek adaptív hangolása fuzzy logikával[14] vagy különböző CF-re alapuló orientáció szűrő architektúrák, amelyeket következő fejezetek során mutatok be.

4.2.2. Kaszkád komplementens szűrő

A CF alapú kaszkád architektúra (CCF)[17] megoldást nyújthat a NCF szűrő PI paramétereinek hangolásában rejlő problémákra. A szűrő két fő részre bontható, az előzőleg ismertetett NCF-re és LCF-re amelyek kaszkád kapcsolásba rendezettek.

A CCF architektúra kiaknázza az NCF-ben rejlő lehetőséget a giroszkóp bias kompenzálására, míg a LCF segítségével javítja a rendszer válaszát az egész frekvencia tartományban, így függetlenné téve a becslés pontosságát a szűrő paramétereitől. Ezt oly módon tehetjük meg, hogy a nem lineáris szűrő segítségével kompenzált, a visszacsatolásban is felhasznált pózvektort, illetve a magnetométer és gyorsulásmérő adataiból meghatározott pózvektort egy LCF-be tápláljuk be, amely kimenete a végső becslést szolgáltatja[17]. Ez a módszer az alábbi ábrán látható:



ábra 11

Az eredeti szerzők által az NCF és CCF szűrőkön végzett validáció eredményei alapján[17] megállapítható, hogy a szűrő visszacsatolásának integrális és proporcionális paramétereitől, sőt, α paramétertől is független a becslés pontossága.

4.2.3. Mahony szűrő

Mahony et al. [15][16] egy olyan szűrő architektúrát javasoltak, amely a klasszikus CF formáját tükrözve, mátrix alakban implementál egy orientáció szűrőt.

Az algoritmus tárgyalása előtt szükséges két új fogalmat ismertetni.

Vektor szorzat operátor(CPO)

Adott két vektor, $a = [a_x \ a_y \ a_z]^T$ és $b = [b_x \ b_y \ b_z]^T$, amelyek vektor szorzata:

$$a \times b = \begin{bmatrix} a_y b_z - b_y a_z \\ -a_x b_z + a_z b_x \\ a_x b_y - b_x a_y \end{bmatrix} \in R^3 \quad (4.18)$$

Ekkor $A = [a \times] = a_{\times}$ az ún. CPO, egy lineáris operátor, amely oly módon implementálja a vektort mátrix formában, hogy azt jobbról egy vektorral szorozva, a két vektor keresztszorzatát kapjuk[3].

$$a_{\times} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \in SS(3) \quad (4.19)$$

Ahol $SS(3)$ egy harmad rendű ferde szimmetrikus mátrixot jelöl, amelyre igaz, hogy

$$a_{\times}^T = -a_{\times} \quad (4.20)$$

A (4.19) – (4.20) egyenletek alapján (4.3)-hoz hasonlóan meghatározható egy forgatómátrix kinematikája[3].

Amennyiben $\{A\}$ a világ, míg $\{B\}$ a test koordináta-rendszer ${}^A_B R = R$

$$\dot{R} = [\omega^A \times] R = R [\omega^B \times] \quad (4.21)$$

$$\dot{R}^T = -[\omega^B \times] R^T \quad (4.22)$$

A 4.2.3. fejezetben belül a továbbiakban jelölje a mért szögsebességet a test koordináta-rendszerben

$$\Omega_y = \omega^B \quad (4.23)$$

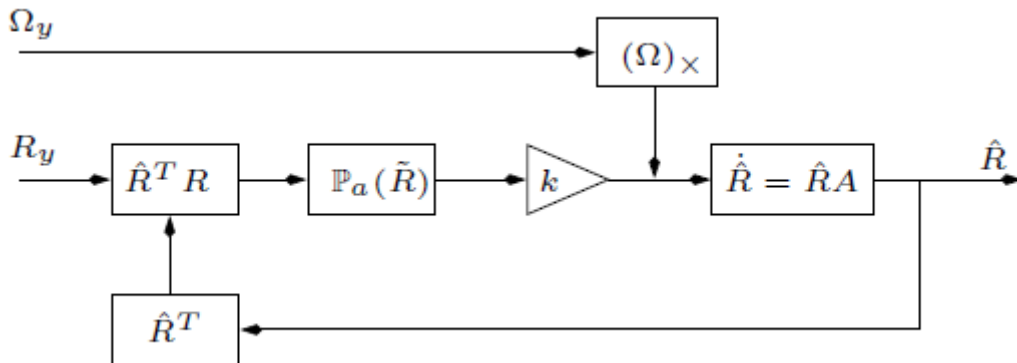
Továbbá a magnetométer és gyorsulásmérő adatokból rekonstruált forgatómátrixot R_y , illetve R mátrix becslését $\hat{R}$.

A világ és test koordináta-rendszerek mellett definiáljuk még $\{E\}$ -t, az állapotbecslő viszonyítási-rendszerét. Ekkor meghatározhatjuk a hiba együtthatót, mint az elforgatást a test és az állapotbecslő koordináta-rendszerek közt.

$$\mathcal{R} = \hat{R}^T R_y: E \rightarrow B \quad (4.24)$$

Ebből kifolyólag a szűrő célja, hogy az állapotbecslő és a test koordináta-rendszerek közt az elforgatást megszüntesse.

A fenti fogalmak segítségével felírható az említett cikkekben bemutatott ún. egyszerűsített passzív CF, amely alább látható:



ábra 12

és ahol $P_a(\mathcal{R})$ a hiba mátrix anti-szimmetrikus vetülete:

$$P_a(R) = \frac{1}{2}(R - R^T) \quad (4.26)$$

illetve k a szűrő egyetlen paramétere.

4.2.4. Madgwick szűrő

A Madgwick-szűrő egy hiba visszacsatolásos algoritmus, amely kvaternió reprezentáción implementál egy orientáció szűrőt.

Határozzuk meg az orientáció változását kvaternió formában az alábbi egyenlettel:

$$\dot{q}_{\omega,t} = \frac{1}{2} \hat{q}_{est,t-1} \otimes s_{\omega,t} \quad (4.27)$$

ahol $s_{\omega,t} = [0 \quad \omega_x \quad \omega_y \quad \omega_z]^T$ a giroszkóp mérések kvaternió formában.

A kvaternió derivált integrálásával az orientáció meghatározható a giroszkóp mérési hibáiból származó zavarok kompenzálása után.

A kompenzációhoz szükséges hibát referencia irányvektorok segítségével hozhatjuk létre.

$$f(\hat{q}_E, \hat{d}^E, \hat{s}_s) = \hat{q}_E \otimes \hat{d}^E \otimes \hat{q}_E - \hat{s}_s \quad (4.28)$$

Ahol E a világ míg S a szenzor koordináta-rendszert jelöli. Továbbá $\hat{d}^E$ a referencia irányvektor becslése világkoordináta-rendszerben, $\hat{s}_s$ pedig a vektoron végzett mérések szenzor koordináta-rendszerben. Az algoritmus feladata biztosítani, hogy a hiba, vagyis az elforgatás a valós és a becsült orientáció közt, nullához konvergáljon.

Ezt a legegyszerűbben és a legkisebb számítási igénnyel az ún. gradiens süllyedéssel módszerrel tehetjük meg[19]:

$$q_{\nabla,t} = \hat{q}_{est,t-1} - \mu_t \frac{\nabla f}{\|\nabla f\|} \quad (4.29)$$

A fenti egyenlet f vektorfüggvény és annak Jakobiánusával kiszámítható. Amennyiben a referencia vektor a Föld mágneses és gravitációs tere, a vektorfüggvények egyszerűsített formái és Jakobiánusai az alábbi ábrákon láthatók.

$$\begin{aligned} \mathbf{f}_g(\hat{\mathbf{q}}_E^S, \hat{\mathbf{a}}^S) &= \begin{bmatrix} 2(q_2q_4 - q_1q_3) - a_x \\ 2(q_1q_2 + q_3q_4) - a_y \\ 2(\frac{1}{2} - q_2^2 - q_3^2) - a_z \end{bmatrix} \\ \mathbf{J}_g(\hat{\mathbf{q}}_E^S) &= \begin{bmatrix} -2q_3 & 2q_4 & -2q_1 & 2q_2 \\ 2q_2 & 2q_1 & 2q_4 & 2q_3 \\ 0 & -4q_2 & -4q_3 & 0 \end{bmatrix} \end{aligned} \quad (4.30), (4.31)$$

$$\mathbf{f}_b(\hat{\mathbf{q}}_E^S, \hat{\mathbf{b}}^E, \hat{\mathbf{m}}^S) = \begin{bmatrix} 2b_x(0.5 - q_3^2 - q_4^2) + 2b_z(q_2q_4 - q_1q_3) - m_x \\ 2b_x(q_2q_3 - q_1q_4) + 2b_z(q_1q_2 + q_3q_4) - m_y \\ 2b_x(q_1q_3 + q_2q_4) + 2b_z(0.5 - q_2^2 - q_3^2) - m_z \end{bmatrix} \quad (4.32)$$

$$\mathbf{J}_b(\hat{\mathbf{q}}_E^S, \hat{\mathbf{b}}^E) = \begin{bmatrix} -2b_zq_3 & 2b_zq_4 & -4b_xq_3 - 2b_zq_1 \\ -2b_xq_4 + 2b_zq_2 & 2b_xq_3 + 2b_zq_1 & 2b_xq_2 + 2b_zq_4 \\ 2b_xq_3 & 2b_xq_4 - 4b_zq_2 & 2b_xq_1 - 4b_zq_3 \\ & & -4b_xq_4 + 2b_zq_2 \\ & & -2b_xq_1 + 2b_zq_3 \\ & & 2b_xq_2 \end{bmatrix} \quad (4.33)$$

Továbbá a teljes orientáció meghatározásához kvaternió formában a mágneses és gyorsulási méréseket az alábbi módon kombinálhatjuk:

$$\begin{aligned} \mathbf{f}_{g,b}(\hat{\mathbf{q}}_E^S, \hat{\mathbf{a}}^S, \hat{\mathbf{b}}^E, \hat{\mathbf{m}}^S) &= \begin{bmatrix} \mathbf{f}_g(\hat{\mathbf{q}}_E^S, \hat{\mathbf{a}}^S) \\ \mathbf{f}_b(\hat{\mathbf{q}}_E^S, \hat{\mathbf{b}}^E, \hat{\mathbf{m}}^S) \end{bmatrix} \\ \mathbf{J}_{g,b}(\hat{\mathbf{q}}_E^S, \hat{\mathbf{b}}^E) &= \begin{bmatrix} \mathbf{J}_g^T(\hat{\mathbf{q}}_E^S) \\ \mathbf{J}_b^T(\hat{\mathbf{q}}_E^S, \hat{\mathbf{b}}^E) \end{bmatrix} \end{aligned} \quad (4.34)$$

Amennyiben mindkét mérés rendelkezésünkre áll, a hibafüggvény gradiense, vagyis az orientáció deriváltjának hibája:

$$\nabla \mathbf{f} = \begin{cases} \mathbf{J}_g^T(\hat{\mathbf{q}}_{est,t-1}^S) \mathbf{f}_g(\hat{\mathbf{q}}_{est,t-1}^S, \hat{\mathbf{a}}_t^S) \\ \mathbf{J}_{g,b}^T(\hat{\mathbf{q}}_{est,t-1}^S, \hat{\mathbf{b}}^E) \mathbf{f}_{g,b}(\hat{\mathbf{q}}_{est,t-1}^S, \hat{\mathbf{a}}^S, \hat{\mathbf{b}}^E, \hat{\mathbf{m}}^S) \end{cases} \quad (4.35)$$

A visszacsatolási ág ismeretében a kompenzált adatokból számolt orientációt az alábbi egyenletek határozzák meg:

$$\hat{\mathbf{q}}_{est,t}^S = \hat{\mathbf{q}}_{est,t-1}^S + \dot{\hat{\mathbf{q}}}_{est,t}^S \Delta t \quad (4.36)$$

5. Kísérleti eredmények

A kísérleteim során MATLAB szoftvercsomagban reprodukáltam a kaszkád komplementis szűrő architektúrát, illetve a harmadik fejezetben ismertetett módszerrel több adathalmazt generáltam különböző paraméterekkel (A és B mellékletek).

A reprodukált szűrőt annak paramétereinek változtatása után minden adathalmazon lefuttattam és meghatároztam azok pontosságát.

A kísérletek során az algoritmusok pontosságát az erre a célra gyakran használt módszerrel végeztem, amely a becslések és a referencia jel ismeretében azok négyzetes közepének meghatározását jelentik:

$$RMSE = \sqrt{\frac{1}{n} \sum_{k=0}^n} \quad (5.1)$$

A programkódok és kísérletek elvégzéséhez generált adathalmazok megtalálhatóak a hagyományos és elektronikus mellékletben, míg az adatok generálásához, továbbá a reprodukcióhoz felhasznált szűrő paraméterek, illetve az egyes eredmények vizuális összehasonlítása a fejezetben található meg.

5.1. Adatgeneráció paraméterei

		Constant Bias	Bias Instability	Misalignment	Noise Density	Random Walk	Temp Bias	Accel Bias
Alap értékek	Gyroscope	$\begin{bmatrix} 0.15 \\ 0.2 \\ 0.1 \end{bmatrix}$	$\begin{bmatrix} 0.015 \\ 0.03 \\ 0.022 \end{bmatrix}$	$m * \begin{bmatrix} 4 \\ 2.5 \\ 3 \end{bmatrix}$	$\begin{bmatrix} 0.0061 \\ 0.0062 \\ 0.0095 \end{bmatrix}$	$\begin{bmatrix} 0.006 \\ 0.009 \\ 0.0065 \end{bmatrix}$	$\begin{bmatrix} 0.003 \\ 0.006 \\ 0.005 \end{bmatrix}$	$\begin{bmatrix} 0.025 \\ 0.0235 \\ 0.027 \end{bmatrix}$
	Acclerometer	$\begin{bmatrix} 0.15 \\ 0.16 \\ 0.34 \end{bmatrix}$	$\begin{bmatrix} 0.02 \\ 0.01 \\ 0.03 \end{bmatrix}$	$m * \begin{bmatrix} 4 \\ 6 \\ 5 \end{bmatrix}$	$\begin{bmatrix} 0.07 \\ 0.06 \\ 0.095 \end{bmatrix}$	$\begin{bmatrix} 0.006 \\ 0.009 \\ 0.0065 \end{bmatrix}$	$\begin{bmatrix} 0.003 \\ 0.002 \\ 0.002 \end{bmatrix}$	
	Magnetometer	$\begin{bmatrix} 0.1 \\ 0.16 \\ 0.19 \end{bmatrix}$	$\begin{bmatrix} 0.0015 \\ 0.0018 \\ 0.0025 \end{bmatrix}$	$m * \begin{bmatrix} 6 \\ 5 \\ 4 \end{bmatrix}$	$\begin{bmatrix} 0.0096 \\ 0.024 \\ 0.015 \end{bmatrix}$	$\begin{bmatrix} 0.0028 \\ 0.0023 \\ 0.0024 \end{bmatrix}$	$\begin{bmatrix} 0.0103 \\ 0.015 \\ 0.097 \end{bmatrix}$	
AH1	Gyroscope	0.01 * Alap paraméterek, kivéve misalignment						
	Acclerometer							
	Magnetometer							
AH2	Gyroscope	0.02 * Alap paraméterek, kivéve misalignment						
	Acclerometer							
	Magnetometer							
AH3	Gyroscope	0.03 * Alap paraméterek, kivéve misalignment						
	Acclerometer							
	Magnetometer							

Táblázat 1

5.2. Eredmények

AH1 - kivizsgálása

α	RMSE	ω_n 0.001	ω_n 0.002	ω_n 0.003	ω_n 0.004	ω_n 0.005	ω_n 0.006	ω_n 0.007	ω_n 0.008	ω_n 0.009	ω_n 0.01
0.1	Roll	1.038	1.038	1.038	1.038	1.038	1.038	1.038	1.038	1.038	1.038
	Pitch	1.512	1.512	1.512	1.512	1.512	1.512	1.512	1.512	1.512	1.512
	Yaw	9.904	9.904	9.904	9.904	9.904	9.904	9.904	9.904	9.904	9.904
0.2	Roll	1.032	1.032	1.032	1.032	1.032	1.032	1.032	1.032	1.032	1.032
	Pitch	1.447	1.447	1.447	1.447	1.447	1.447	1.447	1.447	1.447	1.447
	Yaw	9.888	9.888	9.888	9.888	9.888	9.888	9.888	9.888	9.888	9.888
0.3	Roll	1.0275	1.027	1.027	1.027	1.027	1.027	1.027	1.027	1.027	1.027
	Pitch	1.3873	1.387	1.387	1.387	1.387	1.387	1.387	1.387	1.387	1.387
	Yaw	9.8735	9.873	9.873	9.873	9.873	9.873	9.873	9.873	9.873	9.873
0.4	Roll	1.0231	1.023	1.023	1.023	1.023	1.023	1.023	1.023	1.023	1.023
	Pitch	1.3331	1.333	1.333	1.333	1.333	1.333	1.333	1.333	1.333	1.333
	Yaw	9.8611	9.861	9.861	9.861	9.861	9.861	9.861	9.861	9.861	9.861
0.5	Roll	1.0194	1.019	1.019	1.019	1.019	1.019	1.019	1.019	1.019	1.019
	Pitch	1.2852	1.285	1.285	1.285	1.285	1.285	1.285	1.285	1.285	1.285
	Yaw	9.8506	9.850	9.850	9.850	9.850	9.850	9.850	9.850	9.850	9.850
0.6	Roll	1.0167	1.016	1.016	1.016	1.016	1.016	1.016	1.016	1.016	1.016
	Pitch	1.2443	1.244	1.244	1.244	1.244	1.244	1.244	1.244	1.244	1.244
	Yaw	9.8420	9.842	9.842	9.842	9.842	9.842	9.842	9.842	9.842	9.842
0.7	Roll	1.0152	1.015	1.015	1.015	1.015	1.015	1.015	1.015	1.015	1.015
	Pitch	1.2109	1.211	1.211	1.211	1.211	1.211	1.211	1.211	1.211	1.211
	Yaw	9.8353	9.835	9.835	9.835	9.835	9.835	9.835	9.835	9.835	9.835
0.8	Roll	1.0155	1.015	1.015	1.015	1.015	1.015	1.015	1.015	1.015	1.015
	Pitch	1.1853	1.185	1.185	1.185	1.185	1.185	1.185	1.185	1.185	1.185
	Yaw	9.83	9.83	9.83	9.83	9.83	9.83	9.83	9.83	9.83	9.83
0.9	Roll	1.0211	1.021	1.021	1.021	1.021	1.021	1.021	1.021	1.021	1.021
	Pitch	1.1650	1.165	1.165	1.165	1.165	1.165	1.165	1.165	1.165	1.165
	Yaw	9.8239	9.823	9.823	9.823	9.823	9.823	9.823	9.823	9.823	9.823

AH2 - kivizsgálása

α	RMSE	ω_n 0.01	ω_n 0.02	ω_n 0.03	ω_n 0.04	ω_n 0.05	ω_n 0.06	ω_n 0.07	ω_n 0.08	ω_n 0.09	ω_n 0.1
0.9	Roll	1.0210	1.021	1.021	1.021	1.021	1.021	1.021	1.021	1.021	1.021
	Pitch	1.1650	1.165	1.165	1.165	1.165	1.165	1.165	1.165	1.165	1.165
	Yaw	9.8239	9.824	9.824	9.824	9.824	9.824	9.824	9.824	9.824	9.824

AH3 - kivizsgálása

α	RMSE	ω_n 1	ω_n 2	ω_n 3	ω_n 4	ω_n 5	ω_n 6	ω_n 7	ω_n 8	ω_n 9
0.9	Roll	1.0195	1.0175	1.0157	1.0144	1.0135	1.0129	1.0127	1.0127	1.0131
	Pitch	1.1671	1.1696	1.1721	1.1744	1.1766	1.1787	1.1808	1.183	1.1854
	Yaw	9.8252	9.8268	9.8283	9.8295	9.8305	9.8313	9.832	9.8326	9.8332

Konklúzió

A szakdolgozat keretein belül áttekintettük az orientáció reprezentációjának leggyakoribb módszereit, amely alapján kijelenthető, hogy az Euler-szög reprezentáció korlátoltan vagy nem alkalmas az orientációt illető számítások elvégzésére. Azokra legalkalmasabb a kvaternió reprezentáció, mivel teljesen szingularitásoktól mentesek.

A munkám során implementáltam egy Euler-szög reprezentáción alapuló szűrő architektúrát, amelyet olyan adatokon teszteltem, amikkel a számítások során nem lépnek fel az említett szingularitások.

Az Euler-szög reprezentációt alkalmazó architektúrák a legegyszerűbb módja egy AHRS rendszer szemléltetésének és implementálásának, azonban más, mint a kvaternió alapú alkalmazások a gyakorlatban sokkal célszerűbbek azok stabilabb jellegükből kifolyólag.

Bemutatásra kerültek az orientációnak és annak deriváltjának kinyerésére alkalmas módszerek, amelyek szintén szingularitásokat tartalmaznak, így célszerűbb olyan forgatómátrix vagy kvaternió reprezentációra alkalmazható módszereket használni, amelyek ezektől mentesek.

A MATLAB szoftvercsomag által biztosított szenzor mérési modell bemutatásra és alkalmazásra került a munkám során. A kísérletek alapján elmondható, hogy bizonyos fizikai mennyiségek sikerrel szintetizálhatók megfelelő részletességű matematikai modell segítségével. Ily módon az algoritmus validálása elvégezhető bonyolult módszerek, illetve költséges céleszközök nélkül is.

Melléklet A

Referencia adatok létrehozása

A kísérletek kiértékeléséhez használt adatok generálása egy egység magnitúdójú irányvektor és az a körül értelmezett elfordulás által alkotott tengely-szög rotációs pár létrehozásával történt. Majd azokat Euler-szög formába transzformálásuk után deriváltam, amely deriváltakból a [3]-ban definiált direkt kinematikai módszerrel meghatároztam a szögsebességet test koordináta-rendszerben.

Az előre definiált gravitációs- és mágneses térvektorokat test koordináta-rendszerbe alakítom, az esetleges összevetés szempontjából, azonban a továbbiakban nincsenek felhasználva.

Az orientáció kvaternió formában való meghatározása után a B mellékletben leírt módon szintetizálhatók a szenzor adatok.

Forráskód

```
clear, close all
delta_T = 7e-3;
L = 24000;
time = 0 : delta_T : (L - 1) * delta_T;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%% Generate Ground Truth Data %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%% Create Axis - Angle paramters for accurate Euler-Angle generation %%%%%%%%%%
rotDirection = [1; 0; 0];
rm = eul2rotm([0 pi/4 0], 'XYZ');
rotDirection = rm * rotDirection;

deltaVtheta = (0.25*pi) / (L/3);
Vtheta = zeros(1, L);
axang = zeros(L, 4);
for i = 1 : L
    axang(i, :) = [transpose(rotDirection) 0];
end
for i = (L/3) : ((2 * L)/3)
    Vtheta(i) = Vtheta(i - 1) + deltaVtheta;
    axang(i, :) = [transpose(rotDirection) Vtheta(i)];
end
for i = ((2 * L)/3) : L
    axang(i, :) = axang(((2 * L)/3), :);
end
refRotM = axang2rotm(axang);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%% Generate Euler angles %%%%%%%%%%
refEulAngles = rotm2eul(refRotM, 'XYZ');

testRoll = transpose(refEulAngles(:, 1));
testPitch = transpose(refEulAngles(:, 2));
testYaw = transpose(refEulAngles(:, 3));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%% Differentiate angles %%%%%%%%%%
AngVel = zeros(3, L);
```

```

EulerAngDot = zeros(3, L);
DirectJacobian = zeros(3, 3, L);
for i = 2 : L
    EulerAngDot(1, i) = (testRoll(i) - testRoll(i - 1))/delta_T; %[rad/s]
    EulerAngDot(2, i) = (testPitch(i) - testPitch(i - 1))/delta_T; %[rad/s]
    EulerAngDot(3, i) = (testYaw(i) - testYaw(i - 1))/delta_T; %[rad/s]
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Attitude Direct Kinematics %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Calculate Body Frame Angular Velcity from Euler Angle time derivative%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i = 1 : L
    DirectJacobian(:, :, i) = [1 0 -sin(testPitch(i)); ...
                                0 cos(testRoll(i)) sin(testRoll(i))*cos(testPitch(i)); ...
                                0 -sin(testRoll(i)) cos(testRoll(i))*cos(testPitch(i))];
    AngVel(:, i) = DirectJacobian(:, :, i) * EulerAngDot(:, i);
end
AngVel = transpose(AngVel);

%Generate magnetic and gravity vector
RefGravitationalField = [0; 0; 9.81]; %[m/s2]
RefMagneticField = [22.76; 0; 40.72]; % [uT] / Data based on the magnetic field calculator of the
                                %National Centers for Envireomental Information
                                %National Oceanic and Atmospheric Administration
                                %Lat :45.614922 Long: 20.035061 [deg] SeaLvL : 77[m]

qRefGravitationalField = [0 0 0 1];
qRefMagneticField = [0 22.76 0 40.72];
qRefMagneticField = qRefMagneticField / norm([0 22.76 0 40.72]);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Determine Orientation in Rotation Matrices & Quaternions %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
eulerAngles = zeros(L, 3);
eulerAngles(:, 1) = testRoll; %[rad]
eulerAngles(:, 2) = testPitch; %[rad]
eulerAngles(:, 3) = testYaw; %[rad]
rotationMatricesXYZ = eul2rotm(eulerAngles, 'XYZ');
qOrientation = eul2quat(eulerAngles, 'XYZ');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Transform field vectors %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
qOrientationConjugate = quatconj(qOrientation);
BodyGravity = quatmultiply(qOrientationConjugate, quatmultiply(qRefGravitationalField,
qOrientation));
BodyMagField = quatmultiply(qOrientationConjugate, quatmultiply(qRefMagneticField, qOrientation));

```

Melléklet B

Szenzor adat szintézis

A MATLAB Navigation Toolboxa által biztosított IMU objektum létrehozása után meghatározhatók a szenzorok különböző paraméterei, majd a referencia tér és szögsebesség vektorok betáplálása után megkapjuk a test koordináta-rendszerben értelmezett szenzor adatokat a megfelelő mérési hibákkal terhelve.

Forráskód

```
##### Call Ground Truth Data Generator #####

createNewGroundTruth = 0;

if createNewGroundTruth
    AHRS_GroundTruthData
end

SET_MISALIGNMENT = 1;

misalignment_scalar = 0.5;

##### Generate IMU Object #####

IMU = imuSensor('accel-gyro-mag');
IMU.SampleRate = 1/delta_T;
IMU.MagneticField = transpose(RefMagneticField);

##### Set Gyroscope parameters #####
GyroParamScalar = 0.01;
IMU.Gyroscope.AccelerationBias = GyroParamScalar * [0.025 0.0235 0.027]; IMU.Gyroscope.ConstantBias = GyroParamScalar * [0.15 0.2 0.1];
IMU.Gyroscope.BiasInstability = GyroParamScalar * [0.015 0.03 0.022];
if SET_MISALIGNMENT IMU.Gyroscope.AxesMisalignment = misalignment_scalar * [4 2.5 3]; end
IMU.Gyroscope.NoiseDensity = GyroParamScalar * [0.0061, 0.0062, 0.0095]; IMU.Gyroscope.RandomWalk = GyroParamScalar * [0.004 0.005 0.006];
IMU.Gyroscope.TemperatureBias = GyroParamScalar * [0.003 0.006 0.005];

##### Set Accelerometer parameters #####
AccelParamScalar = 0.01;
IMU.Accelerometer.ConstantBias = AccelParamScalar * [0.15 0.16 0.34]; IMU.Accelerometer.BiasInstability = AccelParamScalar * [0.02 0.01 0.03];
if SET_MISALIGNMENT IMU.Accelerometer.AxesMisalignment = misalignment_scalar * [4 6 5]; end
IMU.Accelerometer.NoiseDensity = AccelParamScalar * [0.07, 0.06, 0.095]; %4 3 5
IMU.Accelerometer.RandomWalk = AccelParamScalar * [0.006 0.009 0.0065];
IMU.Accelerometer.TemperatureBias = AccelParamScalar * [0.003 0.002 0.002];

##### Set Magnetometer parameters #####
MagmeterParamScalar = 0.01;
IMU.Magnetometer.ConstantBias = MagmeterParamScalar * [0.1 0.16 0.19];
IMU.Magnetometer.BiasInstability = MagmeterParamScalar * [0.0015 0.0018 0.0025];
if SET_MISALIGNMENT IMU.Magnetometer.AxesMisalignment = misalignment_scalar * [6 5 4]; end
IMU.Magnetometer.NoiseDensity = MagmeterParamScalar * [0.096, 0.024, 0.015]; %3 5 4
IMU.Magnetometer.RandomWalk = MagmeterParamScalar * [0.0028 0.0023 0.0024];
IMU.Magnetometer.TemperatureBias = MagmeterParamScalar * [0.0103 0.015 0.097];

##### Generate measurements with IMU model and Ground Truth data #####
MagField = zeros(L, 3);
Accel = zeros(L, 3);
for i = L
    Accel(i, :) = transpose(RefGravitationalField);
```

```
end  
[Accel, AngVel, MagField] = IMU(Accel, AngVel, rotationMatricesXYZ); %[m/s/s]; [rad/s]; [uT]
```

Melléklet C

Orientáció szűrő, pontosság kiértékelése és eredmények vizuális összevetése

Forráskód

```
close all
rad2deg = 57.29578;
deg2rad = 0.0174532925;

delta_t = 7e-3;
time = 0 : delta_t : delta_t * (L - 1);

omega_n = 0.001;
alpha = 0.98;

NormAccel = zeros(L, 3);
NormMagField = zeros(L, 3);

for i = 1 : L
    norm = sqrt(Accel(i, 1) * Accel(i, 1) + Accel(i, 2) * Accel(i, 2) + Accel(i, 3) * Accel(i, 3));
    NormAccel(i, 1) = Accel(i, 1) / norm;
    NormAccel(i, 2) = Accel(i, 2) / norm;
    NormAccel(i, 3) = Accel(i, 3) / norm;

    norm = sqrt(MagField(i, 1) * MagField(i, 1) + MagField(i, 2) * MagField(i, 2) + MagField(i, 3) *
MagField(i, 3));
    NormMagField(i, 1) = MagField(i, 1) / norm;
    NormMagField(i, 2) = MagField(i, 2) / norm;
    NormMagField(i, 3) = MagField(i, 3) / norm;
end

Phi = zeros(L, 1);
Theta = zeros(L, 1);
Psi = zeros(L, 1);
for i = 1 : L
    Phi(i, 1) = atan2(-NormAccel(i, 2), sign(NormAccel(i, 3)) * sqrt(NormAccel(i, 3)^2 + NormAccel(i,
1)^2));
    Theta(i, 1) = atan2(NormAccel(i, 1), sqrt(NormAccel(i, 2)^2 + NormAccel(i, 3)^2));
end
inclination = [Phi, Theta, Psi];
rotM_YX = eul2rotm(inclination, 'ZYX');
inclinationCompensatedMagField = zeros(L, 3);
for i = 1 : L
    inclinationCompensatedMagField(i, :) = rotM_YX(:, :, i) * [NormMagField(i, 1); NormMagField(i,
2); NormMagField(i, 3)];
    Psi(i, 1) = atan2(inclinationCompensatedMagField(i, 2), inclinationCompensatedMagField(i, 1));
end

kp = sqrt(2) * omega_n;
ki = omega_n^2;
AttitudeMeasured = [Phi, Theta, Psi];
AttitudeEstimated = zeros(L, 3);
AttitudeDot = zeros(L, 3);
errorFeedback = zeros(L, 3);
biasEstimation = zeros(L, 3);
for i = 2 : L
    SS3 = [1 sin(AttitudeEstimated(i - 1, 1))*tan(AttitudeEstimated(i - 1, 2))
cos(AttitudeEstimated(i - 1, 1))*tan(AttitudeEstimated(i - 1, 2)); ...
0 cos(AttitudeEstimated(i - 1, 1)) -sin(AttitudeEstimated(i - 1, 1)); ...
0 sin(AttitudeEstimated(i - 1, 1))/cos(AttitudeEstimated(i - 1, 2))
cos(AttitudeEstimated(i - 1, 1))/cos(AttitudeEstimated(i - 1, 2))];
    AttitudeDot(i, :) = SS3 * [AngVel(i, 1); AngVel(i, 2); AngVel(i, 3)];

    errorFeedback(i, :) = AttitudeMeasured(i, :) - AttitudeEstimated(i - 1, :);

    biasEstimation(i, :) = ki * (errorFeedback(i - 1, :) + errorFeedback(i, :) * delta_t);

    AttitudeEstimated(i, :) = AttitudeEstimated(i - 1, :) + delta_t * (AttitudeDot(i, :) +
(biasEstimation(i, :) + kp * errorFeedback(i, :)));
```

```

        AttitudeEstimated(i, :) = alpha * AttitudeEstimated(i, :) + (1 - alpha) * AttitudeMeasured(i,
:);
end

sum1 = 0;
sum2 = 0;
sum3 = 0;
for i = 1 : L
    sum1 = sum1 + ((testRoll(1, i) * 57.29578) - (AttitudeEstimated(i, 1) * 57.29578))^2;
    sum2 = sum2 + ((testPitch(1, i) * 57.29578) - (AttitudeEstimated(i, 2) * 57.29578))^2;
    sum3 = sum3 + ((testYaw(1, i) * 57.29578) - (AttitudeEstimated(i, 3) * 57.29578))^2;
end
RMSE_Roll = sqrt(sum1 / L);
RMSE_Pitch= sqrt(sum2 / L);
RMSE_Yaw= sqrt(sum3 / L);

figure
subplot(3, 1, 1), plot(time, 57.29578 * testRoll, time, 57.29578 * AttitudeEstimated(:, 1)), grid,
title('Roll'), legend('Reference', 'Reconstructed');
subplot(3, 1, 2), plot(time, 57.29578 * testPitch, time, 57.29578 * AttitudeEstimated(:, 2)), grid,
title('Pitch'), legend('Reference', 'Reconstructed');
subplot(3, 1, 3), plot(time, 57.29578 * testYaw, time, 57.29578 * AttitudeEstimated(:, 3)), grid,
title('Yaw'), legend('Reference', 'Reconstructed');

figure
subplot(3, 1, 1), plot(time, AttitudeMeasured(:, 1), time, testRoll), grid, title('Roll'),
legend('Measured', 'Reference')
subplot(3, 1, 2), plot(time, AttitudeMeasured(:, 2), time, testPitch), grid, title('Pitch'),
legend('Measured', 'Reference')
subplot(3, 1, 3), plot(time, AttitudeMeasured(:, 3), time, testYaw), grid, title('Yaw'),
legend('Measured', 'Reference')

figure
subplot(3, 1, 1), plot(time, NormMagField(:, 1), time, inclinationCompensatedMagField(:, 1)), grid,
title('Magnetic X - Axis'), legend('Normalized Body', 'Inclination Compensated')
subplot(3, 1, 2), plot(time, NormMagField(:, 2), time, inclinationCompensatedMagField(:, 2)), grid,
title('Magnetic Y - Axis'), legend('Normalized Body', 'Inclination Compensated')
subplot(3, 1, 3), plot(time, NormMagField(:, 3), time, inclinationCompensatedMagField(:, 3)), grid,
title('Magnetic Z - Axis'), legend('Normalized Body', 'Inclination Compensated')

figure
subplot(3, 1, 1), plot(time, AttitudeDot(:, 1), time, EulerAngDot(1, :)), grid, title('RollDot'),
legend('Measured', 'Reference')
subplot(3, 1, 2), plot(time, AttitudeDot(:, 2), time, EulerAngDot(2, :)), grid, title('PitchDot'),
legend('Measured', 'Reference')
subplot(3, 1, 3), plot(time, AttitudeDot(:, 3), time, EulerAngDot(3, :)), grid, title('YawDot'),
legend('Measured', 'Reference')

```

Irodalom

- [1] - Judy, Jack. (2001). Microelectromechanical systems (MEMS): Fabrication, design and applications. Smart Materials and Structures. 10. 1115-1134. 10.1088/0964-1726/10/6/301.
- [2] - Ahmad, Norhafizan & Raja Ghazilla, Raja Ariffin & Khairi, Nazirah & Kasi, Vijayabaskar. (2013). Reviews on Various Inertial Measurement Unit (IMU) Sensor Applications. International Journal of Signal Processing Systems. 1. 256-262. 10.12720/ijsp.1.2.256-262.
- [3] - Olguín Díaz, Ernesto (2019). 3D Motion of Rigid Bodies: A Foundation for Robot Dynamics Analysis. 2198-4182 10.1007/978-3-030-04275-2.
- [4] – Fletcher Dunn, Ian Parberry (2011). 3D Math Primer for Graphics and Game Development: 2nd edition. 978-1-4398-6981-9.
- [5] – Kuipers, Jack B. (1921). Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality. 0-691-05872-5.
- [6] – Madgwick, Sebastian O.H. (2014). AHRS algorithms and calibration solutions to facilitate new applications using low-cost MEMS. University of Bristol. Ph.D. Thesis.
- [7] - D. K. Shaeffer, "MEMS inertial sensors: A tutorial overview," in IEEE Communications Magazine, vol. 51, no. 4, pp. 100-109, April 2013, doi: 10.1109/MCOM.2013.6495768.
- [8] – The MathWorks Inc. (2019-2022). Navigation Toolbox Reference.
- [9] –Poddar, Shashi & Kumar, Vipin & Kumar, Amod. (2016). A Comprehensive Overview of Inertial Sensor Calibration Techniques. Journal of Dynamic Systems, Measurement, and Control. 139. 10.1115/1.4034419.
- [10] – Kok, Manon & Hol, J.D. & Schön, Thomas & Gustafsson, F. & Luinge, Hendrik. (2012). Calibration of a magnetometer in combination with inertial sensors. 15th International Conference on Information Fusion, FUSION 2012. 787-793.
- [11] – Sarcevic, Péter. (2018). New methods in application of inertial and magnetic sensors in online pattern recognition problems. UNIVERSITY OF SZEGED FACULTY OF SCIENCE AND INFORMATICS DEPARTMENT OF TECHNICAL INFORMATICS. Doctoral School of Computer Science.
- [12] – Pedley, Mark. (2013). Tilt Sensing Using a Three-Axis Accelerometer. NXP AN3461

- [13] – Ozyagcilar, Talat. (2015). Implementing a Tilt Compensated eCompass using Accelerometer and Magnetometer Sensors. NXP AN4248
- [14] – Hong, Sung. (2003). Fuzzy logic based closed-loop strapdown attitude system for unmanned aerial vehicle (UAV). *Sensors and Actuators A: Physical*. 107. 109-118. 10.1016/S0924-4247(03)00353-4.
- [15] - Mahony, Robert & Hamel, T. & Pflimlin, J.-M. (2006). Complementary filter design on the special orthogonal group SO(3). 1477 - 1484. 10.1109/CDC.2005.1582367.
- [16] - R. Mahony, Tarek Hamel, Jean-Michel Pflimlin. Nonlinear Complementary Filters on the Special Orthogonal Group. *IEEE Transactions on Automatic Control*, Institute of Electrical and Electronics Engineers, 2008, 53 (5), pp.1203-1217. 10.1109/TAC.2008.923738. hal-00488376
- [17] – Narkhede, Parag & Poddar, Shashi & Walambe, Rahee & Ghinea, George & Kotecha Ketan. (2021). Cascaded Complementary Filter Architecture for Sensor Fusion in Attitude Estimation. *Sensors* 2021, 21(6), 1937; <https://doi.org/10.3390/s21061937> (elérhető 2022. 06. 07)
- [18] - Hyung Gi Min & Eun Tae Jeung. (2015). Complementary Filter Design for Angle Estimation using MEMS Accelerometer and Gyroscope. Robotics Lab., NTREX Ltd., co., Juan-Dong 5-38, Nam-Gu, Incheon, Korea Department of Control and Instrumentation, Changwon National University, Changwon, 641-773, Korea
- [19] - Madgwick, Sebastian & Harrison, Andrew & Vaidyanathan, Ravi. (2011). Estimation of IMU and MARG orientation using a gradient descent algorithm. *IEEE ... International Conference on Rehabilitation Robotics : [proceedings]*. 2011. 5975346. 10.1109/ICORR.2011.5975346.