ÍNDEX

1. INTRODUCCIÓ

1.1. Antecedents

Un INS és un sistema de navegació inercial que utilitza la informació que li proporcionen diversos sensors (acceleròmetres i giroscopis) per calcular els canvis en posició, orientació i velocitat d'un objecte determinat. És usat en vaixells, aeronaus, submarins, míssils, naus espacials, etc.

Els INS es composen essencialment de dues parts: una unitat de mesura inercial (IMU), que està formada per tres acceleròmetres i tres giroscopis muntats de forma ortogonal de manera que permeten prendre mesures en les tres dimensions de l'espai; i una unitat de processament de la navegació, que, mitjançant les equacions de navegació, és l'encarregada de determinar la posició, orientació i velocitat del vehicle amb les dades que proporciona la IMU.

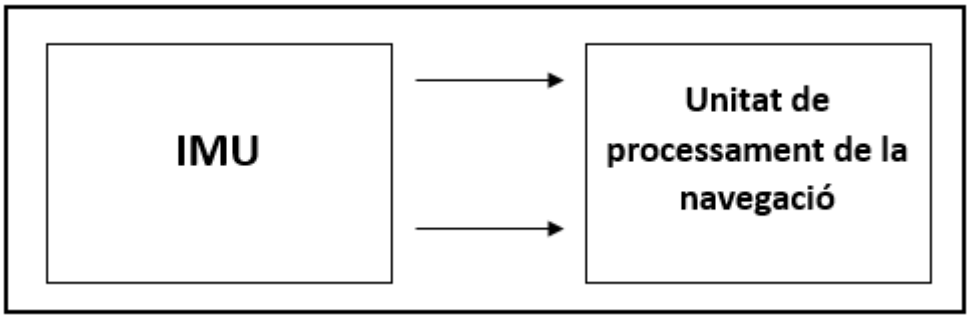


Figura 1.1: Esquema d'un INS

Conèixer les tècniques de processat de les dades és interessant perquè ens permet construir un INS a partir d'una IMU (que té un cost menor), però també perquè, posteriorment, ens ajudarà a una millor integració amb altres sensors o sistemes de navegació prèviament existents (GPS, velocímetres, etc). Això ens permetrà aconseguir, entre d'altres coses, una millor precisió.

La navegació inercial té un sèrie d'avantatges. Opera de forma continua i ofereix dades a una freqüència alta (mínim 50Hz). Proporciona valors eficaços de l'orientació, la velocitat angular, l'acceleració, la posició i la velocitat. També és invulnerable a les interferències (la qual cosa és important per la utilització militar). No obstant això, la navegació inercial requereix una inicialització de les dades de posició, orientació i velocitat, aquestes s'han d'obtenir amb un sensor extern. A més, la seva precisió es degrada amb el temps degut als errors instrumentals que s'integren a través de les equacions de navegació. Per aquests motius és molt útil integrar els sistemes de navegació inercial amb altres sistemes de mesura (GPS, sensors de velocitat, sensors de pressió, etc) que ens oferiran propietats complementàries i ens ajudaran a reduir l'error.

En el cas de la UdG, al Centre d'Investigació en Robòtica Submarina (CIRS) disposar d'un INS pot ser molt útil per millorar la precisió del sistema de navegació dels seus vehicles autònoms submarins. Un robot submarí quan es submergeix no pot rebre dades del GPS. Per tant, sota l'aigua necessita sistemes de navegació complementaris per guiar-se. Una solució típica és la utilització d'un velocímetre Doppler (DVL) per estimar el desplaçament, un sensor de pressió per mesurar la profunditat i un compàs per determinar l'orientació. La incorporació de la informació provinent d'una INS mitjançant tècniques de fusió sensorial permet millorar la precisió del sistema tot reduint la deriva i atenua els efectes dels errors que pateixen els sensors tradicionals, com per exemple dades corruptes en el DVL o pertorbacions magnètiques que puguin afectar al compàs. Recíprocament, les tècniques de fusió sensorial, gràcies a la redundància de dades, també permeten estimar els biaixos que afecten els sensors inercials i per tant millorem l'estimació de la INS.

1.2. Objecte

L'objecte d'aquest treball final de grau és l'estudi i la implementació de les equacions de navegació que ens permeten trobar la posició, la orientació i velocitat d'un objecte determinat a partir de les dades que ens proporciona una IMU de baix cost. D'aquesta manera aconseguiríem un sistema de navegació inercial (INS).

Addicionalment, s'estudiarà i implementarà una versió preliminar de l'algorisme de fusió sensorial basat en el filtre de Kalman que permetrà la posterior integració de la solució inercial amb altres sistemes de navegació ja existents.

1.3. Especificacions i abast

Implementar les equacions de navegació necessàries per processar les dades de la IMU per així obtenir un sistema de navegació inercial (INS). Simular les equacions en un entorn Matlab. Aplicar un filtre de Kalman que ens permetrà estimar els possibles biaixos que pateix la IMU i possibles errors en l'estima de la posició, velocitat i orientació que pateix el INS. Integrar el sistema de navegació inercial amb un receptor GPS mitjançant el filtre de Kalman. D'aquesta manera s'aconseguirà reduir la deriva que pateix el sistema de navegació inercial. Avaluar els resultats obtinguts tant en proves estàtiques com en proves en moviment.

2. DESCRIPCIÓ TECNOLOGIA INERCIAL

En aquest capítol es farà una descripció dels elements més importants que intervenen en els sistemes de navegació inercial. Es començarà fent una explicació de quin tipus de sensor és una IMU i explicarem que és una "Attitude and heading reference Systems" (AHRS). Parlarem de com utilitzant les dades d'aquests sensors es pot arribar a dissenyar un sistema de navegació inercial. La part final del capítol es destinarà a introduir les equacions de navegació necessàries per implementar un INS.

2.1. Descripció IMU

Una unitat de mesura inercial (IMU) és un aparell electrònic format per acceleròmetres i giroscopis que proporcionen dades sobre les acceleracions i les velocitats angulars que pateix un vehicle determinat. Aquesta informació pot ser usada posteriorment per trobar la velocitat, posició i orientació en que es troba el vehicle.

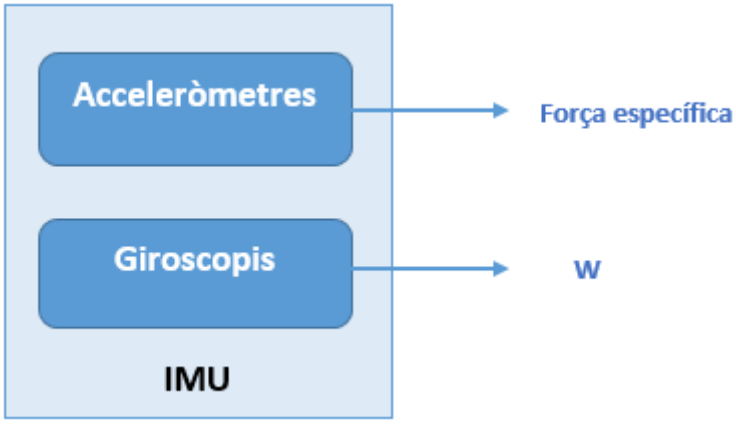


Figura 2.1: Estructura IMU

Els tres acceleròmetres que formen la IMU estant col·locats ortogonalment entre si per així poder prendre les mesures necessàries en les tres dimensions de l'espai. Aquests tres sensors són els encarregats de mesurar la acceleració inercial, també anomenada força específica, que pateix un cos.

La força específica (f) és la força no gravitacional que experimenta un cos per unitat de massa. Això significa que si deixem una IMU quieta veurem com en l'eix vertical enlloc de donar-nos una acceleració de zero ens mostrarà una mesura d'igual magnitud que la força gravitacional ($9.8\ m/s^2$) però en sentit oposat. Per tant, haurem d'aplicar una correcció de la gravetat a l'hora de fer els càlculs, més endavant ho veurem amb més detall.

Els giroscopis també estan col·locats en una forma ortogonal semblant a la dels acceleròmetres. Són els encarregats de mesurar la velocitat angular (w) que pateix l'objecte. Integrant els seus valors podrem determinar els canvis en l'orientació del cos, canvis en el seu roll, pitch i yaw.

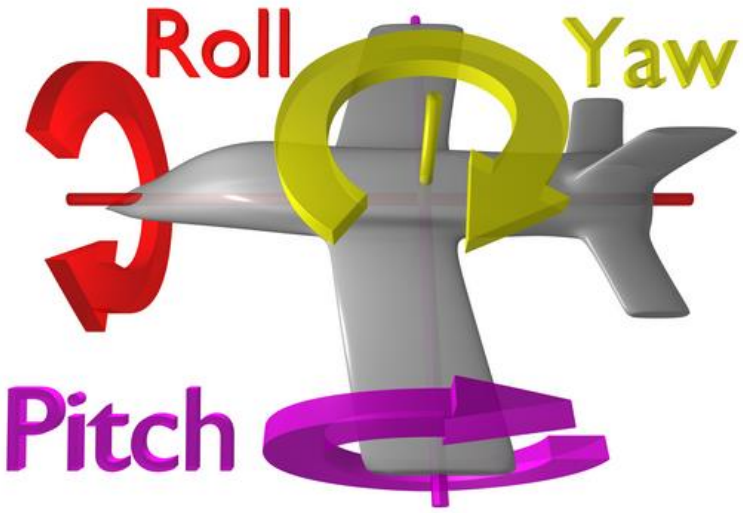


Figura 2.2: Roll, Pitch i Yaw d'una aeronau

Gran part de la deriva que pateix un sistema de navegació inercial prové de la IMU i dels seus petits errors en els mesuraments dels acceleròmetres i giroscopis. És molt important conèixer aquests errors i saber d'on provenen per poder-los corregir i així obtenir uns resultats més fiables.

Actualment, les IMU solen estar construïdes amb sistemes microelectromecànics (MEMS). Això no és res més que tecnologia electrònica i mecànica a una escala molt petita (màxim 1mm).

Tradicionalment les IMU tenen només tres acceleròmetres i tres giroscopis. Per això se les coneix com a IMU de 6DoF (degrees of freedom). Tot i això, cada cop es més comú trobar-se amb IMU de 9DoF. És a dir, aparells que a banda dels sensors tradicionals incorporen també tres magnetòmetres col·locats ortogonalment per mesura vectors de camp magnètic. Com veurem a continuació, aquest tipus d'unitats són particularment adequades per a la construcció de "Attitude and heading reference Systems" (AHRS).

2.2. Descripció AHRS

Tal com el seu nom indica, una AHRS serveix per determinar l'orientació d'un cos en els tres graus de llibertat que té l'espai (roll, pitch i yaw, veure figura 2.2) utilitzant la informació que li proporciona una IMU. És fàcil comprendre que es poden estimar els canvis d'orientació mitjançant la integració de les velocitats angulars mesurades pels giroscopis. Ara bé, al tractar-se de mesures incrementals, sempre es necessari disposar d'alguna mesura absoluta que es pugui utilitzar com a punt de partida pel procés d'estimació. Una opció, es obtenir aquestes referencies d'instruments externs, tot i que si el que es vol és una solució auto continguda el millor és emprar una IMU de 9DoF. En aquest cas, utilitzarem la força específica mesurada en repòs pels acceleròmetres per determinar la direcció del vector gravetat i per tant, per determinar el pitch i roll del sensor (tècnica de leveling, secció 3.5.3.1). D'altra banda, utilitzant els magnetòmetres és possible mesurar el camp magnètic de la Terra i per tant, determinar el yaw respecte el nord magnètic (tècnica de magnètic heading, secció 3.5.3.2).

Tots aquests processos no estan lliures d'errors. Per exemple, la integració al llarg del temps dels errors als giroscopis provocarà un error de deriva en les estimacions dels angles. D'altre banda, els acceleròmetres no perceben només la força gravitacional si el sensor està sotmès a acceleracions. Per tant, les estimacions absolutes de pitch i roll no seran sempre vàlides. Igualment passarà amb el yaw provinent dels magnetòmetres en presencia de pertorbacions electromagnètiques.

Per atenuar aquests efectes adversos, és important la utilització d'avançades tècniques de fusió sensorial.

2.3. Descripció INS

Un sistema de navegació inercial (INS) va un pas més enllà que una AHRS, la qual només estima l'orientació, i és capaç també de mesurar la posició i velocitat d'un cos. El sistema consta d'una IMU que li proporciona dades sobre les acceleracions i les velocitats angulars que pateix el vehicle. Processant aquestes dades i utilitzant la tècnica de la navegació per estima (navegació per estima, capítol 3.1) podem obtenir la velocitat a la que es mou, la posició a la que es troba i l'orientació que té. La navegació per estima es basa en el següent concepte: coneixent la posició a la que es troba un vehicle i la velocitat i direcció en la que es mou, podrem estimar la nova posició al cap de x temps. Més endavant hi entrarem en més detall.

Així doncs, la navegació per estima que executa un INS es basa en lleis de cinemàtica bàsiques. Integrant els valors que ens donen els acceleròmetres i giroscopis i, utilitzant les condicions inicials (velocitat i posició), trobem la velocitat i orientació a l'instant actual. Després, integrant la velocitat aconseguim la posició a la que es troba el vehicle.

Les mesures dels acceleròmetres i giroscopis estan inherentment afectades per errors en la mesura. El procés de navegació per estima integra aquests errors i els converteix en una deriva respecte a la posició real. Aquesta deriva es va acumulant i creixent amb el temps. Aconseguir reduir al mínim l'error és una de les preocupacions més grans alhora de dissenyar un sistema de navegació inercial.

La precisió dependrà en gran part de la qualitat dels sensors inercials. Els INS es poden classificar en cinc grans grups (Paul D. Groves, 2013).

- Marine grade: són els més precisos i solen tenir us militar (míssils, vaixells, submarins...). Poden costar més de 800000 € i el seu error és menor a 1,8 Km després de tot un dia de navegació sense l'ajuda de cap altre sistema de navegació extern.

- Aviation grade: s'utilitzen en avions militars i comercials. Tenen un preu d'uns 80000 €. L'error és menor a 1,5 Km a la primera hora de funcionament.

- Intermediate grade: en fan us els helicòpters i avions petits. El seu cost va des de 16000 € a 40000€.

- Tactical grade: aquests sistemes només són capaços de proporcionar estimacions vàlides per la navegació (operant sense ajudes externes) durant uns quants minuts. Això fa que sigui interessant combinar-los amb un altre sistema de navegació com pot ser un GPS. El seu preu oscil·la des dels 1600€ als 25000€.

- Consumer grade: els que tenen un preu i precisió més baix. Els sensors inercials d'aquesta categoria difícilment es poden emprar en la navegació, però són adequats per a la fabricació de AHRS.

Actualment, les consumer grade es fabriquen amb tecnologia MEMS (sistemes microelectromecànics) (David H. Titterton and John L. Weston, 2004). Els sensors fabricats amb aquesta tecnologia han esdevingut molt populars gràcies al seu ús en tot tipus de dispositius quotidians (mòbils o consoles de videojocs) i professionals (captura de moviment, estabilització de plataformes, patins segway, etc). Això ha propiciat el desenvolupament de sensors cada cop més precisos. Per aquest motiu, avui en dia és possible trobar tactical grade fets amb tecnologia MEMS.

Els sistemes de navegació inercial tenen diversos avantatges. El primer és que poden treballar a una freqüència elevada (50 - 100 Hz). Això vol dir que poden donar moltes més mesures de la posició i velocitat per segon que altres sistemes de navegació com poden ser els GPS (10 Hz). Aquest aspecte és útil en vehicles que es mouen a gran velocitat o que requereixen un control ràpid i precís, per exemple míssils, avions... En segon lloc, els INS a part de mesurar la posició i la velocitat també mesuren l'orientació del vehicle en que es troben. Això és essencial en vaixells, submarins, aeronaus, naus especials, míssils... Finalment, i com ja s'ha dit, els sistemes de navegació inercial, un cop inicialitzats, no necessiten cap ajuda externa per funcionar. No han de rebre cap tipus de senyal de l'exterior. Per tant, problemes que tindria un GPS que perd la senyal en un túnel o en una

zona amb edificis força alts els INS no els tindrien. Això també els converteix en eines molt útils per treballar a sota l'aigua.

Els INS són sistemes de navegació molt útils per ser integrats a altres sistemes de navegació. El motiu és que els problemes que presenten els sistemes de navegació inercial són complementaris a les mancances d'altres sistemes de navegació. Si fem un exemple amb un GPS. Un INS té deriva que augmenta amb el pas del temps, però un GPS presenta un error absolut; un INS té un funcionament continu, un GPS pot perdre la senyal; un INS treballa a freqüència alta, un GPS a freqüència baixa. Un altre motiu per integrar un sistema de navegació inercial amb altres sensors és que, igual que passava amb l'AHRS, una INS necessita una inicialització absoluta, és a dir, conèixer l'orientació i posició inicials (compàs, girocompàs, sensor de pressió, GPS...).

A continuació, mostrarem la figura 2.3 on es pot apreciar millor les diferències entre una IMU, una AHRS i un INS.

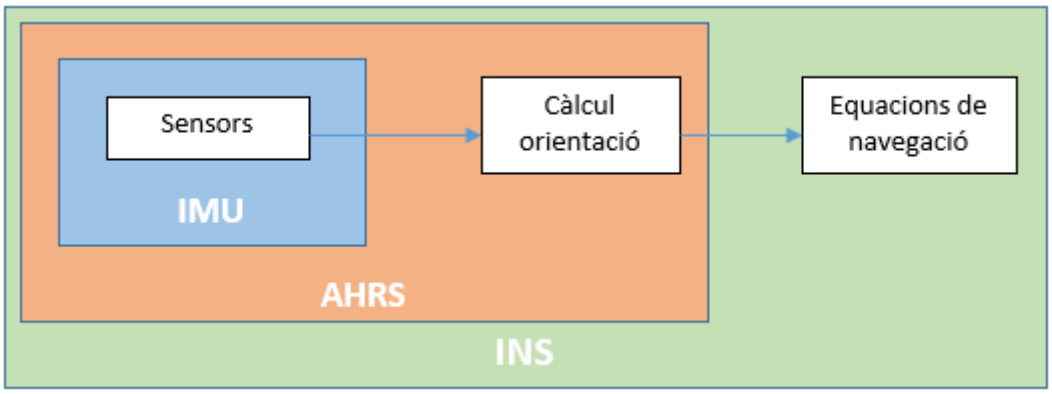


Figura 2.3: IMU, AHRS i INS

Finalment, recordem que els INS estant formats per dos parts: una IMU i una unitat amb les equacions de navegació. A continuació farem una breu descripció d'aquestes equacions. Més endavant, a la secció 3.6, es desenvoluparan en detall.

2.4. Equacions de navegació

Les equacions de navegació són les encarregades de, gràcies a les dades que ens proporciona la IMU, determinar la velocitat, posició i orientació del vehicle. El seu principi de funcionament és molt senzill. Si tenim les acceleracions que ens donen els acceleròmetres, les integrem i obtindrem la velocitat, que si alhora l'integrem aconseguirem la posició en què es troba el vehicle. Amb els giroscopis és semblant, si tenim la velocitat angular i l'integrem obtindrem la orientació en que es troba el cos.

Podem dividir les equacions de navegació en dos apartats. Unes són les encarregades de trobar l'orientació del vehicle i les altres les encarregades de trobar la velocitat i posició. Tal, com veurem a continuació, estan molt relacionades les unes amb les altres.

2.4.1. Orientació

Quan parlem d'orientació ens referim a la rotació que té la nostra IMU i, conseqüentment la del vehicle que l'utilitza, respecte un sistema de coordenades fixe. Alhora de calcular la posició d'un objecte mitjançant un INS definim aquesta posició respecte un sistema de coordenades determinat. De la mateixa manera, podem definir l'orientació del INS respecte aquest mateix sistema de coordenades com un conjunt de rotacions.

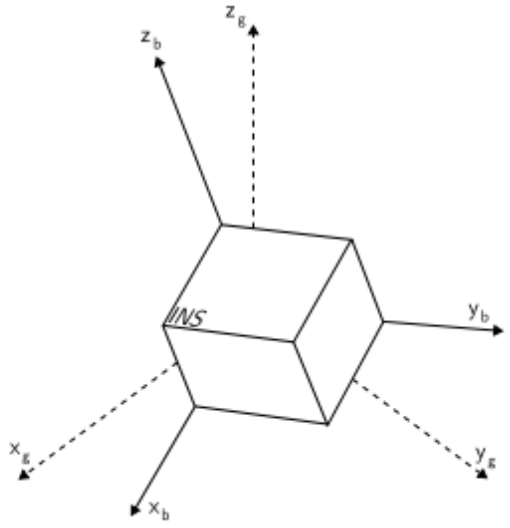


Figura 2.4: Orientació del INS respecte el sistema de coordenades g (Oliver J. Woodman, 2007)

A la figura 2.4 veiem com el sistema de coordenades de referència (G) no coincideix amb el sistema del INS (B) sinó que hi ha una certa rotació. Aquesta rotació/orientació que té l'INS s'ha de tenir en compte alhora fer els càlculs de velocitat i posició. Els acceleròmetres estant col·locats ortogonalment seguint els eixos de coordenades del INS (B), per tant, quan ens donen l'acceleració ens la donen respecte B i per trobar la posició i velocitat la necessitem respecte els eixos de coordenades G ja que són els quals representarem la posició i la velocitat. S'ha de fer una transformació, un canvi de sistema de referència, de les dades dels acceleròmetres per passar-les de B a G mitjançant una matriu de rotació (veure figura 2.5).

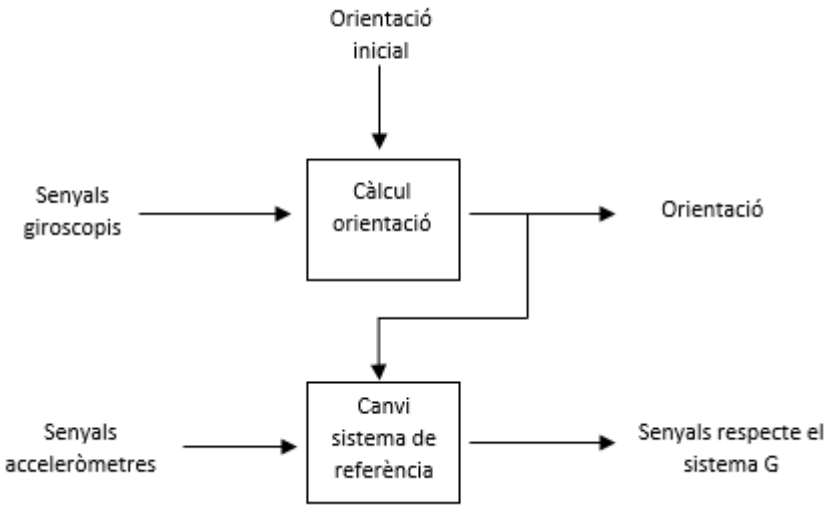


Figura 2.5: Càlcul orientació i transformació a les dades dels acceleròmetres

En el càlcul de l'orientació hem d'integrar la velocitat angular que ens dona els giroscopis respecte el temps per tal d'actualitzar-ne el seu valors. Com en el cas dels acceleròmetres, considerem que els valors de velocitat angular que ens proporcionen els giroscopis són constants dins l'interval de temps. Més endavant entrarem en més detall sobre el càlcul de la orientació (capítol 3.3).

2.4.2. Posició i velocitat

Primer de tot, recordem que els acceleròmetres mesuren la força específica per tant, haurem d'introduir el terme corresponenet a la gravetat per obtenir les acceleracions reals que pateix la IMU. Per fer-ho s'aplica la següent fórmula:

$$f = a - g \qquad \textit{(Eq. 2.1)}$$

On:

f: és la força específica, el que mesuren els acceleròmetres.

a: és les acceleracions reals que pateix la IMU, les que necessitem.

g: l'acceleració de la força de la gravetat.

Un cop hem aplicat la correcció de la gravetat ja ens podem disposar a integrar la acceleració per obtenir la velocitat i després fer una altre integral per trobar la posició. Per fer-ho necessitarem unes condicions inicials (velocitat i posició inicials).

$$v(t) = v(0) + \int_{t_0}^{t} a(t)\, dt \qquad \textit{(Eq. 2.2)}$$

$$r(t) = r(0) + \int_{t_0}^{t} v(t)\, dt \qquad \textit{(Eq. 2.3)}$$

On:

r: és la posició.

v: la velocitat.

a: l'acceleració.

Gràcies a la alta freqüència de treball de la IMU podem obtenir moltes dades dels acceleròmetres per segon. Això fa que podem considerar que l'acceleració dins l'interval de t_0- t és constant, per tant, les equacions per calcular la velocitat i la posició (2.2 i 2.3) quedarien de la següent forma.

$$v = v_0 + a \cdot \Delta t \qquad \textit{(Eq. 2.4)}$$

$$r = r_0 + v \cdot \Delta t \qquad \textit{(Eq. 2.5)}$$

Un cop hem fet el primer càlcul a la següent iteració v i r passaran a ser v_0 i r_0.

A continuació mostrem un diagrama de blocs (figura 2.6) on es veu tot el procés que s'ha de seguir per trobar la orientació, posició i velocitat mitjançant les dades de la IMU.

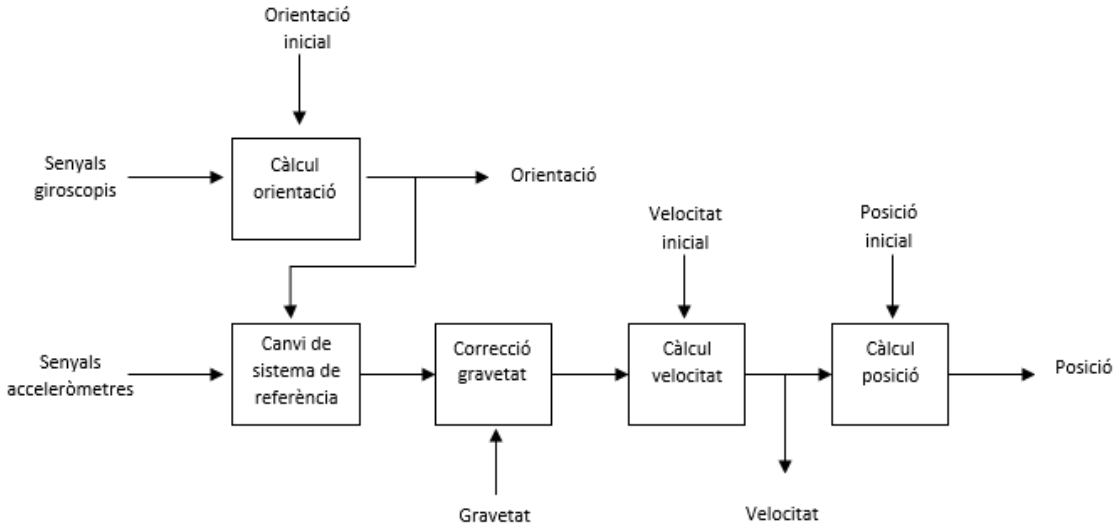


Figura 2.6: Procés equacions de navegació

Més endavant en aquest treball (capítol 3.6) es mostraran amb detall les equacions de navegació necessàries per poder estimar la posició, velocitat i orientació amb un sistema de navegació inercial.

2.5. Descripció sensors utilitzats

Tot seguit mostrarem els sensors inercials utilitzats per realitzar aquest treball.

Primer de tot, hem utilitzat una AHRS anomenada MTi de la marca XSENS (figura 2.7). Es tracte d'un sensor de la categoria de consumer grade de 9DoF (acceleròmetres, giroscopis i magnetòmetres). És fàcil d'usar i interpretar ja que ve amb un software de suport (MT

Manager) que facilita molt la feina. Com a AHRS el sensor MTi proporciona mesures d'orientació, però a banda d'això, també permet obtenir les dades sense tractar dels giroscpois, acceleròmetres i magnetòmetres que són les que necessitem pel nostre estudi. Les dades d'orientació que proporcionava la MTi s'han usat com a referència. S'han comparat amb les orientacions que obteníem dels nostres càlculs per comprovar si eren correctes.

Degut a la poca presició dels seus sensors la solució inercial era de molt mala qualitat. Tot i que es va desenvolupar el treball igualment, quan es va tenir disponible un sensor millor, es va fer el canvi.

Figura 2.7: MTi

Més endavant, un cop avançat el treball hem pogut comptar amb un altre sensor inercial. En aquest cas tracte d'una IMU. És el model ADIS16488 de l'empresa Analog Devices. Es tracta d'un sensor inercial de la categoria tactical grade de 10DoF (acceleròmetres, giroscopis, magnetòmetres i baròmetre). Aquest sensor és de més bona qualitat que la MTi i té més precisió. Això ens ha permès obtenir més bons resultats.

Figura 2.8: IMU ADIS16488

Ambdós sensors inercials estan fabricats amb tecnologia MEMS. Les seves especificacions tècniques es podran trobar a l'annex D on es troben els catàlegs amb les especificacions tècniques dels sensors inercials.

3. NAVEGACIÓ INERCIAL

En aquest capítol s'entrarà amb detall sobre tots els passos que es segueixen amb les equacions de navegació per trobar la posició, velocitat i orientació d'un objecte mitjançant les dades que ens proporciona la unitat de mesura inercial (IMU). També s'explicaran tots els conceptes genèrics necessaris per comprendre millor les tècniques de navegació inercial.

És important esmentar que per realitzar aquest treball es segueix aquest llibre (Paul D. Groves, 2013). La gran majoria d'equacions utilitzades en aquest treball s'extreuen d'aquest llibre. Per tant, l'objectiu és implementar aquestes equacions en Matlab i enriquir els coneixements en el camp de la navegació inercial. Aquest treball pot ser considerat com un pas preliminar a una futura integració real d'aquest tipus de sistemes de navegació.

3.1. Navegació per estima

El "dead reckoning" o navegació per estima és la tècnica que es fa servir en un sistema de navegació inercial per trobar la posició. Es basa en el següent: coneguda la posició anterior i la velocitat i direcció en que ens hem desplaçat durant un interval de temps, podem determinar la nova posició en que ens trobem. Amb la següent imatge il·lustrarem millor el concepte.

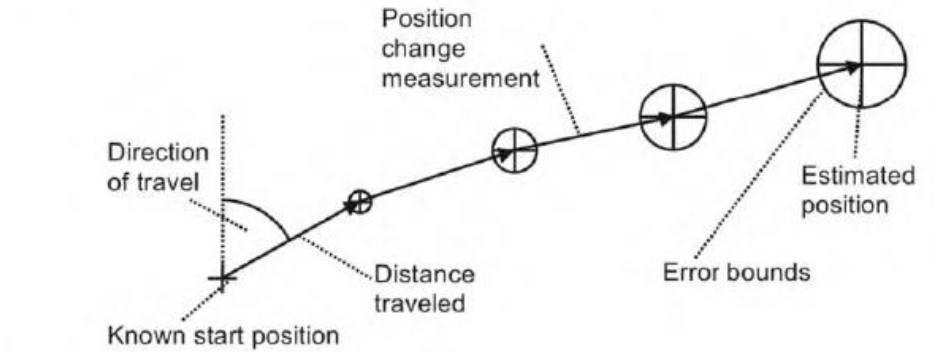


Figura 3.1: Navegació per estima (Paul D. Groves, 2008)

Un dels problemes de la navegació per estima és que l'error que afecta a les mesures amb les quals determinem els desplaçaments es va acumulant i cada vegada es va fent més gran. D'aquesta manera, a mesura que avanci el temps l'estimació de la posició serà més incerta. Els errors solen venir dels instruments de mesura, en el nostre cas de la IMU. Una unitat de mesura inercial més bona permetrà que els errors siguin més petits i que, per tant, obtinguem una millor estimació de la posició.

El gran avantatge de la navegació per estima és que, tret del moment d'inicialitzar el procés (velocitat i posició inicials), no es necessita cap ajuda externa al INS per seguir calculant la posició.

3.2. Sistemes de referència

Quan volem determinar la posició o la velocitat d'un objecte necessitem un sistema de coordenades o de referència on representar-ho. Per tant, alhora de treballar amb un INS hem d'especificar un sistema de referència en el qual podrem mostrar quina posició, velocitat i orientació té un vehicle.

En el cas d'aquest estudi s'utilitzen quatre sistemes de coordenades diferents: el Body, ECI, ECEF i NED (Paul D. Groves, 2013). A continuació es fa una descripció detallada de cada un d'ells.

3.2.1. Sistema de referència Body

El sistema de referència Body són els propis eixos de coordenades del vehicle o la IMU, té el seu origen a la pròpia unitat de mesura inercial. Aquests eixos es mouen segons es mogui o roti la IMU com podem observar a la figura 1.

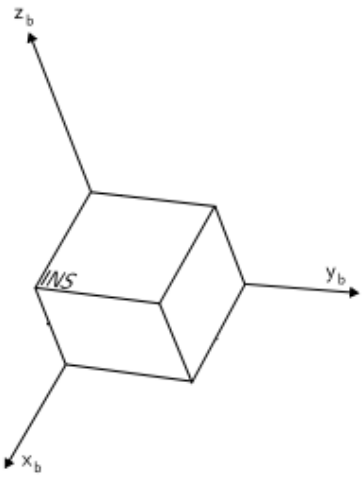


Figura 3.2: Sistema de coordenades Body (Oliver J. Woodman, 2007)

Generalment, les direccions de cadascun dels tres acceleròmetres i giroscopis que porta la unitat de mesura inercial coincideix amb un dels eixos del Body. Així podem prendre mesures en les tres dimensions de l'espai. Per tant, totes les dades que rebem de la IMU (acceleracions i velocitats angulars) estant representades en el Body.

Aquest sistema de coordenades sempre es mou amb el vehicle, això fa que sigui necessària una segona referència on poder representar la posició i velocitats relativa a la que es troba o mou el vehicle. Per aquest motiu, es fa necessària la utilització d'altres sistemes de referència per representar-hi la velocitat i la posició. Tot seguit, es mostraran els tres sistemes de coordenades més utilitzats per fer aquesta funció, els quals s'han implementat tots tres en aquest treball.

3.2.2. Sistema de referència ECI

Les seves sigles en anglès signifiquen Earth-Centered Inertial. ECI és un sistema de coordenades que té el seu origen en el centre de la Terra. Com el seu nom indica es tracta d'un sistema de referència inercial, això vol dir que no accelera ni rota respecte la resta de l'univers. Per tant, el sistema no rota amb la Terra. Els seus eixos x i y estant col·locats en el pla que forma l'equador. L'eix z és ortogonal a x i y i va en direcció al pol Nord. La Terra rota al respecte l'eix z.

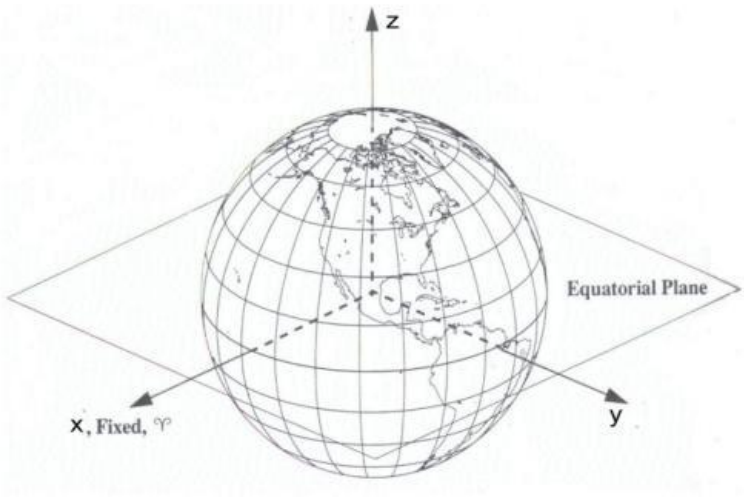


Figura 3.3: Representació del sistema de coordenades ECI

Aquest sistema de coordenades és el que té les equacions més senzilles d'implementar. El problema que té és que el fet de que la Terra roti al seu voltant fa que els diferents punts de la Terra no tinguin sempre les mateixes coordenades sinó que van variant depenen de quin moment del dia ens trobem. Per exemple, amb ECI un punt determinat de Girona no tindrà ara les mateixes coordenades que d'aquí una hora.

3.2.3. Sistema de referència ECEF

El seu significat és Earth Centered Earth Fixed. El sistema de coordenades ECEF també té el seu origen al centre de la Terra. Els seus eixos x i y es troben al pla de l'equador. L'eix x apunta al meridià de Greenwich i z és ortogonal a x i y i apunta cap el pol Nord. La diferència entre ECI i ECEF és que ECEF si que rota amb la Terra, està fixat a ella. O sigui, l'eix x de ECEF sempre apuntarà al meridià de Greenwich perquè a mesura que la Terra vagi rotant el sistema de coordenades també girarà amb ella.

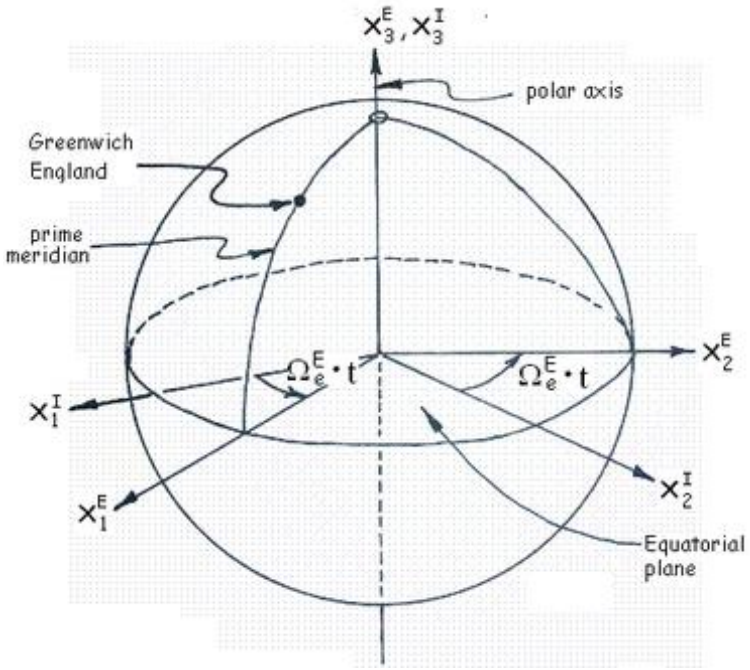


Figura 3.4: Comparació entre els dos sistemes de coordenades

A la figura 3.4 es pot apreciar la diferència entre ECI i ECEF ja que a mesura que avança el temps ECEF rota seguint el gir de la Terra i ECI no. Això fa que la diferència entre els dos vagi augmentant fins a que passa un dia sencer moment on tornen a coincidir.

Aquest sistema de coordenades és el que s'utilitzarà en la majoria de càlculs d'aquest treball. El motiu és que els seus càlculs són prou senzills d'implementar i també perquè aquest sistema ja dóna a l'usuari la seva posició relativa a la Terra.

3.2.4. Sistema de referència NED

Aquest és l'últim sistema de referència que veurem i que utilitzarem en aquest treball. Les seves sigles volen dir North, East i Down i es refereixen a la direcció en que apunta cadascun dels eixos de coordenades del sistema. El seu origen és el propi objecte que porta l'INS. I com s'ha dit, els seus eixos apunten un en direcció al Nord (x), un en direcció a l'est (y) i un perpendicular al pla tangent a la superfície de la Terra en direcció al seu interior (z). En la següent imatge s'apreciarà molt millor com és aquest sistema de referència.

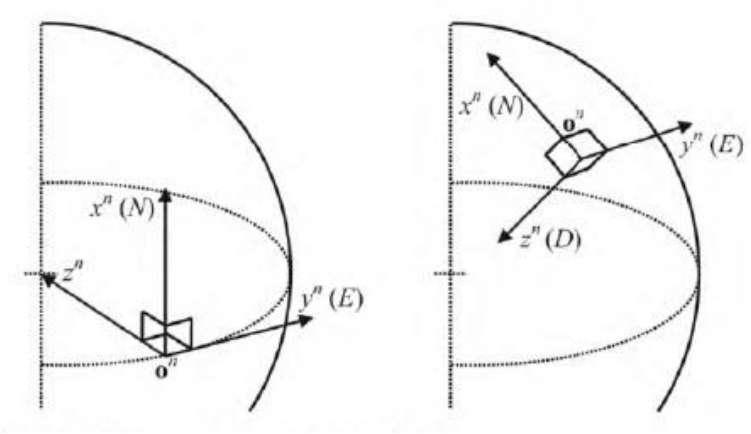


Figura 3.5: NED

El NED té l'origen sobre el vehicle i, per tant, es mou amb ell. És com un Body que no rota amb el vehicle sinó que es manté horitzontal i apuntant al nord. A la secció 3.6.4 s'explica amb detall com s'utilitza el NED per representar la posició, velocitat i orientació.

En aquest treball es mostraran tres implementacions diferents de les equacions de navegació utilitzant com a referència els sistemes ECI, ECEF i NED. Tot i haver-los implementat tots tres, alhora de fer experiments s'usarà l'implementació amb el sistema de referència ECEF ja que és el que va més bé per la seva senzillesa a l'hora d'interpretar resultats i perquè permet una implementació del filtre de Kalman relativament simple.

Igualment, en aquest treball s'ha decidit descriure el tres sistemes i fer-ne la seva implementació.

3.2.5. Canvis de sistemes de referència

En diverses ocasions en el treball ens és necessari transformar dades que estant representades en un sistema de referència per representar-les en un altre sistema. O sigui, aplicar un canvi de sistema de referència o de coordenades. Aquesta transformació és necessària en l'etapa de inicialització (capítol 3.5) i també ens pot ser molt útil quan volem comparar resultats obtinguts amb els diferents sistemes. Per exemple volem comparar les

estimacions de la posició que fa el INS treballant amb ECI i amb ECEF. Per poder comparar les dues estimacions les hem de transformar al mateix sistema de coordenades.

Les matrius necessàries per realitzar els canvis entre els diferents sistemes de coordenades es troben detallades a l'annex B, secció B.2 . La seva implementació en Matlab es troba a l'annex C, secció C.3.

3.3. Orientació

Quan treballem amb un sistema de navegació inercial i volem localitzar un vehicle/objecte (representat pel sistema de referència Body) respecte a un altre sistema que hàgim escollit com a referència base de la nostre navegació, per exemple ECI o ECEF, cal conèixer la posició relativa entre ells, però també la seva orientació relativa.

La posició es pot representar de manera simple utilitzant coordenades cartesianes, tot i que a vegades pot ser convenient utilitzar coordenades esfèriques (latitud i longitud) per representar la posició d'un vehicle que es desplaça sobre la superfície terrestre. Representar la orientació relativa entre dos sistemes de coordenades és una tasca més complexa.

Una de les millors maneres de representar l'orientació és dir els angles que s'han de rotar cadascun dels tres eixos d'un sistema de referència per alinear-lo amb l'altre. Il·lustrem millor aquesta idea mostrant una imatge (figura 3.6) que ho explica amb dos dimensions.

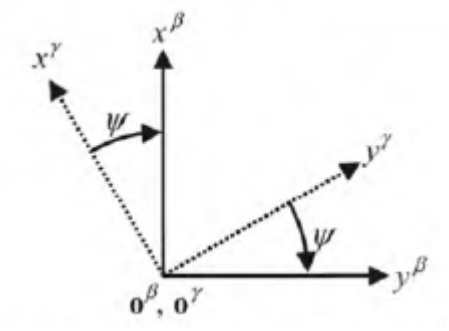


Figura 3.6: Rotació sistemes de referència (Paul D. Groves, 2008)

Veiem com el sistema de referència γestà rotat un angle Ψ respecte el sistema β. El mateix concepte es trasllada a les tres dimensions.

La orientació d'un vehicle o objecte determinat es pot representar de diverses maneres. Les tres més importants són:

- Angles d'Euler
- Matriu de rotació
- Quaternions

En el nostre cas, per representar la orientació s'ha fet seguint els angles d'Euler. Però, alhora de transformar dades representades en un sistema de referència determinat a un altre (per exemple de Body a NED) s'han utilitzat les matrius de rotació. El motiu és que són un sistema conceptualment senzill, amb una interpretació prou intuïtiva i que es pot implementar de manera simple.

De totes les orientacions relatives entre sistemes que podem estudiar, la que hi ha entre el Body i el NED és particularment interessant perquè ens permet definir la inclinació del sensor/vehicle respecte el pla horitzontal (definit per la direcció del vector gravetat o de la direcció Down del NED) i el rumb (orientació del vehicle respecte el nord).

3.3.1. Angles d'Euler

Amb els angles d'Euler l'orientació d'un objecte és desglossada en tres rotacions angulars successives. Aquestes tres rotacions són cadascuna respecte un dels tres eixos (x, y o z) d'un sistema de referència i es fan sobre els sistema de referència obtingut en la rotació prèvia. Amb la següent imatge (figura 3.7) veiem el concepte.

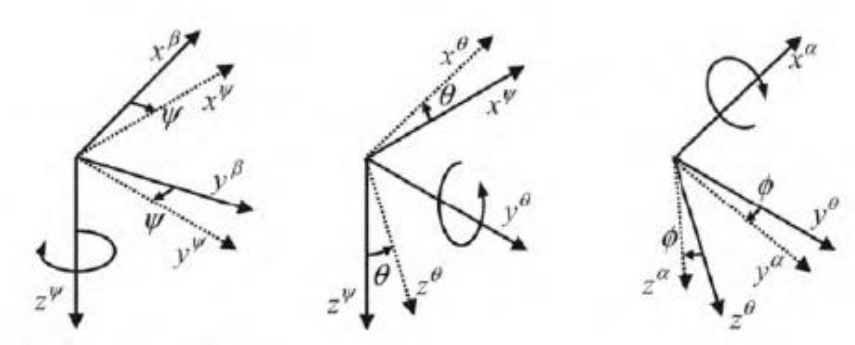


Figura 3.7: rotacions successives

Primer s'ha aplicat una rotació d'angle Ψ al voltant de l'eix z, s'anomena *Yaw.* En segon lloc, una rotació d'angle θ sobre l'eix y, rep el nom de *Pitch*. I finalment, la que s'anomena rotació de *Roll*, és d'angle ϕ al voltant de l'eix x. Aquests són els anomenats angles d'Euler:

Roll (ϕ) Pitch (θ) Yaw (Ψ)

En navegació per convenció es fa primer la rotació en Yaw, després en Pitch i finalment en Roll. Hi ha altres combinacions vàlides que s'utilitzen en altres àmbits.

Per tant, quan en navegació es parla del Roll, Pitch i Yaw d'un objecte o vehicle es refereix a la rotació que hi ha a cadascun dels eixos del sistema de coordenades fixe per arribar fins aquella orientació. En la següent il·lustració (figura 3.8) mostrem el Roll, Pitch i Yaw d'un avió:

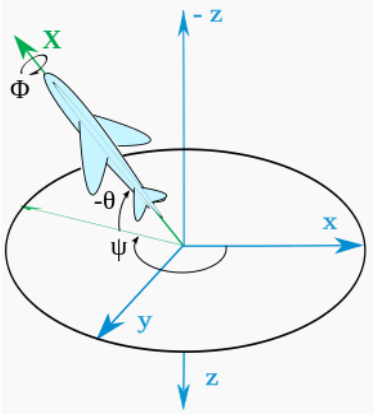


Figura 3.8: Roll, Pitch i Yaw d'un avió

En blau el sistema de referència fixe, en verd el de l'avió (Body).

Com s'ha dit abans, en aquest treball quan volem representar l'orientació que té la nostre IMU en un moment determinat s'utilitzaran els angles d'Euler.

3.3.2. Matriu de rotació

Les matrius de rotació o transformació són matrius de 3x3. La seva utilitat és la següent. Quan tenim unes dades representades en un sistema de referència determinat i les volem representar en un altre el que hem de fer és multiplicar-les per la matriu de rotació que va d'un sistema a l'altre. En aquest treball les matrius de rotació s'anomenaran amb la lletra C i la seva notació serà la següent:

$$X^{\beta} = C_{\alpha}^{\beta} \cdot X^{\alpha} \qquad \textit{(Eq. 3.1)}$$

Sent X^{α} una dada qualsevol representada respecte el sistema de referència α. Es multiplica per la matriu de rotació C_{α}^{β} que transforma les dades representades a α amb dades representades sobre β. Finalment, obtenim X^{β}.

Aquesta és la principal utilitat que tindran les matrius de rotació en el nostre treball. S'utilitzaran per passar dades (velocitat, posició, velocitat angular, força específica, acceleracions...) representades en un sistema de coordenades a un altre. Aquest procés té una importància cabdal ja que s'utilitzarà molt.

Com hem comentat abans, els angles d'Euler ens permeten representar orientacions relatives entre sistemes de referència com una seqüència de tres rotacions (primer Yaw, després Pitch i finalment Roll). Aquesta transformació es pot representar mitjançant matrius de rotació com:

$$C_{\beta}^{n} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi_{\beta n} & \sin\varphi_{\beta n} \\ 0 & -\sin\varphi_{\beta n} & \cos\varphi_{\beta n} \end{pmatrix} \cdot \begin{pmatrix} \cos\theta_{\beta n} & 0 & -\sin\theta_{\beta n} \\ 0 & 1 & 0 \\ \sin\theta_{\beta n} & 0 & \cos\theta_{\beta n} \end{pmatrix} \cdot \begin{pmatrix} \cos\Psi_{\beta n} & \sin\Psi_{\beta n} & 0 \\ -\sin\Psi_{\beta n} & \cos\Psi_{\beta n} & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \textit{(Eq. 3.2)}$$

Les tres matrius que es mostren són les matrius de rotació al voltant de l'eix x, y i z respectivament i les rotacions que s'hi apliquen són les de Roll, Pitch i Yaw. Aquesta matriu ens serà molt útil per passar les dades que provenen directament de la IMU (representades en el Body) a un dels sistemes de referència en que treballarem, com és el NED.

3.4. Models càlcul gravetat

Com hem vist anteriorment (secció 2.4.2), en els càlculs de les equacions de navegació és necessari aplicar-hi una correcció de la gravetat per així poder obtenir el valor de l'acceleració que experimenta el cos. Per tant, per treballar amb un INS necessitem tenir el coneixement de quina força de la gravetat estem experimentant. Com que la gravetat no és igual en tots els punts de la Terra, necessitem un model que ens digui quin valor té segons el punt de la Terra on ens trobem.

Quan treballem en ECI es calcula l'acceleració de la gravetat segons al punt de la Terra on ens trobem. Quan treballem en ECEF o NED a aquesta acceleració de la gravetat se li ha de sumar una component de acceleració centrípeta deguda a que el sistema de referència rota solidari amb la Terra. Les equacions pel càlcul de la gravetat les hem extret de (Paul D. Groves, 2013).

La implementació en Matlab dels diferents models pel càlculs de la gravetat es troben a l'annex C, secció C.4. Les equacions detallades es troben a l'annex B, secció B.5.

3.5. Inicialització

Un cop repassats els conceptes generals que hem vist en els capítols anteriors anem a entrar de ple en el que és el cos del treball, la implementació de les equacions de navegació en un entorn Matlab.

La primera fase que fa un sistema de navegació inercial és la de inicialització (Paul D. Groves, 2013). És on s'obtenen la posició, la velocitat i l'orientació inicials. Totes tres són dades necessàries per poder començar el càlculs. En la següent imatge (figura 3.9) sobre el procés que segueix un INS es podrà apreciar millor a què correspon exactament l'etapa d'inicialització.

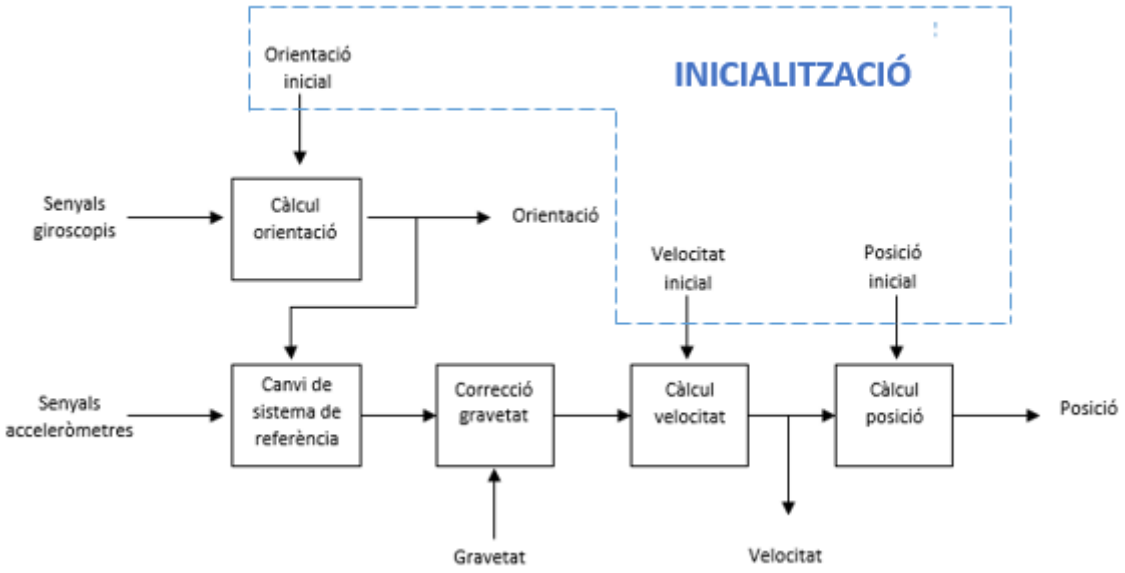


Figura 3.9: Etapa d'inicialització

A continuació, veurem com obtenim els valors inicials de posició, velocitat i orientació en aquest treball.

3.5.1. Posició inicial

Amb un sistema de navegació inercial no podem estimar la posició inicial. L'hem de trobar amb un mitjà extern, el més utilitzat sol ser un GPS. La inicialització és l'únic moment en que el INS necessita una ajuda externa per estimar la posició, després la pot estimar de forma autònoma. Tot i que a la pràctica, a no ser que treballem amb sistemes de tipus marine grade (secció 2.3), el més recomanable és utilitzar també ajudes externes per limitar el creixement de la deriva en la posició, velocitat i orientació. La integració d'aquestes ajudes

externes, ja sigui GPS o algun altre tipus de sensor, es pot fer amb sistemes com el que es descriu a la secció 4.2.

En el nostre cas, obtenim els valors de posició d'un GPS i les seves coordenades de longitud i latitud s'introdueixen en el programa Matlab. A partir d'aquí, si volem fer els càlculs respecte els sistema ECEF havíem de passar les dades de esfèriques (com les dona el GPS) a cartesianes. I si volem fer els càlculs respecte ECI, un cop hem passat les dades a cartesianes hem d'aplicar un canvi de sistema de referència.

3.5.2. Velocitat inicial

En el cas de la velocitat inicial ens trobem amb el mateix problema que amb la posició, no la podem calcular amb el INS. Per tant, ha de venir d'un mitjà extern. Podem trobar la velocitat inicial amb un GPS o amb un radar/sonar Doppler per exemple. De la mateixa manera que la posició, un cop es té la velocitat inicial, el sistema de navegació inercial pot estimar les següents velocitats de forma autònoma i sense cap ajuda externa. Tot i que, evidentment, el rendiment del sistema millorarà si podem fusionar periòdicament dades d'un sensor extern.

En el treball el que es fa és començar els experiments amb la unitat de mesura inercial quieta, d'aquesta manera sabem que la velocitat inicial és igual a zero.

3.5.3. Orientació inicial

A diferència de la velocitat i la posició, per determinar l'orientació inicial no es necessita cap ajuda externa, es pot fer amb les mateixes dades que ens proporciona la IMU, sempre i quan sigui una unitat de mesura inercial de 9DoF, es a dir, equipada amb magnetòmetres. Per fer-ho utilitzarem dos tècniques determinades: el leveling i el magnetic heading (Paul D. Groves, 2013). Amb el leveling podem trobar el Roll i el Pitch inicials. Els càlculs del magnetic heading són els encarregats de trobar el Yaw. D'aquesta manera podem aconseguir el Roll, Pitch i Yaw del vehicle.

La tècnica del leveling utilitza les dades dels acceleròmetres, mentre que el magnètic heading fa servir la dels magnetòmetres. El leveling s'ha de fer primer ja que el magnètic heading es calcula sobre el pla horitzontal i per tant, cal compensar el pitch i roll existents amb el resultat del leveling.

Hi ha una altre forma per trobar el Yaw amb la qual no es necessiten magnetòmetres, sinó que s'utilitzen els giroscopis. Es tracte del girocompassing, que calcula el Yaw mitjançant el vector gravetat i el vector de rotació de la Terra. El seu problema és que perquè funcioni correctament necessita que els giroscopis siguin molt bons, com poden ser els giroscopis de fibra òptica, sinó no pot trobar el vector de rotació de la Terra. Els giroscopis dels quals disposem no són de prou qualitat i, per tant, fem servir el magnètic heading.

Tot seguit, s'expliquen més detalladament les dos tècniques utilitzades i es mostren algunes proves que s'han fet per verificar el seu correcte funcionament.

3.5.3.1. Leveling

El principi de funcionament del leveling és que quan la IMU està estacionaria o movent-se a velocitat constant l'única força específica que mesuren els acceleròmetres és la força de la gravetat. La gravetat sempre apunta en la direcció "Down" del sistema de coordenades NED explicat anteriorment. Per tant, amb la tècnica leveling trobarem el roll i pitch de la IMU respecte el NED. A la següent imatge (figura 3.10) es mostra millor el concepte de leveling.

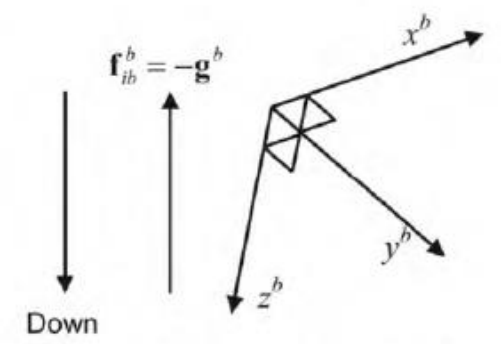


Figura 3.10: principi Leveling (Paul D. Groves, 2013)

És important recalcar que per obtenir una correcte mesura del leveling la unitat de mesura inercial ha d'estar o aturada o movent-se a velocitat constant. Sinó és així els acceleròmetres no només percebran la força de la gravetat, també mesuraran altres acceleracions i això provocarà error en el resultat.

A continuació és mostren les equacions utilitzades per fer els càlculs de Roll (ϕ) i Pitch (θ).

$$\varphi_{nb} = \arctan2(-f^b_{ib,y}, f^b_{ib,z}) \qquad \textit{(Eq. 3.3)}$$

$$\theta_{nb} = \arctan\left(\frac{f^b_{ib,x}}{\sqrt{{f^b_{ib,y}}^2 + {f^b_{ib,z}}^2}}\right) \qquad \textit{(Eq. 3.4)}$$

On:

f^b_{ib}: força específica que mesura l'acceleròmetre respecte x, y o z.

φ_{nb}: angle Roll en rad.

θ_{nb}: angle Pitch en rad.

Els subíndex *nb* signifiquen que és l'angle de *b* (Body) respecte *n* (NED).

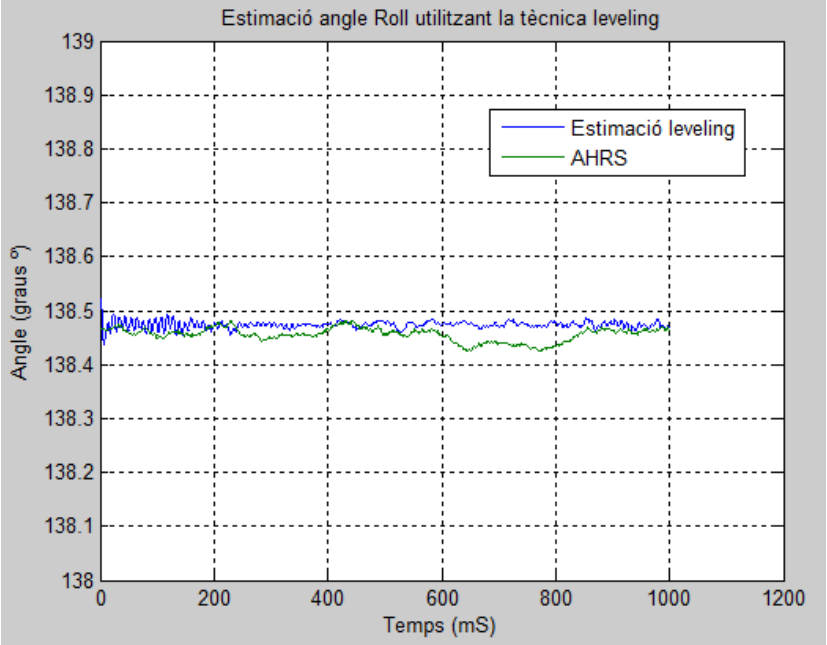
3.5.3.1.1. Proves Leveling

En aquest apartat mostrarem un resum de les diferents proves que s'han fet amb el Matlab i la IMU per determinar que la implementació de les equacions de leveling funcionava correctament i que, per tant, estava a punt per ser utilitzada en un sistema de navegació inercial.

Aquestes proves s'han realitzat amb la MTi. Al ser una AHRS ens dona els angles que ella estima i per tant, els podem utilitzar com a referència per veure si els que estimem nosaltres són correctes.

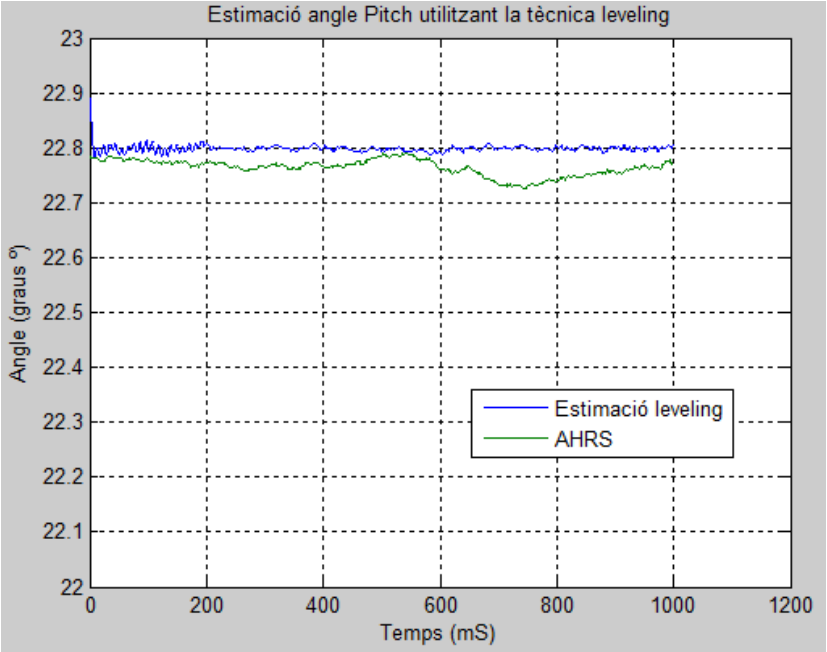
La primera de les proves ha estat totalment estàtica. S'ha deixat la unitat de mesura inercial amb una cert angle de Roll i Pitch. S'han calculat aquests angles mitjançant el leveling i els hem comparat amb els valors que ja ens donava la AHRS.

En total s'han realitzat 7 proves en la mateixa posició, les quals tenen una durada de 10 segons cadascuna. A continuació, mostrarem la mitjana de les 7 proves. Comencem amb l'estimació del Roll (gràfica 3.1).



Gràfica 3.1: Estimació Roll

Com podem veure l'angle calculat mitjançant el leveling (blau) és pràcticament igual a l'angle que ens dona la AHRS (verd). Veiem ara l'estimació de Pitch (gràfica 3.2).



Gràfica 3.2: Estimació Pitch

A continuació, una petita taula (taula 3.1) que recull els errors mitjà i màxim. L'error es calcula fent la diferència entre el valor de leveling hem estimat i el valor que ens dona la AHRS.

	Error mitjà (graus º)	**Error màxim (graus º)**
Roll	0.018	0.0554
Pitch	0.0344	0.1099

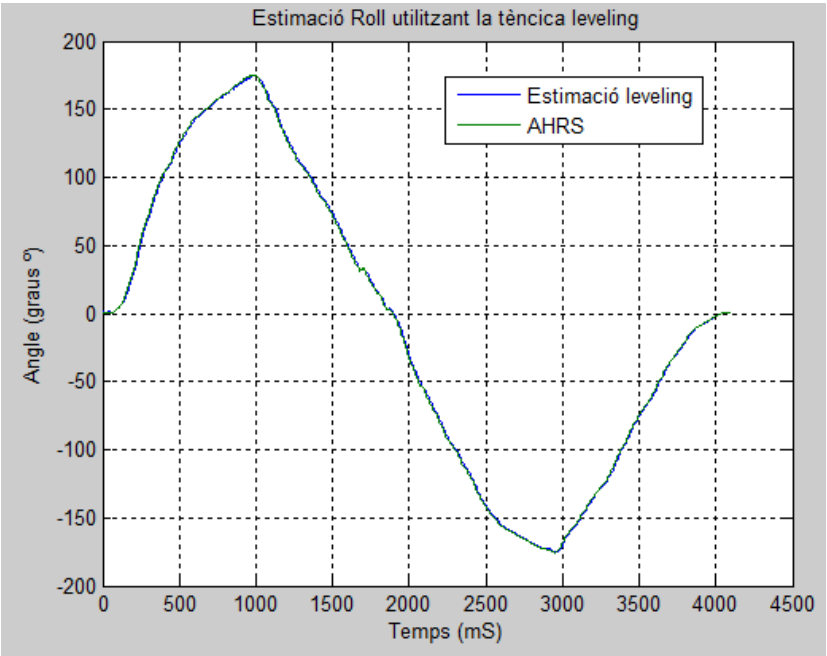
Taula 3.1: Error leveling

Els resultats són molt satisfactoris. Com a màxim trobem l'error d'una dècima de grau.

Si ens fixem en detall en les dues gràfiques (3.1 i 3.2) veurem com abans no han transcorregut unes dos dècimes de segon es pot veure una petita variació aleatòria en els valors de Roll i Pitch que estimem nosaltres (blau). Aquesta petita variació la provoca el que s'anomena soroll. Que no és res més que petites variacions aleatòries en les mostres de dades de la unitat de mesura inercial.

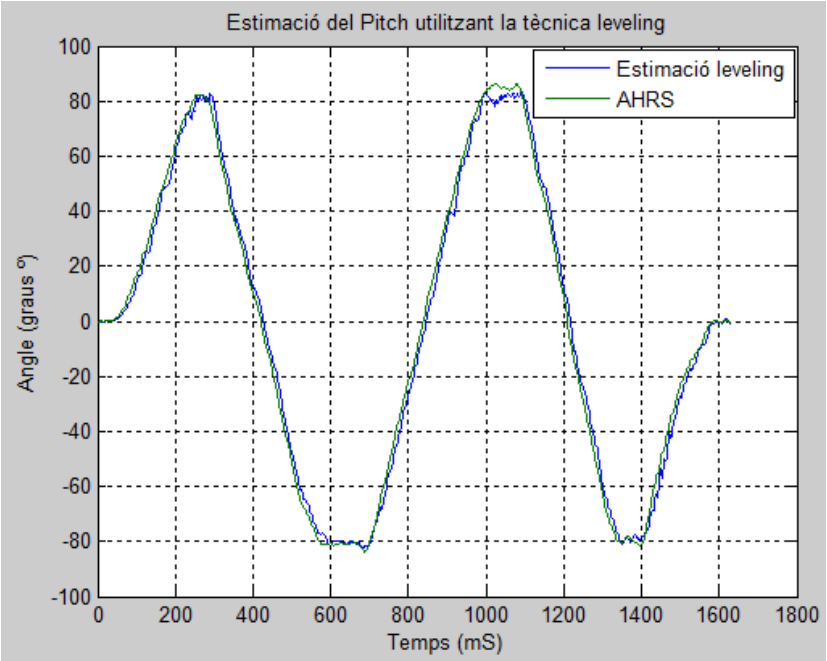
Per corregir el soroll el que fem és el següent. Quan s'apliquen les equacions de leveling no s'agafa el valor de força específica que correspon a aquell instant de temps, sinó que s'agafa el valor d'aquell instant i el d'uns quants anteriors i se'n fa la mitjana. En el nostre cas, em vist que 15 instants de temps era un número adequat. D'aquesta manera, les variacions aleatòries que prenen els valors de la força específica es compensen entre si fent una mitjana. Per aquest motiu en els primers instants de temps el valor de leveling no es gaire estable, perquè la mitjana és de poques dades. A mesura que passa el temps la mitjana ja és de suficients dades i el soroll queda eliminat.

El segon tipus de proves que hem fet ha estat fer rotacions al voltant dels eixos x i y del Body per veure com varien el Roll i el Pitch. D'aquesta manera hem pogut comprovar com s'estimen les variacions d'angle de Roll i Pitch amb les equacions de leveling. Comencem amb un variació de l'angle Roll (gràfica 3.3)



Gràfica 3.3: Estimació Roll en moviment

La variació de l'angle Pitch (gràfica 3.4):



Gràfica 3.4: Estimació de l'angle Pitch en moviment

Finalment, tornem a mostrar en una taula (taula 3.2) els error obtinguts en els càlculs.

	Error mitjà (graus º)	**Error màxim (graus º)**
Roll	1.3168	3.9616
Pitch	3.364	8.8557

Taula 3.2: Error de les proves de leveling en moviment

En aquest prova en moviment veiem com l'error ha augmentat, això és principalment degut a dos motius. El primer és que quan es gira el sensor durant l'experiment s'indueixen acceleracions (no desitjades), però que afecten a la força específica mesurada. Per tant, l'estimació de la direcció del vector gravetat es veu contaminada per aquestes components no esperades. La segona és que per fer l'estimació de l'angle la AHRS executa un algoritme de fusió sensorial que incorpora les dades dels giroscopis amb el leveling dels acceleròmetres. A causa d'això, el temps de reacció i la precisió en front dels girs millora. Tot i així els resultats són acceptables.

Amb aquestes proves hem pogut comprovar com les equacions per aplicar la tècnica de leveling funcionen correctament. També veiem que per aconseguir una bona inicialització dels angles de Roll i Pitch és molt recomanable que la unitat de mesura inercial estigui com més quieta millor. D'aquesta manera obtenim una precisió molt més elevada que no pas si ho féssim en moviment.

3.5.3.2. Magnetic Heading

El magnètic heading és la tècnica utilitzada en aquest treball per trobar l'angle Yaw o de heading en que es troba la unitat de mesura inercial. Recordem que és l'angle que hi ha entre la direcció en que el vehicle apunta i el nord (figura 3.11). D'aquesta manera, trobant aquest angle i juntament amb els calculats amb el leveling completaríem la inicialització de l'orientació.

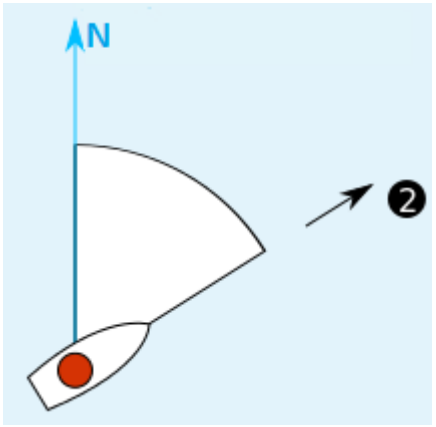


Figura 3.11: Il·lustració d'angle Yaw.

El principi de funcionament del magnètic heading és el següent. Els magnetòmetres utilitzen el camp magnètic de la Terra per trobar on és el nord. El segon pas és, un cop sabem on és el nord, mirar la diferència d'angle que hi ha entre l'eix x del sistema Body i el nord que recordem que és un dels eixos del sistema NED. Amb aquests passos s'obté l'angle Yaw o heading (Ψ).

Per fer-ho s'apliquen les següents fórmules (Eq. 3.5 i Eq. 3.6):

$$\Psi_{mb} = \arctan2\begin{pmatrix} -m^b_{m,y} \cdot \cos\varphi_{nb} + m^b_{m,z} \cdot \sin\varphi_{nb} \\ m^b_{m,x} \cdot \cos\theta_{nb} + m^b_{m,y} \cdot \sin\varphi_{nb} \cdot \sin\theta_{nb} + m^b_{m,y} \cdot \cos\varphi_{nb} \cdot \sin\theta_{nb} \end{pmatrix} \quad \textit{(Eq. 3.5)}$$

On:

$m^b_{m,x}$: és el valor que mesuren els magnetòmetres (Gauss) en cada eix (x, y, z).

θ_{nb} i φ_{nb}: són els angles de Pitch i Roll respectivament calculats amb el leveling.

Ψ_{mb}: angle Yaw respecte el nord magnètic.

Recordem que el Pol Nord magnètic i el Pol Nord geogràfic no es troben en el mateix punt, sinó que hi ha una diferència entre els dos. Amb els magnetòmetres trobem el nord magnètic i nosaltres necessitem trobar el nord geogràfic. Per tant, hem d'aplicar una petita correcció. Aquesta petita correcció s'anomena declinació magnètica (α_{nE}). La declinació magnètica no és res més que la diferència entre en nord magnètic i el nord verdader. La declinació magnètica varia al llarg del temps i segons al punt de la Terra on et trobes. Aquest paràmetre s'ha pogut modelar existeixen cartografies que es poden consultar. Sol aparèixer en qualsevol carta nàutica, però actualment també es pot consultar per internet en planes especialitzades com ara (NOAA). Un cop apliquem la correcció ja tenim l'angle de Yaw (Ψ_{nb}) que buscàvem.

$$\Psi_{nb} = \Psi_{mb} + \alpha_{nE} \quad \textit{(Eq. 3.6)}$$

Aplicant les anteriors equacions ja hem torbat l'angle de Yaw. Tot i així aquest angle pot tenir un cert error. Aquesta desviació pot ser deguda a la influència d'altres camps magnètics, que no siguin el de la Terra, en el nostres magnetòmetres. Aquest camps magnètics poden provenir d'aparells electrònics, cablejat elèctric, etc proper a la unitat de mesura inercial. O també, el simple fet que hi hagi materials ferromagnètics, per exemple metalls, a prop de la IMU pot provocar errors en els magnetòmetres. Per solucionar aquest problema s'ha de fer el que anomenem un calibratge de les dades. En el següent capítol s'explicarà aquest concepte.

3.5.3.2.1. Calibratge de les dades dels magnetòmetres

Primer de tot, anem a veure perquè és necessari realitzar un calibratge de les dades dels magnetòmetres. La finalitat dels magnetòmetres és mesurar el camp magnètic de la Terra per poder trobar el nord. Tot i així, aquestes mesures es poden veure distorsionades per la presència d'altres camps magnètics que no són el de la Terra. Aquestes distorsions provenen principalment de materials i elements que es mouen amb la unitat de mesura inercial. Per exemple, una IMU muntada en un avió patirà els efectes de les pertorbacions creades pels materials ferromagnètics, els components electrònics de l'aparell, etc. L'objectiu del calibratge de les dades és compensar aquestes distorsions per poder aconseguir una bona mesura del nord amb el magnètic heading.

També és necessari conèixer els límits del calibratge de dades. Tornant a l'exemple anterior de l'avió, en aquest cas seria possible corregir les distorsions del camp magnètic mitjançant un calibratge de dades perquè els materials que pertorben el camp magnètic viatjarien amb la IMU i sempre es trobarien en la mateixa posició respecte ella. Però en canvi, no es podria compensar la distorsió del camp magnètic creada per un element extern a l'avió. És important entendre aquest concepte per saber perquè ens pot ser útil el calibratge de les dades del magnetòmetre.

Hi ha dos tipus principals de distorsió: la hard-iron i la soft-iron (Christopher Konvalin, 2009).

Hard-iron. Les distorsions de hard-iron provoquen un camp magnètic constant que es suma al camp magnètic de la Terra. D'aquesta manera els magnetòmetres mesuren el camp magnètic de la Terra més el valor constant de hard-iron. Mentre la posició i l'orientació de la font de distorsió de hard-iron siguin constants respecte la IMU els valors de hard-iron també seran constants. L'imant d'un altaveu, per exemple, produirà una distorsió de hard-iron.

Soft-iron. A diferència del hard-iron on es genera un camp magnètic que es suma al camp magnètic de la Terra, la distorsió de soft-iron és el resultat de l'efecte d'un material en el camp magnètic de la Terra. Materials com el ferro o el níquel poden provocar distorsions de soft-iron si es troben a prop de la IMU. La distorsió de soft-iron depèn de l'orientació del

material respecte la unitat de mesura inercial i el camp magnètic. Aquest tipus de distorsió és més difícil de compensar que la de hard-iron.

A continuació mostrem un model que descriu la relació entre la mesura real i les dades pertorbades.

$$m_m^b = (I + D)^{-1} \cdot (m + b) \qquad \textit{(Eq. 3.7)}$$

On:

m_m^b: són els valors que mesuren els magnetòmetres.

D: la matriu de 3x3 de soft-iron.

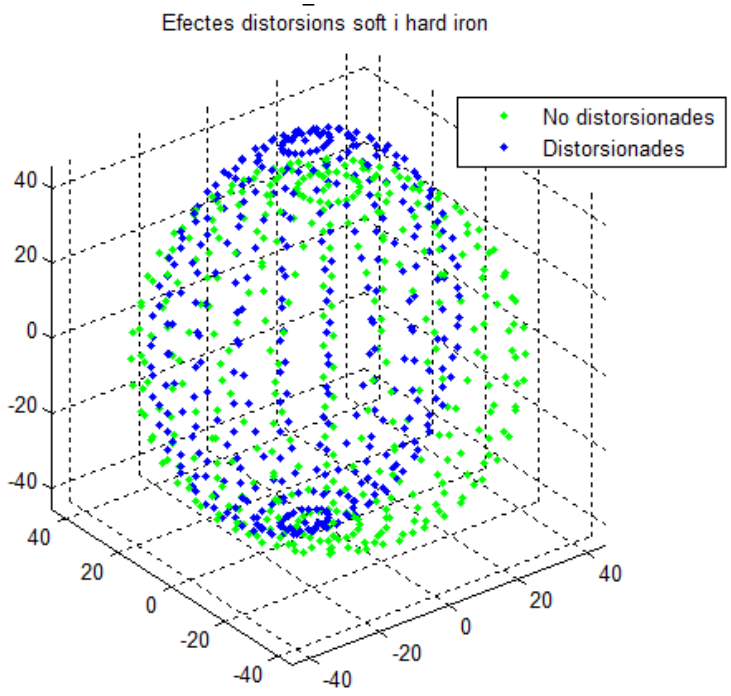
b: el vector de 3x1 de hard-iron.

m: valor real del camp magnètic terrestre no pertorbat.

Si es troba la matriu de soft-iron i el vector de hard-iron es possible desfer l'efecte de les pertorbacions i obtenir unes dades calibrades.

Abans d'explicar com trobem la matriu i vector de soft-iron i hard-iron és necessari descriure els efectes de les pertorbacions. En presencia d'un camp magnètic no pertorbat, tres magnetòmetres col·locats ortogonalment serien capaços de mesurar el vector del camp en les tres dimensions. Si rotéssim el sensor en tots els seus eixos, obtindríem una seqüència de dades on la direcció del vector canviarà respecte el sistema de coordenades body, però no la seva magnitud. Si representéssim el conjunt d'aquestes mesures, obtindríem una forma esfèrica de radi igual a la magnitud del camp i centrada a l'origen (veure dades en verd a la gràfica 3.5). Ara bé, aquesta esfera només es pot obtenir si no hi ha pertorbacions presents. En presència de pertorbacions de hard-iron, un camp magnètic additiu que es mou amb el sensor es reflectirà en les dades com un offset en les mesures de cada eix. Coma conseqüència d'això, observarem que l'esfera desplaça el seu centre fora de l'origen de dades.

D'altre banda, una pertorbació de soft-iron, es a dir la deformació del camp causada per elements ferromagnètics que es mouen amb el sensor provocarà variacions direccionals en la magnitud del camp. Com a resultat d'això, el conjunt de dades abandonaran la forma esfèrica per convertir-se en una mena d'el·lipsoide. Les dades blaves a la gràfica 3.5 mostren els efectes del hard i soft-iron.



Gràfica 5: Efectes distorsions de soft i hard iron

Cal comentar que la gràfica 3.5 és una imatge sintètica, no esta feta amb dades reals. Aquesta gràfica 3.5 representa els efectes de les distorsions de hard i soft iron de forma exagerada per facilitar-ne la visualització i comprensió. A la secció 3.5.3.2.2 es pot veure una mostra de dades real.

Un cop hem descrit els efectes de les pertorbacions sobre un conjunt de dades, és més simple comprendre el procediment de calibratge.

El primer pas d'aquest procediment consisteix en prendre dades mentre rotem el sensor en totes les direccions de l'espai. D'aquesta manera generem un núvol de punts com el que hem vist a la grafica 3.5. Llavors, el següent pas serà determinar els offsets que cal eliminar

per aconseguir que el centre del núvol quedi a l'origen. Aquests valors de offset determinen el valor de la variable b que descriu la pertorbació de hard-iron en el model descrit. Finalment, caldrà ajustar l'el·lipsoide a una esfera tot estimant el valor de la matriu D del soft-iron. Aquesta matriu és responsable de l'escalat de les dades en cada eix i cal estimar-la iterativament. Hi ha múltiples mètodes per realitzar l'estimació d'aquests paràmetres. En aquest projecte s'ha utilitzat una implementació en matlab de l'algorisme TWOSTEP (Alonso R. and Shuster M., 2002) tal i com apareix a (Justin Peter Dinale, 2013).

Amb un experiment, anem a veure fins a quin punt poden afectar les distorsions a les mesures del heading.

3.5.3.2.2. Proves magnètic Heading

Una de les proves que hem fet per veure els afectes de les pertorbacions magnètiques en l'estimació del Yaw ha estat la següent. S'ha agafat les dades de la IMU en un mateix lloc amb quatre orientacions de Yaw diferents utilitzant una referència física per assegurar-ne la seva correcció (figura 3.12). Aquestes orientacions tenien una diferència de 90º les unes amb les altres, per tant, quan calculem el heading utilitzant els magnetòmetres les diferències entre les orientacions també hauria de ser 90º. Però veurem com, degut a l'efecte de les pertorbacions no és així.

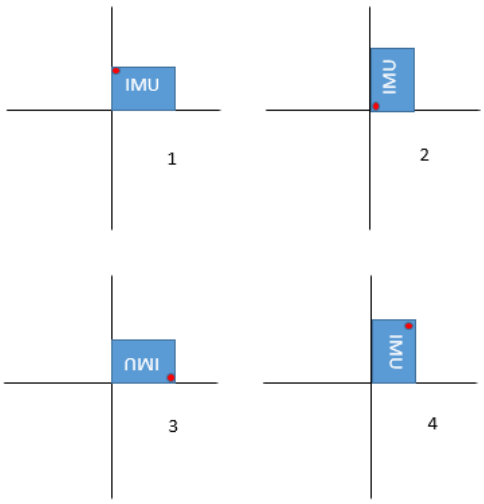
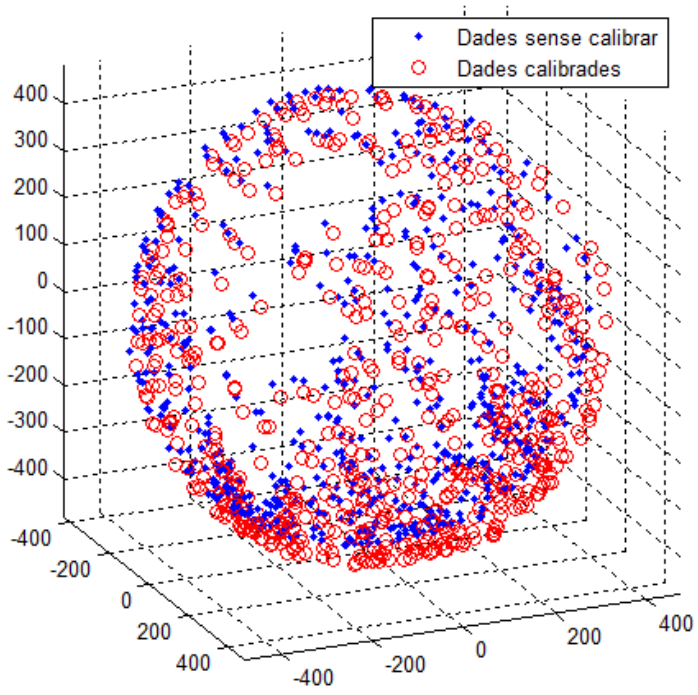
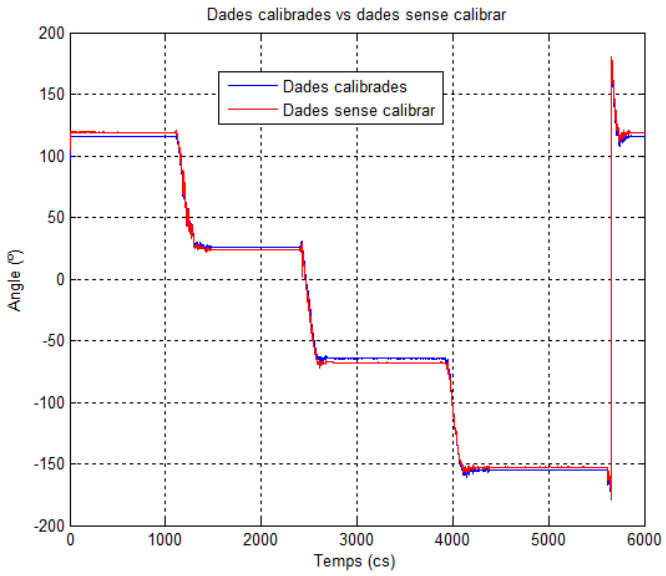


Figura 3.12: Esquema de la prova de Heading

Per comprovar l'efecte de les pertorbacions primer s'ha calculat l'angle Yaw amb les dades sense calibrar, després s'han calibrat les dades i s'ha tornat a calcular l'angle Yaw de cada posició. Cada un dels esglaons de la gràfica 3.7 és una de les posicions que es mostren a la figura 3.12. A la taula 3.3 es mostra una comparativa de les mesures d'angles que s'han pres amb les dades calibrades o sense calibrar. També mostrem les dades calibrades del nostre sensor i les dades sense calibrar. Si els magnetòmetres no patissin cap tipus de pertorbació els punts blaus es trobarien dins dels cercles vermells en la gràfica 3.6, però veiem que no és així.



Gràfica 3.6: Dades calibrades vs dades sense calibrar del sensor ADIS16488



Gràfica 3.7: Evolució de les dades dels magnetòmetres en les diferents posicions

Prova	**Angles**			
	Posició 1	**Posició 2**	**Posició 3**	**Posició 4**
Dades sense calibrar	119.05	23.87	-67.85	-152.80
Dades calibrades	115.80	26.02	-64.51	-154.98

Taula 3.3: Dades calibrades vs dades sense calibrar

Posicions	**Diferència d'angle entre les dos posicions (º)**			
	Dades sense calibrar		**Dades calibrades**	
	Diferència	**Error***	**Diferència**	**Error***
1-2	95.18	5.18	89.78	0.22
2-3	91.72	1.72	90.53	0.53
3-4	84.95	5.05	90.47	0.47
4-1	88.15	1.85	89.22	0.78
Mitjana*		3.45		0.5

Taula 3.4: Diferència d'angles entre les posicions

**Alhora de calcular l'error es fa en valor absolut, sense tenir en compte si es passa o falta per arribar a 90. La mitjana es fa de l'error en valor absolut.*

Amb aquesta prova es pot veure com un sensor pertorbat no mesura correctament els angles reals i que, després d'un calibratge de les dades, hi ha una gran millora de la precisió. L'error de les dades calibrades no arriba a 1º que és un valor acceptable per a sensors com el que hem utilitzat per a l'experiment.

S'ha de comentar que aquesta prova està pensada per demostrar com es compensen el hard i el soft-iron, però no permet determinar com de precís és el heading respecte el nord. Per analitzar això, faria falta algun equip més precís (no basat en magnetòmetres per evitar patir les mateixes pertorbacions) contra el que comparar les dades (un girocompàs, un sistema GPS centimètric amb dues antenes, etc), però són equipaments cars i no n'hem pogut disposar per aquest projecte.

La implementació de les tècniques de leveling i magnètic heading amb Matlab s'anomenen "leveling.m" i "magneticHeading.m". El seu codi es pot trobar a l'annex C, secció C.5.

3.6. Equacions de navegació

Anem a endinsar-nos al cor de l'estudi d'aquest treball, el que anomenem les equacions de navegació. Les equacions de navegació són les encarregades de, un cop passada l'etapa d'inicialització, anar calculant un nou valor de la velocitat, la posició i l'orientació gràcies a les dades que proporciona la IMU.

Les equacions de navegació es troben dins la unitat de processament de la informació d'un INS. Mitjançant la navegació per estima (secció 3.1) estimen la nova posició, velocitat i orientació de cada iteració. Les equacions de navegació s'executen cada vegada que la IMU proporciona un nou conjunt de dades (giroscopis i acceleròmetres) i que per tant, l'interval de temps en que s'integren les dades ve determinat per la freqüència del sensor. Si la unitat

de mesura inercial treballa a una freqüència de 100Hz envia cent mesures per segon de cada sensor. A cada una d'aquestes mesures les equacions de navegació integren el valor dels acceleròmetres i giroscopis per trobar el nou valor de la velocitat, posició i orientació.

Per la implementació de les equacions de navegació s'han utilitzat els sistemes de referència ECI i ECEF, i llavors, també hi ha una tercera implementació on s'utilitza el sistema NED per resoldre velocitats i orientacions, però on utilitzem l'ECEF per descriure la posició en termes de latitud, longitud i alçada. Aquesta última implementació s'anomena sistema local de navegació. Les equacions de navegació varien segons si s'utilitza un sistema de referència o un altre. En les següents pàgines s'explicaran les equacions necessàries per implementar els tres sistemes en Matlab, així com també s'observaran els seus resultats.

Recordar que totes les equacions de navegació implementades en aquest capítol s'han extret del llibre (Paul D. Groves, 2013).

3.6.1. Nomenclatura

Quan a continuació s'expliquin les equacions de navegació s'utilitzaran termes amb una nomenclatura una mica complicada d'entendre. Per això fem aquest subcapítol previ, per així poder explicar petits conceptes que serviran per una millor comprensió de les equacions de navegació.

Primer de tot, quan ens volem referir a un sistema de referència determinat, ECI, ECEF, NED o Body ho farem amb les següents lletres i, e, n, b respectivament.

Es veuran moltes variables amb la següent nomenclatura.

$$x_{\beta\alpha}^{\gamma}$$

On:

x: és la magnitud de la variable que volem representar (força específica, velocitat, posició...)

β: sistema de referència respecte el qual es mou.

α: el sistema al qual afecta la magnitud de la variable x. O sigui, si x és velocitat seria el sistema que es mou a velocitat x.

γ: eixos en el qual es representat la magnitud de la variable.

Anem a fer-ho amb un exemple perquè sigui més entenedor.

$$v_{ib}^{i}$$

On:

v: valor de la velocitat a la que es mou el INS.

b: sistema de referència que es mou a la velocitat v. En aquest cas el sistema de referència Body de la IMU.

i: sistema de referència respecte el qual ens movem a velocitat v (ECI).

i: eixos en els quals representem la velocitat (ECI).

Quan parlem de matriu de rotació es representaran de la següent forma:

$$C_{\beta}^{\alpha}$$

On:

C: és la matriu de rotació o orientació.

α: és el nou sistema en el qual volem representar les dades.

β: és el sistema en el qual tenim les dades referenciades.

Finalment, en les diverses equacions que es mostraran a continuació veurem moltes variables acompanyades d'un (+) o (-). Això indica si és el valor nou de l'actual iteració o el valor que hem calculat a la iteració anterior respectivament. Per exemple, v(+) seria el valor de velocitat actual i v(-) seria el valor de velocitat calculat a la iteració anterior.

Feta aquesta petita explicació ja podem començar a explicar les equacions de navegació.

3.6.2. Equacions de navegació de ECI

Les equacions de navegació de l'ECI són les més senzilles. Tot i així l'avaluació dels seus resultats és confusa. El motiu és perquè, com s'ha explicat anteriorment, el sistema de referència ECI és un sistema de coordenades col·locat al centre de la Terra però que no gira amb ella, sinó que la seva orientació es manté fixa respecte la resta de l'univers. Això fa que els punts de la Terra no tinguin sempre les mateixes coordenades sinó que van variant amb el temps (degut a la rotació de la Terra), per tant, quan obtenim unes coordenades de posició calculades respecte ECI es fa difícil saber d'on són realment. La solució és aplicar una transformació al valor de posició calculat respecte ECI per representar-lo respecte un altre sistema de coordenades (ECEF per exemple).

Per poder arribar a estimar la posició s'han de seguir quatre grans passos que representem en el diagrama de blocs (figura 3.13) que mostrem a continuació.

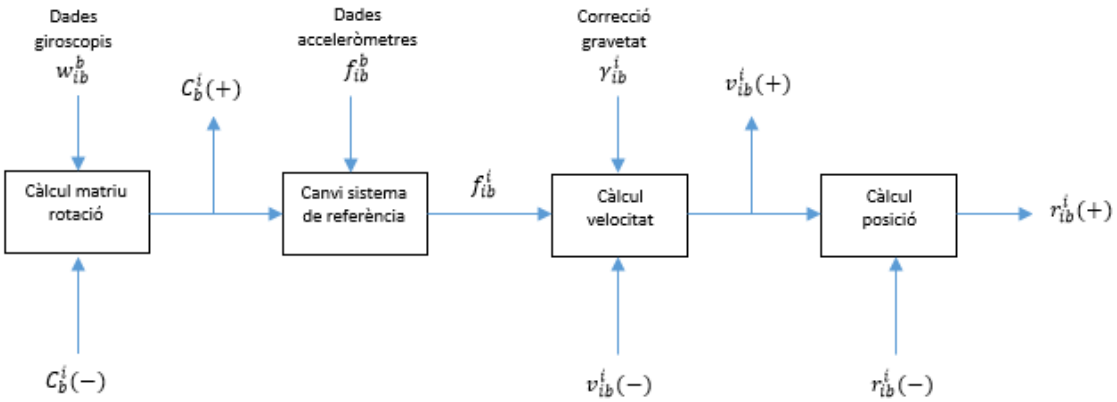


Figura 3.13: Diagrama de blocs ECI

Tot seguit, mostrarem cada un dels blocs individualment, així com les equacions que s'han d'aplicar a cadascun d'ells.

3.6.2.1. Càlcul matriu de rotació

El primer que hem de fer és calcular la matriu de rotació (C). Aquesta matriu ens permetrà transformar les dades que obtenim de la IMU, que estant representades respecte el Body, al sistema de referència en el qual volem treballar. En el nostre cas, aquest sistema serà l'ECI. A més, aquesta matriu de rotació C també ens permetrà trobar la orientació en que es troba la

unitat de mesura inercial. Anem a veure l'equació (Eq. 3.8) necessària per trobar la nova matriu de rotació a cada iteració.

$$C_b^i(+) = C_b^i(-) \cdot \left(I_3 + \Omega_{ib}^b \cdot t\right) \qquad \textit{(Eq. 3.8)}$$

On:

$C_b^i(+)$: és la matriu de rotació que volem calcular que aplica una rotació de Body a ECI.

$C_b^i(-)$: és la matriu de rotació calculada en la iteració anterior.

I_3: matriu identitat de 3x3.

Ω_{ib}^b: és la matriu antisimètrica amb els valors de la velocitat angulars (rad/s) extrets de la IMU.

t: temps de durada d'una iteració (s).

3.6.2.2. Canvi de sistema de referència de la força específica

Recordem que els tres acceleròmetres de la nostre unitat de mesura inercial mesuren el que anomenem força específica. La mesura que fan els sensors de la IMU es troba representada en el sistema de referència Body i per fer els càlculs la necessitem respecte ECI. Per tant, hem d'aplicar una rotació a les dades dels acceleròmetres per canviar-les de sistema de referència. Això es fa amb la següent equació (Eq. 3.9).

$$f_{ib}^i = \frac{1}{2} \cdot \left(C_b^i(-) + C_b^i(+)\right) \cdot f_{ib}^b \qquad \textit{(Eq. 3.9)}$$

On:

f_{ib}^i: són les dades dels acceleròmetres que referenciades sobre el ECI.

f_{ib}^b: són les dades que ens donen els acceleròmetres de la IMU. Estan referenciades respecte el Body.

3.6.2.3. Càlcul velocitat

Un cop ja tenim la matriu de rotació i la forca específica representada sobre el sistema que volem, podem procedir a fer el càlcul de velocitat. En aquest càlcul també extraiem el valor de la gravetat. Recordem que s'havia de fer una correcció de la gravetat perquè la mesura dels acceleròmetres no era l'acceleració pròpiament dita sinó el que anomenem força específica que conté totes les forces no gravitacionals per unitat de massa, és a dir, el que mesura quan està en repòs és la força que contraresta la força de la gravetat. Així doncs per calcular la velocitat apliquem (Eq. 3.10):

$$v_{ib}^{i}(+) = v_{ib}^{i}(-) + \left(f_{ib}^{i} + \gamma_{ib}^{i}(r_{ib}^{i}(-))\right) \cdot t \qquad \textit{(Eq. 3.10)}$$

On:

$v_{ib}^{i}(+)$: és la velocitat (m/s) respecte ECI que volem calcular a l'instant actual.

$v_{ib}^{i}(-)$: la velocitat (m/s) de la iteració anterior.

$\gamma_{ib}^{i}(r_{ib}^{i}(-))$: acceleració de la gravetat (m/s^2). Entre parèntesis la posició de la Terra a la qual s'ha de calcular l'acceleració de la gravetat.

3.6.2.4. Càlcul posició

Finalment, l'últim pas és el càlcul de la posició. Per realitzar-lo aplicarem la següent fórmula (Eq. 3.11):

$$r_{ib}^{i}(+) = r_{ib}^{i}(-) + v_{ib}^{i}(+) \cdot t - \left(f_{ib}^{i} + \gamma_{ib}^{i}\left(r_{ib}^{i}(-)\right)\right) \cdot \frac{t^2}{2} \qquad \textit{(Eq. 3.11)}$$

On:

$r_{ib}^{i}(+)$: és la posició respecte ECI que volem calcular (m).

$r_{ib}^{i}(-)$: és la posició respecte ECI de la iteració anterior.

Aquesta és l'última equació necessària. Seguint aquests passos podem estimar la posició, la velocitat i l'orientació respecte el sistema ECI gràcies a les dades que extraiem de la nostre

IMU. L'arxiu Matlab que conté la implementació d'aquestes equacions té el nom de "equacionsNavegacioECI.m". El codi informàtic també es pot trobar a l'annex C, secció C.6. Els seus càlculs detallats es poden trobar a l'annex B, secció B.7.

3.6.3. Equacions de navegació de ECEF

L'avantatge d'implementar les equacions de navegació respecte l'ECEF en lloc de l'ECI és que els seus resultats són més fàcilment interpretables, ja que cada posició de la Terra té sempre les mateixes coordenades. El problema és que el fet que el sistema de coordenades estigui fixa a la Terra i roti amb ella afegeix una mica de complexitat als càlculs.

Els passos a seguir són semblants als d'abans però, amb petits canvis ques es poden apreciar al següent diagrama de blocs (figura 3.14).

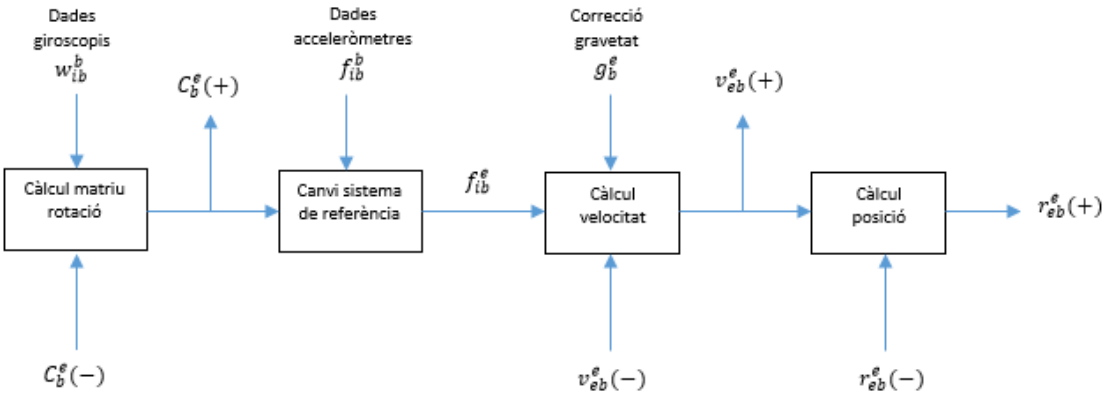


Figura 3.14: Diagrama de blocs ECEF

3.6.3.1. Càlcul matriu de rotació

Aquí ja trobem la primera diferència respecte el cas anterior. En el càlcul de la matriu de rotació que va de Body al sistema de referència ECEF hem de tenir en compte l'efecte de la rotació de la Terra. Això significa que cal substraure l'efecte de la rotació de la Terra sobre les mesures dels giroscopis, els quals mesuren la rotació respecte un sistema inercial i no respecte un sistema de coordenades giratori com ara l'ECEF. Com podem veure l'equació canvia:

$$C_b^e(+) = C_b^e(-) \cdot \left(I_3 + \Omega_{ib}^b \cdot t\right) - \Omega_{ie}^e \cdot C_b^e(-) \cdot t \qquad \textit{(Eq. 3.12)}$$

On:

$C_b^e(+)$: és la matriu de rotació actual de Body a ECEF.

$C_b^e(-)$: és la matriu de rotació de la iteració anterior.

Ω_{ie}^e: és la matriu antisimètrica del vector de rotació de la Terra (rad/s).

3.6.3.2. Canvi de sistema de referència de la força específica

En aquest cas, no hi ha gairebé diferència respecte els càlculs amb el sistema ECI. És tracta de la mateixa equació però simplement canviant la matriu de rotació.

$$f_{ib}^e = \frac{1}{2} \cdot (C_b^e(-) + C_b^e(+)) \cdot f_{ib}^b \qquad \textit{(Eq. 3.13)}$$

On:

f_{ib}^e: és el valor de la força específica referenciat a l'ECEF.

3.6.3.3. Càlcul velocitat

Degut a la rotació del sistema de referència ECEF en l'equació (Eq. 3.14) del càlcul de la velocitat hi apareix el terme de la força de Coriolis. A més, la correcció de la gravetat també pateix un canvi respecte a la correcció que s'aplicava en les equacions de navegació respecte ECI. La correcció g_b^e que s'aplica és la suma de la força de la gravetat en aquell punt i un petit terme d'acceleracions centrífugues.

$$v_{eb}^e(+) = v_{eb}^e(-) + \left(f_{ib}^e + g_b^e(r_{eb}^e(-)) - 2 \cdot \Omega_{ie}^e \cdot v_{eb}^e(-)\right) \cdot t \qquad \textit{(Eq. 3.14)}$$

On:

$v_{eb}^e(+)$: és el nou vector velocitat que volem calcular (m/s).

$v_{eb}^e(-)$: és el vector de velocitat calculat a la iteració anterior (m/s).

$g_b^e(r_{eb}^e(-))$: correcció de la gravetat en el punt r_{eb}^e de la Terra.
$2 \cdot \Omega_{ie}^e \cdot v_{eb}^e(-)$: component de Coriolis.

3.6.3.4. Càlcul posició

Finalment, l'equació (Eq. 3.15) del càlcul de la posició on s'hi afegeix l'efecte provocat per la força de Coriolis.

$$r_{eb}^e(+) = r_{eb}^e(-) + v_{eb}^e(-) \cdot t + \left(f_{ib}^e + g_b^e(r_{eb}^e(-)) - 2 \cdot \Omega_{ie}^e \cdot v_{eb}^e(-)\right) \cdot \frac{t^2}{2} \qquad \textit{(Eq. 3.15)}$$

On:
$r_{eb}^e(+)$: la posició en coordenades ECEF que volem calcular (m).
$r_{eb}^e(-)$: la posició en coordenades ECEF obtinguda en la iteració anterior (m).

Fins aquí, les equacions de navegació que s'han d'aplicar per estimar la posició, la velocitat i l'orientació en coordenades ECEF. La implementació d'aquestes equacions es troben en l'arxiu Matlab anomenat "equacionsNavegacioECEF.m". A l'annex C, secció C.6 també podeu trobar el codi informàtic. Els seus càlculs detallats es poden trobar a l'annex B, secció B.7.

3.6.4. Equacions de navegació en el sistema local de navegació

A continuació s'explicarà l'últim bloc d'equacions de navegació. El sistema de referència NED s'utilitza per resoldre la velocitat i l'orientació, però la posició es representa respecte l'ECEF amb les coordenades de latitud, longitud i alçada respecte el geoide. Són les equacions més complicades d'implementar dels tres sistemes de referència amb els quals hem treballat. El seu avantatge recau en la fàcil interpretació de les dades, sobretot a nivell humà. Per exemple, la posició es representa en coordenades de longitud i latitud, que en termes de posició de vehicles és la forma més senzilla d'interpretar. En termes d'orientació, la representa respecte el nord i el pla horitzontal i això és una forma més natural de definir les rotacions.

Aquest és el diagrama de blocs (figura 3.15) que s'ha de seguir.

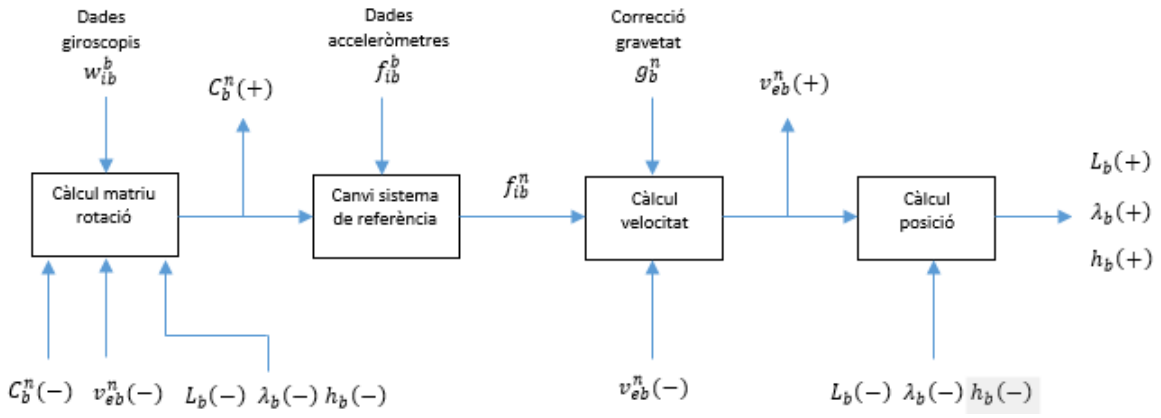


Figura 3.15: Diagrama de blocs NED

Tot seguit mostrem les equacions de navegació que s'han implementat.

3.6.4.1. Càlcul matriu de rotació

A diferència dels dos grups d'equacions de navegació que hem vist fins ara, pel càlcul de la matriu de rotació en el sistema local de navegació, a part de les dades dels giroscopis també s'utilitza la posició i la velocitat. Això és necessari perquè l'orientació dels eixos nord, est i down del sistema NED varia segons els sistema de navegació es mou respecte la Terra.

$$C_b^n(+) = C_b^n(-) \cdot \left(I_3 + \Omega_{ib}^b \cdot t\right) - \left(\Omega_{ie}^n(-) + \Omega_{en}^n(-)\right) \cdot C_b^n(-) \cdot t \qquad \textit{(Eq. 3.16)}$$

On:

$C_b^n(+)$: Matriu de rotació que volem trobar. Transforma les dades de Body a NED.

$C_b^n(-)$: Matriu de rotació del càlcul immediatament anterior.

$\Omega_{ie}^n(-)$: Té en compte la rotació de la Terra segons a la posició en que ens trobem.

$\Omega_{en}^n(-)$: Matriu antisimètrica amb les derivades de la longitud i la latitud.

3.6.4.2. Canvi de sistema de referència de la força específica

En aquest apartat la fórmula (Eq. 3.17) segueix sense canviar.

$$f_{ib}^{n} = \frac{1}{2} \cdot (C_b^n(-) + C_b^n(+)) \cdot f_{ib}^{b} \qquad \textit{(Eq. 3.17)}$$

On:

f_{ib}^{n}: és el valor de la força específica respecte el sistema NED.

3.6.4.3. Càlcul velocitat

Si ens fixem amb els subíndex de la variable de velocitat podem veure com aquesta es referencia respecte l'ECEF però està resolta en els eixos del NED. Això és així per una senzilla raó, el NED té la particularitat que es mou sempre amb la IMU, per tant, sempre es mou a la mateixa velocitat que la unitat de mesura inercial. O sigui, que si volguéssim referenciar la velocitat respecte el sistema de coordenades NED aquesta sempre ens donaria zero. Per tant, es referencia respecte uns altre origen, el del ECEF.

$$v_{eb}^{n}(+) = v_{eb}^{n}(-) + \left(f_{ib}^{n} + g_b^n(L_b(-), h_b(-)) - \left(\Omega_{en}^{n}(-) + 2 \cdot \Omega_{ie}^{n}(-)\right) \cdot v_{eb}^{n}(-)\right) \cdot t \qquad \textit{(Eq. 3.18)}$$

On:

$v_{eb}^{n}(+)$: el nou valor de velocitat que volem calcular (m/s).

$v_{eb}^{n}(-)$: el valor anterior de velocitat (m/s).

$g_b^n(L_b(-), h_b(-))$: gravetat segons els paràmetres de latitud i altura respecte del geoide.

$\left(\Omega_{en}^{n}(-) + 2 \cdot \Omega_{ie}^{n}(-)\right) \cdot v_{eb}^{n}(-)$: són termes per contraresta la rotació de la Terra.

3.6.4.4. Càlcul posició

En el càlcul de la posició és on aquestes equacions pateixen la diferència més gran respecte les representades en els altres dos sistemes de coordenades. A diferència d'ECEF i ECI en què el valor de la posició es representava en coordenades cartesianes, amb les equacions de

navegació del sistema local de navegació la posició es representa en forma de latitud (L), longitud (λ) i altura respecte del geoide (h). La latitud i la longitud s'expressen en radians i l'altura en metres.

$$h_b(+) = h_b(-) - \frac{t}{2} \cdot \left(v^n_{eb,D}(-) + v^n_{eb,D}(+) \right) \qquad \textit{(Eq. 3.19)}$$

$$L_b(+) = L_b(-) + \frac{t}{2} \cdot \left(\frac{v^n_{eb,N}(-)}{R_N\left(L_b(-)\right) + h_b(-)} + \frac{v^n_{eb,N}(+)}{R_N\left(L_b(-)\right) + h_b(+)} \right) \qquad \textit{(Eq. 3.20)}$$

$$\lambda_b(+) = \lambda_b(-) + \frac{t}{2} \cdot \left(\frac{v^n_{eb,E}(-)}{\left(R_E\left(L_b(-)\right) + h_b(-)\right) \cdot \cos L_b(-)} + \frac{v^n_{eb,E}(+)}{\left(R_E\left(L_b(+)\right) + h_b(+)\right) \cdot \cos L_b(+)} \right) \qquad \textit{(Eq. 3.21)}$$

On:

$h_b(+)$, $h_b(-)$: són l'altura respecte el geoide que volem trobar i la que ja tenim de la iteració anterior. (m)

$v^n_{eb,D}$: és la component down (valor de la 3a fila) de la velocitat calculada a l'apartat anterior. Si enlloc de una D és una N voldrà dir component nord (1a fila) i si és una E component est (2a fila). (m/s)

$L_b(+)$ i $L_b(-)$: valor de latitud que volem calcular i valor de latitud calculat a la iteració anterior. (rad)

$R_N\left(L_b(-)\right)$: valor del radi de curvatura del meridià segons la latitud en què ens trobem.

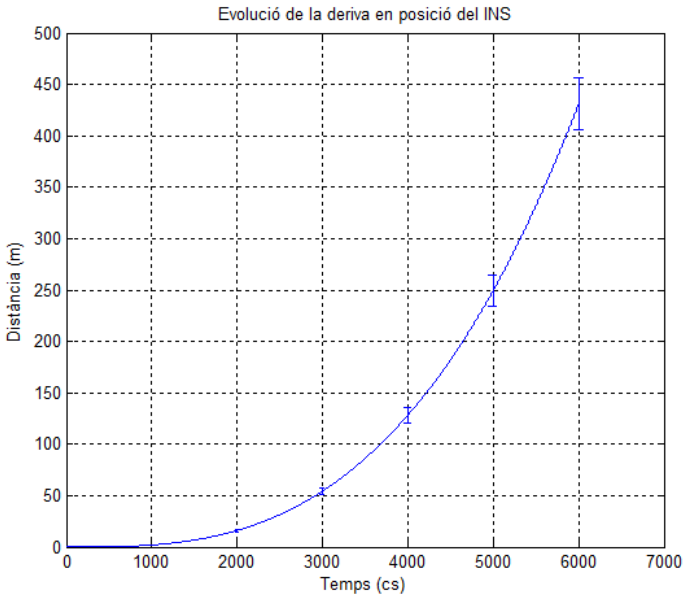
$R_E\left(L_b(-)\right)$: el valor del radi de curvatura transversal.

$\lambda_b(+)$ i $\lambda_b(-)$: valor de longitud que volem calcular i valor de longitud calculat a la iteració anterior. (rad)

Fins aquí les formules necessàries per implementar les equacions de navegació respecte el sistema local de navegacio. L'arxiu Matlab equacionsNavegaciónED conté aquest procediment per estimar el valor de la posició, velocitat i orientació. El codi informàtic també es pot trobar a l'annex C, secció C.6. Els seus càlculs detallats es poden trobar a l'annex B, secció B.7.

3.6.5. Proves equacions de navegació

Un cop tenim les equacions de navegació implementades en Matlab i és hora de fer proves per avaluar el funcionament global de la INS. El primer seguit de proves que s'han fet han consistit en recollir dades de la IMU (en aquest cas l'ADIS16488) durant un minut mentre aquest romania quieta i plana en una taula. Teòricament, el seu desplaçament final hauria de ser zero ja que la unitat de mesura inercial no s'ha mogut. Anem a veure els resultats obtinguts en una gràfica. La gràfica 3.8 dibuixa el desplaçament mitjà que han estimat les equacions de navegació després de fer tot un seguit de 10 proves. Les barres d'error mostren la desviació estànder de les proves.

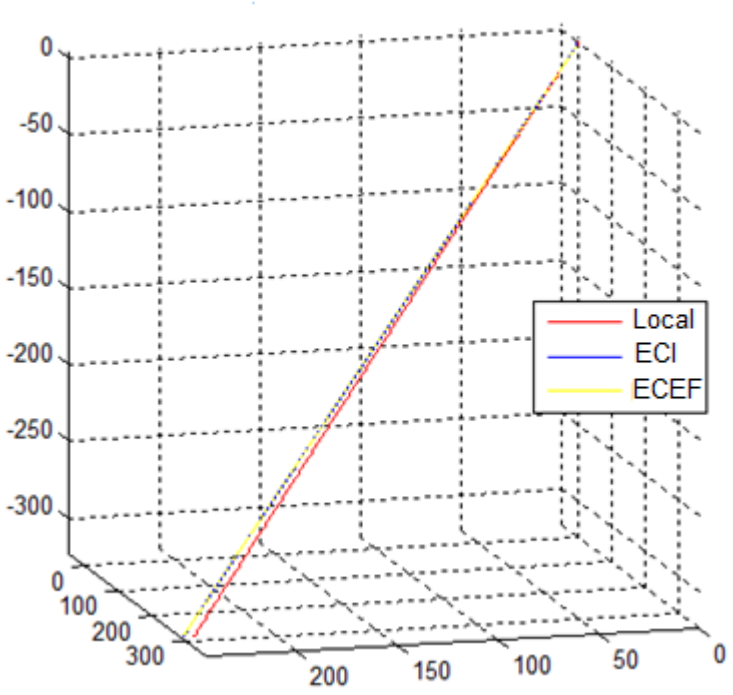


Gràfica 3.8: evolució de la dervia en un INS amb errobars que mostren la desviació de les 10 proves realitzades

Veiem que després d'un minut les equacions de navegació estimen que la IMU s'ha mogut uns 420m de la seva posició inicial. Estem parlant d'un error molt gran ja que el desplaçament hauria de ser zero.

Una altre de les proves que hem fet ha estat calcular l'estimació de la posició amb els tres tipus d'equacions de navegació que s'han vist (ECI, ECEF i sistema local de navegació), però utilitzant el mateix conjunt de dades extretes de la IMU. De manera que, si les equacions

funcionen bé totes haurien de donar el mateix resultat. Per poder comparar els resultats hem passat les estimacions de posició fetes amb ECI i el sistema local de navegació a ECEF utilitzant els canvis de sistemes de coordenades esmentats en la secció 3.2.5 d'aquest treball.



Gràfica 3.9: Comparació de l'estimació de la posició amb els diferents tipus d'equacions

Com podem veure a la gràfica 3.9 els tres tipus d'equacions navegació donen una estimació de la posició pràcticament igual. Això és un indicador que funcionen correctament.

El fet de l'error que va augmentant en el temps que pateixen els INS era un terme que ja es coneixia, per tant, els nostres esforços a partir d'ara hauran d'anar enfocats en reduir el màxim possible aquest error. Els següents capítols tractaran sobre els mètodes i tècniques que s'aplicaran per reduir la deriva.

4. DESCRIPCIÓ I CORRECCIÓ DE LA DERIVA DE L'ERROR

Com s'ha comentat anteriorment una de les preocupacions més grans alhora de dissenyar un sistema de navegació inercial és reduir el màxim possible la seva deriva de l'error. Hem vist que sinó es fa cap correcció a les dades l'error pot ser molt gran en poc temps (per exemple, amb l'ADIS 420m en un minut).Per tant, en aquest capítol ens centrarem en aquest tema. Explicarem les tècniques utilitzades per reduir la deriva del nostre INS i n'examinarem els resultats.

Per reduir la deriva utilitzarem un petit filtre de Kalman. És un algorisme de fusió sensorial que ens permet mesclar informacions de diverses fonts per tal d'obtenir una millor estimació de la trajectòria realitzada. En aquest treball només utilitzarem el GPS i zero Updates en el filtre de Kalman, però la mateixa arquitectura que mostrarem es pot utilitzar com a base per la integració final en el vehicle on s'haurien d'incorporar dades d'altres sensors de navegació (magnetòmetres, baròmetres, DVL, USBL, GPS...). Però abans d'explicar-lo ens centrarem en fer una petita explicació sobre quines són les causes de l'error.

4.1. Models d'error

L'error que pateix un sistema de navegació inercial prové principalment del mostreig de dades que fa la IMU. Com de més bona qualitat sigui la unitat de mesura inercial més afinarà alhora d'agafar les dades i per tant, menys error hi haurà. Els errors en el mostreig de dades pot ser degut a diferents causes (Paul D. Groves, 2013).

Biaix: és un error que té components constants i variables exhibit per la mesura dels acceleròmetres i els giroscopis. Sol ser el terme dominant en l'error que pateix un INS. El biaix pot ser diferent cada vegada que arranca el sensor. Estimar-lo serà una de les funcions del filtre de Kalman, llavors es farà una feedback per corregir la IMU amb aquesta estimació. El biaix s'anomenarà amb la lletra b.

Factor d'escala: és un error proporcional a la mesura del sensor. Com més gran és la mesura del sensor més gran és l'error.

Desalineament d'eixos: que els sensors (acceleròmetres i giroscopis) no estiguin perfectament ortogonals entre si.

Soroll: les dades que donen els sensors tenen el que s'anomena soroll. El soroll és completament aleatori i a la pràctica es pot aproximar a un soroll blanc. La variància d'aquest soroll depèn de la qualitat del sensor.

Tots aquests errors es poden introduir dins d'aquest model (Eq. 4.1). La mateixa equació es pot aplicar als giroscopis també.

$$\widehat{f_{ib}^{b}} = b_a + (I_3 + M_a) \cdot f_{ib}^{b} + w_a \qquad \textit{(Eq. 4.1)}$$

On:

$\widehat{f_{ib}^{b}}$: són les dades de força específica que ens donen els acceleròmetres-

b_a: el biaix dels acceleròmetres.

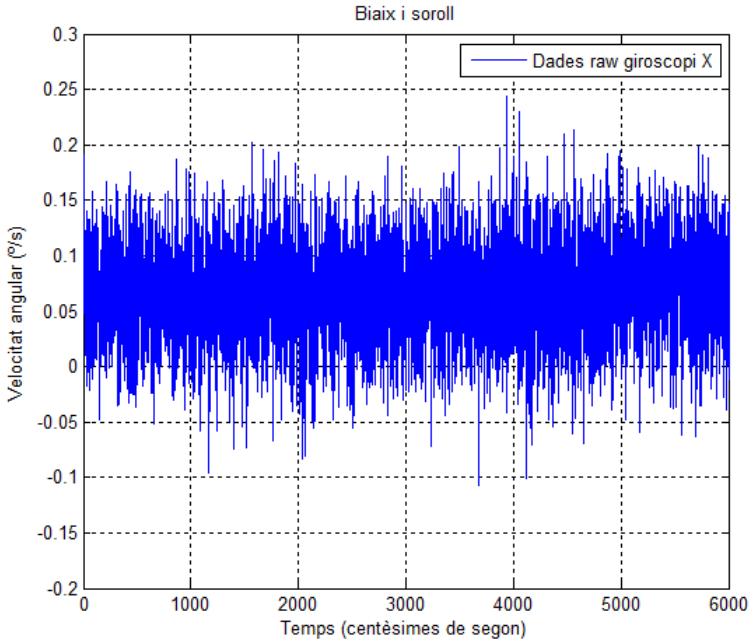
M_a: la matriu que conté el factor d'escala i l'error provocat pel desalineament.

f_{ib}^{b}: la veritable força específica que pateix la IMU.

w_a: el soroll de les dades dels acceleròmetres.

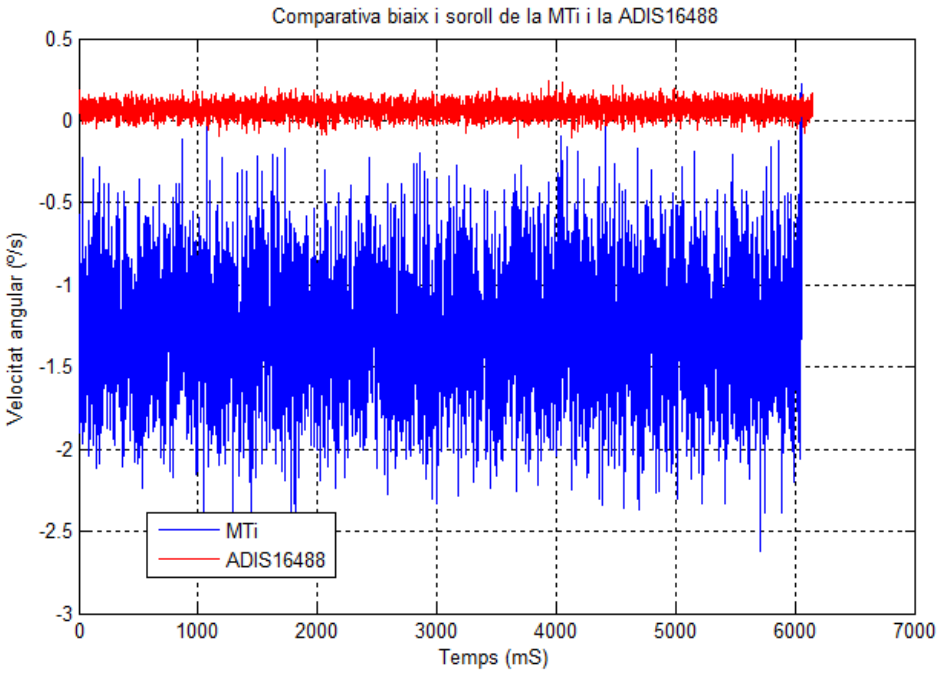
Part d'aquests errors són fàcilment compensables, amb l'equipament adequat, pels propis fabricants del sensor. Per aquest motiu, la principal font d'errors en les mesures d'una IMU correspon a aquelles componenets que varien durant l'operació i que per tant no es poden compensar a fàbrica. Aquests bàsicament són el biaix i el soroll. L'objectiu del filtre de Kalman serà l'estimació d'aquests biaixos, així com determinar els errors de navegació acumulat a causa del soroll i altres pertorbacions no compensades.

A continuació una mostra (gràfica 4.1) de dades de la IMU ADIS16488 que hem utilitzat on es poden apreciar els conceptes de biaix i soroll. En aquest mostreig la unitat de mesura inercial estava en repòs, o sigui, velocitats angulars i acceleracions haurien de ser nul·les.



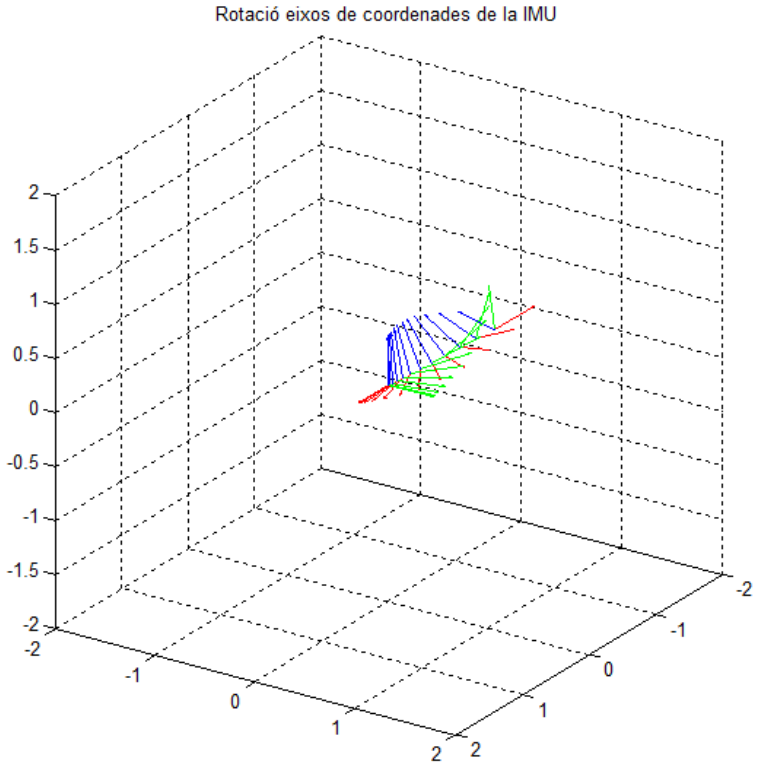
Gràfica 4.1: Mostra biaix i soroll

Després d'agafar les dades del giroscopi de l'eix x podem veure com aquestes no estan centrades en el zero sinó que ronden al voltant de 0.06 º/s. Això és una clar exemple de biaix. A banda del biaix, també podem veure una component aleatòria de la senyal causada per un soroll de variància. A la següent gràfica 4.2 podem veure la comparativa del biaix i del soroll de les dos IMU utilitzades per fer el treball (la MTi i la ADIS16488). Es pot apreciar com la MTi no té la matiexa qualitat que la ADIS16488. La MTi té un biaix i un soroll més gran.



Gràfica 4.2: Mostres dels giroscopis de la MTi i la ADIS16488

Una altre qüestió important és saber quins sensors són els principals causants de l'error, els acceleròmetres o els giroscopis. Tot i que a primera impressió ens pot semblar que els acceleròmetres són els principals causats dels errors en posició és tot el contrari, els principals causant d'error en l'estimació de la posició, velocitat i orientació són els giroscopis. L'explicació és molt senzilla. Imaginem que fem una prova i agafem les dades que ens dona la unitat de mesura inercial quan la deixem quieta sobre una taula. Quan executem les equacions de navegació, el biaix dels giroscopis provoca que el INS percebi erròniament un canvi en la seva orientació, quan realment no és així. En la següent gràfica hi ha dibuixats els eixos de coordenades del sensor. Podem veure com l'estimació obtinguda amb les equacions de navegació mostra una rotació inexistent causada pel biaix dels giroscopis.

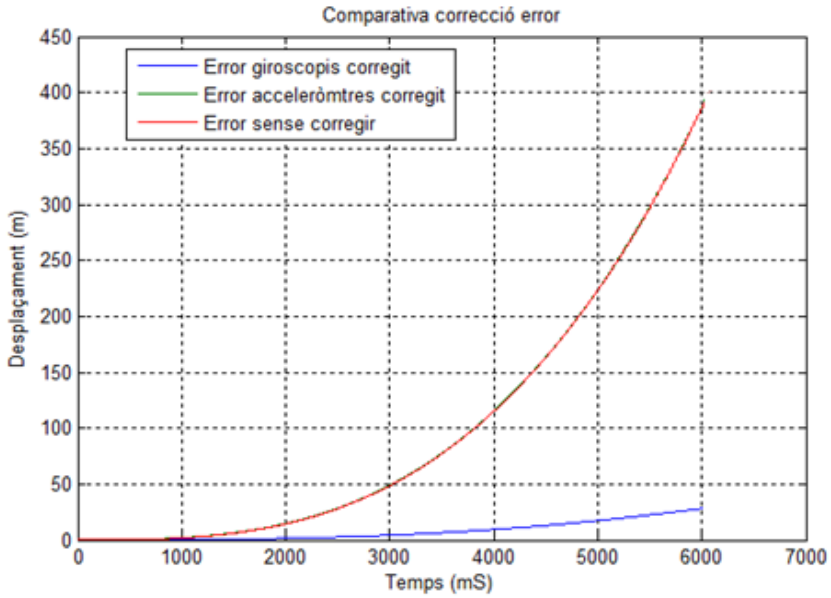


Gràfica 4.3: Evolució orientació

Aquesta estimació errònia de l'orientació té implicacions importants, ja que quan se sostrau el vector gravetat de la lectura de la força específica dels acceleròmetres, aquest està mal alineat i no es compensa adequadament. Enlloc d'aplicar-la només a l'eix z, que és com hauria de ser ja que la IMU es troba quieta i plana en una taula, l'aplica als altres eixos segons com cregui que estar rotada la unitat de mesura inercial. Això provoca que l'acceleració de la gravetat no quedi compensada correctament i per tant, el programa integri aquests residus com si fos una acceleració que esta patint el sensor. Com a resultat d'aquest procés, l'estimació de la posició derivarà cada cop més ràpidament.

Tot seguit mostrem una gràfica 4.4 on es veu l'evolució de l'error després de corregir el bias dels acceleròmetres, els giroscopis o, simplement, no corregir-lo. En aquest cas simple on el sensor no es mou (es troba quiet i pla en una taula), les mesures dels acceleròmetres i giroscopis són constants i l'orientació coneguda, és possible obtenir el biaix simplement fent la mitja de la seqüència de dades. En el cas de l'acceleració z primer se li ha de substraure la componenet de la gravetat.

Per els casos on aquestes condicions ideals no es compleixen, el filtre de Kalman que es presentarà a la secció 4.2 és una eina adequada per l'estimació del biaix.



Gràfica 4.4: Comparació de la deivia segons si corregim el biaix o no

Els resultats segons si corregim el biaix dels acceleròmetres o no són pràcticament iguals (la línia verda i vermella són gairebé coincidents). En canvi, si corregim el biaix dels giroscopis l'error disminueix molt.

Tot seguit, anem a veure què és un filtre de Kalman i com l'utilitzarem per trobar i poder corregir el biaix dels sensors.

4.2. Filtre de Kalman

Un filtre de Kalman és un algorisme que permet fusionar dades de dos o més sensors (Paul D. Groves, 2013). És molt utilitzat en el camp de la navegació ja que ens permet integrar les dades de diferents sensors (velocímetre Doppler, baròmetre...) i també de diferents sistemes de navegació (INS, GPS...). Aquesta fusió s'utilitza per corregir errors que pugui haver en la mesura dels sensors o en l'estimació dels sistemes de navegació i així aconseguir un millor resultat.

El filtre de Kalman té la següent estructura (figura 4.1) formada per dos grans etapes: la de predicció i la de "Measurement Update". Aquestes dues etapes s'executen cíclicament.

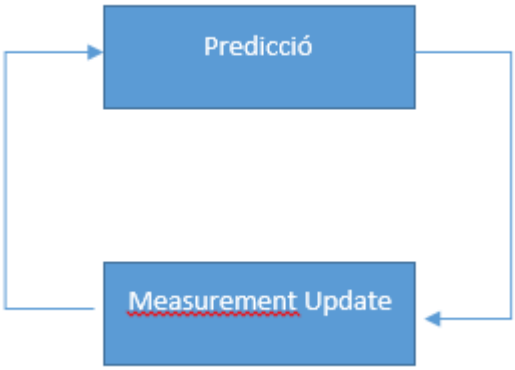


Figura 4.1: etapes del filtre de Kalman

Per poder explicar millor les dues etapes primer hem d'explicar que és el que en el filtre de Kalman s'anomenen el vector d'estat i la matriu de covariància P. El vector d'estat conté totes les variables que volem estimar mitjançant el filtre de Kalman. Aquest vector conté el valor mig estimat d'aquestes variables que volem conèixer. Per la seva banda, P és una matriu de covariàncies que a la diagonal conté les variàncies de cada element del vector d'estat i a fora de la diagonal conté les correlacions entre ells.

Un cop sabem això ja podem dir que l'etapa de predicció és l'encarregada de, mitjançant un model matemàtic, fer una predicció de quins valors prendran, en el següent instant de temps, les variables del vector d'estat així com la seva matriu de covariàncies.

L'etapa del "Measurement Update" és el moment en que s'incorpora la informació provinent de les dades dels sensors. En aquesta etapa s'actualitza la estimació del vector d'estat obtinguda durant l'etapa de predicció amb la informació més actual provinent de les mesures dels sensors. La contribució de cadascun (la predicció i les mesures) en l'estimació final ve determinada per les seves respectives variàncies.

Per poder entendre millor el filtre de Kalman i el seu funcionament anem a veure pas per pas com és la implementació del filtre utilitzat en aquest treball.

4.2.1. Implementació

Per explicar la implementació del nostre filtre de Kalman seguirem pas per pas les dos etapes principals de les que consta l'algorisme, entrant en detall en tots els aspectes importants.

Comentar que la implementació del filtre s'ha fet en el sistema de coordenades ECEF.

4.2.1.1. Etapa de Predicció

El primer pas que fem és introduir el vector d'estat que s'utilitza en aquest filtre de Kalman així com les variables que el formen. Recordem que el vector d'estat és el conjunt de variables que volem saber i estimar gràcies el filtre. És important afegir que, amb el filtre de Kalman, el que farem és intentar obtenir una estimació el més acurada possible d'aquest vector d'estat (no podem saber el seu valor exacte perquè no existeixen els sistemes de mesura perfectes). Aquesta es representa mitjançant la mitja de l'estimació de l'estat $\hat{x}^e_{INS}$ i la seva matriu de covariància associada P.

$$\hat{x}^e_{INS} = \begin{pmatrix} \delta\varphi^e_{eb} \\ \delta v^e_{eb} \\ \delta r^e_{eb} \\ b_a \\ b_g \end{pmatrix} \qquad \textit{(Eq. 4.2)}$$

On:

$\delta\varphi^e_{eb}$: vector 3x1 que conté l'error en l'orientació.

δv^e_{eb}: vector 3x1 que conté l'error de velocitat a cada eix.

δr^e_{eb}: vector 3x1 que conté l'error de posició a cada eix.

b_a: vector 3x1 on hi ha el biaix dels tres acceleròmetres de la IMU.

b_g: vector 3x1 amb el biaix dels tres giroscopis de la IMU.

Aquests errors acabats d'anomenar que s'intenten estimar amb el filtre de Kalman són la discrepància entre l'angle, velocitat i posició calculats per la INS i l'angle, velocitat i posició reals del vehicle.

La lògica podria fer pensar que el vector d'estat hauria d'estar format per les variables que es volen estimar amb el INS o sigui, la posició, la velocitat i l'orientació. Però no és així. El motiu és perquè si volguéssim que el vector d'estat estigues format per la posició, la velocitat i l'orientació, el model matemàtic de l'etapa de predicció del filtre hauria d'estar format per les equacions de navegació. Aquestes equacions contenen equacions no lineals i això faria que el model matemàtic fos molt més complicat i que s'hagués d'implementar una versió del filtre de Kalman anomenada filtre de Kalman estès, que és més inestable i computacionalment costosa que la implementació tradicional. Per aquest motiu s'ha escollit que el vector d'estat contingui l'estimació de l'error de posició, velocitat i orientació, ja que com veurem més endavant, dona lloc a un model lineal fàcilment implementable. Un cop estimat aquest error s'ha restat de les estimacions de posició, velocitat i orientació fetes pel sistema de navegació inercial. A més també s'ha estimat el biaix dels acceleròmetres i els giroscopis per així restar-lo de les mesures de la IMU i així evitar futures derives.

En aquest treball el vector d'estat s'inicialitzarà a zero. Inicialitzem la navegació en mesures absolutes, per tant, en l'instant zero el INS no haurà començat a derivar, o sigui que els errors del vector d'estat seran zero.

Un cop tenim clar el concepte del vector d'estat que utilitzarem ja podem aplicar la primer equació del filtre de Kalman. Aquesta equació ens relaciona la predicció amb l'estat anterior.

$$\hat{x}_k^- = \Phi_{k-1} \cdot \hat{x}_{k-1}^+ \qquad \textit{(Eq. 4.3)}$$

On:

$\hat{x}_k^-$: és la predicció del vector d'estat que realitzem en aquesta etapa del filtre.

Φ: és l'anomenada matriu de transició. És la que conté el model matemàtic per fer la predicció del nou vector d'estat. El càlcul de la matriu de transició el podrem trobar a l'annex B, secció 8.1.

$\hat{x}_{k-1}^+$: l'anterior estimació del vector d'estat que ha fet el filtre de Kalman.

Per obtenir la predicció del vector d'estat, no en fem prou de calcular-ne la mitja, sinó que també ens cal obtenir una predicció de la covariància associada. Aquí és on entra en joc la segona equació (Eq. 4.4) del filtre de Kalman. És l'encarregada de calcular la variància de les variables del vector d'estat.

$$P_k^- = \Phi_{k-1} \cdot P_{k-1}^+ \cdot \Phi_{k-1}^T + Q_{k-1} \quad \textit{(Eq. 4.4)}$$

On:

P_k^-: matriu d'error de covariància de l'etapa de predicció.

P_{k-1}^+: matriu d'error de covariància calculada al final de l'etapa measurement Update de la iteració anterior. A la seva diagonal trobem les variàncies de les variables del vector d'estat.

Q_{k-1}: és la matriu de covariància del soroll.

Els valors de la matriu de covariància P s'inicialitzen d'acord amb la precisió que tenen els mètodes utilitzats per inicialitzar (acceleròmetres pel leveling, GPS per la posició...). Per exemple, el nostre error en posició pot ser inicialitzat a 0, però si el nostre GPS té un error de 3m, és possible que la nostra posició real estigui a una distància similar de la posició correcta. La covariància ha de poder explicar això.

Com a part inherent del procés de fer una predicció, la incertesa de la predicció creix, ja que hi ha el terme additiu Q. La matriu Q és una matriu de covariàncies que descriu la incertesa inherent en el procés de predicció. Els valors que formen la matriu Q depenen de la qualitat de la IMU, per tant, les especificacions del nostre sensor en determinarà els seu valor.

Un cop explicades aquestes dos equacions, arribem al final de l'etapa de predicció del nostre filtre de Kalman. Hem vist quin vector d'estat tenim i per quines variables estar formades, hem vist com calcular la seva predicció mitjançant un model matemàtic. També hem vist com calcular la variància de les variables del vector d'estat.

Un cop acabada aquesta etapa ja podem seguir cap a la segona etapa que forma el filtre de Kalman, la de "Measurement Update".

4.2.1.2. Etapa de "Measurement Update"

Aquesta és l'etapa del filtre on s'introdueix el valor dels sensors o altres sistemes de navegació i es compara amb els valors del vector d'estat amb l'objectiu d'estimar com evolucionen. En aquest treball parlarem de com integrar informació de la INS amb un sensor comú com és el GPS.

Anem a veure quines equacions són les que s'han implementat en aquesta etapa del filtre de Kalman. El primer càlcul (Eq. 4.5) que s'ha introduït és el de la matriu K de Kalman. És l'encarregada de ponderar la informació que ve de la predicció (l'estat que es calcula a partir de l'estat anterior mitjançant un model matemàtic) amb la que ve dels sensors (en el nostre cas, la diferència entre la INS i el GPS). Aquesta ponderació es realitza en base a la credibilitat que tenen les dades, és a dir, la variància P de l'estimació i la incertesa R de les mesures del GPS.

$$K_k = P_k^- \cdot H_k^T \cdot (H_k \cdot P_k^- \cdot H_k^T + R_k)^{-1} \quad \textit{(Eq. 4.5)}$$

On:

K_k: és la matriu K de Kalman que volem calcular.

H_k: és l'anomenada measurement matrix. Defineix com el vector amb les mesures fetes pels sensors o altres sistemes de navegació es relacionen amb els elements del vector d'estat. La matriu H implementada en aquest treball es pot trobar a l'annex B, secció 8.2.

R_k: és la matriu de covariància del soroll del vector amb els mesuraments. És la incertesa de la mesura del sensor que fa l'Update. Els valors de R depenen de la incertesa en les mesures del GPS. La matriu implementada en aquest treball es trobarà a l'annex B, secció 8.2.

Un cop ja tenim la K de Kalman ja podem fer l'estimació actualitzada del vector d'estat. En aquesta part és on mitjançant la ponderació de la informació de la predicció i la informació provinent dels sensors (diferència entre INS i GPS) s'obté la nova mitja de l'estimació del vector d'estat $\hat{x}_k^+$. Veiem com en aquesta estimació final del vector d'estat (Eq. 4.6) i entra en joc la K de Kalman.

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \delta z_k^- \quad \textit{(Eq. 4.6)}$$

On:

$\hat{x}_k^+$: vector d'estat resultant després d'aplicar el filtre de Kalman.

δz_k^-: és la diferència entre la dada del GPS i la del INS. És aquesta matriu (Eq. 4.7).

$$(GPSr_{eb}^e - r_{eb}^e) \quad \textit{(Eq. 4.7)}$$

Ja només falta realitzar l'últim pas del filtre de Kalman, que és calcular la variància de cada un de les variables del vector d'estat que acabem d'estimar. Per fer-ho apliquem aquesta última equació (Eq. 4.8).

$$P_k^+ = (I - K_k \cdot H_k) \cdot P_k^- \qquad \textit{(Eq. 4.8)}$$

On:

P_k^+: és la matriu d'error de covariància resultant després d'aplicar un cicle del filtre de Kalman.

Un cop hem aplicat tots aquest passos ja tenim l'estimació final del vector d'estat. Recordem que les variables del vector d'estat són estimacions de l'error d'orientació, de velocitat i de posició i l'estimació dels biaixos dels acceleròmetres i giroscopis.

4.2.1.3. Integració loosely coupled

En aquest apartat explicarem com el INS queda integrat amb el GPS mitjançant el filtre de Kalman. La forma de integració utilitzada rep el nom de loosely coupled. Segueix l'esquema (figura 4.2) següent.

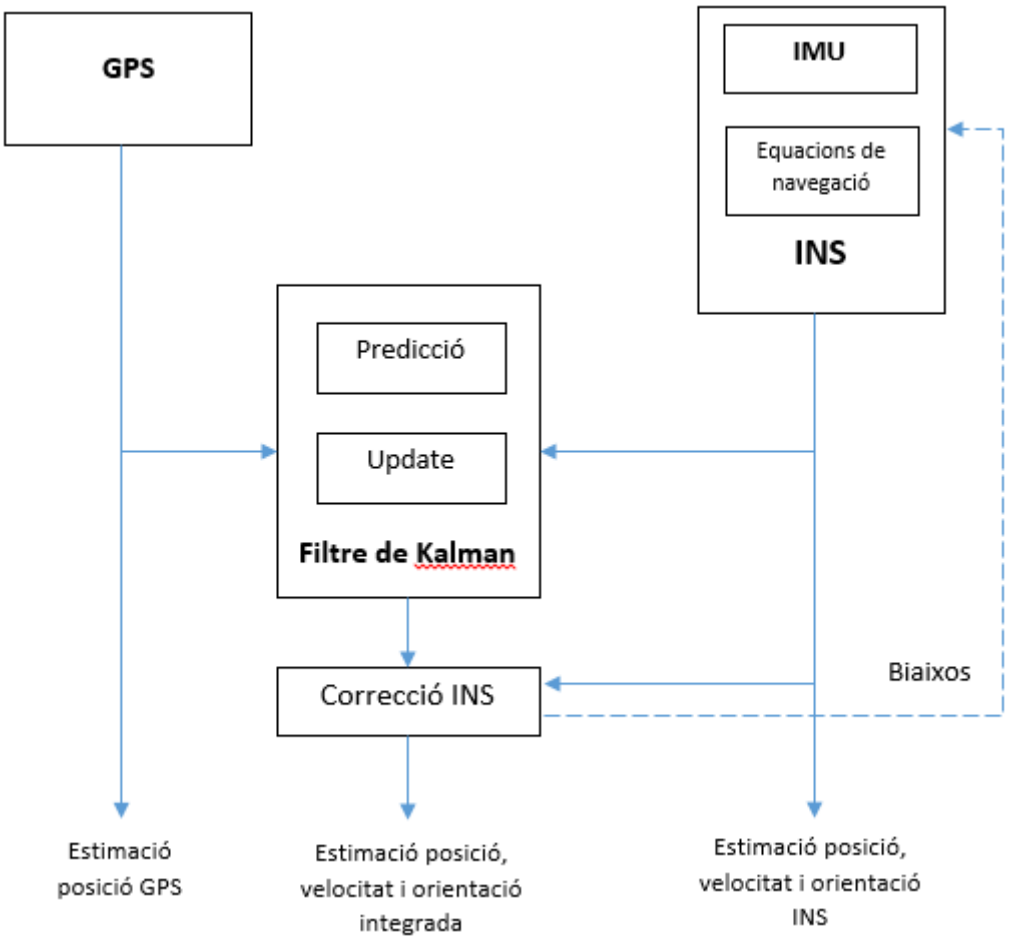


Figura 4.2: Esquema de la integració loosely coupled utilitzat en aquest treball

Com podem veure en la imatge anterior, per una banda tenim el GPS proporcionant dades de la posició i per l'altre el INS proporcionant estimacions de la posició, velocitat i orientació. Integrem la informació que ens proporcionen els dos sistemes de navegació (INS i GPS) mitjançant el filtre de Kalman que acabem d'explicar. Gràcies al filtre de Kalman obtenim una estimació dels errors en posició, velocitat i orientació, a més també obtenim una estimació del biaix dels acceleròmetres i giroscopis. En l'etapa de correcció INS utilitzem les estimacions dels errors fetes pel filtre de Kalman i la restem de l'estimació de posició, velocitat i orientació fetes pel INS. D'aquesta manera obtenim l'estimació de posició, velocitat i orientació integrada. A més, podem utilitzar l'estimació dels biaixos feta pel filtre de Kalman perquè, en la següent iteració, restar els biaixos de la IMU per així ja poder aconseguir unes millors estimacions a partir del INS. Aquesta realimentació per restar els

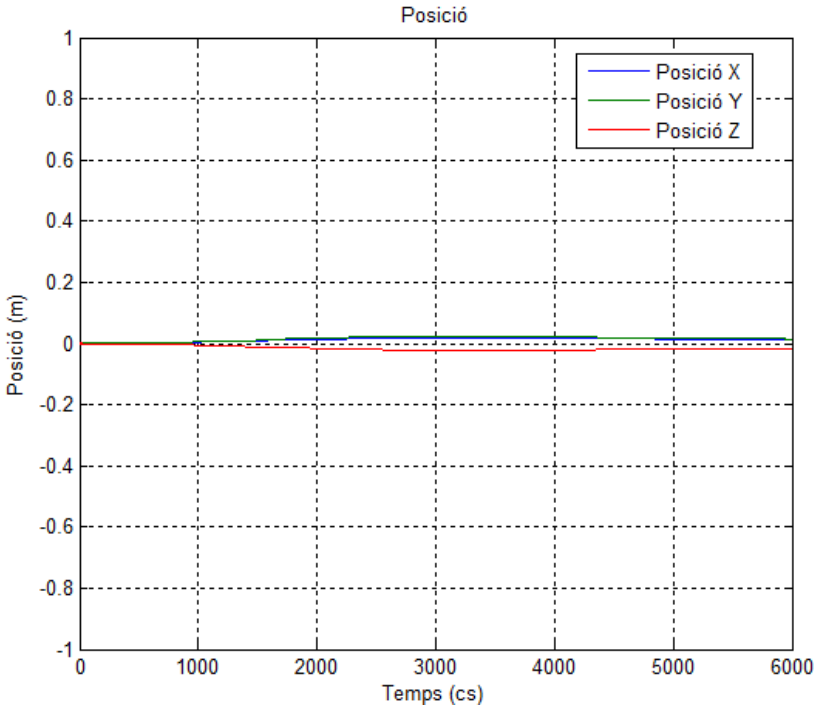
errors estimats del sistema de navegació és opcional, però en sensors que deriven molt com el nostre és recomanable.

4.2.2. Proves preliminar de funcionament del filtre de Kalman

Per comprovar el correcte funcionament del filtre de Kalman es va realitzar una petita prova. Com que en el moment de fer la prova no disposàvem d'un sistema integrat que ens permetés obtenir dades de GPS i només teníem la IMU per separat vam decidir fer una prova simple. Aquesta prova consistia en deixar el sensor quiet (d'aquesta manera podem conèixer amb exactitud la posició, velocitat i orientació del sensor durant l'experiment i comparar-ho amb la solució obtinguda) i simular un sensor GPS que ens doni mesures de la posició i la velocitat del sensor. Com que sabíem que el sensor no es movia el GPS sempre donava la mateixa posició i una velocitat zero. L'objectiu era estimar els errors, però també estimar el bias que ens ajudarà a compensar els errors de les dades de la IMU.
Detalls importants de l'experiment:

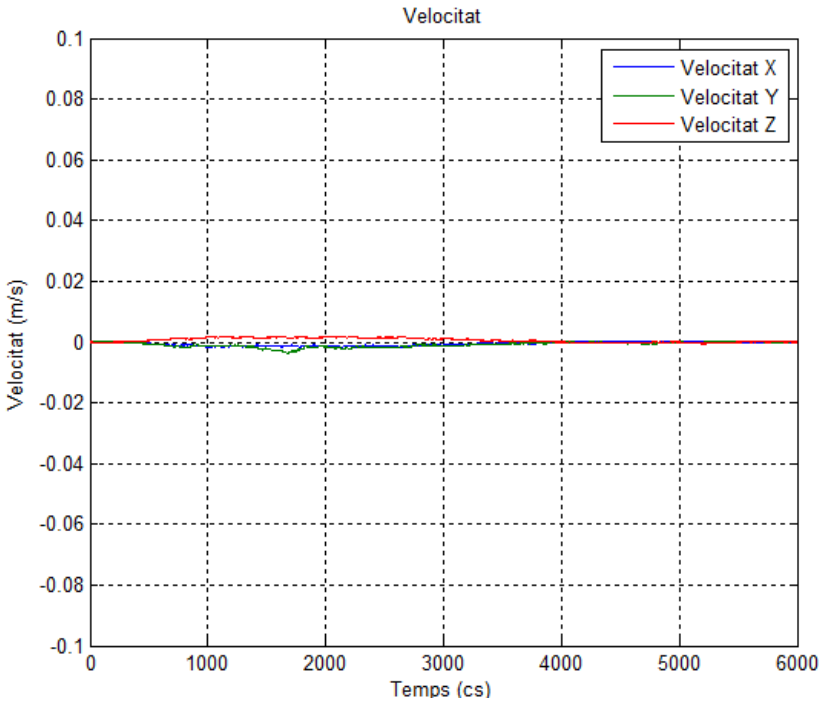
- Sensor: ADIS16488
- Durada: 1 minut
- Sistema de referència utilitzat: ECEF

A continuació anem a veure les estimacions obtingudes de posició, velocitat, orientació i biaixos després de fer aquest petit experiment amb el filtre de Kalman. Comencem per la posició.



Gràfica 4.5: Estimació de la posició mitjançant un Filtre de Kalman

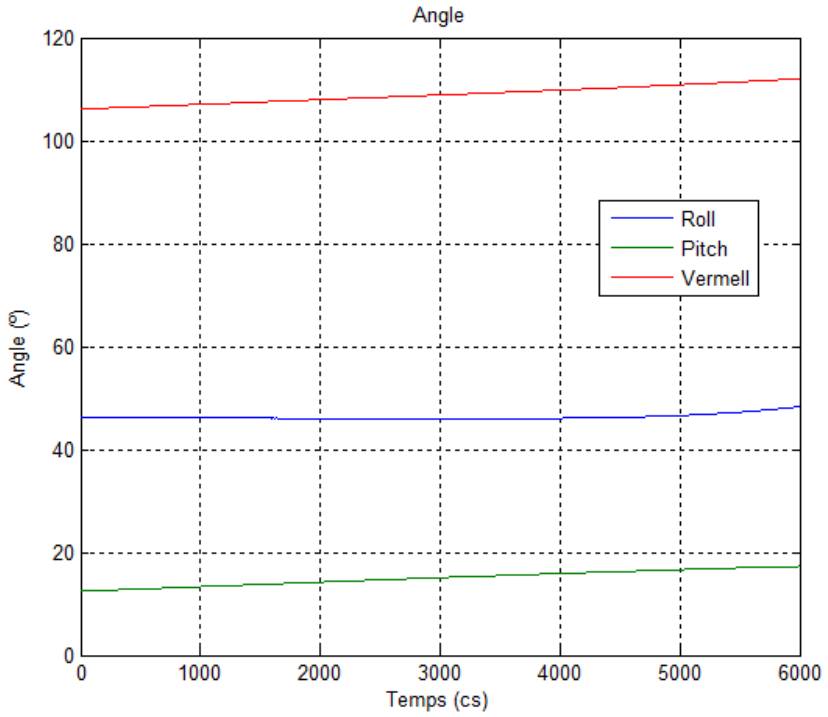
La posició s'estima molt bé, el seu desplaçament és pràcticament zero. Que el valor de la posició sigui tant bo és perquè la seva estimació prové d'una observació directa com és el GPS. Anem a veure l'estimació de la velocitat.



Gràfica 4.6: Estimació de la velocitat amb el filtre de Kalman

Amb la velocitat obtenim els mateixos resultats que amb la posició, el seu valor és gairebé zero. El motiu d'una estimació tant bona torna a ser que les dades de velocitat provenen GPS.

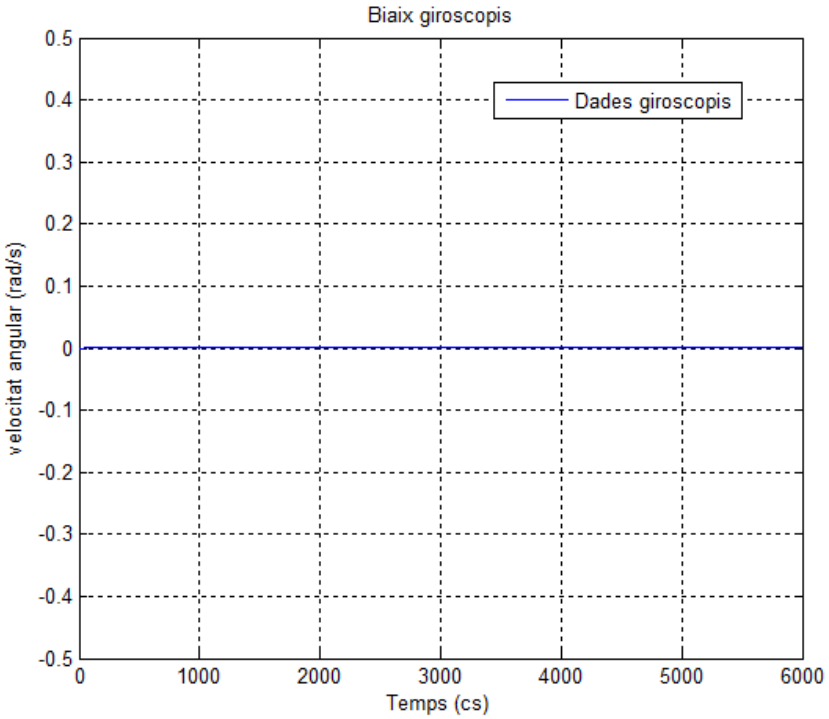
Seguidament, observem com han evolucionat la orientació (gràfica 4.7), els angles de Roll, Pitch i Yaw.



Gràfica 4.7: Estimació del filtre de Kalman

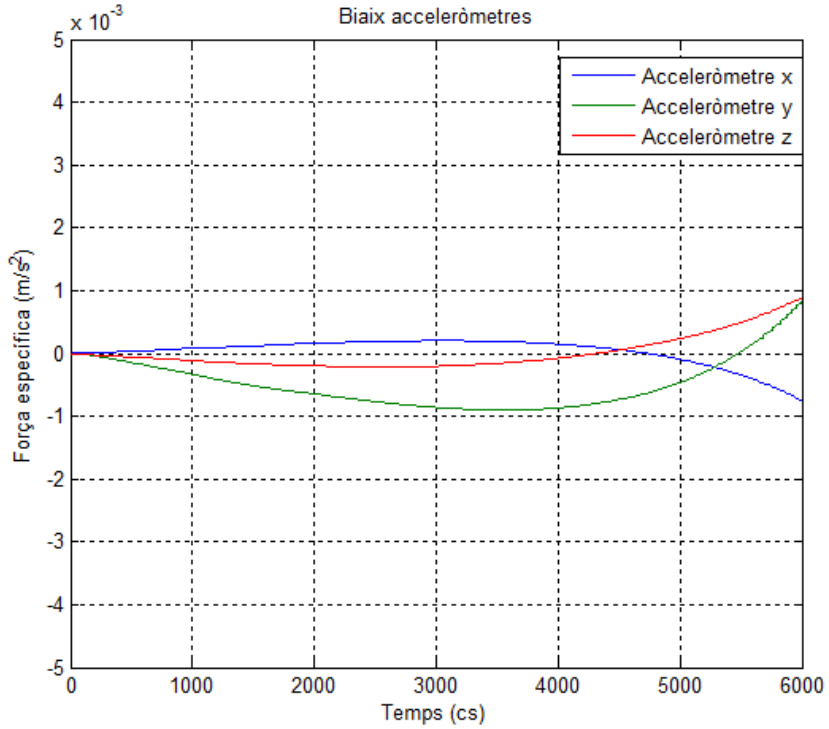
Recalcar que els angles que es mostren aquí estant representats en ECEF.

S'observa una variació al llarg del temps dels angles. Això no hauria de ser així ja que el sensor ha estat quiet durant tot l'experiment. El fet que hi hagi una variació en els angles pot ser molt perillós, ja que com s'ha explicat en el capítol 4.1 d'aquest treball els errors en l'estimació de la posició i velocitat venen provocats en gran part per errors en l'estimació de l'orientació. Si els biaixos s'han estimat bé no s'hauria de veure cap variació de la orientació. Comprovem si els biaixos s'han estimat correctament.



Gràfica 4.8: Estimació del biaix dels giroscopis mitjançant el filtre de Kalman

Els biaixos dels giroscopis no s'ha calculat bé, ja que ens diu que tots són zero i això no és així. Observem que passa amb els biaixos dels acceleròmetres (gràfica 4.9).



Gràfica 4.9: Estimació del biaix dels acceleròmetres amb el filtre de Kalman

A diferència dels giroscopis, el biaix dels acceleròmetres no és zero. Però els seus resultats no són bons ja que l'estimació no s'estabilitza en cap valor.

Després de fer aquestes proves preliminars arribem a la conclusió que hi ha alguna cosa en el filtre que no acaba de funcionar correctament. El problema és que la informació que ens diu el GPS (posició i velocitat) no ens diu res sobre la orientació. Per tant, quan la IMU està quieta, qualsevol orientació és compatible amb les mesures que obtenim. Bàsicament, el filtre no pot determinar l'orientació del sensor a partir de mesures de velocitat i posició si aquest no s'esta desplaçant. Aquesta falta d'observabilitat del problema permet a l'angle derivar i llavors, les estimacions del biaix prenen valors erronis en un intent de compensar els errors provinents de l'angle.

Cal esmentar que si el vehicle es mou, els errors que succeeixen en els angles tenen un efecte directe en la trajectòria. En aquest cas, les observacions del GPS si que ens permetrien estimar correctament els angles.

Per solucionar el problema patit en les proves preliminars hem utilitzat una tècnica anomenada Zero Update, la qual s'explicarà detalladament en el següent capítol.

4.2.3. Zero Updates

Hem tingut aquest problema d'observabilitat, necessitem més informació, un altre tipus de mesura, per poder obtenir una bona estimació del biaixos i de tot l'estat en general. Aquí és on entra en joc el que s'anomena un Zero Update. El Zero Update és com un sensor virtual basat en heurística (coneixement del sistema).

Hi ha moltes situacions on sabem o podem percebre que una INS roman immòbil (un cotxe aparcat o un avió parat, un robot abans d'executar una missió, un míssil abans de ser llançat...). Podem aprofitar el coneixement d'aquestes situacions per entrar updates en el filtre de Kalman on diguem que les velocitats tant lineals com angulars són iguals a zero. Això és coneix com a Zero Update.

Afegir Zero Updates en el filtre de Kalman comporta petits canvis que comentarem a continuació.

Per afegir un Zero Update la component δz_k^- variarà una mica i quedarà així (Eq. 4.9):

$$\begin{pmatrix} GPSr_{eb}^{e} - r_{eb}^{e} \\ GPSv_{eb}^{e} - v_{eb}^{e} \\ 0 - w_{ib}^{b} \end{pmatrix} \quad \textit{(Eq. 4.9)}$$

Els dos elements superiors de l'equació calculen la diferència dels valors de posició i velocitat del GPS amb l'estimació de posició i velocitat del INS. L'element inferior és el Zero Update de velocitat angular que introduïm. El zero representa el coneixement que tenim sobre el fet que el sensor esta quiet. Llavors l'error vol estimar el filtre ve donat per la diferència amb el valor de velocitat angular que rebem del INS.

Si el nostre GPS només ens donés lectures de posició (com és el cas en les proves amb el cotxe que veurem més endavant) llavors també podríem introduir Zero Updates de velocitat lineal. La component δz_k^- quedaria de la següent manera (Eq. 4.10):

$$\begin{pmatrix} GPSr_{eb}^{e} - r_{eb}^{e} \\ 0 - v_{eb}^{e} \\ 0 - w_{ib}^{b} \end{pmatrix} \quad \textit{(Eq. 4.10)}$$

La introducció dels Zero Updates també provocarà canvis en la matriu H (Eq. 11) i en la matriu R (Eq. 12). La matriu H del cas en que el GPS només dona mesures de posició quedarà de la següent forma:

$$H = \begin{pmatrix} 0_3 & 0_3 & -I_3 & 0_3 & 0_3 \\ 0_3 & -I_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & -I_3 \end{pmatrix} \quad \textit{(Eq. 4.11)}$$

La matriu R tindrà els següents canvis:

$$R = \begin{pmatrix} I_3 \cdot Gp & 0_3 & 0_3 \\ 0_3 & I_3 \cdot \sigma_v^2 & 0_3 \\ 0_3 & 0_3 & I_3 \cdot \sigma_w^2 \end{pmatrix} \quad \textit{(Eq. 4.12)}$$

On:

σ_v^2: la variància de la mesura de Zero Update de velocitat lineal.

σ_w^2: la variància de la mesura de Zero Update de velocitat angular.

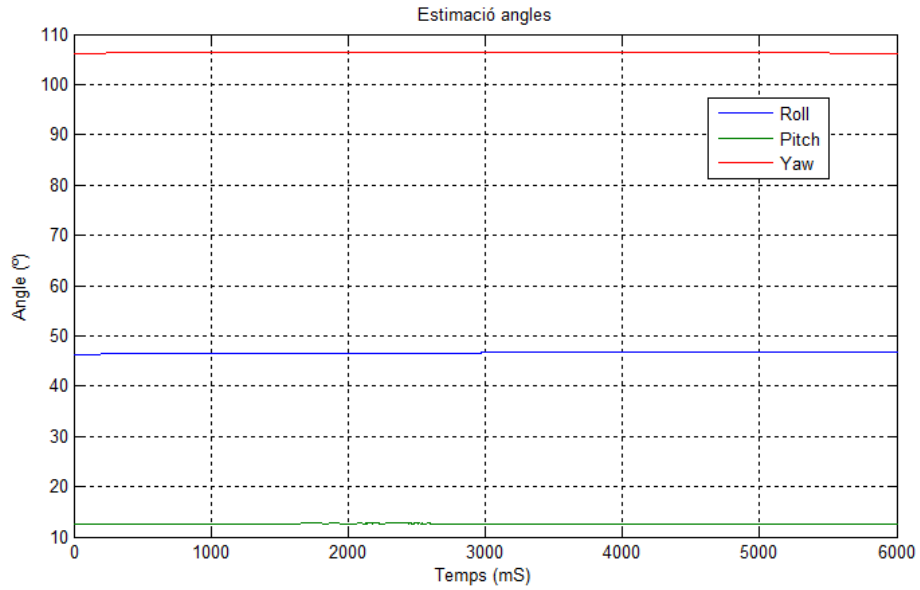
Gp: la variància de la mesura del GPS.

Aquestes variàncies s'han de fixar d'acord amb la incertesa que tenim sobre la mesura de velocitats (angular i lineal) zero. Per exemple, un cotxe tot i estar aturat pot patir vibracions. Per tant, a les mesures de Zero Updates també se'ls hi ha d'afegir una incertesa. Per calcular la σ_v^2 i la σ_w^2 s'ha agafat la variància de les dades dels acceleròmetres i giroscopis respectivament quan la IMU està quieta.

D'aquesta manera solucionem el problema anterior d'observabilitat per els casos en que la INS no es desplaça. Repetim les proves preliminars anteriors per confirmar que efectivament el problema estar resolt.

Tant l'estimació de la posició com la de la velocitat dona valors pràcticament iguals als obtinguts sense aplicar Zero Updates, per tant, són correctes.

Tot seguit, anem a veure si amb la introducció dels Zero Updates els angles s'estimen bé. Recordem que abans no era així.

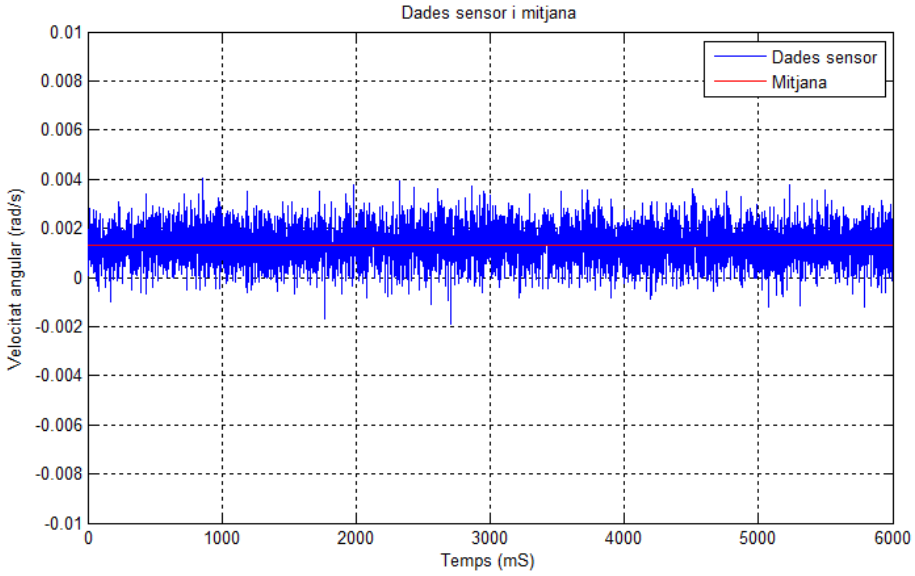


Gràfica 4.10: Estimació dels angles amb la millora del filtre de Kalman

Efectivament, la variació (continua havent una mica de deriva) dels angles és molt petita i es mantenen estables. Seguidament, comprovem si s'ha estimat bé el biaix dels giroscopis.

Per comprovar si el filtre de Kalman estima correctament el biaix el que hem fet és el següent. Després de recollir les dades amb la IMU quan aquesta estava totalment quieta n'hem estimat el biaix fent la mitjana de les dades que han donat cadascun dels seus sensors

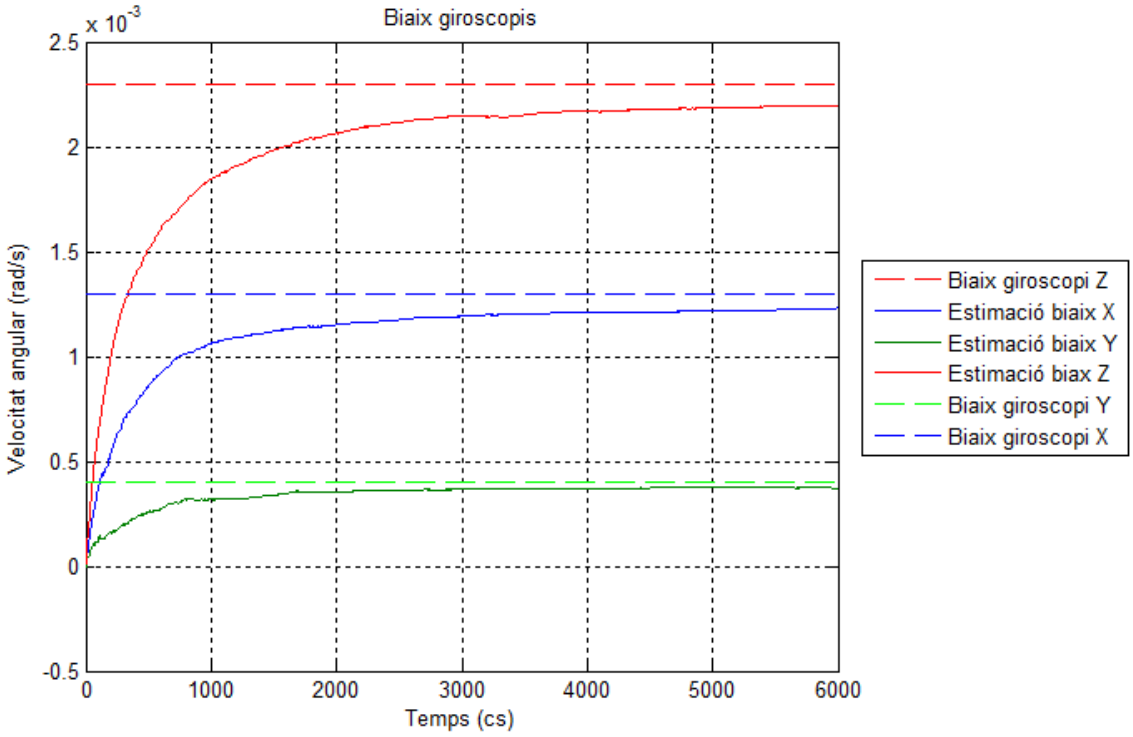
(gràfica 4.11). Si no hi hagués cap tipus de biaix, aquesta mitjana hauria de donar zero. El valor que doni serà el biaix que té el sensor. Això és possible perquè com que l'experiment és curt i fet en condicions ambientals estables, podem considerar que el biaix és constant durant tota la prova i per tant el podem obtenir fent la mitjana tal i com hem explicat. És important remarcar que el biaix pot canviar durant períodes llargs de temps, a més és susceptible a alteracions a causa de les condicions ambientals, pertorbacions d'altres equipaments... Per això és molt interessant poder-lo estimar en línia mitjançant un filtre de Kalman.



Gràfica 4.11: Dades d'un giroscopi amb soroll i la seva mitjana

Giroscopi	**X**	**Y**	**Z**
Biaix (rad/s)	0.0013	0.0004027	0.0023

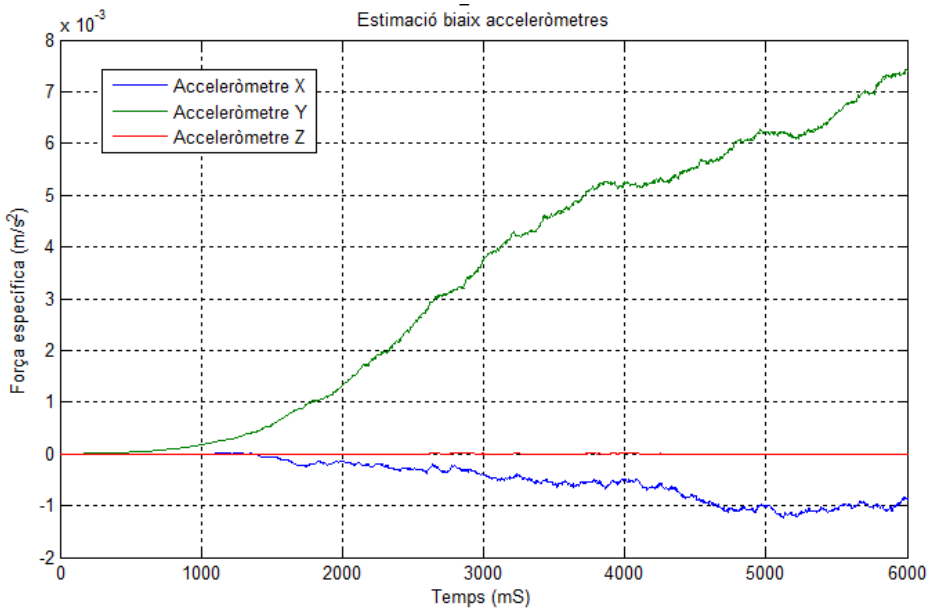
Taula 4.1: Biaix dels giroscopis



Gràfica 4.12: Comparació del biaix estimat amb el filtre de Kalman amb el biaix real

Podem comprovar com l'estimació del biaix dels giroscopis amb el filtre de Kalman és prou bona.

No podem comentar el mateix del biaix dels acceleròmetres que continua sense estimar-se correctament.



Gràfica 4.13: Estimació del biaix dels acceleròmetres amb el filtre de Kalman amb Zero Updates

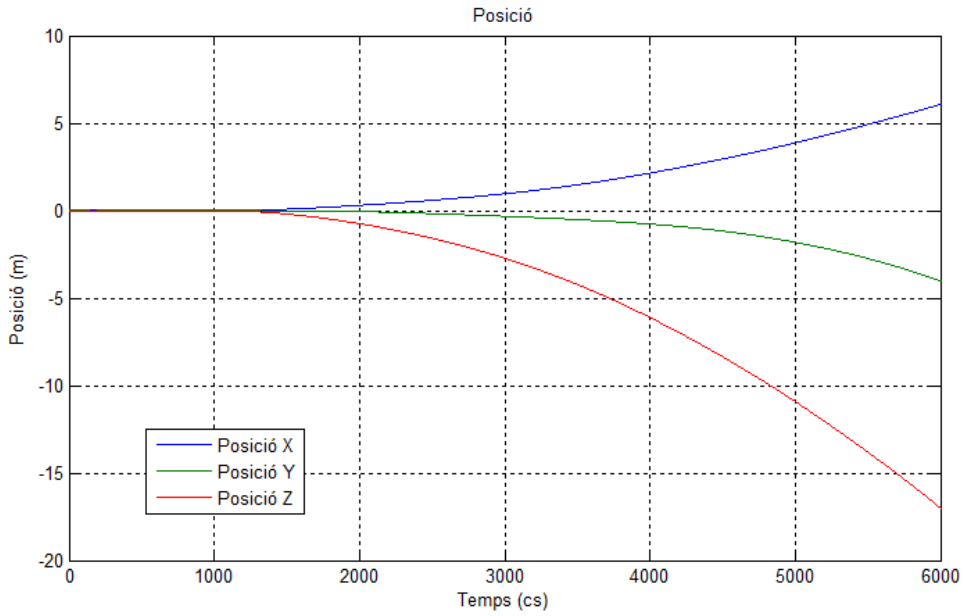
El motiu de que no s'estimi correctament el biaix dels acceleròmetres és el següent. Com hem vist els angles s'estimen bé, però tot i així pateixen una petita deriva. Aquesta deriva provoca que la gravetat no quedi ben compensada. Això fa que hi hagi acceleracions ocasionades perquè no s'ha compensat bé la gravetat. El filtre de Kalman creu que aquestes acceleracions són degudes a un biaix i per això no l'estima bé.

Tot seguit mostrarem com varien els resultats d'aquest experiment si fem un petit canvi. L'experiment ha consistit en veure com s'estima la posició, velocitat, orientació i biaixos gràcies a l'ajuda del filtre de Kalman. A més el filtre de Kalman s'aplicava durant tot el temps que durava la prova (1 minut). Ara farem updates amb el filtre de Kalman durant els instants inicials i passat aquest temps inicial deixarem de fer updates. El motiu és perquè volem observar quin rendiment obtindríem amb un INS que operi sense ajuda externa, però amb una estimació raonable del biaix dels seus sensors.

Aquesta variant de l'experiment anterior tindrà aquests detalls importants:

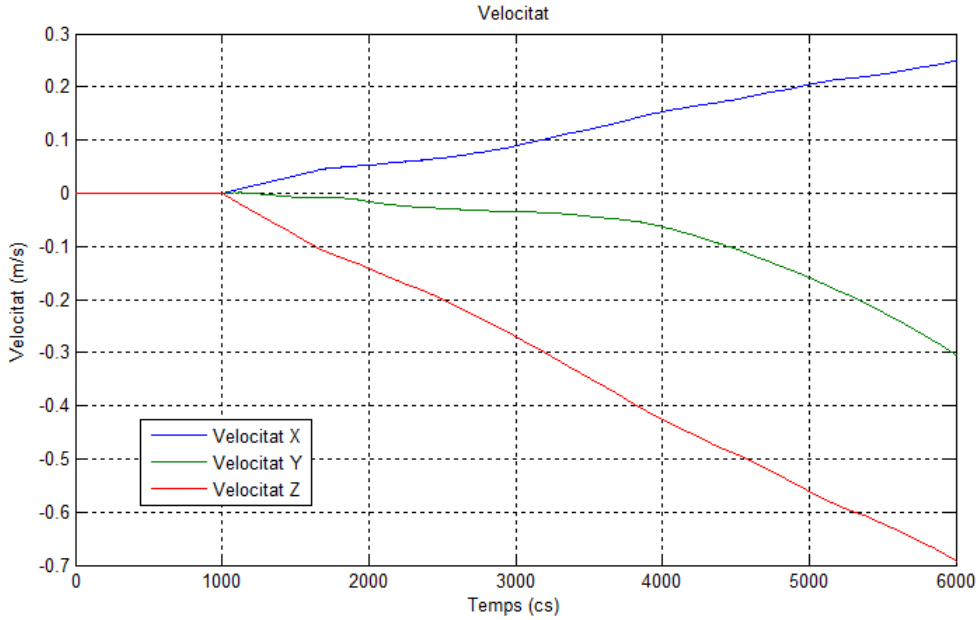
- Sensor: ADIS16488
- Durada: 1 minut
- Durada dels updates del filtre de Kalman: 10 segons
- Sistema de referència utilitzat: ECEF

Comencem per l'estimació de la posició.



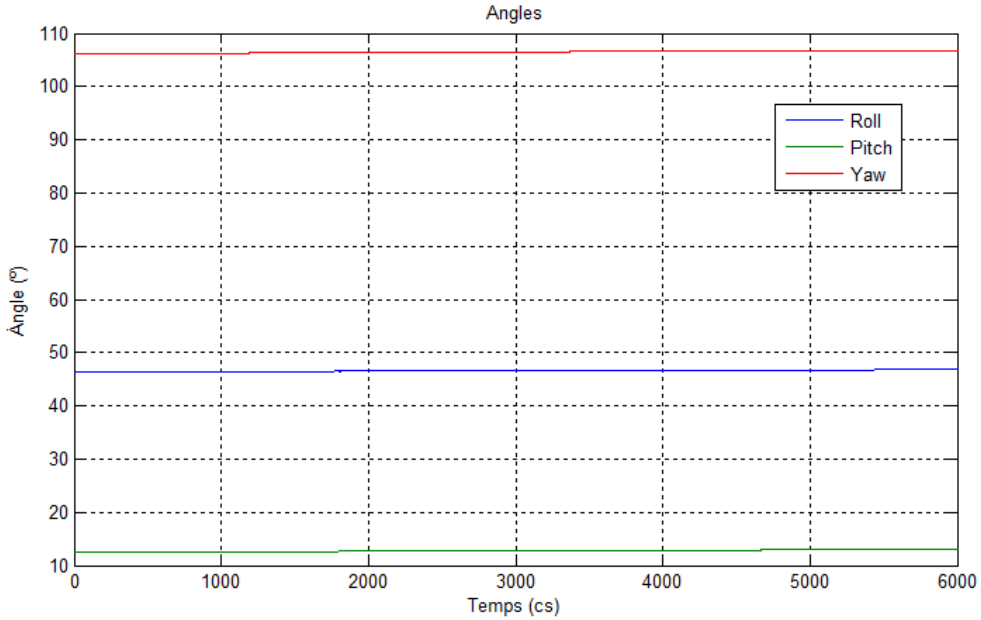
Gràfica 4.14: Estimació de la posició amb updates els 10 primers segons

Podem observar clarament com fins el moment que hi ha updates del Filtre la posició és correcte, a partir de que deixen d'haver updates la posició comença a derivar. A més, també s'observa com a mesura que passa el temps la deriva es fa més i més gran. A continuació podem veure com amb la velocitat passa el mateix.



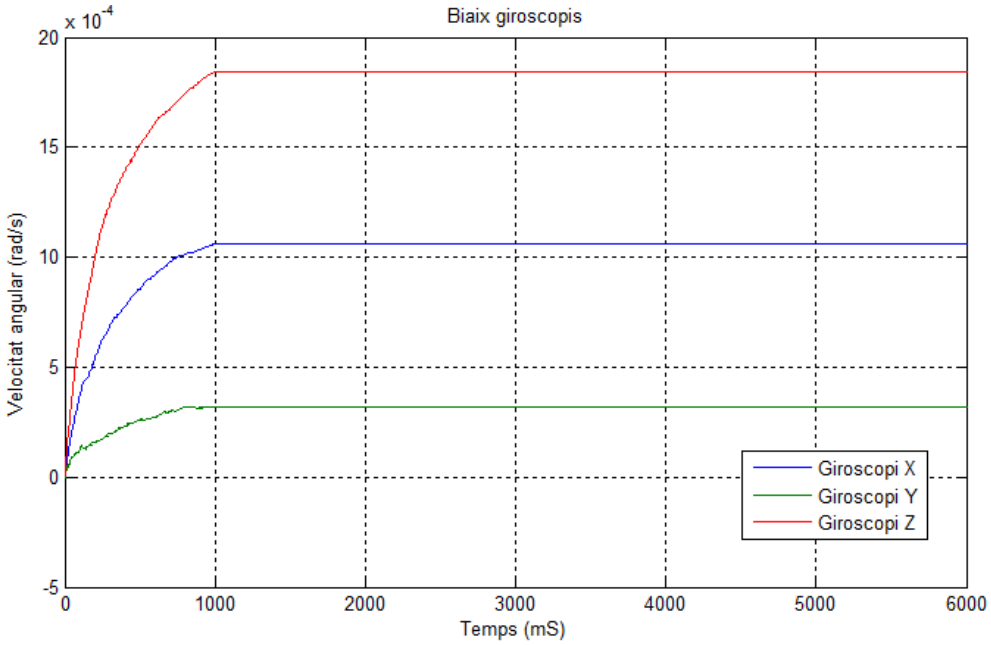
Gràfica 4.15: Estimació de la velocitat amb updates durant els 10 primers segons

Seguidament, observem l'evolució dels angles.



Gràfica 4.16: Estimació dels angles amb updates durant els 10 primers segons

Pateixen una lleugera deriva. Tot i així, la seva estimació és molt correcte.

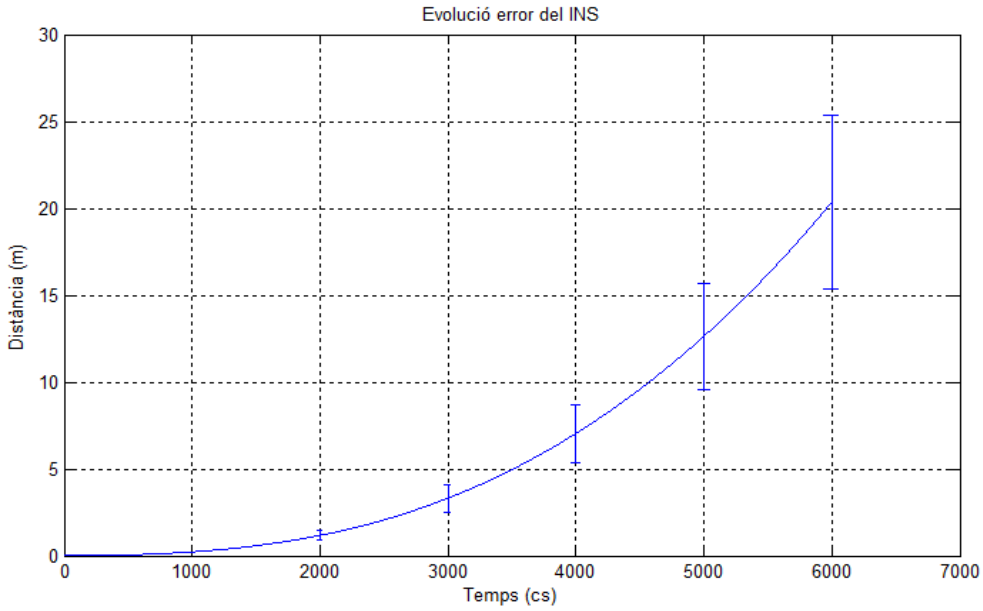


Gràfica 4.17: Estimació del biaix dels giroscopis amb updates durant els 10 primers segons

Veiem com el filtre estava estimant correctament el biaix dels giroscopis, però li ha falta temps per poder-lo acabar d'estimar més acuradament. D'aquí prové la deriva en la posició i la velocitat.

El biaix dels acceleròmetres segueix sense poder-se estimar bé del tot.

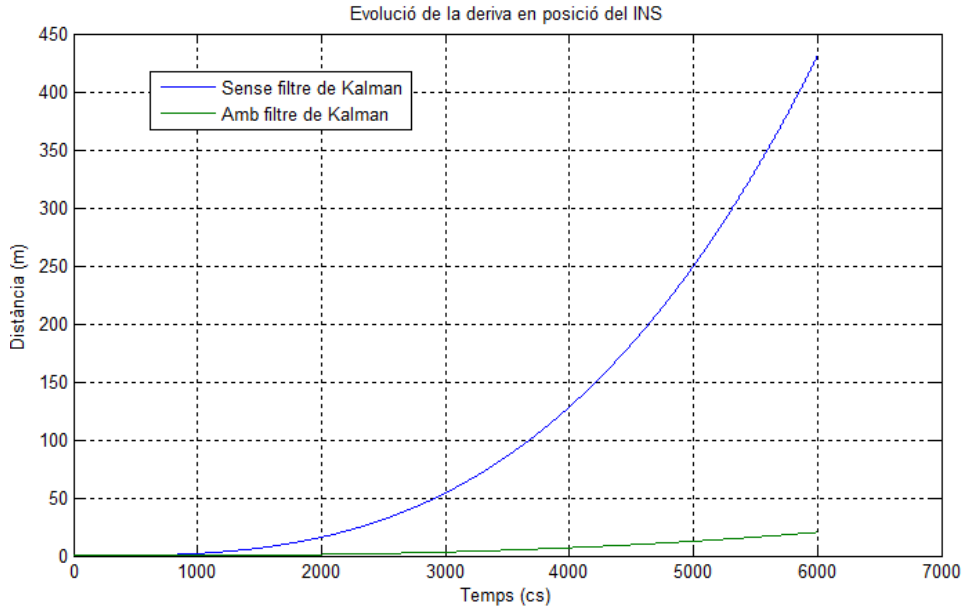
Finalment, i ja per acabar les proves preliminars del filtre de Kalman mostrarem els últims resultats. Deixant la IMU quieta altre vegada. Hem agafat les seves dades i hem realitzar updates de posició i Zero Updates de velocitat i velocitat angular durant uns 10 segons, tot seguit, no s'han fet més updates amb el filtre i el INS ha operat sense cap ajuda externa durant un minut. Hem repetit aquesta prova 10 cops.



Gràfica 4.18: Deriva en l'estimació de la posició amb errobars que mostren la variància

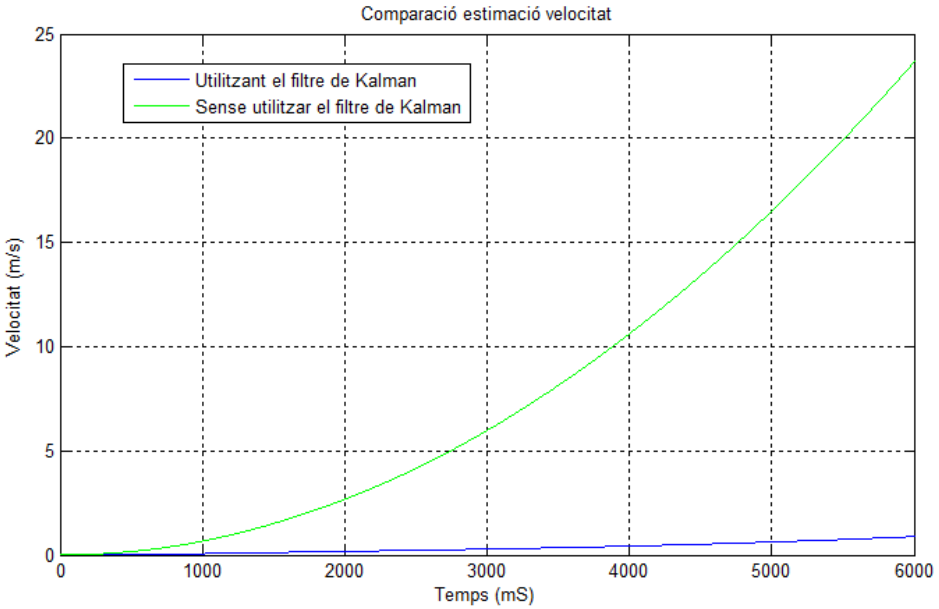
En aquesta gràfica (gràfica 4.18) podem observar com deriva la posició de la INS. Degut a que s'ha realitzat l'experiment 10 vegades mostrem la mitjana de la deriva i unes barres que representen la seva desviació estàndard.

Si els comparem amb l'estimació l'error que obteníem quan no es corregir cap tipus de biaix (capítol 3.6.5) es podrà apreciar millor la millora i l'afecte positiu del filtre de Kalman. Veiem que, tot i que la INS navega sense ajuda externa, com que hem estimat prèviament els biaixos mitjançant el filtre de Kalman els resultats milloren molt.



Gràfica 4.19: Comparació de de la deriva en la posició

Amb aquesta gràfica (gràfica 4.19) queda corroborat que un filtre de Kalman ens ajuda molt alhora de millorar els resultats d'un sistema de navegació inercial. El mateix ens succeeix amb la velocitat.



Gràfica 4.20: Comparació de la deriva en la velocitat

Tos els passos explicats anteriorment es troben en els arxius Matlab "filtreKalman_I_Prediccio.m", "filtreKalman_I_MeasurementUpdate_GPS", "filtreKalman_I_MeasurementUpdate_GPS_ZU" i

"filtreKalman_I_MeasurementUpdate_ZU". Així com la integració del filtre de Kalman amb el INS es troba a l'arxiu "programaECEF.m". Aquestes funcions també es podran trobar a l'annex C, secció C.7.

4.2.4. Resultats de l'aplicació filtre de Kalman en el INS

Un cop acabades les proves preliminars i havent comprovat que el filtre de Kalman funciona correctament s'han realitzat els experiments finals per veure com funciona el INS. S'ha integrat el sistema de navegació inercial amb un GPS real mitjançant el filtre de Kalman tal i com s'ha explicat anteriorment (capítol 4.2.1). Un cop fet això s'han realitzat una sèrie de recorreguts amb cotxe i hem comprovat si el INS estimava bé els valors de posició velocitat, orientació i biaix. D'aquesta manera hem examinat la resposta del INS en moviment.
S'han fet tres recorreguts diferents amb el cotxe. Tots tres sortint i acabant al mateix lloc (al CIRS del Parc Tecnològic de la UdG).

Comentar, que en el moment de realitzar els recorreguts i prendre les dades, primer de tot hem pres dades amb la IMU durant 10 segons amb el vehicle parat. Un cop passats aquests 10 segons hem engegat el vehicle i hem fet els recorreguts. Aquests segons inicials són molt útils per fer una primera estimació dels biaixos dels sensors.

El primer ha consistit en fer tres voltes en un carrer que dona la volta a l'edifici del CIRS. L'experiment a durat 2 minuts. A la següent imatge (imatge 4.1) es pot veure el recorregut en vermell.

Imatge 4.1: Recorregut al voltant del CIRS

El segon recorregut a consistit en fer dues voltes a tot el parc tecnològic. La durada de la prova també és de 2 minuts. A la imatge 4.2 podem veure el recorregut marcar en vermell.

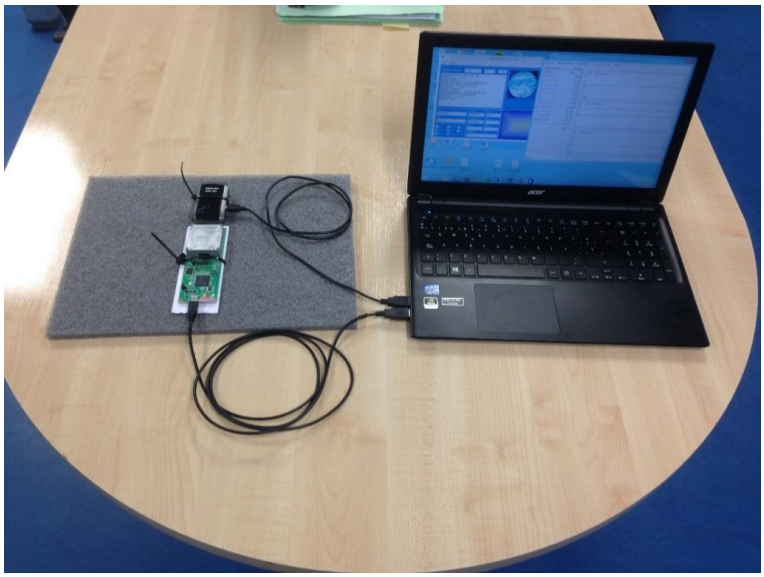
Imatge 4.2: Recorregut al voltant del Parc Tecnològic

Finalment, la última prova i la més llarga a consistit amb fer una volta a tot el campus de Montilivi de la UdG. Sortint i arribant al CIRS. L'experiment ha tingut una durada de 4 minuts.

Imatge 4.3: Volta al campus de Montilivi

El muntatge realitzat per poder prendre les dades, tant de la IMU com el GPS, d'aquests recorreguts és el que es mostra a la següent imatge. On es veu la ADIS16488 amb la seva tarja d'adquisició de dades i just a sobre el dispositiu GPS. Ambdós connectats a l'ordinador mitjançant USB.

És important remarcar que les dades del GPS ja venen marcades amb una dada de temps sincronitzada amb el temps UTC. En canvi, el software de captura de dades de la INS agafa el temps del rellotge del PC. Per això abans de començar l'experiment s'ha sincronitzat el PC amb un servidor NTP online que l'ha sincronitzar amb el temps UTC. Tot i això, el software de la INS només guarda el temps d'inici de cada experiment amb resolució d'un segon i per tant, podem esperar un error de sincronia entre les dades d'aproximadament +/- 1 segon.



Imatge 4.4: Muntatge de GPS i IMU per recollir dades

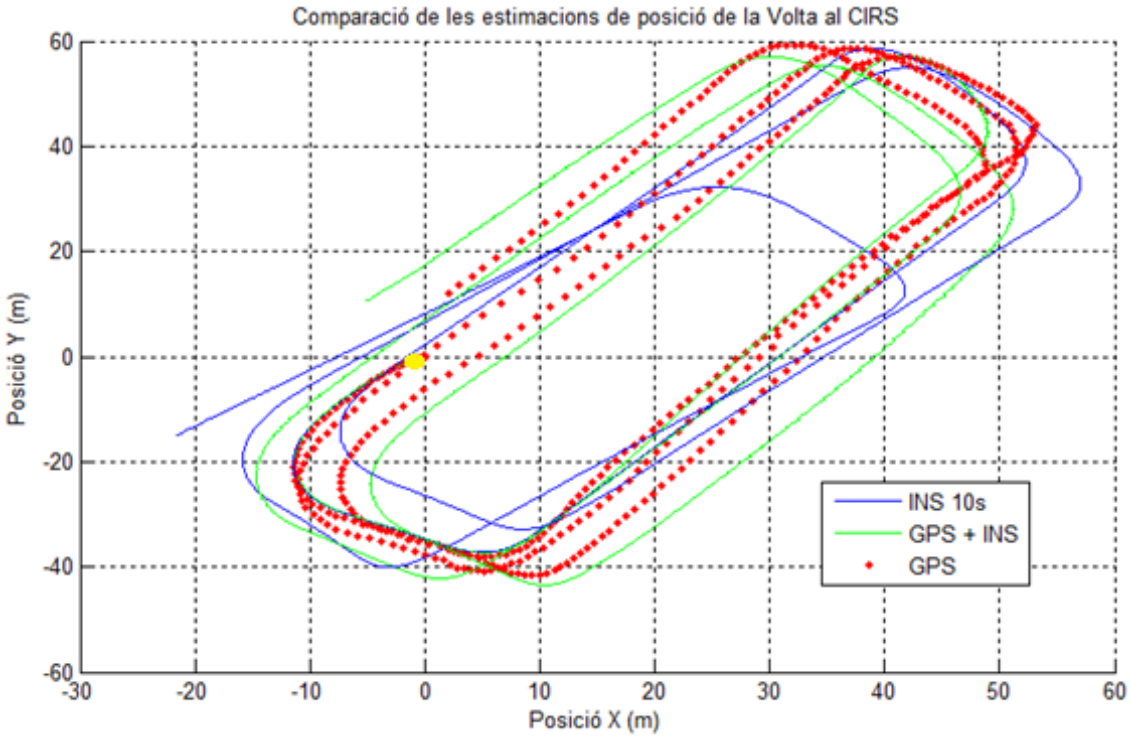
Acabem de veure els recorreguts que s'han realitzat i el muntatge necessari de sensors per poder prendre les dades i posteriorment realitzar els càlculs. Tot seguit comencem a observar els primers resultats.

4.2.4.1. Comparativa estimacions de la posició

Primer de tot mostrarem una comparativa entre les diferents formes en que hem estimat la posició:

- Dades del GPS: es mostra el recorregut que es fa segons les dades que rebem del GPS.
- INS amb Updates de GPS: s'estima el recorregut utilitzant les dades de la IMU, però amb Updates de posició cada vegada que el GPS en envia una dada (cada 0.2 segons).
- INS amb 10 segons de Updates: durant els 10 segons inicials en que estem parats introduïm Zero Updates de velocitat lineal i angular i Updates de GPS. Un cop passats aquests 10 segons, el vehicle es comença a moure i el INS ja no rep cap ajuda externa.

Tot seguit mostrem la comparativa (gràfica 4.21) de les estimacions de posició en el recorregut que hem fet tres voltes al CIRS. El punt groc marca l'inici del recorregut.



Gràfica 4.21: Estimacions de la posició de les tres voltes al CIRS

D'aquesta gràfica 4.21 es pot extreure força informació. Primer de tot, que el GPS utilitzat no és de gaire bona qualitat. A més, al ser una unitat portàtil, té una antena petita que té problemes alhora de rebre senyals en entorns complexes, com per exemple, quan està envoltada d'edificis. Fixem-nos que tot i que fem tres voltes i passem pel mateix lloc diverses vegades, el GPS mostra diferències de fins a 4 o 5 metres entre els recorreguts seguits a cada una de les voltes.

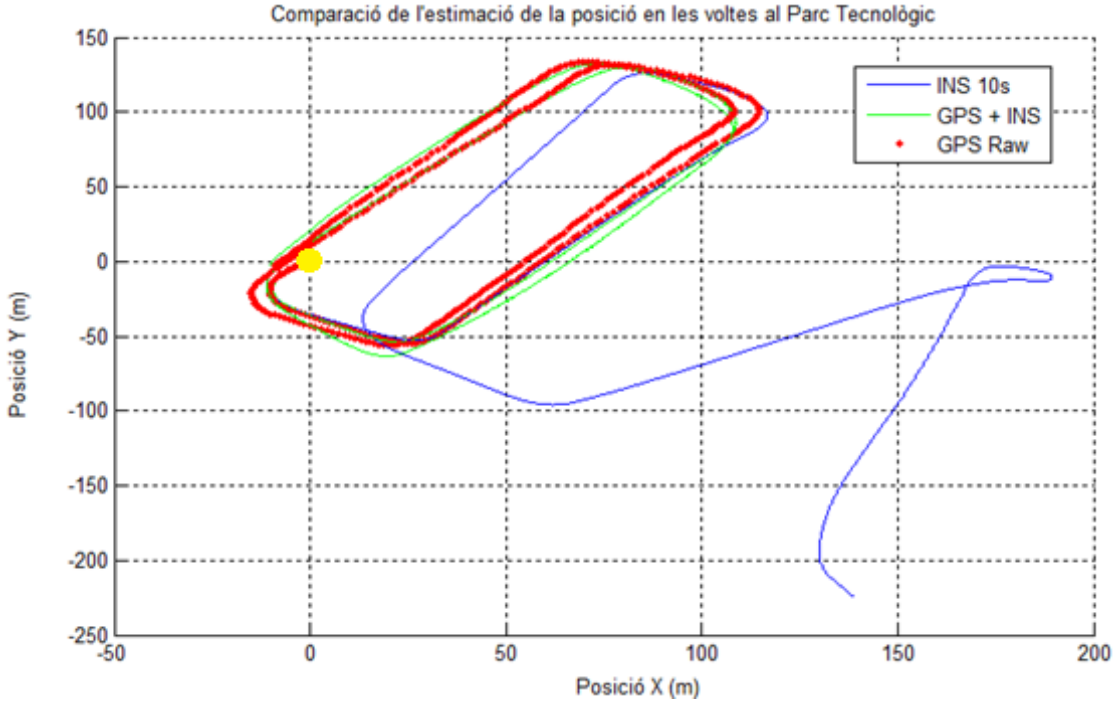
També podem observar la diferència de freqüència en que treballa una IMU i un GPS (el nostre a 5Hz). Cada punt vermell representa una dada de posició que dona el GPS, en canvi les estimacions fetes amb el INS mostren un traçat continu. Aquesta és una de les utilitats de treballar amb una INS, entre les dades rebudes del GPS podem continuar tenint una estimació de la posició gràcies al sistema de navegació inercial ja que treballa a una freqüència molt més alta.

Entre les dades del GPS (vermell) i les dades del INS amb Updates del GPS (verd) veiem una diferència important que teòricament no hauria de ser tant gran. Això ho podem atribuir a múltiples factors, o a una combinació d'ells. Per exemple, podem tenir un petit problema de sincronia entre les dades, o potser els errors presents tant al GPS com a la INS són

inconsistents i dificulten la convergència del filtre. Això fa que l'estimació final no sigui correcte del tot.

Finalment, és important recalcar que si ens fixem amb el traçat blau, el INS treballant sense ajudes externes, podem veure com la primera volta de les tres realitzades té una trajectòria força semblant a la del GPS, i això és bona senyal. L'estimació de posició coincideix gairebé tota la primera volta amb les dades del GPS (vermell). Un cop acabada la primera volta l'estimació de la posició comença a derivar.

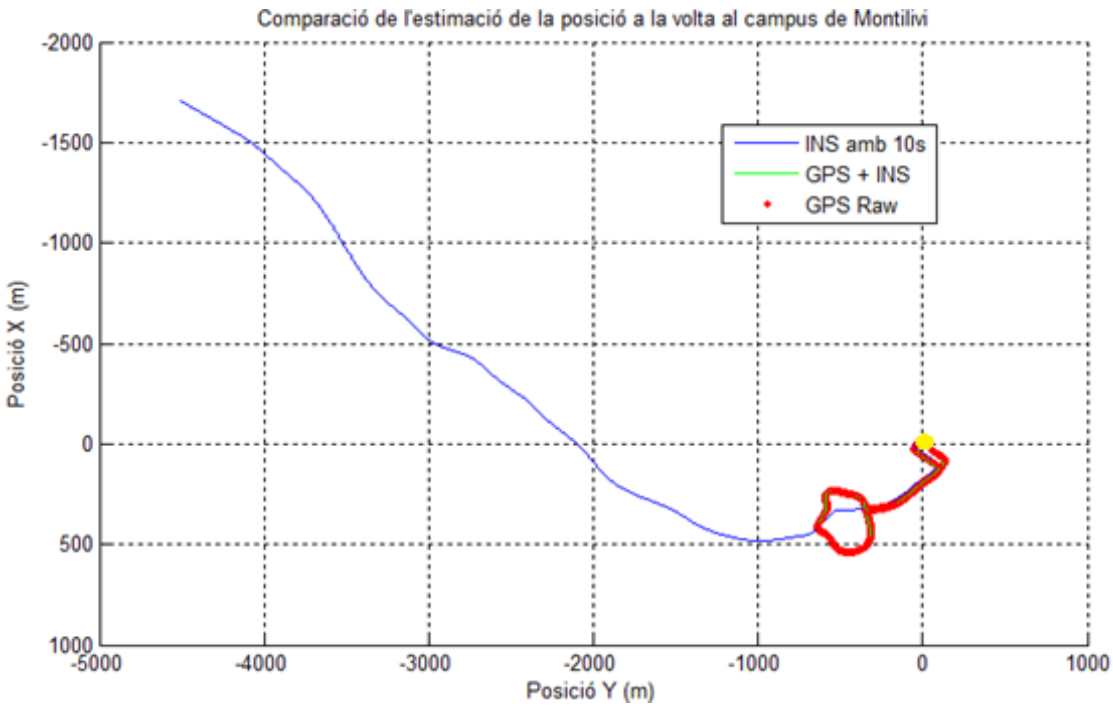
Tot seguit, anem a fer la mateixa comparativa amb el segon recorregut realitzat amb cotxe, les dues voltes al Parc Tecnològic. El punt groc marca l'inici de la trajectòria.



Gràfica 4.22: Estimacions de la posició en les dues voltes al Parc Tecnològic

En aquest cas l'estimació de la INS amb updates de GPS si que s'assembla molt al recorregut que mostren les dades raw del GPS. També veiem com en aquest cas l'estimació del INS sense ajuda externa és bona fins els primers tres quarts de volta, a partir d'aquí comença a derivar fins a acabar amb un error de posició molt gran. En el principi del recorregut la deriva de l'angle és petita i la compensació del vector gravetat es fa de manera adequada, però a mesura que l'error creix, el vector es desalinea i això provoca l'aparició de residus en l'acceleració que fan que la deriva en posició creixi cada cop més ràpidament.

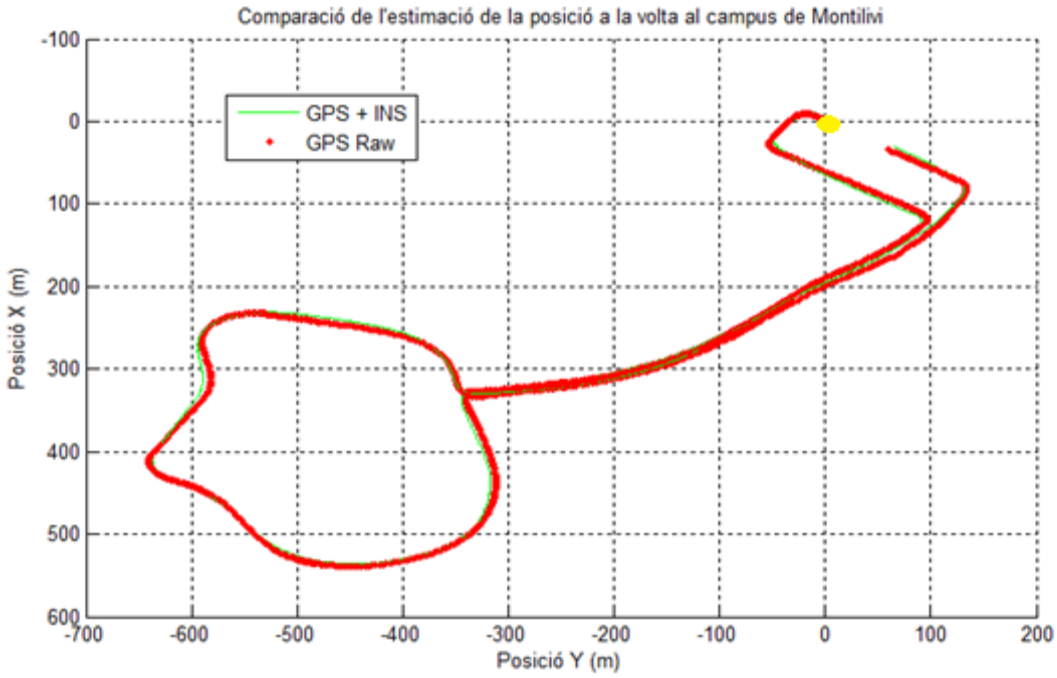
I ja per acabar les comparatives, mostrem la que fem amb l'últim recorregut, la volta a tot el campus de Montilivi de la UdG. El punt groc marc l'inici.



Gràfica 4.23: Recorregut fent la volta al campus de Montilivi

A simple vista podem veure com la deriva de la INS treballant sola és enorme. Tot i així estima correctament la posició fins la sortida del Parc Tecnològic i la pujada fins el campus de Montilivi. A partir d'aquí, l'error en l'estimació de la posició creix molt.

Fem un zoom per comparar les dades raw del GPS i l'estimació feta amb la INS i el GPS treballant de forma integrada.



Gràfica 4.24: Zoom volta al campus de Montilivi

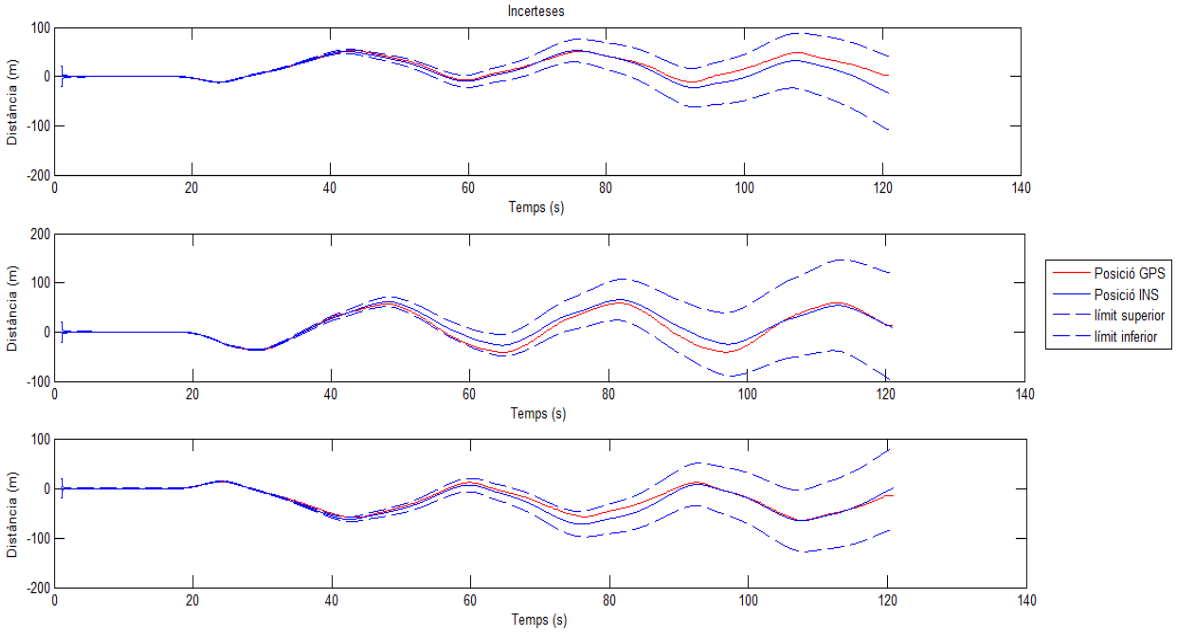
Veiem que els dos recorreguts són pràcticament iguals.

4.2.4.2. Incertesa

L'estimació realitzada pel filtre de Kalman no dona només una dada de posició, velocitat, orientació i biaix, sinó que també proporciona una estimació de la variància d'aquestes dades (matriu P). Aquesta variància mostra el nivell d'incertesa que tenim en l'estimació, és a dir, la nostra estimació pren el valor mig com a referència, però la posició, velocitat o orientació real del vehicle pot trobar-se dins dels límits descrits per aquesta variància. El fet de no rebre updates del GPS converteix el procés en una estimació per deadreckoning (navegació per estima), és a dir, susceptible d'acumular erros de deriva. Quan passa això veiem com la nostra incertesa creix en el temps.
Utilitzant el primer recorregut, el de les tres voltes al voltant del CIRS, i l'estimació que n'hem fet en que la INS treballa sense ajuda externa ensenyarem aquest concepte.

A continuació mostrem tres gràfiques corresponents a l'estimació dels valors de la posició x, y i z respectivament i de la seva incertesa. En blau tenim el valor de la posició estimada mitjançant el INS. Les línies discontinues corresponen a la incertesa del valor de la posició. Aquesta incertesa la calculem amb la matriu P del filtre de Kalman. Hem col·locat la incertesa

o desviació estàndard a 2·σ (95.4%). Això vol dir que en una 95.4% dels casos la posició real hauria de trobar-se dins els intervals marcats per la posició calculada amb la INS i la seva incertesa. Com que no tenim manera de conèixer la posició real, en el nostre cas hem agafat la posició que ens dona el GPS (marcada en vermell) que tot i tenir errors, no acumula deriva i és l'opció més raonable per fer la comparació.



Gràfica 4.25: Incertesa de les estimacions de posició en eixos x (superior), y (mig) i z (inferior)

Veiem com efectivament, la gran majoria del temps la posició del GPS es troba dins els intervals d'incertesa. Els quals amb el temps creixen i reflecteixen el procés de navegació per estima.

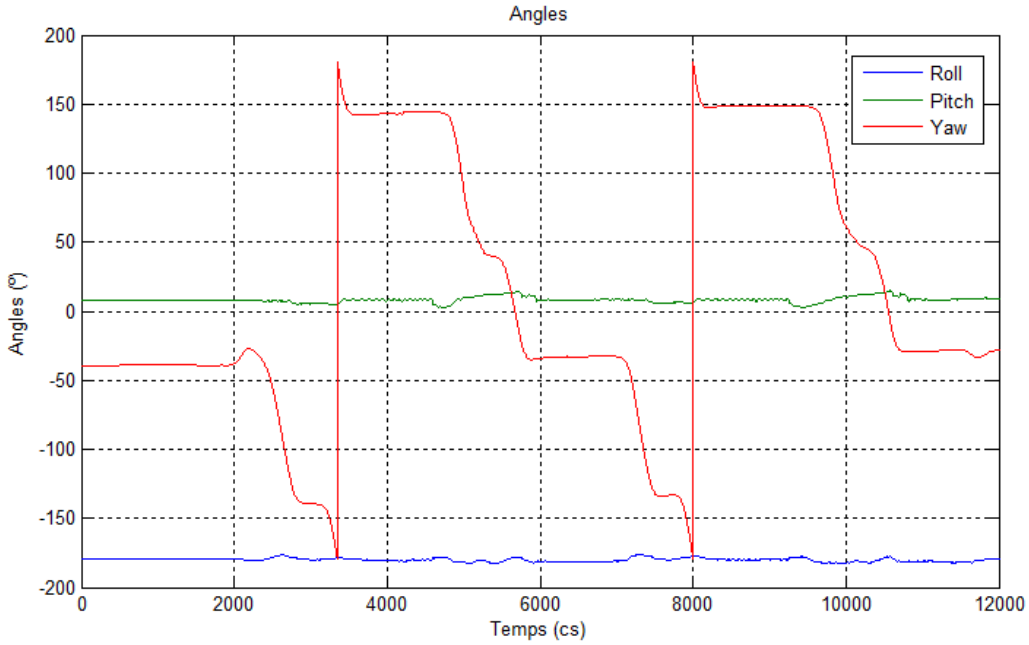
4.2.4.3. Estimació de l'orientació

Un cop hem comprovat com és comporten les estimacions de posició fetes amb el INS anem a veure com s'estimen altres variables com per exemple la orientació, els angles de Roll Pitch i Yaw.

Per examinar les estimacions de l'orientació hem agafat el segon recorregut, les dues voltes al Parc Tecnològic. El motiu és que al fer dues voltes podem comparar els angles obtinguts a la primera volta amb els de la segona. Comparem els angles de la part del trajecte on el

vehicle avança en línia recta. Si l'estimació dels angles és correcte els d'una volta haurien de ser pràcticament iguals als de l'altre.

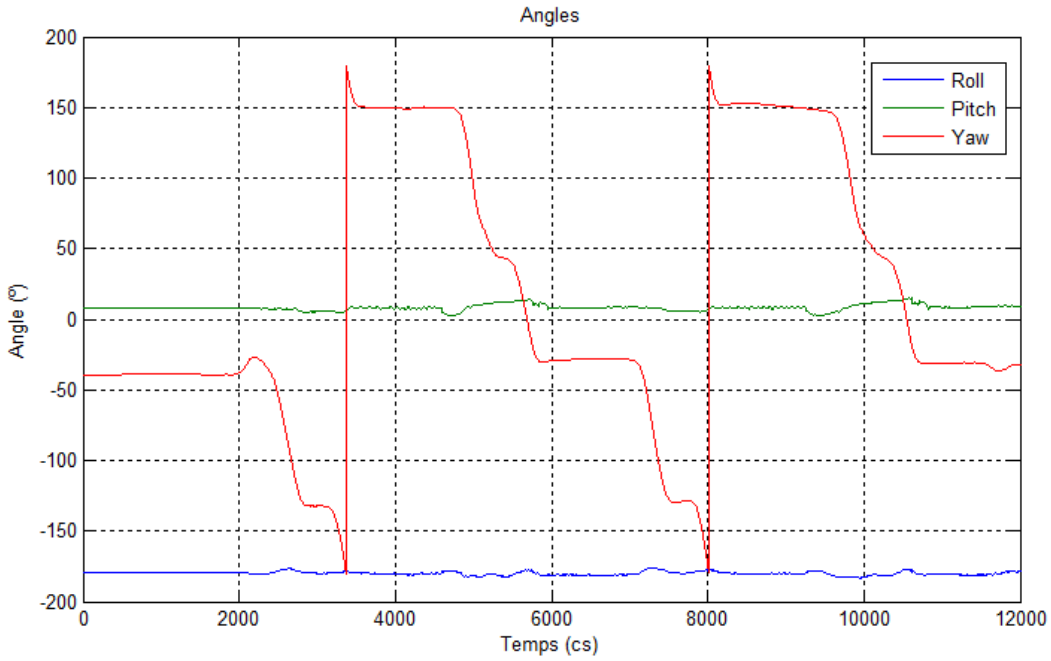
Els següents angles estant representats en el sistema de coordenades NED. El motiu és perquè són els més fàcils d'interpretar. Aquests angles (gràfica 4.26) són els que s'han estimat amb el INS sense rebre cap tipus d'ajuda externa tret dels 10 segons inicials.



Gràfica 4.26: Estimacions angle de les dues voltes al Parc Tecnològic

Amb aquesta gràfica es veuen clarament els dos cicles que representen les dos voltes fetes al Parc Tecnològic. Al voltant dels 4000 i 9000 mS es pot apreciar l'angle del vehicle al passar per la mateixa zona i es veu com hi ha una diferència de pocs graus (uns 5º aproximadament) en l'angle Yaw. Per tant, hi ha una mica de deriva en l'estimació de l'orientació. Tot i així, la deriva és petita i aquesta estimació està feta sense ajudes externes o sigui, que els resultats són molt bons.

Observem com és l'estimació de l'orientació (gràfica 4.27) si durant les dues voltes del recorregut anem fent Updates de GPS.



Gràfica 4.27: Estimació orientació de les dues voltes al Parc Tecnològic amb updates de GPS en tot el recorregut

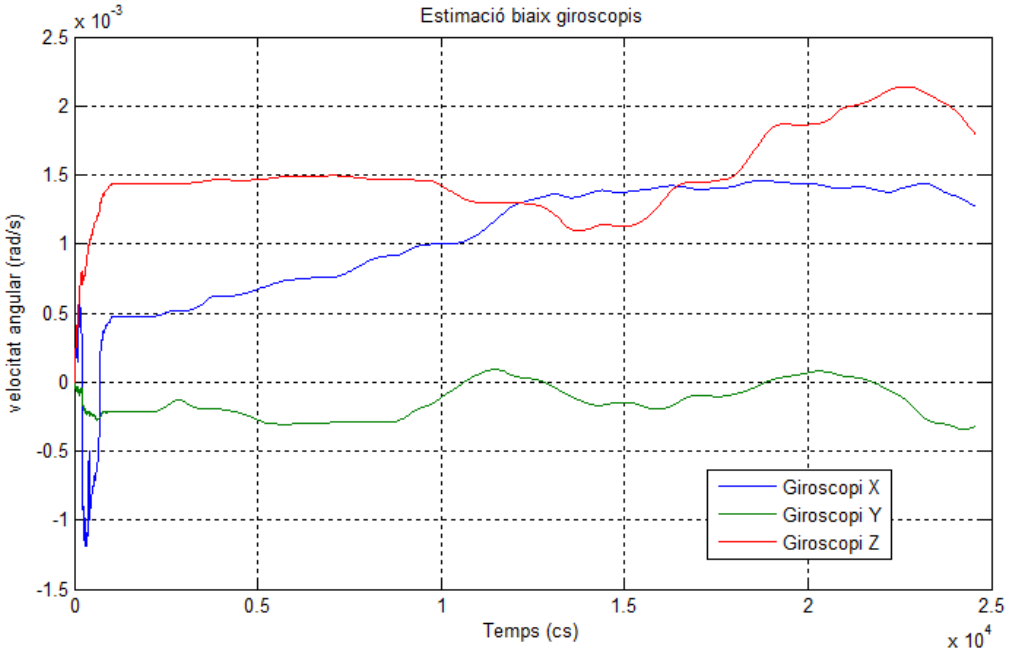
L'estimació dels angles millora, en els mateixos punts d'abans no s'apracia cap error, pràcticament no hi ha deriva. Els resultats són molt bons.

Amb aquests bons resultats podem confirmar que podríem utilitzar la IMU utilitzada (ADIS16488) més les equacions de navegació implementades com una AHRS. Les estimacions aconseguides sense updates de GPS durant les dues voltes ja són prou bones. Només faltaria integrar-hi updates de leveling i magnètic heading per corregir la petita deriva. D'aquesta manera podríem aconseguir una AHRS.

4.2.4.4. Estimació biaixos

A continuació comprovem com s'han comportat les estimacions el biaix en aquestes proves en moviment. Per comprovar l'estimació del biaix ens hem fixat en l'evolució del biaix del recorregut en que es fa la volta al campus de Montilivi. D'aquesta manera, com que agafem un recorregut llarg podem veure com el biaix varia al llarg del temps. Evidentment, s'ha utilitzat l'estimació en que hi ha updates de GPS tota l'estona perquè sinó només estimaríem el biaix durant els 10 primers segons.

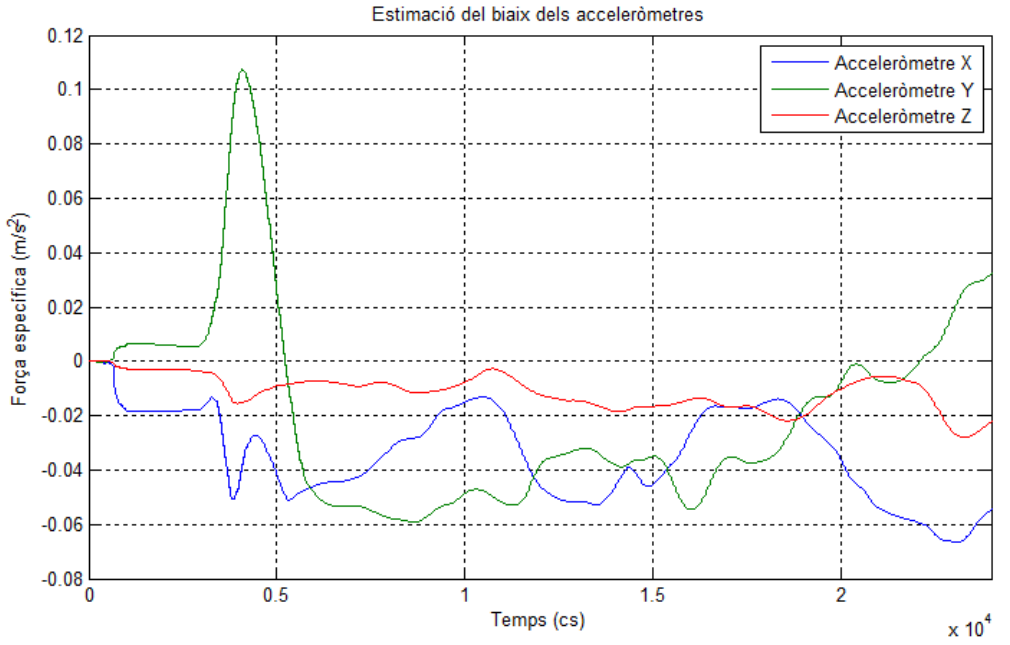
Comencem pels giroscopis.



Gràfica 4.28: Biaixos dels giroscopis durant la volta al campus

Veiem com sembla que el biaix dels giroscopis s'estabilitza en un valor però amb el temps va patint petites variacions.

Comprovem ara com es comporta el biaix dels acceleròmetres.



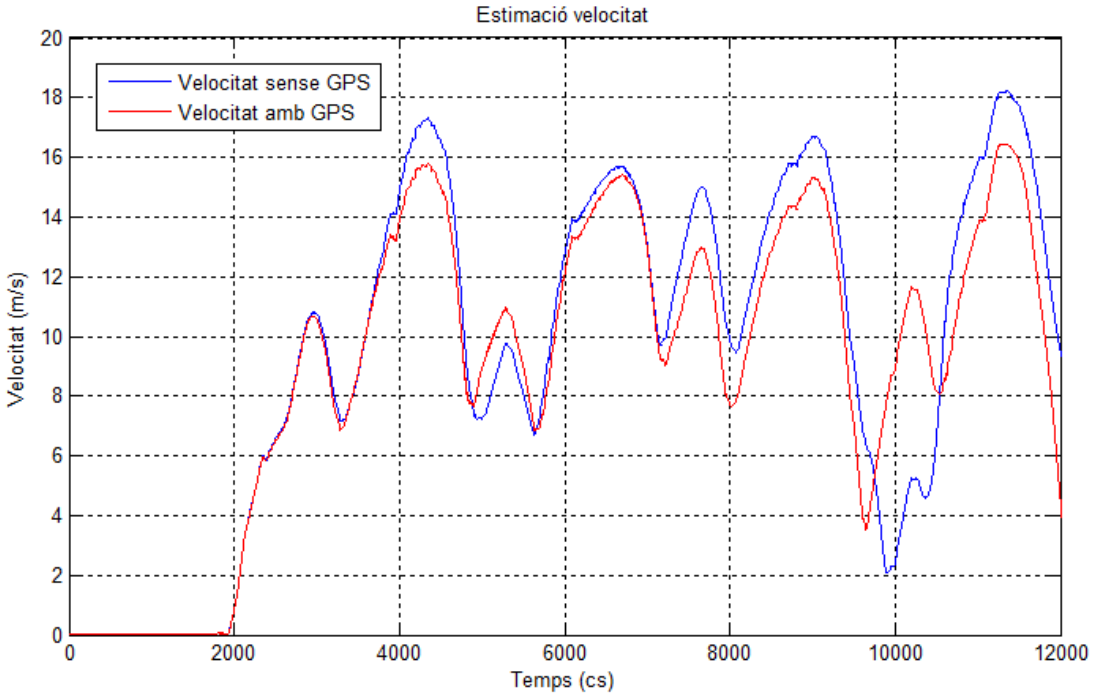
Gràfica 4.29: Biaixos acceleròmetres estimats amb el filtre de Kalman

Podem veure com el biaix dels acceleròmetres segueix sense poder-se estimar. La causa principal en aquest cas és que el GPS té força error, aquest causa discrepàncies amb la

informació obtinguda de la IMU. Aquestes discrepàncies no només es reflecteixen en canvis en l'estimació de l'error en posició, sinó que també es propaga en certa manera a l'estimació del biaix per intentar explicar les discrepàncies entre GPS i IMU.

4.2.4.5. Estimació velocitat

Finalment, queda comprovar com s'ha estimat la velocitat. Mostrem els resultats obtinguts amb la prova del recorregut en el Parc Tecnològic. Ensenyem l'estimació de la velocitat feta amb el INS i sense ajudes externes i de l'estimació feta amb updates de GPS.



Gràfica 4.30: Comparació de l'estimació de la velocitat

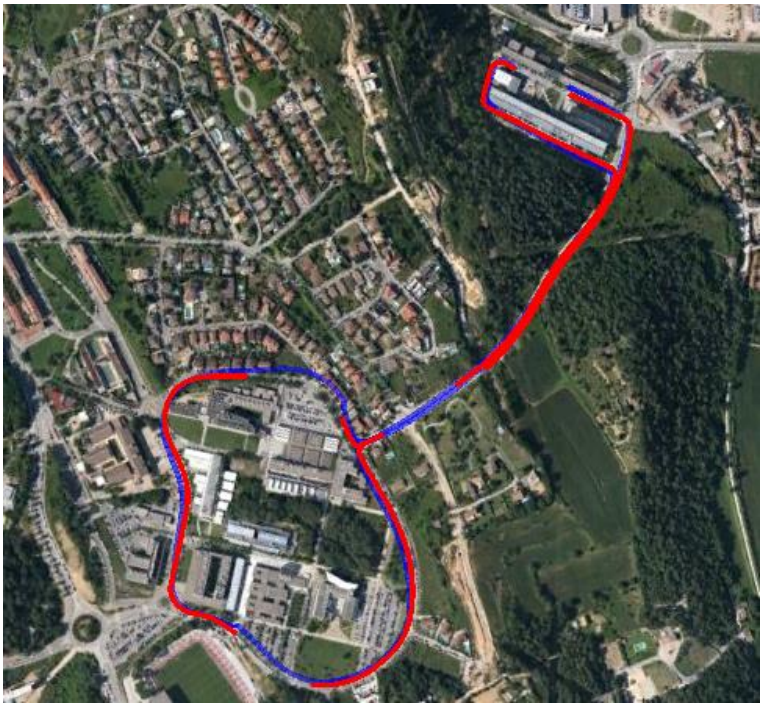
Si l'estimació fos del tot correcte els valors de velocitat haurien d'acabar a zero ja que al final de la prova es parava el cotxe. Veiem com l'estimació feta amb updates de GPS (línia vermella) si que tendeix a anar cap a zero tot i que no hi acaba d'arribar. L'estimació de la velocitat sense updates de GPS també tendeix a zero però presenta una deriva més gran.

4.2.4.6. Buits de cobertura GPS

Per acabar farem unes últimes proves. Volem veure com respon el INS quan tenim fallades de cobertura del GPS. O sigui, quan durant un petit període de temps ens quedem sense

rebre dades del GPS i la INS ha de navegar sola. Aquesta és un tipus de fallada que s'ajusta força a la realitat, per exemple quan passem per un túnel tindríem aquest buit de cobertura. A més el fet que el GPS estigui fent updates tota l'estona tret dels petits buits de cobertura permet que el biaix s'estimi més correctament.
Aquestes proves s'han fet amb el recorregut en que es fa una volta al campus de Montilivi.

En la primera prova fem 3 zones sense cobertura GPS, els buits de cobertura duren 10 segons cada un. En vermell trobem la part del recorregut on hi ha cobertura GPS. En blau l'estimació del INS. Podem veure com en les zones on no es reben dades del GPS l'estimació de la posició es continua fent, i amb força bons resultats. Si hi ha una corba el INS segueix la seva traçada.

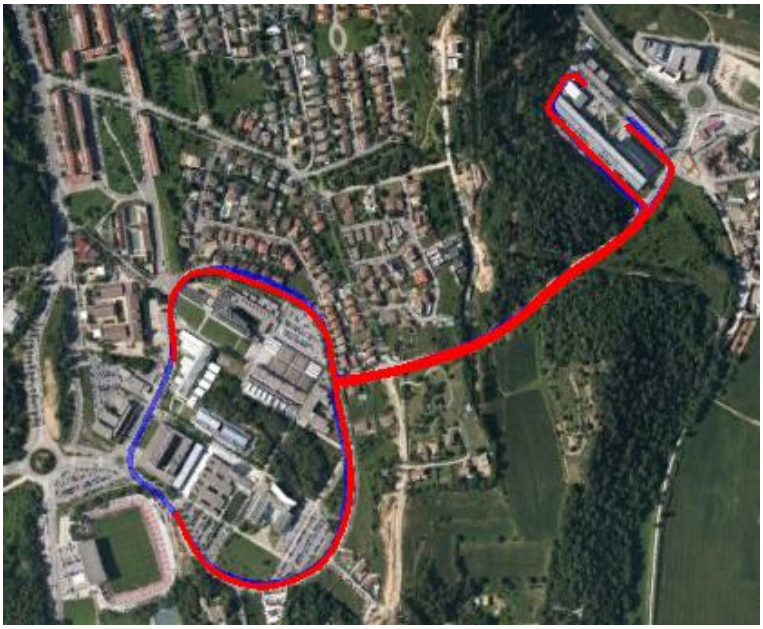


Imatge 4.5: Tres buits de cobertura GPS (només blau)

Una altre buit de GPS que s'ha provat ha estat durant la pujada. S'ha deixat de rebre dades de GPS durant tota la pujada fins al campus de Montilivi, uns 30 segons. Com podem veure en la següent imatge, tot i així el INS ha seguit estimant correctament la posició.

Imatge 4.6: Buit de cobertura GPS en la pujada

Finalment, per acabar de comprovar com responia el INS davant d'una corba tancada s'ha fet el següent buit de cobertura (de 20 segons de durada). Com podem veure el INS aconsegueix traçar perfectament la corba i estimar correctament la posició durant els 20 segons en que no rep cap tipus de dades del GPS.



Imatge 4.7: Buit cobertura GPS en una corba

5. RESUM DEL PRESSUPOST

Puja el Pressupost d'Execució per Contracta a falta d'iva la quantitat de DEU MIL DOS-CENTS DEU EUROS AMB QUINZE CÈNTIMS (10210,15€).

6. CONCLUSIONS

Un cop arribats al final d'aquest treball podem començar a extreure les conclusions, analitzar si hem aconseguit els objectius proposats i en quin grau de satisfacció.
En primer lloc, comentar que s'ha complert l'objectiu principal d'aquest treball que era l'estudi i implementació de les equacions de navegació. Aquestes equacions s'han comprés, escrit i simulat en Matlab obtenint uns resultats que demostraven el seu correcte funcionament.

Tot i el correcte funcionament de les equacions de navegació, l'estima de la posició, velocitat i orientació pateix una deriva important. Per corregir-la hem implementat un filtre de Kalman que ha funcionat correctament hi ha reduït la deriva i per tant, millorat l'estimació de la posició, velocitat i orientació.

Amb la implementació del filtre de Kalman s'han aconseguit resultats molt satisfactoris en l'estima de l'orientació. Això ens fa arribar a la conclusió que hem aconseguit crear una AHRS a partir de la IMU i les equacions de navegació. Faltaria introduir updates de leveling i magnètic heading amb el filtre de Kalman per acabar de reduir la petita deriva i aconseguir una estimació de l'orientació més acurada.

Per altre banda els resultats en l'estima de la posició i velocitat tenen marge de millora, tot i que són prometedors i adequats per a un sensor de categoria tactical (ADIS16488). Durant uns segons aconseguim una estima de la posició acceptable, però un cop passats aquests segons inicials la deriva comença a augmentar de forma considerable. Els sensors de la categoria tactical poden estimar correctament la posició durant uns pocs minuts, la ADIS16488 és de la gamma biaxa de tactical, per tant compleix les espectetives. Es necessiten sensors de més bona qualitat i amb una precisió més elevada per aconseguir una bona navegació inercial. Aquests sensors es troben disponibles en el mercat però, evidentment, amb un preu molt més elevat.

Altres formes de reduir aquest error serien introduir els updates de leveling i magnètic heading perquè no hi hagi deriva en l'orientació. Recordar que la deriva en l'estimació dels

angles provoca que no es compensi correctament la gravetat i això afecti negativament a l'estimació de la posició i velocitat. També seria productiu realitzar un estudi de fiabilitat de cada sensor de la IMU per separat (giroscopis i acceleròmetres) per així poder aconseguir els paràmetres de desviació en la mesura de cada sensor i poder ajustar el filtre de Kalman més correctament. En el cas d'integrar-ho en un robot submarí, una altre mesura per reduir la deriva seria integrar el INS amb altres sensors com un sonar Doppler per obtenir mesures de velocitat i un sensor de pressió per estimar la profunditat. Això ens permetri obtenir una navegació més robusta.

El que no s'han pogut realitzar han estat les proves de les equacions de navegació amb un robot. Els motius han sigut que per fer-ho es necessitava integrar la IMU i les equacions de navegació amb altres sensors per aconseguir una estimació més bona de la posició, velocitat i orientació i així obtenir una navegació més fiable. A més, tots aquests sensors (IMU, baròmetre, velocímetre, etc) s'haurien d'instal·lar en el robot. El que significa programar els drivers dels sensors, així com les equacions de navegació i el filtre de Kalman dins de l'arquitectura software del robot. Aquestes tasques requerien un temps i uns coneixements dels quals no desponíem i per tant, no les hem pogut dur a terme.

Aquest treball es podria ampliar o continuar de la forma que acabem de comentar. Un cop realitzat aquest projecte els següents passos serien la integració del INS resultant d'aquest treball (IMU + equacions de navegació) amb altres sensors per poder aconseguir una millor navegació, sempre parlant en el camp dels robots autònoms submarins. Els sensors que es podrien integrar podrien ser; un baròmetre per mesurar la profunditat; un sonar Doppler per prendre mesures de la velocitat a la que es mou el submarí; un compàs per estimar l'orientació que té el submarí en tot moment i un GPS per tenir mesures de la posició així que el vehicle submarí pugi a la superfície. Un cop s'aconseguís integrar tots aquests sensors es podria aconseguir una navegació molt més fiable i amb una deriva reduïda.

Data: 04/09/2014

7. BIBLIOGRAFIA

ALONSO, R., SHUSTER, M. D. Complete linear attitude-independent magnetòmetre calibration, The Journal of the astronautical sciences, 50(4), pp. 477-490. Eng. 2002.

GROVES, P. D. Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems (2a ed.). Artech House Publishers. Boston. 2013.

JUSTIN PETER DINALE. Magnetic Test Facility - Sensor and Coil Calibrations, Research Report DSTO-RR-0396. Maritime Division Defence Science and Technology Organisation. 2013.

KONVALIN, C. Compensating for Tilt, Hard-Iron, and Soft-Iron Effects. 2009.

NOAA. Magnetic Declination. (http://www.ngdc.noaa.gov/geomag/declination.shtml , 1 de juny de 2014)

TITTERTON, DAVID H., WESTON, JOHN L. Strapdown Inertial Navigation Technology (2a ed.). The Institution of Electrical Engineers. Herts. 2004.

WELCH, G., BISHOP, G. An Introduction to the Kalman Filter. University of North Carolina at Chapel Hill. 2001.

WOODMAN, OLIVER J. An Introduction to inertial navigation. University of Cambridge. 2007.

ANNEXOS

A. ANNEX PRESSUPOST

Anem a explicar en detall el pressupost necessari per la realització d'aquest treball.

A.1. Preus unitaris

Concepte	Preu unitari
Enginyer	36 €/h
Tècnic	24 €/h
AHRS MTi	1750 €
IMU ADIS16488	1400 €
GPS	60€
Placa adquisició de dades per ADIS16488	355 €
Gasolina	0.25€/Km

A.2. Pressupostos parcials

Estudi i implementació de les equacions de navegació:

Mà d'obra	Preu unitari	Hores	Total
Enginyer	36 €/h	90 h	3240 €

Realització de proves:

Mà d'obra	Preu unitari	Hores	Total
Tècnic	24 €/h	8 h	192 €
Material	**Preu unitari**	**Unitats**	**Total**
AHRS MTi	1750 €	1	1750 €
IMU ADIS16488	1400 €	1	1400 €
Placa adquisició de dades	355 €	1	355 €
GPS	60 €	1	60€
Gasolina	0.25 €/km	4.6 Km	1.15 €
TOTAL			3758.15 €

Avaluació de resultats:

Mà d'obra	Preu unitari	Hores	Total
Enginyer	36 €/h	40 h	1440 €

A.3. Resum de pressupost

Capítol	Import
Estudi i implementació de les equacions de navegació	3240 €
Realització de proves	3758.15 €
Avaluació de resultats	1440 €
PRESSUPOST D'EXECUCIÓ MATERIAL (PEM)	**8438.15 €**

A.4. Pressupost general

PRESSUPOST D'EXECUCIÓ MATERIAL (PEM)	8438.15
13 % de despeses generals	1096.95
8 % de benefici Industrial	675.05
PRESSUPOST D'EXECUCIÓ PER CONTRACTE A FALTA D'IVA	**10210.15**

Aquest pressupost d'execució per contracte puja a (Deu mil dos-cents deu Euros amb quinze Cèntims més IVA)

B. ANNEX CÀLCUL

B.1. Introducció

En aquest annex si trobaran tots els càlculs necessaris per implementar les equacions de navegació que per motius de volum no s'han introduït directament a la memòria d'aquest treball.

Per estructurar aquest annex s'ha seguit l'ordre que té la memòria. Començant pels càlculs de les matrius de rotació i acabant pels del filtre de Kalman.

B.2. Matrius de canvi de sistema de referència

En aquest capítol s'exposaran totes les matrius necessàries per fer els canvis de sistema de referència entre els sistemes de coordenades utilitzats.

B.2.1. ECI i ECEF

A continuació mostrarem la matriu de rotació necessària per passar de ECI a ECEF.

$$C_i^e = \begin{pmatrix} \cos w_{ie} \cdot (t - t_0) & \sin w_{ie} \cdot (t - t_0) & 0 \\ -\sin w_{ie} \cdot (t - t_0) & \cos w_{ie} \cdot (t - t_0) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Si volem passar de ECEF a ECI.

$$C_e^i = \begin{pmatrix} \cos w_{ie} \cdot (t - t_0) & -\sin w_{ie} \cdot (t - t_0) & 0 \\ \sin w_{ie} \cdot (t - t_0) & \cos w_{ie} \cdot (t - t_0) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

On:

w_{ie}: 7.292115E-5 rad/s. Velocitat de rotació de la Terra.

t: temps.

t_0: temps inicial.

B.2.2. ECEF i NED

Matriu de rotació per passar de ECEF a NED.

$$C_e^n = \begin{pmatrix} -sinL_b \cdot \cos\lambda_b & -sinL_b \cdot \sin\lambda_b & \cos L_b \\ -\sin\lambda_b & \cos\lambda_b & 0 \\ -cosL_b \cdot \cos\lambda_b & -\cos L_b \cdot \sin\lambda_b & -\sin L_b \end{pmatrix}$$

Per passar de NED a ECEF.

$$C_n^e = \begin{pmatrix} -sinL_b \cdot \cos\lambda_b & -\sin\lambda_b & -cosL_b \cdot \cos\lambda_b \\ -sinL_b \cdot \sin\lambda_b & \cos\lambda_b & -\cos L_b \cdot \sin\lambda_b \\ \cos L_b & 0 & -\sin L_b \end{pmatrix}$$

On:

L_b: latitud (rad).

λ_b: longitud (rad).

B.2.3. ECI i NED

Matriu de rotació per passar de ECI a NED.

$$C_i^n = \begin{pmatrix} -sinL_b \cdot \cos(\lambda_b + w_{ie} \cdot (t - t_0)) & -sinL_b \cdot \sin(\lambda_b + w_{ie} \cdot (t - t_0)) & \cos L_b \\ -\sin(\lambda_b + w_{ie} \cdot (t - t_0)) & \cos(\lambda_b + w_{ie} \cdot (t - t_0)) & 0 \\ -cosL_b \cdot \cos(\lambda_b + w_{ie} \cdot (t - t_0)) & -\cos L_b \cdot \sin(\lambda_b + w_{ie} \cdot (t - t_0)) & -\sin L_b \end{pmatrix}$$

Per passar de NED a ECI.

$$C_n^i = \begin{pmatrix} -sinL_b \cdot \cos(\lambda_b + w_{ie} \cdot (t - t_0)) & -\sin(\lambda_b + w_{ie} \cdot (t - t_0)) & -cosL_b \cdot \cos(\lambda_b + w_{ie} \cdot (t - t_0)) \\ -sinL_b \cdot \sin(\lambda_b + w_{ie} \cdot (t - t_0)) & \cos(\lambda_b + w_{ie} \cdot (t - t_0)) & -\cos L_b \cdot \sin(\lambda_b + w_{ie} \cdot (t - t_0)) \\ \cos L_b & 0 & -\sin L_b \end{pmatrix}$$

B.3. Conversió de posicions entre diferents sistemes

Sistemes alternatius per convertir posicions d'un sistema de referència a un altre sense utilitzar la matriu de rotació.

B.3.1. De sistema local de navegació a ECEF

A partir de les coordenades de longitud, latitud i altura podem trobar la posició cartesiana ECEF.

$$x_{eb}^{e} = (R_E(L_b) + h_b) \cdot \cos L_b \cdot \cos \lambda_b$$

$$y_{eb}^{e} = (R_E(L_b) + h_b) \cdot \cos L_b \cdot \sin \lambda_b$$

$$z_{eb}^{e} = [(1 - e^2) \cdot R_E(L_b) + h_b] \cdot \sin L_b$$

On:

$$R_E = \frac{R_0}{\sqrt{1 - e^2 \cdot \sin {L_b}^2}}$$

e = 0.081819191

R_0: Radi de la Terra a l'ecuador 6378137 m.

x_{eb}^{e}, y_{eb}^{e} i z_{eb}^{e}: coordenades cartesianes ECEF. (m)

L_b, λ_bi h_b: coordenades sistema local de navegació (rad), (rad) i (m).

B.3.2. De ECEF a sistema local de navegació (Borkowski)

Mètode per passar de coordenades cartesianes ECEF a les coordenades pseudoesfèriques utilitzades en el sistema local de navegació. S'anomena Borkowski.

$$\lambda_b = atan2(y_{eb}^{e}, x_{eb}^{e})$$

$$B = \sqrt{{x_{eb}^{e}}^2 + {y_{eb}^{e}}^2}$$

$$E = \frac{\sqrt{1 - e^2}|z_{eb}^{e}| - e^2 \cdot R_0}{B}$$

$$F = \frac{\sqrt{1 - e^2}|z_{eb}^{e}| + e^2 \cdot R_0}{B}$$

$$P = \frac{4}{3}(E \cdot F + 1)$$

$$Q = 2 \cdot (E^2 - F^2)$$

$$D = P^3 + Q^3$$

$$V = \left(D^{1/2} - Q\right)^{1/3} - \left(D^{1/2} + Q\right)^{1/3}$$

$$G = \frac{1}{2} \cdot \left(\sqrt{E^2 + V} + E\right)$$

$$T = \sqrt{G^2 + \frac{F - VG}{2G - E}} - G$$

$$L_b = sign(z_{eb}^e) \cdot arctan\left(\frac{1 - T^2}{2 \cdot T \cdot \sqrt{1 - e^2}}\right)$$

$$h_b = (B - R_0 \cdot T) \cdot cosL_b + \left(z_{eb}^e - sign(z_{eb}^e) \cdot R_0 \cdot \sqrt{1 - e^2}\right) \cdot sinL_b$$

Els càlculs de B, E, F, P, Q, D, V, G i T només són càlculs inter mitjos, no tenen cap mena de significat fora d'aquestes equacions.

B.4. Càlcul angles d'Euler

Un cop tenim la matriu de rotació en podem extreure els angles d'Euler que hi ha entre els dos sistemes que representa la matriu de rotació. S'han de seguir aquestes equacions.

Si tenim una matriu de rotació de Body a NED anomenada C_b^n per trobar els angles d'Euler entre els dos sistemes les equacions són:

$$\varphi_{nb} = arctan2(C^n_{b3,2}, C^n_{b3,3})$$

$$\theta_{nb} = -arcsin C^n_{b3,1}$$

$$\Psi_{nb} = arctan2(C^n_{b2,3}, C^n_{b1,1})$$

On:

φ_{nb}, θ_{nb} i Ψ_{nb}: són Roll, Pitch i Yaw respectivament.

B.5. Càlcul gravetat

En aquest capítol s'explicaran els models que s'han utilitzat per poder realitzar una bona estimació de la força de la gravetat. Recordem que les formes de càlcul varien segons si es treballa amb ECI, ECEF o el sistema local de navegació.

B.5.1. Gravetat ECI

Quan treballem amb el sistema de coordenades ECI a la gravetat no hi ha cap component de força centrípeta, o sigui només hem de calcular l'acceleració de la gravetat.

$$\gamma^i_{ib} = -\frac{\mu}{\left|r^i_{ib}\right|^3}\left\{ r^i_{ib} + \frac{3}{2}J_2\frac{R_0^2}{\left|r^i_{ib}\right|^2}\left\{\begin{bmatrix} 1-5\left(r^i_{ib,z}/\left|r^i_{ib}\right|\right)^2 \end{bmatrix}\cdot r^i_{ib,x} \\ \begin{bmatrix} 1-5\left(r^i_{ib,z}/\left|r^i_{ib}\right|\right)^2 \end{bmatrix}\cdot r^i_{ib,y} \\ \begin{bmatrix} 3-5\left(r^i_{ib,z}/\left|r^i_{ib}\right|\right)^2 \end{bmatrix}\cdot r^i_{ib,z}\right\}\right\}$$

On:

γ^i_{ib}: és l'acceleració de la força de la gravetat que volem calcular.

μ: constant gravitatòria de la Terra, el seu valor és de 3.986004418E14 m^3/s^2.

J_2 **:** segona constant gravitatòria de la Terra. 1.082627E-3.

r^i_{ib} **:** la posició que ens trobem en coordenades ECI.

B.5.2. Gravetat ECEF

Quan treballem amb el sistema de coordenades ECEF a la l'acceleració de la gravetat se li ha de sumar una petita component deguda a la força centrípeta provocada per la rotació de la Terra.

$$\gamma_{ib}^{e} = -\frac{\mu}{\left|r_{eb}^{e}\right|^{3}}\left\{ r_{eb}^{e} + \frac{3}{2}J_2\frac{R_0^2}{\left|r_{eb}^{e}\right|^{2}}\left\{\begin{bmatrix}\left[1-5\left(r_{eb,z}^{e}/\left|r_{eb}^{e}\right|\right)^{2}\right]\cdot r_{eb,x}^{e}\\ \left[1-5\left(r_{eb,z}^{e}/\left|r_{eb}^{e}\right|\right)^{2}\right]\cdot r_{eb,y}^{e}\\ \left[3-5\left(r_{eb,z}^{e}/\left|r_{eb}^{e}\right|\right)^{2}\right]\cdot r_{eb,z}^{e}\end{bmatrix}\right\}\right\}$$

$$g_{b}^{e} = \gamma_{ib}^{e} + w_{ie}^{2}\cdot\begin{pmatrix}1&0&0\\0&1&0\\0&0&0\end{pmatrix}\cdot r_{eb}^{e}$$

On:

g_b^e: és la força de la gravetat que volem calcular (m/s^2).

r_{eb}^e: la posició en què ens trobem amb coordenades ECEF (m).

γ_{ib}^e: acceleració de la gravetat. (m/s^2).

B.5.3. Gravetat sistema local de navegació

Pel càlcul de la gravetat quan treballem amb el sistema local de navegació fem servir un model diferent als utilitzats amb ECI i ECEF.

$$g_0(L_b) = 9.7803253359\cdot\frac{(1+0.001931853\cdot sin^2 L)}{\sqrt{1-e^2\cdot sin^2 L}}$$

$$g_{b,N}^{n}(L_b,h_b) = -8.08*10^{-9}\cdot h_b\cdot\sin 2\cdot L_b$$

$$g_{b,D}^{n}(L_b,h_b) = g_0(L_b)\left\{1-\frac{2}{R_0}\left[1+f(1-2sin^2 L_b)+\frac{w_{ie}^2\cdot R_0^2\cdot R_p}{\mu}\right]\cdot h_b+\frac{3}{R_0^2}\cdot h_b^2\right\}$$

$$g_b^n = \begin{pmatrix} g_{b,N}^n \\ 0 \\ g_{b,D}^n \end{pmatrix}$$

On:

e: és una constant que val 0.081819191

R_0: el radi equatorial de la Terra, 6378137 m.

R_p: el radi polar de la Terra, 6356752 m

f: constant que val 1/298.257223563

g_b^n: gravetat en coordenades NED que volem calcular. (m/s^2)

B.6. Velocitat inicial ECI

Veiem la fórmula que hem de fer servir per inicialitzar la velocitat inicial en el sistema ECI. Recordem que té una component de velocitat deguda a la rotació de la Terra.

$$v_{ib}^i = C_e^i \cdot (v_{eb}^e + \Omega_{ie}^e \cdot r_{eb}^e)$$

On:

v_{eb}^e: la velocitat respecte ECEF (m/s).

r_{eb}^e: la posició inicial respecte ECEF (m). Si ens trobem a l'instant inicial la posició ECEF coincideix amb la posició respecte ECI.

C_e^i: matriu de rotació de ECEF a ECI. Si volem calcular la velocitat a l'instant inicial aquesta matriu es converteix en la matriu identitat perquè en aquell moment ECI i ECEF es troben alineats.

$$\Omega_{ie}^e: \begin{pmatrix} 0 & -w_{ie} & 0 \\ w_{ie} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

v_{ib}^i: velocitat respecte ECI (m/s).

B.7. Equacions de navegació

A la memòria s'introdueixen les equacions de navegació, però en alguns casos no s'explica com es calculen algunes components. El motiu és per no fer molt feixuga la lectura i la comprensió del text. En aquest capítol de l'annex s'expliquen, amb més detall, els components les equacions de navegació.

B.7.1. Equacions de navegació utilitzant ECI

Per explicar les components de les equacions de navegació s'anirà seguint l'ordre que s'utilitzava a la memòria. Primer de tot, començarem pel càlcul de la matriu de rotació.

$$C_b^i(+) = C_b^i(-) \cdot \left(I_3 + \Omega_{ib}^b \cdot t\right)$$

On:

$C_b^i(+)$: és la matriu de rotació que volem calcular i que aplica una rotació de Body a ECI.

$C_b^i(-)$: és la matriu de rotació calculada en la iteració anterior.

I_3: matriu identitat de 3x3.

Ω_{ib}^b: és la matriu antisimètrica amb els valors de la velocitat angulars (rad/s) extrets de la IMU.

$$\Omega_{ib}^b = \begin{pmatrix} 0 & -w_{ib,z}^b & w_{ib,y}^b \\ w_{ib,z}^b & 0 & -w_{ib,x}^b \\ -w_{ib,y}^b & w_{ib,x}^b & 0 \end{pmatrix}$$

t: temps de durada d'una iteració (s).

Tot seguit, es realitzava el canvi de sistema de referència de la força específica.

$$f_{ib}^i = \frac{1}{2} \cdot \left(C_b^i(-) + C_b^i(+)\right) \cdot f_{ib}^b$$

On:

f_{ib}^i: són les dades dels acceleròmetres que referenciades sobre el ECI.

f_{ib}^{b}: són les dades que ens donen els acceleròmetres de la IMU. Estan referenciades respecte el Body.

Després fem el càlcul de la velocitat:

$$v_{ib}^{i}(+) = v_{ib}^{i}(-) + \left(f_{ib}^{i} + \gamma_{ib}^{i}(r_{ib}^{i}(-))\right) \cdot t$$

On:

$v_{ib}^{i}(+)$: és la velocitat (m/s) respecte ECI que volem calcular a l'instant actual.

$v_{ib}^{i}(-)$: la velocitat (m/s) de la iteració anterior.

$\gamma_{ib}^{i}(r_{ib}^{i}(-))$: acceleració de la gravetat (m/s^2). Entre parèntesis la posició de la Terra a la qual s'ha de calcular l'acceleració de la gravetat.

I finalment, el càlcul de la posició.

$$r_{ib}^{i}(+) = r_{ib}^{i}(-) + v_{ib}^{i}(+) \cdot t - \left(f_{ib}^{i} + \gamma_{ib}^{i}\left(r_{ib}^{i}(-)\right)\right) \cdot \frac{t^2}{2}$$

On:

$r_{ib}^{i}(+)$: és la posició respecte ECI que volem calcular (m).

$r_{ib}^{i}(-)$: és la posició respecte ECI de la iteració anterior.

B.7.2. Equacions de navegació utilitzant ECEF

Amb els càlculs respecte ECEF també seguirem el mateix ordre en les explicacions. Per tant, el primer pas és el càlcul de la matriu de rotació.

$$C_{b}^{e}(+) = C_{b}^{e}(-) \cdot \left(I_3 + \Omega_{ib}^{b} \cdot t\right) - \Omega_{ie}^{e} \cdot C_{b}^{e}(-) \cdot t$$

On:

$C_{b}^{e}(+)$: és la matriu de rotació actual de Body a ECEF.

$C_{b}^{e}(-)$: és la matriu de rotació de la iteració anterior.

I_3: matriu identitat de 3x3.

Ω_{ib}^{b}: és la matriu antisimètrica amb els valors de la velocitat angulars (rad/s) extrets de la IMU.

$$\Omega_{ib}^{b} = \begin{pmatrix} 0 & -w_{ib,z}^{b} & w_{ib,y}^{b} \\ w_{ib,z}^{b} & 0 & -w_{ib,x}^{b} \\ -w_{ib,y}^{b} & w_{ib,x}^{b} & 0 \end{pmatrix}$$

t: temps de durada d'una iteració (s).

Ω_{ie}^{e}: és la matriu antisimètrica del vector de rotació de la Terra (rad/s).

$$\Omega_{ie}^{e} = \begin{pmatrix} 0 & -w_{ie} & 0 \\ w_{ie} & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

w_{ie}: és la velocitat de rotació de la Terra, $7.292115 * 10^{-5}\text{rad/s}$

Seguidament, apliquem el canvi de sistema de referència.

$$f_{ib}^{e} = \frac{1}{2} \cdot (C_{b}^{e}(-) + C_{b}^{e}(+)) \cdot f_{ib}^{b}$$

On:

f_{ib}^{e}: és el valor de la força específica referenciat a l'ECEF.

A continuació, es calcula la velocitat.

$$v_{eb}^{e}(+) = v_{eb}^{e}(-) + \left(f_{ib}^{e} + g_{b}^{e}(r_{eb}^{e}(-)) - 2 \cdot \Omega_{ie}^{e} \cdot v_{eb}^{e}(-)\right) \cdot t$$

On:

$v_{eb}^{e}(+)$: és el nou vector velocitat que volem calcular (m/s).

$v_{eb}^{e}(-)$: és el vector de velocitat calculat a la iteració anterior (m/s).

$g_{b}^{e}(r_{eb}^{e}(-))$: correcció de la gravetat en el punt r_{eb}^{e} de la Terra. S'utilitza l'equació calculada a l'apartat B.5.2 d'aquest annex.

Finalment, el càlcul de la posició.

$$r_{eb}^{e}(+) = r_{eb}^{e}(-) + v_{eb}^{e}(-) \cdot t + \left(f_{ib}^{e} + g_{b}^{e}(r_{eb}^{e}(-)) - 2 \cdot \Omega_{ie}^{e} \cdot v_{eb}^{e}(-)\right) \cdot \frac{t^2}{2}$$

On:

$r_{eb}^{e}(+)$: la posició en coordenades ECEF que volem calcular (m).

$r_{eb}^{e}(-)$: la posició en coordenades ECEF obtinguda en la iteració anterior (m).

B.7.3. Equacions de navegació en el sistema local de navegació

Finalment, els càlculs amb el sistema local de navegació, l'últim que hem utilitzat. En aquest és on s'haurà d'explicar el càlcul de més components ja que, com hem reiterat forces vegades, els càlculs amb el sistema local de navegació són els més complicats d'implementar.

L'ordre seguit és el mateix que amb ECI i ECEF.

Comencem pel càlcul de la matriu de rotació.

$$C_b^n(+) = C_b^n(-) \cdot \left(I_3 + \Omega_{ib}^b \cdot t\right) - (\Omega_{ie}^n(-) + \Omega_{en}^n(-)) \cdot C_b^n(-) \cdot t$$

On:

$C_b^n(+)$: Matriu de rotació que volem trobar. Transforma les dades de Body a NED.

$C_b^n(-)$: Matriu de rotació del càlcul immediatament anterior.

$\Omega_{ie}^n(-)$: Té en compte la rotació de la Terra segons a la posició en que ens trobem.

$$\Omega_{ie}^n = w_{ie} \cdot \begin{pmatrix} 0 & \sin L_b & 0 \\ -\sin L_b & 0 & -\cos L_b \\ 0 & \cos L_b & 0 \end{pmatrix}$$

L_b: latitud en que es troba la IMU (rad).

$\Omega_{en}^n(-)$: Matriu antisimètrica amb les derivades de la longitud i la latitud.

$$\Omega_{en}^n = \begin{pmatrix} 0 & -w_{en,z}^n & w_{en,y}^n \\ w_{en,z}^n & 0 & -w_{en,x}^n \\ -w_{en,y}^n & w_{en,x}^n & 0 \end{pmatrix}$$

$$w_{en}^n = \begin{pmatrix} v_{eb,E}^n \Big/ (R_E(L_b) + h_b) \\ -v_{eb,N}^n \Big/ (R_N(L_b) + h_b) \\ v_{eb,E}^n \cdot tan(L_b) \Big/ (R_E(L_b) + h_b) \end{pmatrix}$$

h_b: altura respecte el geoide (m).

$v_{eb,E}^n$ i $v_{eb,N}^n$: velocitat lineal de la IMU respecte ECEF però representada en els eixos del NED. Els subíndex N, E i D signifiquen si són la component nord, east o down respectivament.

$R_E(L_b)$: radi de curvatura del paral·lel segons a la latitud en que ens trobem.

$$R_E(L_b) = \frac{R_0}{\sqrt{1 - e^2 \cdot sin^2 L_b}}$$

$R_N(L_b)$: radi de curvatura del meridià.

$$R_N(L_b) = \frac{R_0 \cdot (1 - e^2)}{(1 - e^2 \cdot sin^2 L_b)}$$

R_0: radi equatorial, 6378137 m.

Segueix el canvi de sistema de referència.

$$f_{ib}^n = \frac{1}{2} \cdot (C_b^n(-) + C_b^n(+)) \cdot f_{ib}^b$$

On:

f_{ib}^n: és el valor de la força específica respecte NED .

A continuació el càlcul de la velocitat.

$$v_{eb}^n(+) = v_{eb}^n(-) + \left(f_{ib}^n + g_b^n(L_b(-), h_b(-)) - \left(\Omega_{en}^n(-) + 2 \cdot \Omega_{ie}^n(-)\right) \cdot v_{eb}^n(-)\right) \cdot t$$

On:

$v_{eb}^n(+)$: el nou valor de velocitat que volem calcular (m/s).

$v_{eb}^n(-)$: el valor anterior de velocitat (m/s).

$g_b^n(L_b(-), h_b(-))$: gravetat segons els paràmetres de latitud i altura respecte del geoide. S'ha calculat segons les equacions explicades a l'apartat B.5.3 d'aquest annex.

I finalment, el càlcul de la posició.

$$h_b(+) = h_b(-) - \frac{t}{2} \cdot \left(v_{eb,D}^n(-) + v_{eb,D}^n(+)\right)$$

$$L_b(+) = L_b(-) + \frac{t}{2} \cdot \left(\frac{v_{eb,N}^n(-)}{R_N\left(L_b(-)\right) + h_b(-)} + \frac{v_{eb,N}^n(+)}{R_N\left(L_b(-)\right) + h_b(+)}\right)$$

$$\lambda_b(+) = \lambda_b(-) + \frac{t}{2} \cdot \left(\frac{v_{eb,E}^n(-)}{\left(R_E\left(L_b(-)\right) + h_b(-)\right) \cdot \cos L_b(-)} + \frac{v_{eb,E}^n(+)}{\left(R_E\left(L_b(+)\right) + h_b(+)\right) \cdot \cos L_b(+)}\right)$$

On:

$h_b(+)$, $h_b(-)$: són l'altura respecte el geoide que volem trobar i la que ja tenim de la iteració anterior. (m)

$v^n_{eb,D}$: és la component down (valor de la 3a fila) de la velocitat calculada a l'apartat anterior. Si enlloc de una D és una N voldrà dir component nord (1a fila) i si és una E component est (2a fila). (m/s)

$L_b(+)$ i $L_b(-)$: valor de latitud que volem calcular i valor de latitud calculat a la iteració anterior. (rad)

$\lambda_b(+)$ i $\lambda_b(-)$: valor de longitud que volem calcular i valor de longitud calculat a la iteració anterior. (rad)

B.8. Equacions filtre de Kalman

En el cas de les equacions per implementar el filtre de Kalman passa el mateix que amb les equacions de navegació. Hi ha components que a la memòria no s'han explicat per no haver de fer-la molt feixuga.

L'algorisme del filtre de Kalaman es podia dividir en dos grans etapes: la Predicció i el Measurement Update. A l'annex també es seguirà l'ordre desenvolupat a la memòria on primer s'explicarà l'etapa de Predicció i després la de Measurement Update.

B.8.1. Etapa de Predicció

Recordem que la primera equació que s'aplicava en l'etapa de Predicció era la següent. En aquest pas es feia una predicció del vector d'estat.

$$\hat{x}_k^- = \Phi_{k-1} \cdot \hat{x}_{k-1}^+$$

Per poder aplicar aquesta equació el primer que hem de fer és calcular la matriu que conté el model matemàtic que fa la predicció. És l'anomenada matriu de transició Φ. L'explicació del seu càlcul ve a continuació.

$$\Phi_{INS} = \begin{pmatrix} I_3 - \Omega_{ie}^e \cdot t & 0_3 & 0_3 & 0_3 & C_b^e \cdot t \\ F_{21}^e \cdot t & I_3 - 2 \cdot \Omega_{ie}^e \cdot t & F_{23}^e \cdot t & C_b^e \cdot t & 0_3 \\ 0_3 & I_3 \cdot t & I_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & I_3 \end{pmatrix}$$

On:

I_3: matriu identitat 3x3.

Ω_{ie}^e: matriu rotació de la Terra explicada a la secció B.7.2 d'aquest annex.

C_b^e: matriu de rotació de Body a ECEF.

F_{21}^e: matriu antisimètrica.

$$f_{21}^e = -C_b^e \cdot f_{ib}^b$$

$$F_{21}^e = \begin{pmatrix} 0 & -f_{21}^e(3) & f_{21}^e(2) \\ f_{21}^e(3) & 0 & -f_{21}^e(1) \\ -f_{21}^e(2) & f_{21}^e(1) & 0 \end{pmatrix}$$

F_{23}^e:

$$F_{23}^e = -\frac{2 \cdot g_{ib}^e \cdot {r_{eb}^e}^T}{r_{eS}^e(L_b) \cdot |r_{eb}^e|}$$

g_{ib}^e: gravetat calculada segons s'explica a l'apartat B.5.2 d'aquest annex.

r_{eb}^e: posició a la que ens trobem.

$r_{eS}^e(L_b)$:

$$r_{eS}^e(L_b) = \frac{R_0}{\sqrt{1 - (e \cdot sinL_b)^2}} \cdot \sqrt{cosL_b{}^2 + (1 - e^2)^2 \cdot \sin L_b{}^2}$$

Un cop ja hem definit com es calcula la matriu de transició Φ_{INS} anem a recordar com estava format el vector d'estat.

$$x_{INS}^{e} = \begin{pmatrix} \delta\varphi_{eb}^{e} \\ \delta v_{eb}^{e} \\ \delta r_{eb}^{e} \\ b_a \\ b_g \end{pmatrix}$$

On:

$\delta\varphi_{eb}^{e}$: vector 3x1 que conté l'error en l'orientació.

δv_{eb}^{e}: vector 3x1 que conté l'error de velocitat a cada eix.

δr_{eb}^{e}: vector 3x1 que conté l'error de posició a cada eix.

b_a: vector 3x1 on hi ha el biaix dels tres acceleròmetres de la IMU.

b_g: vector 3x1 amb el biaix dels tres giroscopis de la IMU.

Un cop vist com es calcula el vector d'estat i la matriu de transició ja podem aplicar l'equació anterior.

$$\hat{x}_k^- = \Phi_{k-1} \cdot \hat{x}_{k-1}^+$$

La segona i última equació que s'ha d'aplicar a l'etapa de Predicció és la següent. És l'encarregada de calcular les variàncies de les variables del vector d'estat.

$$P_k^- = \Phi_{k-1} \cdot P_{k-1}^+ \cdot \Phi_{k-1}^T + Q_{k-1}$$

On:

P_k^-: matriu d'error de covariància de l'etapa de predicció.

P_{k-1}^+: matriu d'error de covariància calculada al final de l'etapa measurement Update de la iteració anterior. A la seva diagonal trobem les variàncies de les variables del vector d'estat.

Q_{k-1}: és la matriu de covariància del soroll. Es calcula de la següent manera.

$$Q_{INS} = \begin{pmatrix} S_{rg} \cdot I_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & S_{ra} \cdot I_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & S_{bad} \cdot I_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & S_{bgd} \cdot I_3 \end{pmatrix}$$

On:

S_{rg} i S_{ra}: són paràmetres de la desviació estàndard del soroll dels giroscopis i acceleròmetres respectivament.
S_{bad} i S_{bgd}: són paràmetres de la desviació estàndard del biaix dels giroscopis i acceleròmetres.

Aquests valors es poden determinar per característiques que dona el fabricant de la IMU. Els valors usats en aquest treball es troben en la implementació de codi Matlab (annex C, secció C.7).

La inicialització de la matriu P (abans de començar les iteracions) la trobarem a l'annex C, secció C.8.2.

Un cop ja sabem com es calculen totes aquests paràmetres ja podem aplicar la equació:

$$P_k^- = \Phi_{k-1} \cdot P_{k-1}^+ \cdot \Phi_{k-1}^T + Q_{k-1}$$

Així acabem l'etapa de Predicció.

B.8.2. Etapa de Measurement Update

Un cop vistes en detall les equacions de l'etapa de Predicció anem a fer el mateix amb les equacions de l'etapa de Measurement Update. La primera equació que s'aplica en aquesta secció és la que calcula la K de Kalman.

$$K_k = P_k^- \cdot H_k^T \cdot (H_k \cdot P_k^- \cdot H_k^T + R_k)^{-1}$$

On:
K_k: és la matriu K de Kalman que volem calcular.
H_k: és l'anomenada measurement matrix. Defineix com el vector amb els mesuraments fets pels sensors o altres sistemes de navegació varia amb el vector d'estat.

$$H = \begin{pmatrix} 0_3 & 0_3 & -I_3 & 0_3 & 0_3 \\ 0_3 & -I_3 & 0_3 & 0_3 & 0_3 \\ 0_3 & 0_3 & 0_3 & 0_3 & -I_3 \end{pmatrix}$$

R_k: és la matriu de covariància del soroll del vector amb els mesuraments. És la incertesa de la mesura del sensor que fa l'Update.

$$R = \begin{pmatrix} I_3 \cdot Gp & 0_3 & 0_3 \\ 0_3 & I_3 \cdot Gv & 0_3 \\ 0_3 & 0_3 & I_3 \cdot \sigma^2 \end{pmatrix}$$

On:

Gp: és la variància de les mesures de posició del GPS.

Gv: és la variància de les mesures de velocitat del GPS.

σ^2: la variància de les mesures dels Zero Updates.

Aquestes valors venen determinats per característiques que dona el fabricant. Els valors que hem fet servir es troben a l'annex C, secció C.7.

Un cop vist com es calculen les components de l'equació ja podem calcular la K de Kalman.

La següent equació a aplicar en l'etapa de Measurement Update és la següent. On es calculava el següent vector d'estat.

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \delta z_k^-$$

On:

$\hat{x}_k^+$: vector d'estat resultant després d'aplicar el filtre de Kalman.

δz_k^-: és la diferència entre la dada del GPS i la del INS. És aquesta matriu.

$$\begin{pmatrix} GNSSr_{eb}^e - r_{eb}^e \\ GNSSv_{eb}^e - v_{eb}^e \end{pmatrix}$$

Finalment, faltarà aplicar l'última equació del filtre. On es calcula la variància de cada una de les variables del vector d'estat.

$$P_k^+ = (I - K_k \cdot H_k) \cdot P_k^-$$

On:

P_k^+: és la matriu d'error de covariància resultant després d'aplicar el filtre de Kalman. A la seva diagonal si troben les variàncies de les variables del vector d'estat.

Aquest és l'últim i definitiu pas que s'ha d'aplicar per realitzar una iteració del filtre de Kalman.

C. ANNEX CODI INFORMÀTIC

C.1. Introducció

En aquest annex s'hi troben totes les funcions i programes escrits en codi Matlab necessaris per la implementació del INS incloent el filtre de Kalman. D'aquesta manera el lector del treball pot consultar ràpidament com s'ha implementant una determinada tasca.

L'estructura que s'ha seguit en aquest annex és molt semblant a la que s'ha seguit a la memòria. Es comença per funcions encaminades al càlcul de matrius de rotació i convertir dades d'un sistema de referència a un altre. Es segueix per les tècniques d'inicialització (leveling, magnètic heading). Després es continua amb la implementació de les equacions de navegació per cada sistema de referència amb que s'ha treballat. Tot seguit es mostra el codi del filtre de Kalman i finalment, es mostren els programes que fan tot el procés junt per cada sistema de referència.

C.2. Funcions necessàries per la lectura de dades

Primer de tot un recull de funcions molt bàsiques que tenen la funció de llegir les dades que ens dona la IMU.

C.2.1. Lectura acceleròmetres

```
function [ fbx, fby, fbz] = llegirForcaEspecifica(fila,matriuDades)
%Llegeix la matriu de resultats de sortida de la IMU. Retorna el valor de
la
%forca especifica x, y i z de la fila que li indiquem.
%Inputs:
%       matriuDades: matriu amb les dades que recull la IMU.
%       fila: fila de la matriu de la qual volem extreure les dades.
%Outputs:
%       fbx, fby, fbz: valors que mesuren els acceleròmetres x, y i z.

    fbx = matriuDades(fila, 5);
    fby = matriuDades(fila, 6);
    fbz = matriuDades(fila, 7);

end
```

C.2.2. Lectura giroscopis

```
function [wbx, wby, wbz] = llegirW(fila, matriuDades)
%Llegeix i torna les velocitats angulars mesurades per la IMU i que es
%troben a la matriu de Dades.
%Inputs:
%       matriuDades: matriu de les dades que recull la IMU.
%       fila: fila de la qual es vol extreure les dades.
%Outputs:
%       wbx, wby, wbz: valor dels giroscopis x, y i z respectivament(rad/s)

    wbx =  matriuDades(fila,2);
    wby =  matriuDades(fila,3);
    wbz =  matriuDades(fila,4);
end
```

C.2.3. Lectura magnetòmetres

```
function [mbx, mby, mbz] = llegirMagnetometre(fila, matriuDades )
%Llegeix i retorna els valors que donen els magnetometres de la IMU
%corresponens a la fila fila i a la matriu matriuDades.
%Inputs:
%       matriuDades: matriu de les dades que recull la IMU.
%       fila: fila de la qual es vol extreure les dades.
%Outputs:
%       mbx, mby, mbz: valor dels giroscopis x, y i z respectivament(rad/s)

    mbx = matriuDades(fila, 8);
    mby = matriuDades(fila, 9);
    mbz = matriuDades(fila, 10);

end
```

C.2.4. Lectura orientació

Llegeix el valor de l'orientació que ens donava la MTi que hem utilitzat en algunes parts d'aquest treball, recordem que la MTi era una AHRS i per tant, també calculava l'orientació.

```
function [anglex, angley, anglez] = llegirOrientacio(fila,matriuDades)
%Llegeix i retorna els valors que dona la MTi corresponents a l'orientació
%en que es troba.
%Inputs:
%       matriuDades: matriu de les dades que recull la IMU.
%       fila: fila de la qual es vol extreure les dades.
%Outputs:
%       anglex, angley, anglez: valor de l'orientació de Roll, Pitch i Yaw
%                               respectivament(°).

    anglex = matriuDades(fila, 11);
    angley = matriuDades(fila, 12);
```

```
    anglez = matriuDades(fila, 13);
end
```

C.3. Rotacions, transformacions i angles d'Euler

En aquest capítol ens centrem amb totes les funcions necessàries pels càlculs de les matrius de rotació que passen d'un sistema a un altre. Així com també funcions que transformen dades de posició representades en un sistema, en un altre.

C.3.1. Càlcul matriu de rotació de Body a NED.

Després d'obtenir els angles Roll, Pitch i Yaw amb l'ajuda de les tècniques de leveling i magnetci heading podem trobar la matriu de rotació entre Body i NED.

```
function [ C_ned_b ] = crearMatriuRotacioBody_NED( angleFi_nb,
angleTeta_nb, anglePsi_nb)
% Entrem els angles Roll, pitch i yaw (angleFi_nb, angleTeta_nb,
% anglePsi_nb)i fem la matriu de rotació de body a NED frame.
%Inputs:
%       angleFi_nb: angle de Roll. (rad)
%       angleTeta_nb: angle de Pitch. (rad)
%       anglePsi_nb: angle de Yaw. (rad)
%Outputs:
%       C_ned_b: matriude rotació de Body a NED.

    %Matriu de rotacio z
    C1 = [cos(anglePsi_nb) sin(anglePsi_nb) 0; -sin(anglePsi_nb)...
        cos(anglePsi_nb) 0; 0 0 1];
     %Matriu rotacio y
    C2 = [cos(angleTeta_nb) 0 -sin(angleTeta_nb); 0 1 0;
      sin(angleTeta_nb)...
        0 cos(angleTeta_nb)];
    %Matriu rotacio x
    C3 = [1 0 0; 0 cos(angleFi_nb) sin(angleFi_nb); 0 -...
      sin(angleFi_nb)...
        cos(angleFi_nb)];

    C_ned_b_not = C3*C2*C1;

      %Matriu rotacio per passar de body a NED
    C_ned_b = C_ned_b_not';

end
```

C.3.2. Càlcul matriu de rotació de ECI a ECEF

```
function [C_ecef_eci] = crearMatriuECI_ECEF(t)
%Crea una matriu per passar de ECI a ECEF segons el temps que ha passat des
%de l'inici dels càlculs moment en que els dos frame coincidien.
%Inputs:
%       t: temps que ha passat des de l'inici. És la suma del total
d'iteracions (s)
%Outputs:
%       C_ecef_eci: Matriu que passa de ECI a ECEF frame.
    w_ie = 7.292115e-5;
    t0 = 0;

    C_ecef_eci = [cos(w_ie*(t-t0)) sin(w_ie*(t-t0)) 0; -sin(w_ie*(t-t0))...
        cos(w_ie*(t-t0)) 0; 0 0 1];

end
```

C.3.3. Càlcul matriu de rotació de ECI a NED

```
function [ C_n_i ] = crearMatriuECI_NED( t, L, lambda )
%Crea matriu de rotació de ECI a NED frame.
%Inputs:
%       t: temps acomulat des de l'inici (s).
%       L: latitud (rad).
%       lambda: longitud (rad).
%Outputs:
%       C_n_i: matriu rotació ECI-NED frame.
    w_ie = 7.292115e-5;
    t0 = 0;

    C_n_i = [-sin(L)*cos(lambda + w_ie*(t - t0)) -sin(L)*sin(lambda ...
        + w_ie*(t-t0)) cos(L); -sin(lambda + w_ie*(t - t0)) ...
        cos(lambda + w_ie*(t - t0)) 0; -cos(L)*cos(lambda + w_ie*(t-t0))...
        -cos(L)*sin(lambda + w_ie*(t-t0)) -sin(L)];

End
```

C.3.4. Càlcul matriu de rotació de ECEF a NED

```
function [ C_ned_ecef ] = crearMatriuRotacioECEF_NED( L, lambda)
%Retorna la matriu de transformació de ECEF a NED.
%Inputs:
%       L: latitud (rad)
%       lambda: longitud (rad)
%Outputs:
%       C_ned_ecef: matriu rotació de ECEF a NED.

    C_ned_ecef = [-sin(L)*cos(lambda) -sin(L)*sin(lambda) cos(L);...
        -sin(lambda) cos(lambda) 0; -cos(L)*cos(lambda) -
cos(L)*sin(lambda)...
```

```
        -sin(L)];

End
```

C.3.4. Càlcul matriu de rotació per passar de NED a ECEF

```
function [ C_ecef_ned ] = crearMatriuRotacioNED_ECEF( L, lambda)
%Retorna la matriu de transformació de NED a ECEF.
%Inputs:
%       L: latitud (°)
%       lambda: longitud (°)
%Outputs:
%       C_ecef_ned: matriu rotació de NED a ECEF.

      C_ecef_ned = [ -sind(L)*cosd(lambda) -sind(lambda) -...
      cosd(L)*cosd(lambda);...
        -sind(L)*sind(lambda) cosd(lambda) -cosd(L)*sind(lambda); ...
        cosd(L) 0 -sind(L)];

end
```

C.3.5. Càlcul matriu de rotació per passar de NED a ECI

```
function [ C_eci_ned ] = crearMatriuRotacioNED_ECI( Lg, lambdag, t)
%A partir de les dades de longitud (lambda) i latitud (L) ens retorna les
matriu de
%rotació que passa de NED a ECI frame.
%Inputs:
%       L: latitud (°)
%       lambda: longitud (°)
%       t: temps acomulat des de l'inici (s)
%Outputs:
%       C_eci_ned: matriu rotació de NED a ECI.
    w_ie = 7.292115e-5;
    L = degtorad(Lg);
    lambda = degtorad(lambdag);

    C_eci_ned = [-sin(L)*cos(lambda+w_ie*(t)) -sin(lambda+w_ie*(t))...
        -cos(L)*cos(lambda+w_ie*(t)); -sin(L)*sin(lambda+w_ie*(t))...
      cos(lambda+w_ie*(t)) -cos(L)*sin(lambda+w_ie*(t)); cos(L) 0 -sin(L)];

end
```

C.3.6. Passar una posició del sistema local de navegació a ECEF

Donada una posició en coordenades de latitud, longitud i h en calcula la seva equivalent en coordenades ECEF (x, y i z).

```
function [x_e_eb, y_e_eb, z_e_eb] = posicioNED_ECEF(L, lambda, h)
% Donada una posició de la Terra amb latitud(L),longitud (lambda) i altura
% (h)es retorna la posició segons el sistema ECEF amb coordenades
% cartesianes x, y, z.
%Inputs:
%       L: latitud (rad).
%       lambda: longitud (rad).
%       h: altura (m).
%Outputs:
%       x_e_eb,y_e_eb,z_e_eb: coordenades posició ECEF (m)

    R0 = 6378137;
    e = 0.081819191;

    Re = R0/(sqrt(1-e^2*sin(L)^2));

    x_e_eb = (Re + h)*cos(L)*cos(lambda);
    y_e_eb = (Re + h)*cos(L)*sin(lambda);
    z_e_eb = ((1 - e^2)*Re + h)*sin(L);

end
```

C.3.7. Passar una posició de ECEF a sistema local de navegació.

Mètode matemàtic per passar de coordenades ECEF a coordenades del sistema local de navegació. S'anomena Borkowski.

```
function [L, lambda, h] = Borkowski(x_e_eb, y_e_eb, z_e_eb)
%Mètode per passar de coordenades ECEF (cartesianes) a coordenades NED
%(pseudoesfèriques). S'anomena Borkowski.
%Input:
%       x_e_eb, y_e_eb, z_e_eb: coordenades ECEF. (m)
%Outputs:
%       L: latitud (rad).
%       lambda: longitud (rad)
%       h: altura sobre el nivell del mar (m).

    Ro = 6378137;
    e = 0.08181919;

    lambda = atan2(y_e_eb, x_e_eb);

    B = sqrt(x_e_eb^2 + y_e_eb^2);

    E = (sqrt(1 - e^2)*abs(z_e_eb) - e^2*Ro)/B;

    F = (sqrt(1 - e^2)*abs(z_e_eb) + e^2*Ro)/B;

    P = 4/3*(E*F + 1);

    Q = 2*(E^2 - F^2);

    D = P^3 + Q^2;
```

```
    V = (D^(1/2) - Q)^(1/3) - (D^(1/2) + Q)^(1/3);

    G = 1/2*(sqrt(E^2 + V) + E);

    T = sqrt(G^2 + (F - V*G)/(2*G - E)) - G;

    L = sign(z_e_eb)*atan((1 - T^2)/(2*T*sqrt(1 - e^2)));

    h = (B - Ro*T)*cos(L) + (z_e_eb - sign(z_e_eb)*Ro*sqrt(1-e^2))*sin(L);

end
```

C.3.8. Calcular angles d'Euler

Quan tenim una matriu de rotació entre dos sistemes, és possible trobar els angles d'Euler que hi ha entre aquells dos sistemes segons els valors que té la matriu. Per fer-ho s'utilitzen les fórmules d'aquesta funció

```
function [Roll, Pitch, Yaw] = calcularAnglesEuler( C_n_b )
%Calcula els angles d'Euler utilitzant la matriu C_ned_b. Troba el Roll,
%Pitch i Yaw respecte NED frame.
%Inputs:
%       C_ned_b: matriu de transformació de Body a NED frame.
%Outputs:
%       Roll: angle Roll (graus)
%       Pitch: angle Pitch (graus)
%       Yaw: angle Yaw (graus)

    Roll_rad = atan2(C_n_b(3,2), C_n_b(3,3));
    Pitch_rad = asin(C_n_b(3,1));
    Yaw_rad = atan2(C_n_b(2,1),C_n_b(1,1));

    Roll = radtodeg(Roll_rad);
    Pitch = radtodeg(Pitch_rad);
    Yaw = radtodeg(Yaw_rad);

End
```

C.4. Càlcul gravetat

En aquest capítol es mostren les funcions i mètodes que s'han implementat per calcular la gravetat segons el sistema de referència en que es treballava.

C.4.1. Gravetat ECI

Calcula la gravetat respecte el sistema de coordenades ECI.

```
function [g] = gravetat_ECI(rx, ry, rz)
%Dona la gravetat segons el punt de la Terra on es troba
%Inputs:
%       rx, ry, rz: posició en coordenades ECI (m).
%Outputs:
%       g: gravetat (m/s^2).

    R0 = 6378137;
    mu = 3.986004418E14;
    J_2 = 1.082627E-3;
    norma = sqrt(rx^2 + ry^2 + rz^2);

    r_i_ib = [rx; ry; rz];

    vector_calcul_x = (1 - 5*(rz/norma)^2)*rx;
    vector_calcul_y = (1 - 5*(rz/norma)^2)*ry;
    vector_calcul_z = (3 - 5*(rz/norma)^2)*rz;

    vector_calcul = [vector_calcul_x; vector_calcul_y;
vector_calcul_z];

    gamma = -mu/norma^3*(r_i_ib + (3/2)*J_2*(R0^2/norma^2)*vector_calcul);

    g = gamma;

end
```

C.4.2. Gravetat ECEF

Calcula la gravetat en coordenades ECEF. En aquest cas la gravetat, a més de la seva pròpia acceleració cap al centre de la Terra, conté una component de força centrípeta degut a a la rotació de la Terra.

```
function [g, gamma] = gravetat_ECEF(rx, ry, rz)
%Dona la gravetat segons el punt de la Terra on es troba en coordenades
%ECEF.
%Inputs:
%       rx, ry, rz: posició en coordenades respecte ECEF (m).
%Outputs:
%       g: força de la gravetat amb la componenet d'acceleració
%       centrípeta (m/s^2).
%       gamma: acceleració de la gravetat (m/s^2).

    R0 = 6378137;
    mu = 3.986004418E14;
```

```
    J_2 = 1.082627E-3;
    w_ie = 7.292115e-5;
    norma = sqrt(rx^2 + ry^2 + rz^2);

    r_e_eb = [rx; ry; rz];

    vector_calcul_x = (1 - 5*(rz/norma)^2)*rx;
    vector_calcul_y = (1 - 5*(rz/norma)^2)*ry;
    vector_calcul_z = (3 - 5*(rz/norma)^2)*rz;

    vector_calcul = [vector_calcul_x; vector_calcul_y; vector_calcul_z];

    gamma = -mu/norma^3*(r_e_eb + (3/2)*J_2*(R0^2/norma^2)*vector_calcul);

    M = [1 0 0; 0 1 0; 0 0 0];
    g = gamma + w_ie^2*M*r_e_eb;

end
```

C.4.3. Gravetat NED

Calcula la gravetat i la retorna en coordenades NED.

```
function [G] = gravetatNED( L, h )
%Retorna la gravetat en el NED frame.
%Inputs:
%       L: coordenada de longitud de la posició on ens trobem (rad).
%       h: altura sobre el nivell del mar a que ens trobem. (m).
%Outputs:
%       G: Vector 3x1 amb les componenets Nord, Est i Down de la gravetat.
%       (m/s^2)

    Ro = 6378137;
    Rp = 6356752.31425;
    e = 0.0818191908425;
    f = 1 / 298.257223563;
    mu = 3.986004418E14;
    w_ie = 7.292115E-5;

    g0 = 9.7803253359*(1 + 0.001931853*(sin(L))^2)/sqrt(1 -
e^2*(sin(L))^2);

    gN = -8.08E-9*h*sin(2*L);

    gD = g0*(1-2/Ro*(1 + f*(1 - 2*sin(L)^2) + (w_ie^2*Ro^2*Rp)/mu)*h + ...
        (3/Ro^2)*h^2);

    G = [gN; 0; gD];

end
```

C.5. Funcions inicialització

En aquest capítol de l'annex hi trobem totes les funcions necessaris i utilitzades per poder fer la inicialització del INS (leveling, magnètic heading...). Així com també les necessàries per poder fer un calibratge de dades per aconseguir millor precisió amb la funció magnètic heading.

C.5.1. Leveling

Funció encarregada de calcular el Roll i Pitch en que es troba la IMU. Per calcular-ho es fa la mitjana de 15 iteracions per poder reduir l'efecte que provoca el soroll.

```
function [angleFi_nb, angleTeta_nb, angleFi_nb_graus, angleTeta_nb_graus] =
leveling(matriuDades )
%LEVELING: Aplica el procés de leveling per trobar els angles  angleFi_nb i
%angleTeta_nb. Els torna amb radians i graus.
%Inputs:
%       matriuDades: fitxer de les dades mesurades per la IMU.
%Outputs:
%       angleFi_nb: angle Fi (rad)
%       angleTeta_nb: angle Teta (rad)
%       angleFi_nb_graus: angle Fi (graus)
%       angleTeta_nb_graus: angle Teta (graus)

    %Sumem els 15 primers valors de la força específica per fer-ne la
    %mtijana.
    f_b_ibx_suma = 0;
    f_b_iby_suma = 0;
    f_b_ibz_suma = 0;

    for fila=1:1:15
        [f_b_ibx, f_b_iby, f_b_ibz] =
      llegirForcaEspecifica(fila,matriuDades);
        f_b_ibx_suma = f_b_ibx_suma + f_b_ibx;
        f_b_iby_suma = f_b_iby_suma + f_b_iby;
        f_b_ibz_suma = f_b_ibz_suma + f_b_ibz;
    end

    %En fem la mitjana
    f_b_ibx_mitja = f_b_ibx_suma/15;
    f_b_iby_mitja = f_b_iby_suma/15;
    f_b_ibz_mitja = f_b_ibz_suma/15;

    %Apliquem les equacions del llibre per trobar Fi i Teta. Els
      calculem
    %en rad i graus.

    angleFi_nb = atan2(-f_b_iby_mitja, -f_b_ibz_mitja);
    angleTeta_nb = atan(f_b_ibx_mitja/(sqrt(f_b_iby_mitja^2 + ...
```

```
        f_b_ibz_mitja^2)));

    angleFi_nb_graus = atan2d(-f_b_iby_mitja, -f_b_ibz_mitja);
    angleTeta_nb_graus = atand(f_b_ibx_mitja/(sqrt(f_b_iby_mitja^2 + ...
        f_b_ibz_mitja^2)));

end
```

C.5.2. Magnetic Heading

Càlcul de l'angle Yaw mitjançant la funció magnetic Heading.

```
function [ anglePsi_nb, anglePsi_nb_graus ] = magneticHeading(matriuDades )
% Retorna l'orientació yaw inicial segons la posició del nord. La retorna
%amb radians i graus. Utilitza les dades de la matriuDades.
%Inputs:
%       matriuDades: matriu de dades recollides pels sensors de la IMU.
%Outputs:
%       anglePsi_nb: angle Psi (Yaw) en (rad).
%       anglePsi_nb_graus: angle Psi (Yaw) en (graus).

    %Declinació magnètica a la Bisbal d'Empordà (0.583333 graus) o per ser
    %exactes 32/60. S'ha buscat a internet. La passem a radiants.
    declinacio_magnetica = 32/60;
    declinacio_magnetica_rad = declinacio_magnetica*2*pi/360;

    %Per fer el magnetic Heading primer necessitem els valors de leveling.
    [angleFi_nb, angleTeta_nb, angleFi_nb_graus, angleTeta_nb_graus] = ...
        leveling(matriuDades );

    mbx_suma = 0;
    mby_suma = 0;
    mbz_suma = 0;

    %Fem la mitjana de les 15 primeres dades dels magnetòmetres. D'aquesta
    %manera corregim el possible soroll.
    for fila=1:1:15
        [mbx, mby, mbz] = llegirMagnetometre(fila, matriuDades);
        mbx_suma = mbx_suma + mbx;
        mby_suma = mby_suma + mby;
        mbz_suma = mbz_suma + mbz;
    end

    mbx_mitja = mbx_suma/15;
    mby_mitja = mby_suma/15;
    mbz_mitja = mbz_suma/15;

    %Apliquem les equacions del llibre per fer el magnetic Heading. El
    %calculem en radians i en graus.
    x = -mby_mitja*cos(angleFi_nb) + mbz_mitja*sin(angleFi_nb);
    y = mbx_mitja*cos(angleTeta_nb) + mby_mitja*sin(angleFi_nb)*...
        sin(angleTeta_nb) + mbz_mitja*cos(angleFi_nb)*sin(angleTeta_nb);
    anglePsi_mb = atan2(x,y);

    %Fem la correcció de la declinació magnetica per trobar el nord
```

```
    %geogràfiB.
    anglePsi_nb = anglePsi_mb + declinacio_magnetica_rad;
    anglePsi_nb_graus = 360*anglePsi_nb/(2*pi);
   end
```

C.5.3. Calibratge

Es tracte d'un petit script on es crida la funció TWOSTEP per trobar les matrius de soft i hard iron. A més aquests script també dibuixa les dades dels magnetòmetres abans de calibrar-se i després de calibrar-se.

```
% CALIBRATGE DADES MAGNETÒMETRES
%
% Fa el calibratge de les dades dels magnetòmetres. Troba les dades
% calibrades i les matrius de soft i hard iron.

%Agafa el fitxer de dades que volem.
matriuDades = fitxerDades;

%Posa les dades corresponents als magnetòmetres (columnes 8, 9 i 10) en una
%matriu B (3xn). Agafa les dades de 10 en 10 perquè es puguin representar
%millor els plots.
[n, c] = size(matriuDades);
Ba = matriuDades(1:10:n, [8,9,10]);
B = Ba';

%Represento les dades sense calibrar
figure(1)
plot3(B(1,:),B(2,:),B(3,:),'.b')
hold on

%Creo matriu (3xn) amb el valor de covariança de les dades.
Sigma_noise = ones(3,length(B))*0.1;

%Camp magnètic de la Terra a BCN.
H = [24.96; 0.16; 37.98];

%Cridem funció TWOSTEP que calcula les matrius de soft i hard iron.
[D_est,b_est,n,Cov_est]=TWOSTEP_estimate(B,H, Sigma_noise);

%Amb les matrius que acabem d'aconseguir calibrem les dades anteriors.
for i=1:length(B)
    C(1:3,i)=(eye(3)+D_est)*B(1:3,i)-b_est;
end

%Les representem per poder-les comparar amb les dades sense calibrar
plot3(C(1,:),C(2,:),C(3,:),'or');

axis equal
grid on
```

C.5.4. Calibratge de les dades

Es tracte d'una funció que realitza el calibratge de les dades dels magnetòmetres gràcies a les matrius de soft i hard iron calculades amb la funció TWOSTEP.

```
function [ matriuDadesCorregida ] = calibratgeDades( matriuDades, D_est,
b_est )
%Realitza el calibratge de les dades dels magnetometres.
%Input:
%       matriuDades: matriu de dades del sensor.
%Output:
%       matriuDadesCorregida: matriu de dades del sensor amb les dades dels
%                             magnetometres calibrades.

    [n, s] = size(matriuDades);
    matriuDadesCorregida = matriuDades;

    for i=1:1:n,

        F = matriuDades(i, 8:10)';
        C=(eye(3)+D_est)*F-b_est;
        matriuDadesCorregida(i, 8:10) = C';
    end

end
```

C.5.5. Velocitat Inicial ECI

En el moment que volem inicialitzar la velocitat, si estem treballant amb el sistema ECI, és necessari aplicar aquesta funció. Com que el sistema de coordenades ECI no rota amb la Terra quan estem quiets ja tenim una velocitat lineal respecte el sistema ECI degut a la rotació de la Terra i a la posició en que ens trobem. Per tant, alhora d'inicialitzar la velocitat amb ECI, sempre hem de tenir en compte la velocitat provocada per la rotació de la Terra.

```
function [V_i_ib] = velocitatInicialECI( R_i_ib, V_e_eb )
%Troba la velocitat inicial respecte ECI segons el punt de la Terra on %es
troba. ATENCIÓ: Aquesta funció només serveix per l'instant incial, %ja que
suposem que en aquest moment ECI i ECEF estan alineats. Si
%voléssim que servis per qualsevol instant hauriem de calcular la %matriu
C_eci_ecef segons el temps.
%
%Inputs:
%       R_i_ib: posició respecte ECI (m)
%       V_e_eb: velocitat inicial respecte ECEF (m/s)
%Outputs:
%       V_i_ib: velocitat inicial respecte ECI. (m/s)
    w_ie = 7.292115e-5;
```

```
    Matriu_e_ie = [0 -w_ie 0; w_ie 0 0; 0 0 0];
    C_eci_ecef = [1 0 0; 0 1 0; 0 0 1];

    V_i_ib = C_eci_ecef*(V_e_eb + Matriu_e_ie*R_i_ib);
end
```

C.6. Equacions de navegació

En aquest apartat es trobaran les funcions que contenen les equacions de navegació del INS implementades. També es podrà trobar alguna funció necessària en els càlculs.

C.6.1. Equacions de navegació respecte ECI

Les equacions de navegació respecte el sistema de referència ECI implementades en codi Matlab.

```
function [R_i_ib, V_i_ib, C_eci_b ] = equacionsNavegacioECI( t,
R_i_ib_vella, V_i_ib_vella, C_eci_b_vella, F_b_ib, W_b_ib )
%Conté les equacions de navegació per trobar la posició, velocitat i
%orientació respecte el sistema ECI.
%Inputs:
%       t: temps de durada d'una iteració. (s)
%       R_i_ib_vella: posició resultant de la iteració anterior (m)
%       V_i_ib_vella: velocitat de la iteració anterior (m/s)
%       C_eci_b_vella: matriu de rotació anterior.
%       F_b_ib: lecutra dels acceleròmetres
%       W_b_ib: lectura giroscopis
%Outputs:
%       R_i_ib: nova estimació de la posició. (m)
%       V_i_ib: nova estimació de la velocitat.(m/s)
%       C_eci_b: nova matriu de rotació.

    w_ie = 7.292115e-5;
    wx_b_ib = W_b_ib(1);
    wy_b_ib = W_b_ib(2);
    wz_b_ib = W_b_ib(3);

    %Càlcul de diverses matrius necessàries per dur a terme els càlculs
    MatriuAcce = [0 -wz_b_ib*t wy_b_ib*t; wz_b_ib*t 0 -wx_b_ib*t;...
        -wy_b_ib*t wx_b_ib*t 0];
    Modul_Alfa = sqrt((wx_b_ib*t)^2 + (wy_b_ib*t)^2 + (wz_b_ib*t)^2);

    %Matriu orientació, equacions optimització
    if Modul_Alfa>1.E-8
        C_menys_mes = eye(3) + sin(Modul_Alfa)/Modul_Alfa*MatriuAcce +...
            (1 - cos(Modul_Alfa))/((Modul_Alfa)^2)*MatriuAcce*MatriuAcce;
    else
        C_menys_mes = eye(3) + MatriuAcce;
    end
```

```
    C_eci_b = C_eci_b_vella*C_menys_mes;


    %Càlcul matriu orientació per calcular F_i_ib
    if Modul_Alfa>1.E-8
        C_eci_b_ave = C_eci_b_vella*(eye(3) + (1 - cos(Modul_Alfa))...
            /Modul_Alfa^2*MatriuAcce + (1 - sin(Modul_Alfa)/Modul_Alfa)...
            /Modul_Alfa^2*MatriuAcce*MatriuAcce);
    else
        C_eci_b_ave = C_eci_b_vella;
    end

    F_i_ib = C_eci_b_ave*F_b_ib;

    %Troba la gravetat segons el punt de la Terra on està.
    [G] = gravetat_ECI(R_i_ib_vella(1), R_i_ib_vella(2), R_i_ib_vella(3));

    %Càlculs acceleració, velocitat i posició
    A_i_ib = F_i_ib + G;

    V_i_ib = V_i_ib_vella + A_i_ib*t;

    R_i_ib = R_i_ib_vella + V_i_ib*t - A_i_ib*(t^2)/2;

End
```

C.6.2. Equacions de navegació respecte ECEF

Les equacions de navegació respecte el sistema de coordenades ECEF implementades en Matlab.

```
function [ R_e_eb, V_e_eb, C_ecef_b] = equacionsNavegacioECEF(t,
R_e_eb_vella, V_e_eb_vella, C_ecef_b_vella, F_b_ib, W_b_ib )
%Conté les equacions navegació per trobar posició, velocitat i orientació
%respecte el sistema ECEF.
%Inputs:
%       t: temps de durada d'una iteració (s)
%       R_e_eb_vella: posició resultant de la iteració anterior (m)
%       V_e_eb_vella: velocitat resultant de la iteració anterior (m/s)
%       C_ecef_b_vella: matriu de rotació resultant de la iteració
%       anterior (rad)
%       F_b_ib: lectura dels acceleròmetres per fer els nous càlculs.
%       W_b_ib: lectura dels giroscopis.
%Outputs:
%       R_e_eb: nova posició resultant dels càlculs (m)
%       V_e_eb: nova velocitat resultant dels càlculs (m/s)
%       C_e_eb: nova matriu de rotació.

        w_ie = 7.292115e-5;
        wx_b_ib = W_b_ib(1);
        wy_b_ib = W_b_ib(2);
        wz_b_ib = W_b_ib(3);
```

```
        %Calculem diverses matrius necessàries per dur a terme els càlculs.
        MatriuAcce = [0 -wz_b_ib*t wy_b_ib*t; wz_b_ib*t 0 -wx_b_ib*t;...
            -wy_b_ib*t wx_b_ib*t 0];
        MatriuVelocitatTerra = [0 -w_ie 0; w_ie 0 0; 0 0 0];
        Modul_Alfa = sqrt((wx_b_ib*t)^2 + (wy_b_ib*t)^2 + (wz_b_ib*t)^2);
        C_earth = [cos(w_ie*t) sin(w_ie*t) 0; -sin(w_ie*t) cos(w_ie*t)...
            0; 0 0 1];


        %Matriu rotació, equacions optimització
        if Modul_Alfa>1.E-8
            C_menys_mes = eye(3) + sin(Modul_Alfa)/Modul_Alfa*MatriuAcce...
                + (1 - cos(Modul_Alfa))/((Modul_Alfa)^2)...
                *MatriuAcce*MatriuAcce;
        else
            C_menys_mes = eye(3) + MatriuAcce;
        end

        C_ecef_b = C_earth*C_ecef_b_vella*C_menys_mes;

        %Força específica
        if Modul_Alfa>1.E-8
            C_ecef_b_ave = C_ecef_b_vella*(eye(3) + (1 -
            cos(Modul_Alfa))...
                /Modul_Alfa^2*MatriuAcce + (1 - sin(Modul_Alfa)...
                /Modul_Alfa)/Modul_Alfa^2*MatriuAcce*MatriuAcce)...
                - 0.5*MatriuAcce*C_ecef_b_vella;
        else
            C_ecef_b_ave = C_ecef_b_vella - 0.5*MatriuAcce*C_ecef_b_vella;
        end

        %Passem la força específica a ECEF frame.
        F_e_ib = C_ecef_b_ave*F_b_ib;

        %Calculem la gravetat en funció de la posició
        [G, gamma] = gravetat_ECEF(R_e_eb_vella(1), R_e_eb_vella(2),...
            R_e_eb_vella(3));

        %Càlcul acceleració:
        A_e_eb = F_e_ib + G - 2*MatriuVelocitatTerra*V_e_eb_vella;

        %Càlcul velocitat
        V_e_eb = V_e_eb_vella + A_e_eb*t;

        %Càlcul posició respecte ECEF
        R_e_eb = R_e_eb_vella + V_e_eb_vella*t + A_e_eb*(t^2)/2;

end
```

C.6.3. Equacions de navegació en el sistema local de navegació

Les equacions de navegació implementades en el sistema local de navegació.

```
function [L, lambda, h, V_n_eb, C_ned_b] = equacionsNavegacioNED(t,
L_vella, lambda_vella, h_vella, V_n_eb_vella, C_ned_b_vella, F_b_ib, W_b_ib
)
%Conté les equacions de navegació necessaries per trobar l'orientació, la
%posició i la velocitat. Estant implementades utilitzant el NED frame
%Inputs:
%       t: interval de temps d'una iteració (s).
%       L_vella: latitud iteració anterior (rad).
%       lambda_vella: longitud iteració anterior (rad).
%       h_vella: altura respecte el geoda. (m)
%       V_n_eb_vella: velocitat iteració anterior (m/s).
%       C_ned_b_vella: matriu de rotació de la iteració anterior.
%       F_b_ib: lectura de la força específica dels acceleròmetres. (m/s^2)
%       W_b_ib: lectura de la velocitat angular dels giroscopis. (rad/s)
%Outputs:
%       L: latitud (rad).
%       lambda: longitud (rad).
%       h: altura respecte el geoda (m).
%       V_n_eb: velocitat lineal (m/s).
%       C_ned_b: matriu de rotació.

    w_ie = 7.292115e-5;
    wx_b_ib = W_b_ib(1);
    wy_b_ib = W_b_ib(2);
    wz_b_ib = W_b_ib(3);

    %Velocitats angular (omega_b_ib), acceleracions angulars (alfa_b_ib),
    %modul de les acceleracions angulars (Modul_alfa).

    omega_b_ib = [wx_b_ib; wy_b_ib; wz_b_ib];
    alfa_b_ib = omega_b_ib*t;
    Modul_Alfa = sqrt(alfa_b_ib'*alfa_b_ib);

    %MatriuAcce (Omega_b_ib en el llibre), M_n_ie.
    MatriuAcce = [0 -wz_b_ib*t wy_b_ib*t; wz_b_ib*t 0 -wx_b_ib*t; ...
        -wy_b_ib*t wx_b_ib*t 0];
    M_n_ie = w_ie*[0 sin(L_vella) 0; -sin(L_vella) 0 -cos(L_vella);...
        0 cos(L_vella) 0];

    %Calcular radi de curvatura del merdià (Rn) i radi de curvatura del
    %paral·lel (Re) segons la posició en que ens trobem de la Terra
    [ Rn_vell, Re_vell ] = calcularRn_Re( L_vella );

    %Càlcul de Omega_n_en en el llibre. Aquí Matriu_n_en
    w_e_enx_vella = V_n_eb_vella(2)/(Re_vell + h_vella);
    w_e_eny_vella = -V_n_eb_vella(1)/(Rn_vell + h_vella);
    w_e_enz_vella = -V_n_eb_vella(2)*tan(L_vella)/(Re_vell + h_vella);

    W_e_en_vella = [w_e_enx_vella; w_e_eny_vella; w_e_enz_vella];

    Matriu_n_en_vella = [0 -w_e_enz_vella w_e_eny_vella; w_e_enz_vella 0...
        -w_e_enx_vella; w_e_eny_vella w_e_enx_vella 0];
```

```
%*******Càlcul força específica (F_n_ib)********

%equacions d'optimització pel càlcul de la matriu orientació per
%calcular F_n_ib
if Modul_Alfa>1.E-8
    C_ned_b_ave = C_ned_b_vella*(eye(3) + (1 - ...
        cos(Modul_Alfa))/Modul_Alfa^2*MatriuAcce +...
        (1 - sin(Modul_Alfa)/Modul_Alfa)/Modul_Alfa^2 ...
        *MatriuAcce*MatriuAcce) - 0.5*(M_n_ie  + Matriu_n_en_vella)*...
        C_ned_b_vella*t;
else
    C_ned_b_ave = C_ned_b_vella - 0.5*(M_n_ie + Matriu_n_en_vella)...
        *C_ned_b_vella*t;
end

%Força específica respecte NED frame.
F_n_ib = C_ned_b_ave*F_b_ib;

%Gravetat segons posició de la Terra on ens trobem.
[G] = gravetatNED( L_vella, h_vella );

%Acceleració
A_n_eb = F_n_ib + G - (Matriu_n_en_vella + 2*M_n_ie)*V_n_eb_vella;

%Velocitat
V_n_eb = V_n_eb_vella + A_n_eb*t;

%Altura (h), longitud (L) i latitud (lambda).
h = h_vella - t/2*(V_n_eb_vella(3) + V_n_eb(3));
L = L_vella + t/2*(V_n_eb_vella(1)/(Rn_vell + h_vella) + V_n_eb(1)...
    /(Rn_vell + h));

[ Rn, Re ] = calcularRn_Re(L);
lambda = lambda_vella + t/2*(V_n_eb_vella(2)/((Re + h_vella)*...
    cos(L_vella)) + V_n_eb(2)/((Re + h)*cos(L)));

%***********Càlcul Matriu orientació************

%Tornem a calcular una matriu Matriu_n_en, però amb la nova velocitat
%acabada de calcular.

w_e_enx = V_n_eb(2)/(Re + h);
w_e_eny = -V_n_eb(1)/(Rn + h);
w_e_enz = -V_n_eb(2)*tan(L)/(Re + h);

W_e_en = [w_e_enx; w_e_eny; w_e_enz];
Matriu_n_en = [0 -w_e_enz w_e_eny; w_e_enz 0 -w_e_enx; w_e_eny w_e_enx
  0];



%Equacions optimització per la matriu d'orientació.

if Modul_Alfa>1.E-8
    C_menys_mes = eye(3) + sin(Modul_Alfa)/Modul_Alfa*MatriuAcce...
        + (1 - cos(Modul_Alfa))/((Modul_Alfa)^2)*MatriuAcce*MatriuAcce;
else
    C_menys_mes = eye(3) + MatriuAcce;
end
```

```
    %Nova matriu d'orientació (C_ned_b)
    C_ned_b = (eye(3) - (MatriuAcce + 0.5*Matriu_n_en_vella +...
        0.5*Matriu_n_en)*t)*C_ned_b_vella*C_menys_mes;

End
```

C.6.4. Càlcul radi de curvatura meridià i paral·lel.

Per la implementació de les equacions de navegació utilitzant el sistema local de navegació necessitem la següent funció que calcula el radi de curvatura del paral·lel i del meridià. Ambdós necessaris per realitzar els càlculs.

```
function [ Rn, Re ] = calcularRn_Re( L )
%Calcula Rn (radi curvatura del meridià) i Re (radi de curvatura del
paral·lel).
%Inputs:
%       L: latitud de la posició en que ens trobem (rad)
%Outputs:
%       Rn: radi de curvatura del meridià. O sigui de N a S.
%       Re: radi de curvatura del paral·lel. O sigui de E a O.

    Ro = 6378137;
    e = 0.0818191908425;

    Rn = Ro*(1-e^2)/((1-(e^2)*(sin(L))^2)^(3/2));
    Re = Ro/sqrt(1 - (e^2)*(sin(L))^2);

end
```

C.7. Filtre de Kalman

En aquest capítol mostrem el codi necessari per implementar el filtre de Kalman. El codi integra un INS, un GPS i hi afegeix zero Updates. El filtre està separat en dos etapes: la de Predicció i la de Measurement update.

C.7.1. Etapa de Predicció

Funció que executa l'etapa de predicció del filtre de Kalman.

```
function [P_propagada, x_propagada] = filtreKalman_I_Prediccio( t,...
    C_e_b_vella, f_b_ib, r_e_eb_vella, P_vella)
%FILTREKALMAN_I_PREDICCIO: Executa l'etapa de predicció del filtre de
%Kalman.
%Inputs:
%       t: temps d'iteració (s).
%       C_e_b_vella: Matriu de rotació de body a ECEF.
%       f_b_ib: força específica que llegeix la IMU.
%       r_e_eb_vella: posició en coordenades ECEF (m).
%       P_vella: matriu de covariança anterior.
%
%Outputs:
%       x_propagada: vector d'estat.
%       P_propagada: Covariance matrix de l'etapa de predicció.

    Ro = 6378137;
    e = 0.0818191908425;
    w_ie = 7.292115e-5;
    MatriuVelocitatTerra = [0 -w_ie 0; w_ie 0 0; 0 0 0];

    %Definirem un per un tots els pasos per aplicar correctament l'etapa de
    %predicció del filtre de Kalman.

    %PAS 1: Calcular la matriu de transisió (transition matrix) (Phi).

    f_e_21 = -C_e_b_vella*f_b_ib;
    F_e_21 = [0 -f_e_21(3) f_e_21(2); f_e_21(3) 0 -f_e_21(1); -f_e_21(2)...
        f_e_21(1) 0];

    [L, lambda, h] = Borkowski(r_e_eb_vella(1), r_e_eb_vella(2),...
        r_e_eb_vella(3));

    [g, gamma] = gravetat_ECEF(r_e_eb_vella(1), r_e_eb_vella(2),...
        r_e_eb_vella(3));

    r_e_eS = Ro/sqrt(1 - (e*sin(L))^2)*sqrt(cos(L)^2 + ...
        (1 - e^2)^2*sin(L)^2);

    F_e_23 = -2*g*r_e_eb_vella'/(r_e_eS*sqrt(r_e_eb_vella'*r_e_eb_vella));


    Phi = eye(15);
    Phi(1:3,1:3) = Phi(1:3,1:3) - MatriuVelocitatTerra*t;
    Phi(1:3, 13:15) = C_e_b_vella*t;
    Phi(4:6, 1:3) = F_e_21*t;
    Phi(4:6, 4:6) = Phi(4:6, 4:6) - 2*MatriuVelocitatTerra*t;
    Phi(4:6, 7:9) = F_e_23*t;
    Phi(4:6, 10:12) = C_e_b_vella*t;
    Phi(7:9, 4:6) = eye(3)*t;

    %PAS 2: System noise covariance matrix
```

```
    LC_KF_gyro_noise_PSD = 0.000007^2;
    LC_KF_accel_noise_PSD = 0.002^2;
    LC_KF_accel_bias_PSD = 1.0E-7;
    LC_KF_gyro_bias_PSD = 4.0E-12;

    Q = zeros(15);
    Q(1:3, 1:3) = eye(3)*LC_KF_gyro_noise_PSD * t;
    Q(4:6, 4:6) = eye(3)*LC_KF_accel_noise_PSD * t;
    Q(10:12, 10:12) = eye(3)*LC_KF_accel_bias_PSD * t;
    Q(13:15, 13:15) = eye(3)* LC_KF_gyro_bias_PSD * t;

    %PAS 3: Propagar l'estimació d'estat.

    x_propagada(1:15,1) = 0;

    %PAS 4: Calcular covariance matrix.

    P_propagada = Phi*(P_vella + 0.5*Q)*Phi' + 0.5*Q;

end
```

C.7.2. Etapa de Measurement Update

Segons el moment de la iteració del INS i segons la prova que estàvem realitzant necessitàvem diferents etapes de Measurement Update del filtre de Kalman: una en que només s'apliquèssin Zero Updates, una altre o s'aplicaven Zero Updates i dades del GPS i un altre on només s'entraven dades del GPS. A continuació mostrem les tres implementacions.

C.7.2.1. Measurement Updates amb Zero Updates

```
function [C_e_b_nova, v_e_eb_nova, r_e_eb_nova, IMU_bias_nou, P_nova] = ...
    filtreKalman_I_MeasurementUpdate_ZU( P_propagada, x_propagada, ...
    C_e_b_vella, w_b_ib,  r_e_eb_vella,  v_e_eb_vella, IMU_bias_vell)
%FILTREKALMAN_I_MEASUREMENTUPDATE_ZU: Executa la part de measurement
%Updates del filtre de Kalman en l'etapa de inicialització. Fa Zero Update
%de velocitat angular i velocitat lineal.
%Inputs:
%       x_propagada: vector d'estat.
%       C_e_b_vella: Matriu de rotació de body a ECEF.
%       w_b_ib: velocitat angular que llegeig la IMU.
%       r_e_eb_vella: posició en coordenades ECEF (m).
%       v_e_eb_vella: velocitat en coordenades ECEF (m/s).
%       P_propagada: matriu de covariança anterior.
%       IMU_bias_vell: Estimació del bias de la IMU (acceleròmetres i
%       giroscopis)
%Outputs:
%       C_e_b_nova: Estimació de la matriu de rotació de body a ECEF
%       v_e_eb_nova: Estimació de la velocitat en coordenades ECEF (m/s)
```

```
%       r_e_eb_nova: Estimació de la posició en coordenades ECEF (m/s)
%       IMU_bias_nou: Estimació del bias de la IMU.
%                files 1-3 bias acceleròmetres (m/s^2).
%                files 4-6 bias giroscopis (rad/s).
%       P_nova: Covariance matrix

%MESURAMENT MODEL:

    %PAS 5: Mesurament matrix.

    H = zeros(6, 15);
    H(1:3,4:6) = -eye(3);
    H(4:6,13:15) = -eye(3);

    %PAS 6: Mesurament noise covariance matrix

    LC_KF_pos_meas_SD = 2.5;
    LC_KF_vel_meas_SD = 0.1;

    R(1:3,1:3) = eye(3) * 6e-7;
    R(1:3,4:6) = zeros(3);
    R(4:6,1:3) = zeros(3);
    R(4:6,4:6) = eye(3) * 6e-7;

    %PAS 7: K de Kalaman

    K = P_propagada*H'*inv(H*P_propagada*H' + R);

    %PAS 8: Calcular z.

    delta_z(1:3,1) = -v_e_eb_vella;
    delta_z(4:6,1) = -w_b_ib;

    %PAS 9: Estimació de xk+ a partir de xk-.

    x_nova = x_propagada + K*delta_z;

    %PAS 10: Estimació de Pk- a Pk+.

    P_nova = (eye(15) - K*H)*P_propagada;

    % CORRECCIÓ CLOSED-LOOP

    % Correccions d'orientació, velocitat i posició
    C_e_b_nova = (eye(3) - antisimetrica(x_nova(1:3))) * C_e_b_vella;
    v_e_eb_nova = v_e_eb_vella - x_nova(4:6);
    r_e_eb_nova = r_e_eb_vella - x_nova(7:9);

    % Estimació del bias de la IMU
    IMU_bias_nou = IMU_bias_vell + x_nova(10:15);

end
```

C.7.2.2. Measurement Updates amb Zero Updates i Updates de GPS

```
function [C_e_b_nova, v_e_eb_nova, r_e_eb_nova, IMU_bias_nou, P_nova] = ...
    filtreKalman_I_MeasurementUpdate_GPS_ZU( P_propagada, x_propagada,...
    C_e_b_vella, w_b_ib,  r_e_eb_vella,  v_e_eb_vella, GNSS_r_eb_e,...
    IMU_bias_vell)
%FILTREKALMAN_I_MEASUREMENTUPDATE_GPS_ZU: Executa l'etapa de Measurement
%Update del filtre de Kalman utilitzat en la inicialització on s'entren
%updates de GPS i Zero Updates.
%Inputs:
%       x_propagada: vector d'estat.
%       C_e_b_vella: Matriu de rotació de body a ECEF.
%       w_b_ib: velocitat angular que llegeig la IMU.
%       r_e_eb_vella: posició en coordenades ECEF (m).
%       v_e_eb_vella: velocitat en coordenades ECEF (m/s).
%       P_propagada: matriu de covariança anterior.
%       GNSS_r_eb_e: Posició GPS o zero update.
%       IMU_bias_vell: Estimació del bias de la IMU (acceleròmetres i
%       giroscopis)
%Outputs:
%       C_e_b_nova: Estimació de la matriu de rotació de body a ECEF
%       v_e_eb_nova: Estimació de la velocitat en coordenades ECEF (m/s)
%       r_e_eb_nova: Estimació de la posició en coordenades ECEF (m/s)
%       IMU_bias_nou: Estimació del bias de la IMU.
%                files 1-3 bias acceleròmetres (m/s^2).
%                files 4-6 bias giroscopis (rad/s).
%       P_nova: Covariance matrix

    %MESURAMENT MODEL:

    %PAS 5: Mesurament matrix.

    H = zeros(6, 15);
    H(1:3,7:9) = -eye(3);
    H(4:6,4:6) = -eye(3);
    H(7:9,13:15) = -eye(3);

    %PAS 6: Mesurament noise covariance matrix

    LC_KF_pos_meas_SD = 2.5;
    LC_KF_vel_meas_SD = 0.1;

    R(1:3,1:3) = eye(3) * LC_KF_pos_meas_SD;
    R(1:3,4:6) = zeros(3);
    R(4:6,1:3) = zeros(3);
    R(4:6,4:6) = eye(3) * 6e-7;
    R(7:9,7:9) = eye(3)*6e-7;

    %PAS 7: K de Kalaman

    K = P_propagada*H'*inv(H*P_propagada*H' + R);

    %PAS 8: Calcular z.

    delta_z(1:3,1) = GNSS_r_eb_e - r_e_eb_vella;
    delta_z(4:6,1) = -v_e_eb_vella;
    delta_z(7:9,1) = -w_b_ib;
```

```
    %PAS 9: Estimació de xk+ a partir de xk-.

    x_nova = x_propagada + K*delta_z;

    %PAS 10: Estimació de Pk- a Pk+.

    P_nova = (eye(15) - K*H)*P_propagada;

    % CORRECCIÓ CLOSED-LOOP

    % Correccions d'orientació, velocitat i posició
    C_e_b_nova = (eye(3) - antisimetrica(x_nova(1:3))) * C_e_b_vella;
    v_e_eb_nova = v_e_eb_vella - x_nova(4:6);
    r_e_eb_nova = r_e_eb_vella - x_nova(7:9);

    % Estimació del bias de la IMU
    IMU_bias_nou = IMU_bias_vell + x_nova(10:15);

end
```

C.7.2.3. Measurement Updates de GPS

```
function [C_e_b_nova, v_e_eb_nova, r_e_eb_nova, IMU_bias_nou, P_nova] = ...
    filtreKalman_I_MeasurementUpdate_GPS( P_propagada, x_propagada,...
    C_e_b_vella, r_e_eb_vella,  v_e_eb_vella, GNSS_r_eb_e, IMU_bias_vell)
%FILTREKALMAN_I_MEASUREMENTUPDATE_GPS: Executa l'etapa de measurement
%update del filtre de Kalman. Fa un Update de posició amb les dades que
%arriben d'un GPS.
%Inputs:
%       x_propagada: vector d'estat.
%       C_e_b_vella: Matriu de rotació de body a ECEF.
%       w_b_ib: velocitat angular que llegeig la IMU.
%       r_e_eb_vella: posició en coordenades ECEF (m).
%       v_e_eb_vella: velocitat en coordenades ECEF (m/s).
%       P_propagada: matriu de covariança anterior.
%       GNSS_r_eb_e: Posició GPS o zero update.
%       IMU_bias_vell: Estimació del bias de la IMU (acceleròmetres i
%       giroscopis)
%Outputs:
%       C_e_b_nova: Estimació de la matriu de rotació de body a ECEF
%       v_e_eb_nova: Estimació de la velocitat en coordenades ECEF (m/s)
%       r_e_eb_nova: Estimació de la posició en coordenades ECEF (m/s)
%       IMU_bias_nou: Estimació del bias de la IMU.
%                files 1-3 bias acceleròmetres (m/s^2).
%                files 4-6 bias giroscopis (rad/s).
%       P_nova: Covariance matrix
%MESURAMENT MODEL:

    %PAS 5: Mesurament matrix.

    H = zeros(3, 15);
    H(1:3,7:9) = -eye(3);

    %PAS 6: Mesurament noise covariance matrix
```

```
    LC_KF_pos_meas_SD = 2.5;
    LC_KF_vel_meas_SD = 0.1;

    R(1:3,1:3) = eye(3) * LC_KF_pos_meas_SD;

    %PAS 7: K de Kalaman

    K = P_propagada*H'*inv(H*P_propagada*H' + R);

    %PAS 8: Calcular z.

    delta_z(1:3,1) = GNSS_r_eb_e - r_e_eb_vella;

    %PAS 9: Estimació de xk+ a partir de xk-.

    x_nova = x_propagada + K*delta_z;

    %PAS 10: Estimació de Pk- a Pk+.

    P_nova = (eye(15) - K*H)*P_propagada;

    % CORRECCIÓ CLOSED-LOOP

    % Correccions d'orientació, velocitat i posició
    C_e_b_nova = (eye(3) - antisimetrica(x_nova(1:3))) * C_e_b_vella;
    v_e_eb_nova = v_e_eb_vella - x_nova(4:6);
    r_e_eb_nova = r_e_eb_vella - x_nova(7:9);

    % Estimació del bias de la IMU
    IMU_bias_nou = IMU_bias_vell + x_nova(10:15);

end
```

Per implementar el filtre de Kalman és necessària aquesta petita funció que calcula la matriu antisimètrica d'un vector. S'aplica al final de l'etapa de measurement update.

```
function A = antisimetrica(a)
%Retorna la matriu antisimètrica d'un vector
% Inputs:
%       a: vector 3x1
% Outputs:
%       A: matriu antisimetrica 3x3

A = [    0, -a(3),  a(2);...
      a(3),     0, -a(1);...
     -a(2),  a(1),     0];

end
```

C.8. Programes sencers

En aquest capítol es trobaran els scripts dels tres programes sencers que implementen INS. Des de la inicialització amb el leveling i magnètic heading fins a les equacions de navegació necessàries pel càlcul de la posició, velocitat i orientació. Hi ha el programa que ho resolt en ECI, el que ho resolt en ECEF i el que utilitza el NED. A més en el programa ECEF també si integra el filtre de Kalman.

B.8.1. Programa ECI

```
%
%                PROGRAMA ECI
%
% Script que calcula la posició, velocitat, orientació i acceleració de la
% IMU segons el sistema de referència ECI.

%                Inicialització

clc;
clear all;
load('matlab.mat');

[indat] = data_loader('quiet2.csv');
matriuDades = indat;

%Transformar les dades dels giroscopis de graus/s a rad/s.
matriuDades(:,2:4)=matriuDades(:,2:4)*pi/180;

%Temps d'iteració (s), velocitat rotació Terra (rad/s).
t = 1/102.5;
t_acumulat = 0;
w_ie = 7.292115e-5;

% Troba Roll i Pitch inicials de la IMU utilitzant la funcio LEVELING
[angleFi_nb, angleTeta_nb, angleFi_nb_graus, angleTeta_nb_graus] =
leveling(matriuDades);

%Troba Yaw de la  utilitzant la funcio MAGNETIC HEADING
[anglePsi_nb, anglePsi_nb_graus] = magneticHeading(matriuDades);

%Posició segons paràmetres de latitud, longitud i altitud.
L = input('Ingresa el valor de la latitud (L) inicial: ');
lambda = input('Ingresa el valor de la longitud inicial: ');
h = input('Ingresa el valor de la altitud inicial: ');

L_rad = degtorad(L);
lambda_rad = degtorad(lambda);

%Troba posició respecte ECEF
[rx,ry,rz] = posicioNED_ECEF(L_rad, lambda_rad, h);
```

```
R_i_ib_vella = [rx; ry; rz];

%Crea matriu de rotació de NED a Body frame, després de NED a ECI, i
%finalment, de Body a ECI.
[C_ned_b] = crearMatriuRotacioBody_NED( angleFi_nb, angleTeta_nb,
anglePsi_nb);
[C_eci_ned] = crearMatriuRotacioNED_ECI( L, lambda, t_acumulat);
C_eci_b_vella = C_eci_ned*C_ned_b;

%Velocitat inicial respecte ECEF
V_e_eb = [0; 0; 0];

%Velocitat inicial respecte ECI
[V_i_ib_vella] = velocitatInicialECI( R_i_ib_vella, V_e_eb );

%Creem fitxer de dades
fileID = fopen('PosicioECI.txt','w');
fprintf(fileID,'\n%s %s %s\n\n','rx','ry','rz');
fprintf(fileID,'%f      %f      %f\n', R_i_ib_vella);

%Creem fitxer de dades per l'orientació
fileID2 = fopen('OrientacioECI.txt','w');
fprintf(fileID2,'\n%s %s %s\n\n','Roll','Pitch','Yaw');
fprintf(fileID2,'%f      %f      %f\n', angleFi_nb_graus,
angleTeta_nb_graus,anglePsi_nb_graus);

%**********ITERACIÓ**********
%Ens dona el nombre de files (n) i columnes (c) de la matriu de dades de la
IMU.
[n, c] = size(matriuDades);

for fila=1:1:n,

    t_acumulat = t_acumulat + t;

    %Llegir velocitat angular del fitxer de dades.
    [wx_b_ib, wy_b_ib, wz_b_ib] = llegirW(fila, matriuDades);
    W_b_ib = [wx_b_ib; wy_b_ib; wz_b_ib];

    %Llegim la força específica del fitxer de dades
    [fx_b_ib, fy_b_ib, fz_b_ib] = llegirForcaEspecifica(fila,matriuDades);
    F_b_ib = [fx_b_ib; fy_b_ib; fz_b_ib]*9.80655;

    %   EQUACIONS DE NAVEGACIO ECI
    %Apliquem les equacions de navegació per trobar la posició, la
    %velocitat i la matriu de rotacio.
    [R_i_ib, V_i_ib, C_eci_b ] = equacionsNavegacioECI( t, R_i_ib_vella,
      V_i_ib_vella, C_eci_b_vella, F_b_ib, W_b_ib );

    fprintf(fileID,'%f      %f    %f\n', R_i_ib);

    %Fitxer angles Euler.
    [L, lambda, h] = Borkowski(R_e_eb(1), R_e_eb(2), R_e_eb(3));
    [ C_n_i ] = crearMatriuECI_NED( t_acumulat, L, lambda );
     C_n_b = C_n_i*C_eci_b;
    [Roll, Pitch, Yaw] = calcularAnglesEuler( C_n_b );

    fprintf(fileID2,'%f      %f    %f\n', Roll, Pitch, Yaw);
```

```
    R_i_ib_vella = R_i_ib;
    V_i_ib_vella = V_i_ib;
    C_eci_b_vella = C_eci_b;

end

fclose(fileID);
fclose(fileID2);
```

C.8.2. Programa ECEF

En aquest programa es criden les funcions del filtre de Kalman.

```
%
%               PROGRAMA ECEF
%
% Script que calcula la posició, velocitat, orientació i acceleració de la
% IMU segons el sistema de referència ECEF.

%               Inicialització

clc;
clear all;

%Carraguem dades de la IMU
[ indat ] = data_loader( 'voltagran1.csv' );
matriuDades = indat;

%Carraguem dades del GPS
addpath('voltacotxe/')
addpath('NMEAReader/')

[hour, minute, second, velocity, numsats, height, lat_decimal,...
    long_decimal, quality] = readGPS('GPS1.TXT');

%Transformar les dades dels griscopis de graus/s a rad/s. I càlcul del
%bias.
matriuDades(:,2:4)=matriuDades(:,2:4)*pi/180;

%Temps d'iteració (s) i velocitat de rotació de la Terra (rad/s).
w_ie = 7.292115e-5;
fila = 1;
cops = 0;

t_old = 0;
[ t, label, indexs ] = samplingTimeVoltaGran( matriuDades );

 % Troba Roll, Pitch i Yaw inicials de la IMU
[angleFi_nb, angleTeta_nb, angleFi_nb_graus, angleTeta_nb_graus] = ...
    leveling(matriuDades);
[anglePsi_nb, anglePsi_nb_graus] = magneticHeading(matriuDades);

%Posició segons paràmetres de latitud, longitud i altitud (en graus).
L = lat_decimal(1647,1);
```

```
lambda = long_decimal(1647,1);
h = height(1647,1);

L_rad = degtorad(L);
lambda_rad = degtorad(lambda);

%Troba posició respecte ECEF
[rx,ry,rz] = posicioNED_ECEF(L_rad, lambda_rad, h);
R_e_eb_vella = [rx; ry; rz];
GNSS_r_e_eb = [rx; ry; rz];
PosicioInicial = [rx; ry; rz];

%Creem una matriu de rotació de Body a NED amb els valors dels angles
%trobats amb les funcions de leveling i magnetic Heading. Després fem la
%matriu de rotació de NED a ECEF amb els valors de la posició (L, lambda).
%Finalment, multipliquem les dues matrius calculades i trobem la que va de
%Body a ECEF.
[C_ned_b] = crearMatriuRotacioBody_NED( angleFi_nb, angleTeta_nb, ...
    anglePsi_nb);
[C_ecef_ned] = crearMatriuRotacioNED_ECEF( L, lambda);
C_ecef_b_vella = C_ecef_ned*C_ned_b;

%Velocitat inicial 0.
V_e_eb_vella = [0; 0; 0];
GNSS_v_e_eb = [0; 0; 0];

%Inicialització de dades pel filtre de Kalman (matriu P i bias).
LC_KF_orientacio = degtorad(1);
LC_KF_velocitat = 0.1;
LC_KF_posicio = 10;
LC_KF_biasA = 1000 * 9.80665E-6;
LC_KF_biasG = 10 * 0.01745329252 / 3600;

P =  zeros(15);
P(1:3,1:3) = eye(3) * LC_KF_orientacio^2;
P(4:6,4:6) = eye(3) * LC_KF_velocitat^2;
P(7:9,7:9) = eye(3) * LC_KF_posicio^2;
P(10:12,10:12) = eye(3) * LC_KF_biasA^2;
P(13:15,13:15) = eye(3) * LC_KF_biasG^2;

IMU_bias_vell = zeros(6,1);

%************ITERACIÓ***************
for i=2:length(t),

    At = t(i) - t_old;

    if label(i)==1
        %Llegim el valor dels giròscops.
        [wx_b_ib, wy_b_ib, wz_b_ib] = llegirW(indexs(i), matriuDades);
        W_b_ib = [wx_b_ib; wy_b_ib; wz_b_ib];

        %Llegim la força específica
        [fx_b_ib, fy_b_ib, fz_b_ib] = ...
            llegirForcaEspecifica(indexs(i),matriuDades);
        F_b_ib = [fx_b_ib; fy_b_ib; fz_b_ib]*9.80665;

        %Correcció bias
        F_b_ib = F_b_ib - IMU_bias_vell(1:3);
```

```
        W_b_ib = W_b_ib - IMU_bias_vell(4:6);

        %   EQUACIONS DE NAVEGACIO ECEF
        %Apliquem les equacions de navegació per trobar la posició, la
        %velocitat i la matriu de rotacio.
        [ R_e_eb, V_e_eb, C_ecef_b] = equacionsNavegacioECEF(At,...
            R_e_eb_vella, V_e_eb_vella, C_ecef_b_vella, F_b_ib, W_b_ib );

        PosicioECEF(fila, 1) = R_e_eb(1,1)- PosicioInicial(1,1);
        PosicioECEF(fila, 2) = R_e_eb(2,1)- PosicioInicial(2,1);
        PosicioECEF(fila, 3) = R_e_eb(3,1)- PosicioInicial(3,1);
        VelocitatECEF(fila, 1:3) = V_e_eb';
        fila = fila + 1;
    end

    if label(i)==1 && t(i)<10

        %Etapa de Predicció del Filtre de Kalman
         [P, x_propagada] = filtreKalman_I_Prediccio( At,...
         C_ecef_b_vella, F_b_ib, R_e_eb_vella, P);

         %Measurment Update del Filtre de Kalman amb Zupdates de
         %velocitat lineal i angular.
         [C_ecef_b,  V_e_eb, R_e_eb, IMU_bias, P  ] = ...
         filtreKalman_I_MeasurementUpdate_ZU( P, x_propagada,...
         C_ecef_b_vella,...
         W_b_ib,  R_e_eb_vella,  V_e_eb_vella, IMU_bias_vell);
    end

    if label(i)==2 && t(i)<10

        %Llegim les coordenades del GPS
         L_rad = degtorad(lat_decimal(indexs(i),1));
         lambda_rad = degtorad(long_decimal(indexs(i),1));
         h = height(indexs(i),1);

         %Passem les coordenades de GPS a ECEF
         [rx,ry,rz] = posicioNED_ECEF(L_rad, lambda_rad, h);
         GNSS_r_e_eb = [rx; ry; rz];

         %Etapa de Predicció del Filtre de Kalman
         [P, x_propagada] = filtreKalman_I_Prediccio( At,...
         C_ecef_b_vella, F_b_ib, R_e_eb_vella, P);

         %Measurement Update del Filtre de Kalman amb Zupdates de
         %velocitat lineal i angular i amb dades del GPS
         [C_ecef_b,  V_e_eb, R_e_eb, IMU_bias, P  ] = ...
         filtreKalman_I_MeasurementUpdate_GPS_ZU( P, x_propagada,...
         C_ecef_b_vella, W_b_ib,  R_e_eb_vella,  V_e_eb_vella,...
         GNSS_r_e_eb, IMU_bias_vell);

    end

    if t(i)>10

         %Etapa de Predicció del Filtre de Kalman
         [P, x_propagada] = filtreKalman_I_Prediccio( At,...
         C_ecef_b_vella, F_b_ib, R_e_eb_vella, P);
```

```
    end


    if label(i)==2 && t(i)>10

        %Llegim les coordenades del GPS
         L_rad = degtorad(lat_decimal(indexs(i),1));
         lambda_rad = degtorad(long_decimal(indexs(i),1));
         h = height(indexs(i),1);

         %Passem les coordenades de GPS a ECEF
         [rx,ry,rz] = posicioNED_ECEF(L_rad, lambda_rad, h);
         GNSS_r_e_eb = [rx; ry; rz];

         %Measurement Updates del filtre de Kalman amb Updates del GPS
         [C_ecef_b,  V_e_eb, R_e_eb, IMU_bias, P  ] = ...
        filtreKalman_I_MeasurementUpdate_GPS( P, x_propagada,...
        C_ecef_b_vella, R_e_eb_vella,  V_e_eb_vella, GNSS_r_e_eb,...
        IMU_bias_vell);
    end

    IMU_bias_vell = IMU_bias;
    t_old = t(i);

    %   CÀLCUL ANGLES EULER (Roll, Pitch i Yaw)

    %Passem la posició trobada respecte ECEF a NED frame.
    [L, lambda, h] = Borkowski(R_e_eb(1), R_e_eb(2), R_e_eb(3));

    %Trobem matriu de rotació de ECEF a NED.
    [ C_ned_ecef ] = crearMatriuRotacioECEF_NED( L, lambda);

    %Trobem matriu de Body a NED.
    C_n_b = C_ned_ecef*C_ecef_b;

    %Trobem angles d'Euler.
    [Roll, Pitch, Yaw] = calcularAnglesEuler( C_n_b );

    AnglesEuler(fila, 1:3)=[Roll; Pitch; Yaw];


    R_e_eb_vella = R_e_eb;
    V_e_eb_vella = V_e_eb;
    C_ecef_b_vella = C_ecef_b;

End
```

Dins el programa ECEF que acabem de mostrar, a la part inicial, es crida la funció sampingTime. Aquesta funció crea un vector amb totes els instants de temps ordenats en que s'ha d'entrar una dada de la IMU o una dada del GPS. A continuació la mostrem.

```
function [ t, label, indexs ] = samplingTimeVoltaGran( matriuDades )
%SAMPLING TIME
%Passos per crear el sampling time de la IMU i el GPS.

    %Vectors de temps de la IMU i el GPS
    t_IMU = matriuDades(:,1)*(1/102.5);
    t_IMU = t_IMU';

    for fila=1:1:600
        contador(fila) = fila;
    end

    t_GPS = contador*(1/5);
    t_GPS = t_GPS + 0.0001;

    %Creem un vector de 1 i un altre de 2 per saber si es tracte
    %d'una dada de IMU o de GPS
    label_IMU=ones(size(t_IMU));
    label_GPS=2*ones(size(t_GPS));

    %Concatanem els temps de la IMU i del GPS. El mateix amb els 1 i 2.
    t=[t_IMU t_GPS];
    label=[label_IMU label_GPS];

    %Creem un vector indexs
    indexs=[1:length(t_IMU) 1647:2247];

    %Ordenem el temps i coloquem els 1 i 2 i els indexs segons com
    %està ordenat el temps.
    [t,i]=sort(t);
    label=label(i);
    indexs=indexs(i);


end
```

C.8.3. Programa sistema local de navegació

```
%
%               PROGRAMA SISTEMA LOCAL DE NAVEGACIÓ
%
% Script que calcula la posició, velocitat, orientació i acceleració de la
% IMU segons el sistema de referència NED.

%               Inicialització

clc;
clear all;
load('matlab.mat');

[ indat ] = data_loader( 'quiet2.csv' );
matriuDades = indat;

%Transformar les dades dels griscopis de graus/s a rad/s. I càlcul del
%bias.
```

```
matriuDades(:,2:4)=matriuDades(:,2:4)*pi/180;

%Temps d'iteració (s), velocitat rotació de la Terra (rad/s) i fila.
t = 1/102.5;
w_ie = 7.292115e-5;
t_acumulat = 0;

%Troba Roll i Pitch de la IMU
[angleFi_nb, angleTeta_nb, angleFi_nb_graus, angleTeta_nb_graus] =
leveling(matriuDades);

%Troba Yaw de la IMU
[anglePsi_nb, anglePsi_nb_graus] = magneticHeading(matriuDades);

%Posició segons paràmetres de latitud, longitud i altitud (en graus).
L_graus = input('Ingresa el valor de la latitud (L) inicial: ');
lambda_graus = input('Ingresa el valor de la longitud inicial: ');
h_vella = input('Ingresa el valor de la altitud inicial: ');

%Passem la longitud i la latitud a radians
L_vella = degtorad(L_graus);
lambda_vella = degtorad(lambda_graus);

%Creem matriu de rotació de body a NED frame
[C_ned_b_anterior] = crearMatriuRotacioBody_NED( angleFi_nb, angleTeta_nb,
anglePsi_nb);

%Velocitat inicial respecte NED frame
V_n_eb_vella = [0; 0; 0];

%Creem fitxers de dades
fileID = fopen('PosicioNED.txt','w');
fprintf(fileID,'\n%s %s %s\n\n','L','lambda','h');
fprintf(fileID,'%f      %f      %f\n', L_graus, lambda_graus, h_vella);



%Creem fitxers de dades per guardar angles d'Euler.
fileID2 = fopen('OrientacioNED'.txt','w');
fprintf(fileID2,'\n%s %s %s\n\n','Roll','Pitch','Yaw');
fprintf(fileID2,'%f      %f      %f\n', angleFi_nb_graus,
angleTeta_nb_graus, anglePsi_nb_graus);

%Ens dona el nombre de files (n) i columnes (c) de la matriu de dades de la
IMU.
[n, c] = size(matriuDades);

%*********ITERACIÓ***********


for fila=1:1:n,

    t_acumulat = t_acumulat +1;

    %Agafem velocitat angualr de la matriu de dades.
    [wx_b_ib, wy_b_ib, wz_b_ib] = llegirW(fila, matriuDades);
    W_b_ib = [wx_b_ib; wy_b_ib; wz_b_ib];
```

```
    %Llegim la força específica
    [fx_b_ib, fy_b_ib, fz_b_ib] = llegirForcaEspecifica(fila,matriuDades);
    F_b_ib = [fx_b_ib; fy_b_ib; fz_b_ib]*9.80665;

    %   EQUACIONS DE NAVEGACIO SISTEMA LOCAL DE NAVEGACIÓ
    %Apliquem les equacions de navegació per trobar la posició, la
    %velocitat i la matriu de rotacio.

    [L, lambda, h, V_n_eb, C_ned_b] = equacionsNavegacioNED(t, L_vella,...
        lambda_vella, h_vella, V_n_eb_vella, C_ned_b_anterior, F_b_ib,
      W_b_ib );


    %Passem a graus per grabar-ho al fitxer en coordenades NED.
    L_graus = radtodeg(L);
    lambda_graus = radtodeg(lambda);
    fprintf(fileID,'%f      %f    %f\n', L_graus, lambda_graus, h);


    %Grabar orientació en un fitxer.
    [Roll, Pitch, Yaw] = calcularAnglesEuler( C_ned_b );
    fprintf(fileID2,'%f      %f    %f\n', Roll, Pitch, Yaw);

    %Actualitzem C_ned_b_anterior, L_vella, lambda_vella, h_vella,
    %V_n_eb_vella

    C_ned_b_anterior = C_ned_b;
    L_vella = L;
    lambda_vella = lambda;
    h_vella = h;
    V_n_eb_vella = V_n_eb;

end
fclose(fileID);
fclose(fileID2)
```

D. ANNEX DE CATÀLEGS TÈCNICS DE LA IMU I AHRS

En aquest annex si pot trobar dos catàlegs amb les especificacions tècniques dels sensors que hem utilitzat en aquest treball: la MTi i ADIS16488.

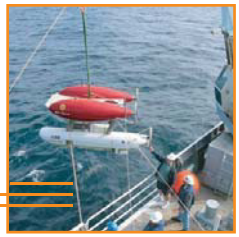
MTi

Miniature Attitude and Heading Reference System

The MTi is a miniature, gyro-enhanced Attitude and Heading Reference System (AHRS). Its internal low-power signal processor provides drift-free 3D orientation as well as calibrated 3D acceleration, 3D rate of turn (rate gyro) and 3D earth-magnetic field data. The MTi is an excellent measurement unit for stabilization and control of cameras, robots, vehicles and other equipment.

Features

- accurate full 360 degrees 3D orientation output (Attitude and Heading)
- highly dynamic response combined with long-term stability
- 3D acceleration, 3D rate of turn and 3D earth-magnetic field data
- all solid state miniature MEMS inertial sensors inside
- compact design
- high update rate
- various digital output modes
- accepts or generates synchronization pulses
- temperature, 3D misalignment and sensor cross-sensitivity compensated



Fields of use

- robotics
- aerospace
- autonomous vehicles
- marine industry
- bore industry

The MTi uses 3 rate gyros to track rapidly changing orientations in 3D and it measures the directions of gravity and magnetic north to provide a stable reference. The systems real-time algorithm fuses the sensor information to calculate accurate 3D orientation, with a highly dynamic response and stable over time.

With the MTi Development Kit, the MTi can easily be integrated in any system or (OEM) application.

xsens

Output

3D orientation (Quaternions/Matrix/Euler angles)
3D acceleration
3D rate-of-turn
3D earth-magnetic field (normalized)
Temperature

Orientation performance

Dynamic Range:	all angles in 3D
Angular Resolution[1]:	0.05 deg
Static Accuracy (Roll/Pitch):	<0.5 deg
Static Accuracy[2] (Heading):	<1 deg
Dynamic Accuracy[3]:	2 deg RMS

Sensor performance

	rate of turn	acceleration	magnetic field	temperature
Dimensions	3 axes	3 axes	3 axes	-
Full Scale (standard)	± 300 deg/s	± 17 m/s²	± 750 mGauss	-55...+125 °C
Linearity	0.1% of FS	0.2% of FS	0.2% of FS	<1% of FS
Bias stability[4] (1σ)	5 deg/s	0.02 m/s²	0.5 mGauss	0.5 °C accuracy
Scale Factor stability[4] (1σ)	-	0.05%	0.5%	-
Noise density	0.1 deg/s/√Hz	0.001 m/s²/√Hz	0.5 mGauss (1σ)	-
Alignment error	0.1 deg	0.1 deg	0.1 deg	-
Bandwidth (standard)	40 Hz	30 Hz	10 Hz	-

Interfacing

Max update rate:	512 Hz (calibrated sensor data) 120 Hz (orientation data)
Digital interface (standard):	RS-232 and USB (external converter)
Operating voltage:	4.5 - 15 V
Power consumption:	360 mW (orientation output)

Housing

Dimensions:	58x58x22 mm (WxLxH)
Weight:	50 g
Ambient temperature operating range:	0 - 55 deg Celsius

Options and product code

Interface:

- RS-232 (RS-232, analog in, sync out, sync in) : 28
- RS-485 (RS-485, sync out, sync in) : 48
- RS-422 (RS-422, sync in) : 68

Full Scale Acceleration:

- 1.7 g (17 m/s²) : A33
- 5 g (50 m/s²) : A53
- 10 g (100 m/s²) : A13

Full Scale Rate of Turn:

- 150 deg/s : G15
- 300 deg/s : G35
- 1200 deg/s : G25

Product code: MTi- ##A##G##
Standard version: MTi- 28A53G35

Other options on request.
Surcharges may apply.

1 1σ standard deviation of zero-mean angular random walk
2 in homogenous magnetic environment
3 may depend on type of motion
4 deviation over operating temperature range (1σ) specifications subject to change without notice

Xsens Technologies B.V. • Pantheon 6a • P.O. Box 559 • 7500 AN Enschede • The Netherlands
phone +31 88 XSENS 00 / +31 88 97367 00 • fax +31 88 97367 01 • e-mail info@xsens.com • internet www.xsens.com

Tactical Grade Ten Degrees of Freedom Inertial Sensor

Data Sheet

ADIS16488

FEATURES

- Triaxial, digital gyroscope, ±450°/sec dynamic range
  - <±0.05° orthogonal alignment error
  - 6°/hr in-run bias stability
  - 0.3°/√hr angular random walk
  - 0.01% nonlinearity
- Triaxial, digital accelerometer, ±18 *g*
- Triaxial, delta angle and delta velocity outputs
- Triaxial, digital magnetometer, ±2.5 gauss
- Digital pressure sensor, 300 mbar to 1100 mbar
- Fast start-up time, ~500 ms
- Factory-calibrated sensitivity, bias, and axial alignment
  - Calibration temperature range: −40°C to +70°C
- SPI-compatible serial interface
- Embedded temperature sensor
- Programmable operation and control
  - Automatic and manual bias correction controls
  - 4 FIR filter banks, 120 configurable taps
  - Digital I/O: data-ready alarm indicator, external clock
  - Alarms for condition monitoring
  - Power-down/sleep mode for power management
  - Optional external sample clock input: up to 2.4 kHz
  - Single-command self-test
- Single-supply operation: 3.0 V to 3.6 V
- 2000 *g* shock survivability
- Operating temperature range: −40°C to +85°C

APPLICATIONS

- Platform stabilization and control
- Navigation
- Personnel tracking
- Instrumentation
- Robotics

GENERAL DESCRIPTION

The ADIS16488 iSensor® device is a complete inertial system that includes a triaxis gyroscope, a triaxis accelerometer, triaxis magnetometer, and pressure sensor. Each inertial sensor in the ADIS16488 combines industry-leading *i*MEMS® technology with signal conditioning that optimizes dynamic performance. The factory calibration characterizes each sensor for sensitivity, bias, alignment, and linear acceleration (gyroscope bias). As a result, each sensor has its own dynamic compensation formulas that provide accurate sensor measurements.

The ADIS16488 provides a simple, cost-effective method for integrating accurate, multiaxis inertial sensing into industrial systems, especially when compared with the complexity and investment associated with discrete designs. All necessary motion testing and calibration are part of the production process at the factory, greatly reducing system integration time. Tight orthogonal alignment simplifies inertial frame alignment in navigation systems. The SPI and register structure provide a simple interface for data collection and configuration control.

The ADIS16488 uses the same footprint and connector system as the ADIS16375, which greatly simplifies the upgrade process. It comes in a module that is approximately 47 mm × 44 mm × 14 mm and has a standard connector interface.

FUNCTIONAL BLOCK DIAGRAM

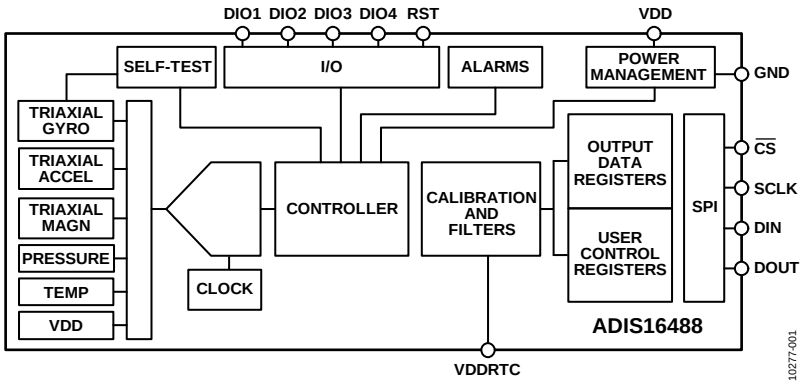


Figure 1.

Rev. G

Document Feedback

Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Specifications subject to change without notice. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices. Trademarks and registered trademarks are the property of their respective owners.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781.329.4700 ©2011–2014 Analog Devices, Inc. All rights reserved.
Technical Support www.analog.com

TABLE OF CONTENTS

REVISION HISTORY

5/14—Rev. F to Rev. G

Changes to Table 71, Table 72, and Table 73 24
Changes to Table 81, Table 82, and Table 83 25

4/14—Rev. E to Rev. F

Change to Features Section 1
Change to Nonlinearity, Barometer Parameter, Endnote 4, and Endnote 11, Table 1 5
Change to t_{SFS} Parameter, Table 2 6
Changes to Table 9 14
Changes to Delta Angles Section 17
Changes to Status Alarm Indicators Section 19
Changes to Magnetometer/Barometer Section 21
Change to Linear Acceleration on Effect on Gyroscope Bias Section 25
Changes to Endnote 1, Table 118 31
Change to Mounting Tips Section 33

1/14—Rev. D to Rev. E

Change to Reset Recovery Time Parameter and Endnote 11, Table 1 5
Changes to Accelerometers Section and Magnetometers Section 25

1/14—Rev. C to Rev. D

Change to t_2 Parameter, Table 2 6
Changes to Delta Angles Section 16
Changes to Delta Velocity Section 17
Changes to Status/Alarm Indicators Section 19
Deleted Prototype Interface Board Section, Mechanical Design Tips Section, Figure 26, Figure 27, Figure 30, and Figure 31; Renumbered Sequentially 33
Added Mounting Tips Section and Figure 26; Renumbered Sequentially 33
Added Evaluation Tools Section, Power Supply Consideration Section, Figure 29, and Figure 30; Renumbered Sequentially 34

2/13—Rev. B to Rev. C

Change to Applications Section 1
Moved Revision History 3
Changes to Table 1 4
Changes to Table 2 and Figure 2 6
Change to Barometric Pressure Parameter, Table 3 7
Changes to Figure 6 8
Changes to Table 9 12
Changes to Table 31, Table 32, Table 33, and Table 34 17
Changes to Linear Acceleration on Effect on Gyroscope Bias Section 25
Changes to Restoring Factory Calibration Section 27
Change to Prototype Interface Board Section 33
Deleted Installation Tips Section 34
Added Mechanical Design Tips Section and Connector Down Mounting Tips Section; Changes to Figure 28 and Figure 29 34
Added Connector Up Design Tips Section, Figure 30, and Figure 31, Renumbered Sequentially 35
Updated Outline Dimensions 36

2/12—Rev. A to Rev. B

Change to Features Section 1
Changes to Table 3 6
Changes to Figure 7 and Figure 8 8
Changes to Delta Angles Section 15
Changes to Delta Velocity Section, Table 31, Table 32, Table 33, and Table 34 16
Change to Status/Alarm Indicators Section 18
Changes to Gyroscopes/Accelerometers Section, Averaging/Decimation Filter Section, Magnetometer/Barometer Section, and Figure 20 20
Changes to Input Sync/Clock Control Section 30
Changes to Prototype Interface Board Section and Figure 26 30

12/11—Rev. 0 to Rev. A

Changes to Specifications Section 3
Changes to System/Alarm Indicators Section 18
Changes to Averaging/Decimation Filter Section 20
Changes to General-Purpose I/O Section 29
Changes to Input Sync/Clock Control Section 30

10/11—Revision 0: Initial Version

SPECIFICATIONS

T_A = 25°C, VDD = 3.3 V, angular rate = 0°/sec, dynamic range = ±450°/sec ± 1 *g*, 300 mbar to 1100 mbar, unless otherwise noted.

Table 1.

Parameter	Test Conditions/Comments	Min	Typ	Max	Unit
GYROSCOPES					
Dynamic Range		±450		±480	°/sec
Sensitivity	x_GYRO_OUT and x_GYRO_LOW (32-bit)		3.052×10^{-7}		°/sec/LSB
Repeatability[1]	−40°C ≤ T_A ≤ +70°C			±1	%
Sensitivity Temperature Coefficient	−40°C ≤ T_A ≤ +70°C, 1 σ		±35		ppm/°C
Misalignment	Axis-to-axis		±0.05		Degrees
	Axis-to-frame (package)		±1.0		Degrees
Nonlinearity	Best-fit straight line, FS = 450°/sec		0.01		% of FS
Bias Repeatability[1, 2]	−40°C ≤ T_A ≤ +70°C, 1 σ		±0.2		°/sec
In-Run Bias Stability	1 σ		6.25		°/hr
Angular Random Walk	1 σ		0.3		°/√hr
Bias Temperature Coefficient	−40°C ≤ T_A ≤ +70°C, 1 σ		±0.0025		°/sec/°C
Linear Acceleration Effect on Bias	Any axis, 1 σ (CONFIG[7] = 1)		0.009		°/sec/*g*
Output Noise	No filtering		0.16		°/sec rms
Rate Noise Density	f = 25 Hz, no filtering		0.0066		°/sec/√Hz rms
3 dB Bandwidth			330		Hz
Sensor Resonant Frequency			18		kHz
ACCELEROMETERS	Each axis				
Dynamic Range		±18			*g*
Sensitivity	x_ACCL_OUT and x_ACCL_LOW (32-bit)		1.221×10^{-8}		*g*/LSB
Repeatability[1]	−40°C ≤ T_A ≤ +70°C			±0.5	%
Sensitivity Temperature Coefficient	−40°C ≤ T_A ≤ +70°C, 1 σ		±25		ppm/°C
Misalignment	Axis-to-axis		±0.035		Degrees
	Axis-to-frame (package)		±1.0		Degrees
Nonlinearity	Best-fit straight line, ±10 *g*		0.1		% of FS
	Best-fit straight line, ±18 *g*		0.5		% of FS
Bias Repeatability[1, 2]	−40°C ≤ T_A ≤ +70°C, 1 σ		±16		m*g*
In-Run Bias Stability	1 σ		0.1		m*g*
Velocity Random Walk	1 σ		0.029		m/sec/√hr
Bias Temperature Coefficient	−40°C ≤ T_A ≤ +85°C		±0.1		m*g*/°C
Output Noise	No filtering		1.5		m*g* rms
Noise Density	f = 25 Hz, no filtering		0.067		m*g*/√Hz rms
3 dB Bandwidth			330		Hz
Sensor Resonant Frequency			5.5		kHz
MAGNETOMETER					
Dynamic Range		±2.5			gauss
Sensitivity			0.1		mgauss/LSB
Initial Sensitivity Tolerance				±2	%
Sensitivity Temperature Coefficient	1 σ		275		ppm/°C
Misalignment	Axis to axis		0.25		Degrees
	Axis to frame (package)		0.5		Degrees
Nonlinearity	Best fit straight line		0.5		% of FS
Initial Bias Error	0 gauss stimulus		±15		mgauss
Bias Temperature Coefficient	−40°C ≤ T_A ≤ +85°C, 1 σ		0.3		mgauss/°C
Output Noise	No filtering		0.45		mgauss
Noise Density	f = 25 Hz, no filtering		0.054		mgauss/√Hz
3 dB Bandwidth			330		Hz

Parameter	Test Conditions/Comments	Min	Typ	Max	Unit
BAROMETER					
Pressure Range		300		1100	mbar
	Extended	10		1200	mbar
Sensitivity	BAROM_OUT and BAROM_LOW (32-bit)		6.1×10^{-7}		mbar/LSB
Error with Supply			0.04		%/V
Total Error			4.5		mbar
Relative Error[3]	−40°C to +85°C		2.5		mbar
Nonlinearity[4]	Best fit straight line, FS = 1100 mbar		0.1		% of FS
	−40°C to +85°C		0.2		% of FS
Linear-*g* Sensitivity	±1 *g*, 1 σ		0.005		mbar/*g*
Noise			0.025		mbar rms
TEMPERATURE SENSOR					
Scale Factor	Output = 0x0000 at 25°C (±5°C)		0.00565		°C/LSB
LOGIC INPUTS[5]					
Input High Voltage, V_{IH}		2.0			V
Input Low Voltage, V_{IL}				0.8	V
$\overline{CS}$ Wake-Up Pulse Width		20			µs
Logic 1 Input Current, I_{IH}	V_{IH} = 3.3 V			10	µA
Logic 0 Input Current, I_{IL}	V_{IL} = 0 V				
All Pins Except $\overline{RST}$				10	µA
$\overline{RST}$ Pin			0.33		mA
Input Capacitance, C_{IN}			10		pF
DIGITAL OUTPUTS					
Output High Voltage, V_{OH}	I_{SOURCE} = 0.5 mA	2.4			V
Output Low Voltage, V_{OL}	I_{SINK} = 2.0 mA			0.4	V
FLASH MEMORY	Endurance[6]	100,000			Cycles
Data Retention[7]	T_J = 85°C	20			Years
FUNCTIONAL TIMES[8]	Time until data is available				
Power-On Start-up Time			500		ms
Reset Recovery Time[9]			500		ms
Sleep Mode Recovery Time			500		µs
Flash Memory Update Time			375		ms
Flash Memory Test Time			50		ms
Automatic Self-Test Time	Using internal clock, 100 SPS		12		ms
CONVERSION RATE			2.46		kSPS
Initial Clock Accuracy			0.02		%
Temperature Coefficient			40		ppm/°C
Sync Input Clock		0.7[10]		2.4	kHz
POWER SUPPLY, VDD	Operating voltage range	3.0		3.6	V
Power Supply Current[11]	Normal mode, VDD = 3.3 V, µ ± σ		254		mA
	Sleep mode, VDD = 3.3 V		12.2		mA
	Power-down mode, VDD = 3.3 V		45		µA
POWER SUPPLY, VDDRTC	Operating voltage range	3.0		3.6	V
Real-Time Clock Supply Current	Normal mode, VDDRTC = 3.3 V		13		µA

[1] The repeatability specifications represent analytical projections based on the following drift contributions and conditions: temperature hysteresis (−40°C to +70°C), electronics drift (High-Temperature Operating Life test: +85°C, 500 hours), drift from temperature cycling (JESD22, Method A104-C, Method N, 500 cycles, −40°C to +85°C), rate random walk (10 year projection), and broadband noise.
[2] Bias repeatability describes a long-term behavior over a variety of conditions. Short-term repeatability is related to the in-run bias stability and noise density specifications.
[3] The relative error assumes that the initial error, at 25°C, is corrected in the end application.
[4] Specification assumes a full scale (FS) of 1000 mbar.
[5] The digital I/O signals use a 3.3 V system.
[6] Endurance is qualified as per JEDEC Standard 22, Method A117, and measured at −40°C, +25°C, +85°C, and +125°C.
[7] The data retention specification assumes a junction temperature (T_J) of 85°C as per JEDEC Standard 22, Method A117. Data retention lifetime decreases with T_J.
[8] These times do not include thermal settling and internal filter response times, which may affect overall accuracy.
[9] The $\overline{RST}$ line must be in a low state for at least 10 µs to ensure a proper reset initiation and recovery.
[10] Device functions at clock rates below 0.7 kHz, but at reduced performance levels.
[11] Supply current transients can reach 600 mA during initial start-up or reset recovery. See Figure 29 and Figure 30.

TIMING SPECIFICATIONS

T_A = 25°C, VDD = 3.3 V, unless otherwise noted.

Table 2.

		Normal Mode			
Parameter	**Description**	**Min[1]**	**Typ**	**Max[1]**	**Unit**
f_{SCLK}	Serial clock	0.01		15	MHz
t_{STALL}	Stall period between data	2			µs
t_{CLS}	Serial clock low period	31			ns
t_{CHS}	Serial clock high period	31			ns
$t_{\overline{CS}}$	Chip select to clock edge	32			ns
t_{DAV}	DOUT valid after SCLK edge			10	ns
t_{DSU}	DIN setup time before SCLK rising edge	2			ns
t_{DHD}	DIN hold time after SCLK rising edge	2			ns
t_{DR}, t_{DF}	DOUT rise/fall times, ≤100 pF loading		3	8	ns
t_{DSOE}	$\overline{CS}$ assertion to data out active	0		11	ns
t_{HD}	SCLK edge to data out invalid	0			ns
t_{SFS}	Last SCLK edge to $\overline{CS}$ deassertion	32			ns
t_{DSHI}	$\overline{CS}$ deassertion to data out high impedance	0		9	ns
t_1	Input sync pulse width	5			µs
t_2	Input sync to data invalid			407	µs
t_3	Input sync period	417			µs

[1] Guaranteed by design and characterization, but not tested in production.

Timing Diagrams

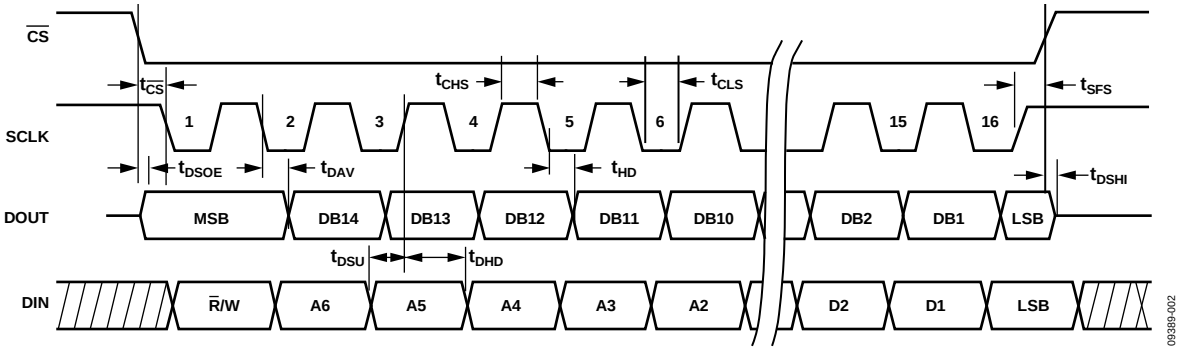


Figure 2. SPI Timing and Sequence

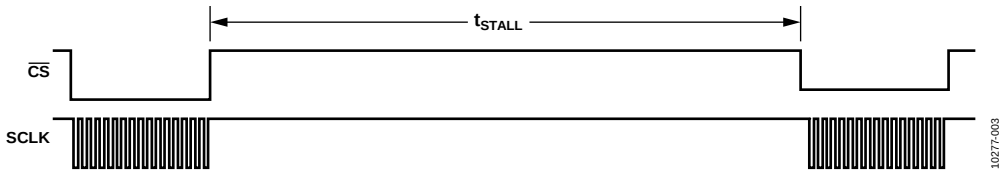


Figure 3. Stall Time and Data Rate

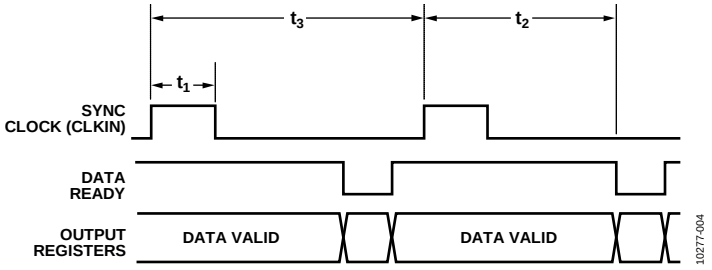


Figure 4. Input Clock Timing Diagram

ABSOLUTE MAXIMUM RATINGS

Table 3.

Parameter	Rating
Acceleration	
Any Axis, Unpowered	2000 *g*
Any Axis, Powered	2000 *g*
VDD to GND	−0.3 V to +3.6 V
Digital Input Voltage to GND	−0.3 V to VDD + 0.2 V
Digital Output Voltage to GND	−0.3 V to VDD + 0.2 V
Operating Temperature Range	−40°C to +85°C
Storage Temperature Range	−65°C to +150°C[1]
Barometric Pressure	2 bar

[1] Extended exposure to temperatures that are lower than −40°C or higher than +105°C can adversely affect the accuracy of the factory calibration.

Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is a stress rating only; functional operation of the device at these or any other conditions above those indicated in the operational section of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Table 4. Package Characteristics

Package Type	θ_{JA}	θ_{JC}	Device Weight
24-Lead Module (ML-24-6)	22.8°C/W	10.1°C/W	48 g

ESD CAUTION

ESD (electrostatic discharge) sensitive device. Charged devices and circuit boards can discharge without detection. Although this product features patented or proprietary protection circuitry, damage may occur on devices subjected to high energy ESD. Therefore, proper ESD precautions should be taken to avoid performance degradation or loss of functionality.

PIN CONFIGURATION AND FUNCTION DESCRIPTIONS

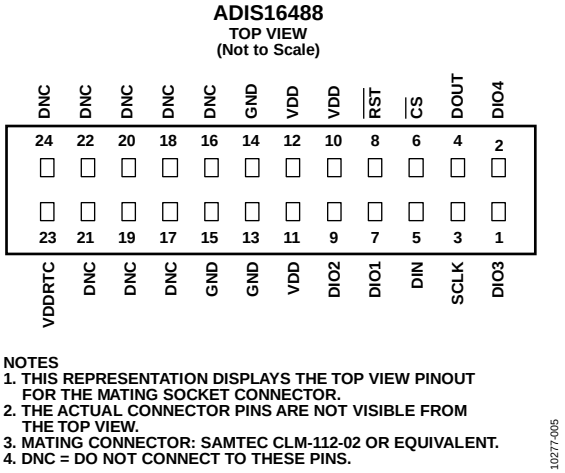


Figure 5. Mating Connector Pin Assignments

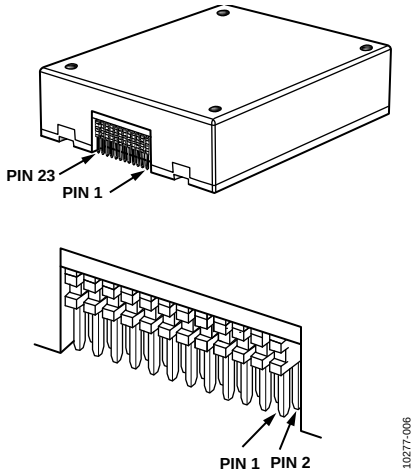


Figure 6. Axial Orientation (Top Side Facing Up)

Table 5. Pin Function Descriptions

Pin No.	Mnemonic	Type	Description
1	DIO3	Input/output	Configurable Digital Input/Output.
2	DIO4	Input/output	Configurable Digital Input/Output.
3	SCLK	Input	SPI Serial Clock.
4	DOUT	Output	SPI Data Output. Clocks output on SCLK falling edge.
5	DIN	Input	SPI Data Input. Clocks input on SCLK rising edge.
6	$\overline{CS}$	Input	SPI Chip Select.
7	DIO1	Input/output	Configurable Digital Input/Output.
8	$\overline{RST}$	Input	Reset.
9	DIO2	Input/output	Configurable Digital Input/Output.
10, 11, 12	VDD	Supply	Power Supply.
13, 14, 15	GND	Supply	Power Ground.
16 to 22, 24	DNC	Not applicable	Do Not Connect to These Pins.
23	VDDRTC	Supply	Real-Time Clock Power Supply.

TYPICAL PERFORMANCE CHARACTERISTICS

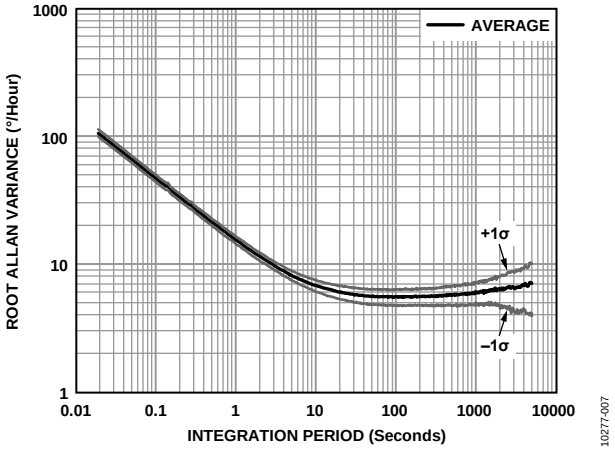


Figure 7. Gyroscope Allan Variance, 25°C

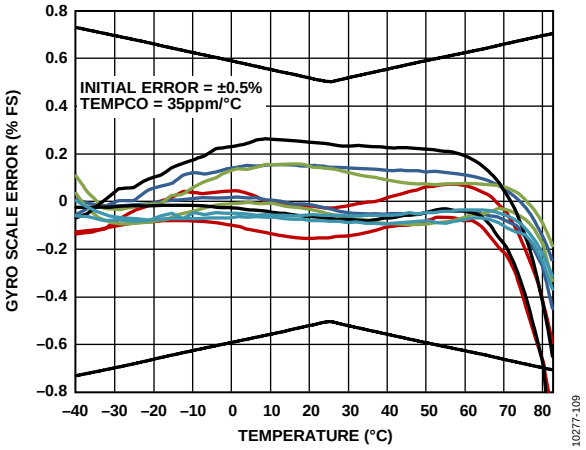


Figure 9. Gyroscope Scale (Sensitivity) Error and Hysteresis vs. Temperature

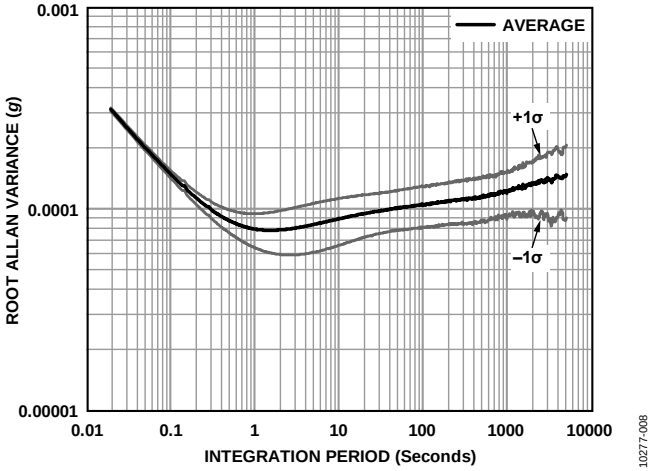


Figure 8. Accelerometer Allan Variance, 25°C

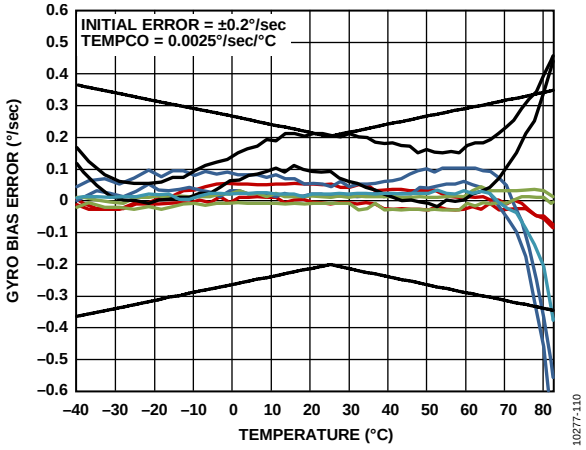


Figure 10. Gyroscope Bias Error and Hysteresis vs. Temperature

BASIC OPERATION

The ADIS16488 is an autonomous sensor system that starts up on its own when it has a valid power supply. After running through its initialization process, it begins sampling, processing, and loading calibrated sensor data into the output registers, which are accessible using the SPI port. The SPI port typically connects to a compatible port on an embedded processor, using the connection diagram in Figure 11. The four SPI signals facilitate synchronous, serial data communication. Connect $\overline{RST}$ (see Table 5) to VDD or leave it open for normal operation. The factory default configuration provides users with a data-ready signal on the DIO2 pin, which pulses high when new data is available in the output data registers.

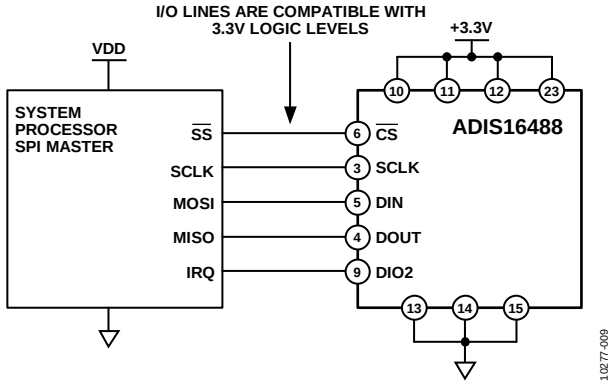


Figure 11. Electrical Connection Diagram

Table 6. Generic Master Processor Pin Names and Functions

Mnemonic	Function
$\overline{SS}$	Slave select
IRQ	Interrupt request
MOSI	Master output, slave input
MISO	Master input, slave output
SCLK	Serial clock

Embedded processors typically use control registers to configure their serial ports for communicating with SPI slave devices such as the ADIS16488. Table 7 provides a list of settings, which describe the SPI protocol of the ADIS16488. The initialization routine of the master processor typically establishes these settings using firmware commands to write them into its serial control registers.

Table 7. Generic Master Processor SPI Settings

Processor Setting	Description
Master	The ADIS16488 operates as a slave.
SCLK ≤ 15 MHz	Maximum serial clock rate.
SPI Mode 3	CPOL = 1 (polarity), and CPHA = 1 (phase).
MSB-First Mode	Bit sequence.
16-Bit Mode	Shift register/data length.

REGISTER STRUCTURE

The register structure and SPI port provide a bridge between the sensor processing system and an external, master processor. It contains both output data and control registers. The output data registers include the latest sensor data, a real-time clock, error flags, alarm flags, and identification data. The control registers include sample rate, filtering, input/output, alarms, calibration, and diagnostic configuration options. All communication between the ADIS16488 and an external processor involves either reading or writing to one of the user registers.

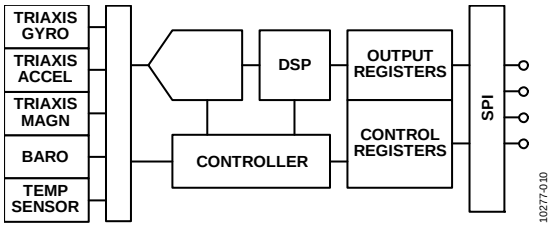


Figure 12. Basic Operation

The register structure uses a paged addressing scheme that is composed of 13 pages, with each one containing 64 register locations. Each register is 16 bits wide, with each byte having its own unique address within that page's memory map. The SPI port has access to one page at a time, using the bit sequence in Figure 17. Select the page to activate for SPI access by writing its code to the PAGE_ID register. Read the PAGE_ID register to determine which page is currently active. Table 8 displays the PAGE_ID contents for each page, along with their basic functions. The PAGE_ID register is located at Address 0x00 on every page.

Table 8. User Register Page Assignments

Page	PAGE_ID	Function
0	0x00	Output data, clock, identification
1	0x01	Reserved
2	0x02	Calibration
3	0x03	Control: sample rate, filtering, I/O, alarms
4	0x04	Serial number
5	0x05	FIR Filter Bank A Coefficient 0 to Coefficient 59
6	0x06	FIR Filter Bank A, Coefficient 60 to Coefficient 119
7	0x07	FIR Filter Bank B, Coefficient 0 to Coefficient 59
8	0x08	FIR Filter Bank B, Coefficient 60 to Coefficient 119
9	0x09	FIR Filter Bank C, Coefficient 0 to Coefficient 59
10	0x0A	FIR Filter Bank C, Coefficient 60 to Coefficient 119
11	0x0B	FIR Filter Bank D, Coefficient 0 to Coefficient 59
12	0x0C	FIR Filter Bank D, Coefficient 60 to Coefficient 119

SPI COMMUNICATION

The SPI port supports full duplex communication, as shown in Figure 17, which enables external processors to write to DIN while reading DOUT, if the previous command was a read request. Figure 17 provides a guideline for the bit coding on both DIN and DOUT.

DEVICE CONFIGURATION

The SPI provides write access to the control registers, one byte at a time, using the bit assignments shown in Figure 17. Each register has 16 bits, where Bits[7:0] represent the lower address (listed in Table 9) and Bits[15:8] represent the upper address. Write to the lower byte of a register first, followed by a write to its upper byte second. The only register that changes with a single write to its lower byte is the PAGE_ID register. For a write command, the first bit in the DIN sequence is set to 1. Address Bits[A6:A0] represent the target address, and Data Command Bits[DC7:DC0] represent the data being written to the location. Figure 13 provides an example of writing 0x03 to Address 0x00 (PAGE_ID [7:0]), using DIN = 0x8003. This write command activates the control page for SPI access.

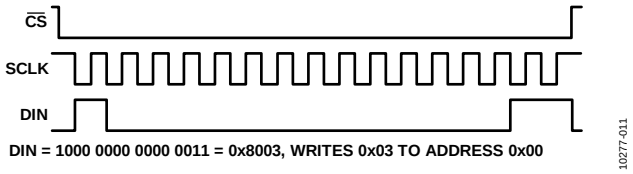


Figure 13. SPI Sequence for Activating the Control Page (DIN = 0x8003)

Dual Memory Structure

Writing configuration data to a control register updates its SRAM contents, which are volatile. After optimizing each relevant control register setting in a system, use the manual flash update command, which is located in GLOB_CMD[3] on Page 3 of the register map. Activate the manual flash update command by turning to Page 3 (DIN = 0x8003) and setting GLOB_CMD[3] = 1 (DIN = 0x8208, then DIN = 0x8300). Make sure that the power supply is within specification for the entire 375 ms processing time for a flash memory update. Table 9 provides a memory map for all of the user registers, which includes a column of flash backup information. A yes in this column indicates that a register has a mirror location in flash and, when backed up properly, automatically restores itself during startup or after a reset. Figure 14 provides a diagram of the dual memory structure used to manage operation and store critical user settings.

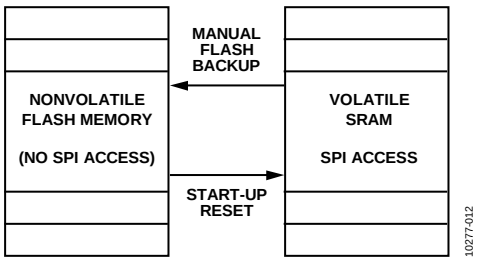


Figure 14. SRAM and Flash Memory Diagram

READING SENSOR DATA

The ADIS16488 automatically starts up and activates Page 0 for data register access. Write 0x00 to the PAGE_ID register (DIN = 0x8000) to activate Page 0 for data access after accessing any other page. A single register read requires two 16-bit SPI cycles. The first cycle requests the contents of a register using the bit assignments in Figure 17, and then the register contents follow DOUT during the second sequence. The first bit in a DIN command is zero, followed by either the upper or lower address for the register. The last eight bits are don't care, but the SPI requires the full set of 16 SCLKs to receive the request. Figure 15 includes two register reads in succession, which starts with DIN = 0x1A00 to request the contents of the Z_GYRO_OUT register and follows with 0x1800 to request the contents of the Z_GYRO_LOW register.

Figure 15. SPI Read Example

Figure 16 provides an example of the four SPI signals when reading PROD_ID in a repeating pattern. This is a good pattern to use for troubleshooting the SPI interface setup and communications because the contents of PROD_ID are predefined and stable.

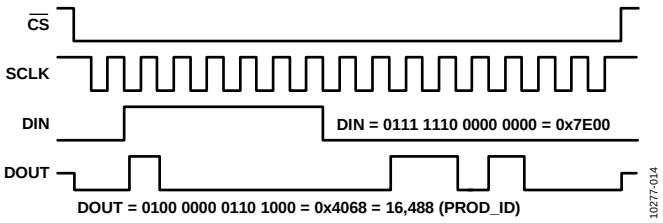
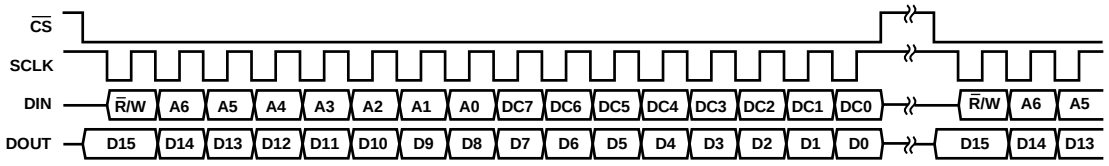


Figure 16. SPI Read Example, Second 16-Bit Sequence



NOTES
1. DOUT BITS ARE PRODUCED ONLY WHEN THE PREVIOUS 16-BIT DIN SEQUENCE STARTS WITH R/W = 0.
2. WHEN CS IS HIGH, DOUT IS IN A THREE-STATE, HIGH IMPEDANCE MODE, WHICH ALLOWS MULTIFUNCTIONAL USE OF THE LINE FOR OTHER DEVICES.

Figure 17. SPI Communication Bit Sequence

USER REGISTERS

Table 9. User Register Memory Map (N/A = Not Applicable)

Name	R/W	Flash	PAGE_ID	Address	Default	Register Description	Format
PAGE_ID	R/W	No	0x00	0x00	0x00	Page identifier	N/A
Reserved	N/A	N/A	0x00	0x02 to 0x04	N/A	Reserved	N/A
SEQ_CNT	R	No	0x00	0x06	N/A	Sequence counter	Table 56
SYS_E_FLAG	R	No	0x00	0x08	0x0000	Output, system error flags	Table 47
DIAG_STS	R	No	0x00	0x0A	0x0000	Output, self-test error flags	Table 48
ALM_STS	R	No	0x00	0x0C	0x0000	Output, alarm error flags	Table 49
TEMP_OUT	R	No	0x00	0x0E	N/A	Output, temperature	Table 45
X_GYRO_LOW	R	No	0x00	0x10	N/A	Output, x-axis gyroscope, low word	Table 14
X_GYRO_OUT	R	No	0x00	0x12	N/A	Output, x-axis gyroscope, high word	Table 10
Y_GYRO_LOW	R	No	0x00	0x14	N/A	Output, y-axis gyroscope, low word	Table 15
Y_GYRO_OUT	R	No	0x00	0x16	N/A	Output, y-axis gyroscope, high word	Table 11
Z_GYRO_LOW	R	No	0x00	0x18	N/A	Output, z-axis gyroscope, low word	Table 16
Z_GYRO_OUT	R	No	0x00	0x1A	N/A	Output, z-axis gyroscope, high word	Table 12
X_ACCL_LOW	R	No	0x00	0x1C	N/A	Output, x-axis accelerometer, low word	Table 21
X_ACCL_OUT	R	No	0x00	0x1E	N/A	Output, x-axis accelerometer, high word	Table 17
Y_ACCL_LOW	R	No	0x00	0x20	N/A	Output, y-axis accelerometer, low word	Table 22
Y_ACCL_OUT	R	No	0x00	0x22	N/A	Output, y-axis accelerometer, high word	Table 18
Z_ACCL_LOW	R	No	0x00	0x24	N/A	Output, z-axis accelerometer, low word	Table 23
Z_ACCL_OUT	R	No	0x00	0x26	N/A	Output, z-axis accelerometer, high word	Table 19
X_MAGN_OUT	R	No	0x00	0x28	N/A	Output, x-axis magnetometer, high word	Table 38
Y_MAGN_OUT	R	No	0x00	0x2A	N/A	Output, y-axis magnetometer, high word	Table 39
Z_MAGN_OUT	R	No	0x00	0x2C	N/A	Output, z-axis magnetometer, high word	Table 40
BAROM_LOW	R	No	0x00	0x2E	N/A	Output, barometer, low word	Table 44
BAROM_OUT	R	No	0x00	0x30	N/A	Output, barometer, high word	Table 42
Reserved	N/A	N/A	0x00	0x32 to 0x3E	N/A	Reserved	N/A
X_DELTANG_LOW	R	No	0x00	0x40	N/A	Output, x-axis delta angle, low word	Table 28
X_DELTANG_OUT	R	No	0x00	0x42	N/A	Output, x-axis delta angle, high word	Table 24
Y_DELTANG_LOW	R	No	0x00	0x44	N/A	Output, y-axis delta angle, low word	Table 29
Y_DELTANG_OUT	R	No	0x00	0x46	N/A	Output, y-axis delta angle, high word	Table 25
Z_DELTANG_LOW	R	No	0x00	0x48	N/A	Output, z-axis delta angle, low word	Table 30
Z_DELTANG_OUT	R	No	0x00	0x4A	N/A	Output, z-axis delta angle, high word	Table 26
X_DELTVEL_LOW	R	No	0x00	0x4C	N/A	Output, x-axis delta velocity, low word	Table 35
X_DELTVEL_OUT	R	No	0x00	0x4E	N/A	Output, x-axis delta velocity, high word	Table 31
Y_DELTVEL_LOW	R	No	0x00	0x50	N/A	Output, y-axis delta velocity, low word	Table 36
Y_DELTVEL_OUT	R	No	0x00	0x52	N/A	Output, y-axis delta velocity, high word	Table 32
Z_DELTVEL_LOW	R	No	0x00	0x54	N/A	Output, z-axis delta velocity, low word	Table 37
Z_DELTVEL_OUT	R	No	0x00	0x56	N/A	Output, z-axis delta velocity, high word	Table 33
Reserved	N/A	N/A	0x00	0x58 to 0x76	N/A	Reserved	N/A
TIME_MS_OUT	R/W	Yes	0x00	0x78	N/A	Factory configuration time: minutes/seconds	Table 124
TIME_DH_OUT	R/W	Yes	0x00	0x7A	N/A	Factory configuration date/time: day/hour	Table 125
TIME_YM_OUT	R/W	Yes	0x00	0x7C	N/A	Factory configuration date: year/month	Table 126
PROD_ID	R	Yes	0x00	0x7E	0x4068	Output, product identification (16,488)	Table 53
Reserved	N/A	N/A	0x01	0x00 to 0x7E	N/A	Reserved	N/A
PAGE_ID	R/W	No	0x02	0x00	0x00	Page identifier	N/A
Reserved	N/A	N/A	0x02	0x02	N/A	Reserved	N/A
X_GYRO_SCALE	R/W	Yes	0x02	0x04	0x0000	Calibration, scale, x-axis gyroscope	Table 71
Y_GYRO_SCALE	R/W	Yes	0x02	0x06	0x0000	Calibration, scale, y-axis gyroscope	Table 72
Z_GYRO_SCALE	R/W	Yes	0x02	0x08	0x0000	Calibration, scale, z-axis gyroscope	Table 73
X_ACCL_SCALE	R/W	Yes	0x02	0x0A	0x0000	Calibration, scale, x-axis accelerometer	Table 81
Y_ACCL_SCALE	R/W	Yes	0x02	0x0C	0x0000	Calibration, scale, y-axis accelerometer	Table 82
Z_ACCL_SCALE	R/W	Yes	0x02	0x0E	0x0000	Calibration, scale, z-axis accelerometer	Table 83

Name	R/W	Flash	PAGE_ID	Address	Default	Register Description	Format
XG_BIAS_LOW	R/W	Yes	0x02	0x10	0x0000	Calibration, offset, gyroscope, x-axis, low word	Table 67
XG_BIAS_HIGH	R/W	Yes	0x02	0x12	0x0000	Calibration, offset, gyroscope, x-axis, high word	Table 64
YG_BIAS_LOW	R/W	Yes	0x02	0x14	0x0000	Calibration, offset, gyroscope, y-axis, low word	Table 68
YG_BIAS_HIGH	R/W	Yes	0x02	0x16	0x0000	Calibration, offset, gyroscope, y-axis, high word	Table 65
ZG_BIAS_LOW	R/W	Yes	0x02	0x18	0x0000	Calibration, offset, gyroscope, z-axis, low word	Table 69
ZG_BIAS_HIGH	R/W	Yes	0x02	0x1A	0x0000	Calibration, offset, gyroscope, z-axis, high word	Table 66
XA_BIAS_LOW	R/W	Yes	0x02	0x1C	0x0000	Calibration, offset, accelerometer, x-axis, low word	Table 78
XA_BIAS_HIGH	R/W	Yes	0x02	0x1E	0x0000	Calibration, offset, accelerometer, x-axis, high word	Table 75
YA_BIAS_LOW	R/W	Yes	0x02	0x20	0x0000	Calibration, offset, accelerometer, y-axis, low word	Table 79
YA_BIAS_HIGH	R/W	Yes	0x02	0x22	0x0000	Calibration, offset, accelerometer, y-axis, high word	Table 76
ZA_BIAS_LOW	R/W	Yes	0x02	0x24	0x0000	Calibration, offset, accelerometer, z-axis, low word	Table 80
ZA_BIAS_HIGH	R/W	Yes	0x02	0x26	0x0000	Calibration, offset, accelerometer, z-axis, high word	Table 77
HARD_IRON_X	R/W	Yes	0x02	0x28	0x0000	Calibration, hard iron, magnetometer, x-axis	Table 84
HARD_IRON_Y	R/W	Yes	0x02	0x2A	0x0000	Calibration, hard iron, magnetometer, y-axis	Table 85
HARD_IRON_Z	R/W	Yes	0x02	0x2C	0x0000	Calibration, hard iron, magnetometer, z-axis	Table 86
SOFT_IRON_S11	R/W	Yes	0x02	0x2E	0x0000	Calibration, soft iron, magnetometer, S11	Table 88
SOFT_IRON_S12	R/W	Yes	0x02	0x30	0x0000	Calibration, soft iron, magnetometer, S12	Table 89
SOFT_IRON_S13	R/W	Yes	0x02	0x32	0x0000	Calibration, soft iron, magnetometer, S13	Table 90
SOFT_IRON_S21	R/W	Yes	0x02	0x34	0x0000	Calibration, soft iron, magnetometer, S21	Table 91
SOFT_IRON_S22	R/W	Yes	0x02	0x36	0x0000	Calibration, soft iron, magnetometer, S22	Table 92
SOFT_IRON_S23	R/W	Yes	0x02	0x38	0x0000	Calibration, soft iron, magnetometer, S23	Table 93
SOFT_IRON_S31	R/W	Yes	0x02	0x3A	0x0000	Calibration, soft iron, magnetometer, S31	Table 94
SOFT_IRON_S32	R/W	Yes	0x02	0x3C	0x0000	Calibration, soft iron, magnetometer, S32	Table 95
SOFT_IRON_S33	R/W	Yes	0x02	0x3E	0x0000	Calibration, soft iron, magnetometer, S33	Table 96
BR_BIAS_LOW	R/W	Yes	0x02	0x40	0x0000	Calibration, offset, barometer, low word	Table 99
BR_BIAS_HIGH	R/W	Yes	0x02	0x42	0x0000	Calibration, offset, barometer, high word	Table 98
Reserved	N/A	N/A	0x02	0x44 to 0x72	N/A	Reserved	N/A
USER_SCR_1	R/W	Yes	0x02	0x74	0x0000	User Scratch Register 1	Table 120
USER_SCR_2	R/W	Yes	0x02	0x76	0x0000	User Scratch Register 2	Table 121
USER_SCR_3	R/W	Yes	0x02	0x78	0x0000	User Scratch Register 3	Table 122
USER_SCR_4	R/W	Yes	0x02	0x7A	0x0000	User Scratch Register 4	Table 123
FLSHCNT_LOW	R	Yes	0x02	0x7C	N/A	Diagnostic, flash memory count, low word	Table 115
FLSHCNT_HIGH	R	Yes	0x02	0x7E	N/A	Diagnostic, flash memory count, high word	Table 116
PAGE_ID	R/W	No	0x03	0x00	0x0000	Page identifier	N/A
GLOB_CMD	W	No	0x03	0x02	N/A	Control, global commands	Table 114
Reserved	N/A	N/A	0x03	0x04	N/A	Reserved	N/A
FNCTIO_CTRL	R/W	Yes	0x03	0x06	0x000D	Control, I/O pins, functional definitions	Table 117
GPIO_CTRL	R/W	Yes	0x03	0x08	0x00X0[1]	Control, I/O pins, general purpose	Table 118
CONFIG	R/W	Yes	0x03	0x0A	0x00C0	Control, clock, and miscellaneous correction	Table 74
DEC_RATE	R/W	Yes	0x03	0x0C	0x0000	Control, output sample rate decimation	Table 55
NULL_CNFG	R/W	Yes	0x03	0x0E	0x070A	Control, automatic bias correction configuration	Table 70
SLP_CNT	R/W	No	0x03	0x10	N/A	Control, power-down/sleep mode	Table 119
Reserved	N/A	N/A	0x03	0x12 to 0x14	N/A	Reserved	N/A
FILTR_BNK_0	R/W	Yes	0x03	0x16	0x0000	Filter selection	Table 57
FILTR_BNK_1	R/W	Yes	0x03	0x18	0x0000	Filter selection	Table 58
Reserved	N/A	N/A	0x03	0x1A to 0x1E	N/A	Reserved	N/A
ALM_CNFG_0	R/W	Yes	0x03	0x20	0x0000	Alarm configuration	Table 110
ALM_CNFG_1	R/W	Yes	0x03	0x22	0x0000	Alarm configuration	Table 111
ALM_CNFG_2	R/W	Yes	0x03	0x24	0x0000	Alarm configuration	Table 112
Reserved	N/A	N/A	0x03	0x26	N/A	Reserved	N/A
XG_ALM_MAGN	R/W	Yes	0x03	0x28	0x0000	Alarm, x-axis gyroscope threshold setting	Table 100
YG_ALM_MAGN	R/W	Yes	0x03	0x2A	0x0000	Alarm, y-axis gyroscope threshold setting	Table 101
ZG_ALM_MAGN	R/W	Yes	0x03	0x2C	0x0000	Alarm, z-axis gyroscope threshold setting	Table 102

Name	R/W	Flash	PAGE_ID	Address	Default	Register Description	Format
XA_ALM_MAGN	R/W	Yes	0x03	0x2E	0x0000	Alarm, x-axis accelerometer threshold	Table 103
YA_ALM_MAGN	R/W	Yes	0x03	0x30	0x0000	Alarm, y-axis accelerometer threshold	Table 104
ZA_ALM_MAGN	R/W	Yes	0x03	0x32	0x0000	Alarm, z-axis accelerometer threshold	Table 105
XM_ALM_MAGN	R/W	Yes	0x03	0x34	0x0000	Alarm, x-axis magnetometer threshold	Table 106
YM_ALM_MAGN	R/W	Yes	0x03	0x36	0x0000	Alarm, y-axis magnetometer threshold	Table 107
ZM_ALM_MAGN	R/W	Yes	0x03	0x38	0x0000	Alarm, z-axis magnetometer threshold	Table 108
BR_ALM_MAGN	R/W	Yes	0x03	0x3A	0x0000	Alarm, barometer threshold setting	Table 109
Reserved	N/A	N/A	0x03	0x3C to 0x76	N/A	Reserved	N/A
FIRM_REV	R	Yes	0x03	0x78	N/A	Firmware revision	Table 50
FIRM_DM	R	Yes	0x03	0x7A	N/A	Firmware programming date: day/month	Table 51
FIRM_Y	R	Yes	0x03	0x7C	N/A	Firmware programming date: year	Table 52
Reserved	N/A	N/A	0x03	0x7E	N/A	Reserved	N/A
Reserved	N/A	N/A	0x04	0x00 to 0x18	N/A	Reserved	N/A
SERIAL_NUM	R	Yes	0x04	0x20	N/A	Serial number	Table 54
Reserved	N/A	N/A	0x04	0x22 to 0x7F	N/A	Reserved	N/A
PAGE_ID	R/W	No	0x05	0x00	0x0000	Page identifier	N/A
FIR_COEF_Axxx	R/W	Yes	0x05	0x02 to 0x7E	N/A	FIR Filter Bank A, Coefficients 0 through 59	Table 59
PAGE_ID	R/W	No	0x06	0x00	0x0000	Page identifier	N/A
FIR_COEF_Axxx	R/W	Yes	0x06	0x02 to 0x7E	N/A	FIR Filter Bank A, Coefficients 60 through 119	Table 59
PAGE_ID	R/W	No	0x07	0x00	0x0000	Page identifier	N/A
FIR_COEF_Bxxx	R/W	Yes	0x07	0x02 to 0x7E	N/A	FIR Filter Bank B, Coefficients 0 through 59	Table 60
PAGE_ID	R/W	No	0x08	0x00	0x0000	Page identifier	N/A
FIR_COEF_Bxxx	R/W	Yes	0x08	0x02 to 0x7E	N/A	FIR Filter Bank B, Coefficients 60 through 119	Table 60
PAGE_ID	R/W	No	0x09	0x00	0x0000	Page identifier	N/A
FIR_COEF_Cxxx	R/W	Yes	0x09	0x02 to 0x7E	N/A	FIR Filter Bank C, Coefficients 0 through 59	Table 61
PAGE_ID	R/W	No	0x0A	0x00	0x0000	Page identifier	N/A
FIR_COEF_Cxxx	R/W	Yes	0x0A	0x02 to 0x7E	N/A	FIR Filter Bank C, Coefficients 60 through 119	Table 61
PAGE_ID	R/W	No	0x0B	0x00	0x0000	Page identifier	N/A
FIR_COEF_Dxxx	R/W	Yes	0x0B	0x02 to 0x7E	N/A	FIR Filter Bank D, Coefficients 0 through 59	Table 62
PAGE_ID	R/W	No	0x0C	0x00	0x0000	Page identifier	N/A
FIR_COEF_Dxxx	R/W	Yes	0x0C	0x02 to 0x7E	N/A	FIR Filter Bank D, Coefficients 60 through 119	Table 62

[1] The GPIO_CTRL[7:4] bits reflect the logic levels on the DIOx lines and do not have a default setting.

OUTPUT DATA REGISTERS

After the ADIS16488 completes its start-up process, the PAGE_ID register contains 0x0000, which sets Page 0 as the active page for SPI access. Page 0 contains the output data, real-time clock, status, and product identification registers.

INERTIAL SENSOR DATA FORMAT

The gyroscope, accelerometer, delta angle, delta velocity, and barometer output data registers use a 32-bit, twos complement format. Each output uses two registers to support this resolution. Figure 18 provides an example of how each register contributes to each inertial measurement. In this case, X_GYRO_OUT is the most significant word (upper 16 bits), and X_GYRO_LOW is the least significant word (lower 16 bits). In many cases, using the most significant word registers alone provides sufficient resolution for preserving key performance metrics.

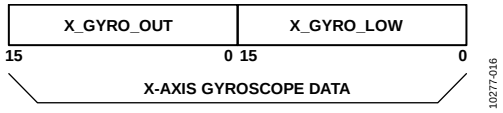


Figure 18. Gyroscope Output Format Example, DEC_RATE > 0

The arrows in Figure 19 describe the direction of the motion, which produces a positive output response in each sensor's output register. The accelerometers respond to both dynamic and static forces associated with acceleration, including gravity. When lying perfectly flat, as shown in Figure 19, the z-axis accelerometer output is 1 *g*, and the x and y accelerometers are 0 *g*.

ROTATION RATE (GYROSCOPE)

The registers that use the x_GYRO_OUT format are the primary registers for the gyroscope measurements (see Table 10, Table 11, and Table 12). When processing data from these registers, use a 16-bit, twos complement data format. Table 13 provides x_GYRO_OUT digital coding examples.

Table 10. X_GYRO_OUT (Page 0, Base Address = 0x12)

Bits	Description
[15:0]	X-axis gyroscope data; twos complement, ±450°/sec range, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 11. Y_GYRO_OUT (Page 0, Base Address = 0x16)

Bits	Description
[15:0]	Y-axis gyroscope data; twos complement, ±450°/sec range, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 12. Z_GYRO_OUT (Page 0, Base Address = 0x1A)

Bits	Description
[15:0]	Z-axis gyroscope data; twos complement, ±450°/sec range, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 13. x_GYRO_OUT Data Format Examples

Rotation Rate	Decimal	Hex	Binary
+450°/sec	+22,500	0x57E4	0101 0111 1110 0100
+0.04/sec	+2	0x0002	0000 0000 0000 0010
+0.02°/sec	+1	0x0001	0000 0000 0000 0001
0°/sec	0	0x0000	0000 0000 0000 0000
−0.02°/sec	−1	0xFFFF	1111 1111 1111 1111
−0.04°/sec	−2	0xFFFE	1111 1111 1111 1110
−450°/sec	−22,500	0xA81C	1010 1000 0001 1100

The registers that use the x_GYRO_LOW naming format provide additional resolution for the gyroscope measurements (see Table 14, Table 15, and Table 16). The MSB has a weight of 0.01°/sec, and each subsequent bit has ½ the weight of the previous one.

Table 14. X_GYRO_LOW (Page 0, Base Address = 0x10)

Bits	Description
[15:0]	X-axis gyroscope data; additional resolution bits

Table 15. Y_GYRO_LOW (Page 0, Base Address = 0x14)

Bits	Description
[15:0]	Y-axis gyroscope data; additional resolution bits

Table 16. Z_GYRO_LOW (Page 0, Base Address = 0x18)

Bits	Description
[15:0]	Z-axis gyroscope data; additional resolution bits

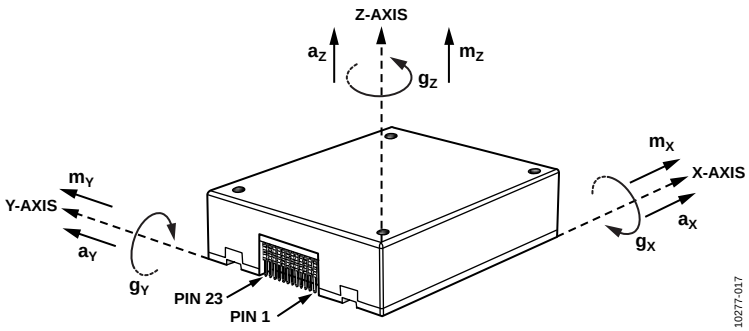


Figure 19. Inertial Sensor Direction Reference Diagram

ACCELERATION

The registers that use the x_ACCL_OUT format are the primary registers for the accelerometer measurements (see Table 17, Table 18, and Table 19). When processing data from these registers, use a 16-bit, twos complement data format. Table 20 provides x_ACCL_OUT digital coding examples.

Table 17. X_ACCL_OUT (Page 0, Base Address = 0x1E)

Bits	Description
[15:0]	X-axis accelerometer data; twos complement, ±18 *g* range, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 18. Y_ACCL_OUT (Page 0, Base Address = 0x22)

Bits	Description
[15:0]	Y-axis accelerometer data; twos complement, ±18 *g* range, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 19. Z_ACCL_OUT (Page 0, Base Address = 0x26)

Bits	Description
[15:0]	Z-axis accelerometer data; twos complement, ±18 *g* range, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 20. x_ACCL_OUT Data Format Examples

Acceleration	Decimal	Hex	Binary
+18 *g*	+22,500	0x57E4	0101 0111 1110 0100
+1.6 m*g*	+2	0x0002	0000 0000 0000 0010
+0.8 m*g*	+1	0x0001	0000 0000 0000 0001
0 m*g*	0	0x0000	0000 0000 0000 0000
−0.8 m*g*	−1	0xFFFF	1111 1111 1111 1111
−1.6 m*g*	−2	0xFFFE	1111 1111 1111 1110
−18 *g*	−22,500	0xA81C	1010 1000 0001 1100

The registers that use the x_ACCL_LOW naming format provide additional resolution for the accelerometer measurements (see Table 21, Table 22, and Table 23). The MSB has a weight of 0.4 m*g*, and each subsequent bit has ½ the weight of the previous one.

Table 21. X_ACCL_LOW (Page 0, Base Address = 0x1C)

Bits	Description
[15:0]	X-axis accelerometer data; additional resolution bits

Table 22. Y_ACCL_LOW (Page 0, Base Address = 0x20)

Bits	Description
[15:0]	Y-axis accelerometer data; additional resolution bits

Table 23. Z_ACCL_LOW (Page 0, Base Address = 0x24)

Bits	Description
[15:0]	Z-axis accelerometer data; additional resolution bits

DELTA ANGLES

The x_DELTANG_OUT registers are the primary output registers for the delta angle calculations. When processing data from these registers, use a 16-bit, twos complement data format (see Table 24, Table 25, and Table 26). Table 27 provides x_DELTANG_OUT digital coding examples.

The delta angle outputs represent an integration of the gyroscope measurements and use the following formula for all three axes (x-axis displayed):

$$\Delta\theta_{x,nD} = \frac{1}{2f_S} \times \sum_{d=0}^{D-1} \left(\omega_{x,nD+d} + \omega_{x,nD+d-1}\right)$$

where:
ω_x is the x-axis rate of rotation (gyroscope).
f_S is the sample rate.
n is the sample time prior to the decimation filter.
D is the decimation rate (D = DEC_RATE + 1)

When using the internal sample clock, f_S is equal to 2,460 SPS. When using the external clock option, f_S is equal to the frequency of the external clock, which is limited to a minimum of 2 kHz to prevent overflow in the x_DELTANG_xxx registers at high rotation rates. See Table 55 and Figure 20 for more information on the DEC_RATE register (decimation filter).

Table 24. X_DELTANG_OUT (Page 0, Base Address = 0x42)

Bits	Description
[15:0]	X-axis delta angle data; twos complement, ±720° range, 0° = 0x0000, 1 LSB = 720°/2^{15} = ~0.022°

Table 25. Y_DELTANG_OUT (Page 0, Base Address = 0x46)

Bits	Description
[15:0]	Y-axis delta angle data; twos complement, ±720° range, 0° = 0x0000, 1 LSB = 720°/2^{15} = ~0.022°

Table 26. Z_DELTANG_OUT (Page 0, Base Address = 0x4A)

Bits	Description
[15:0]	Z-axis delta angle data; twos complement, ±720° range, 0° = 0x0000, 1 LSB = 720°/2^{15} = ~0.022°

Table 27. x_DELTANG_OUT Data Format Examples

Angle (°)	Decimal	Hex	Binary
+720 × (2^{15} − 1)/2^{15}	+32,767	0x7FFF	0111 1111 1110 1111
+1440/2^{15}	+2	0x0002	0000 0000 0000 0010
+720/2^{15}	+1	0x0001	0000 0000 0000 0001
0	0	0x0000	0000 0000 0000 0000
−720/2^{15}	−1	0xFFFF	1111 1111 1111 1111
−1440/2^{15}	−2	0xFFFE	1111 1111 1111 1110
−720	−32,768	0x8000	1000 0000 0000 0000

The x_DELTANG_LOW registers (see Table 28, Table 29 and Table 30) provide additional resolution bits for the delta-angle measurement and combine with the x_DELTANG_OUT registers to provide a 32-bit, twos complement number. The MSBs in the x_DELTANG_ LOW registers have a weight of ~0.011° (720°/2^{16}), and each subsequent bit carries a weight of ½ of the previous one.

Table 28. X_DELTANG_LOW (Page 0, Base Address = 0x40)

Bits	Description
[15:0]	X-axis delta angle data; additional resolution bits

Table 29. Y_DELTANG_LOW (Page 0, Base Address = 0x44)

Bits	Description
[15:0]	Y-axis delta angle data; additional resolution bits

Table 30. Z_DELTANG_LOW (Page 0, Base Address = 0x48)

Bits	Description
[15:0]	Z-axis delta angle data; additional resolution bits

DELTA VELOCITY

The registers that use the x_DELTVEL_OUT format are the primary registers for the delta velocity calculations. When processing data from these registers, use a 16-bit, twos complement data format (see Table 31, Table 32, and Table 33). Table 34 provides x_DELTVEL_OUT digital coding examples.

The delta velocity outputs represent an integration of the accelerometer measurements and use the following formula for all three axes (x-axis displayed):

$$\Delta V_{x,nD} = \frac{1}{2f_S} \times \sum_{d=0}^{D-1} \left(a_{x,n\,D+\,d} + a_{x,nD+d-1} \right)$$

where:
a_x is the x-axis linear acceleration.
f_S is the sample rate.
n is the sample time prior to the decimation filter.
D is the decimation rate (D = DEC_RATE + 1)

When using the internal sample clock, f_S is equal to 2,460 SPS. When using the external clock option, f_S is equal to the frequency of the external clock, which is limited to a minimum of 2 kHz to prevent overflow in the x_DELTVEL_xxx registers at high rotation rates. See Table 55 and Figure 20 for more information on the DEC_RATE register.

Table 31. X_DELTVEL_OUT (Page 0, Base Address = 0x4E)

Bits	Description
[15:0]	X-axis delta velocity data; twos complement, ±200 m/sec range, 0 m/sec = 0x0000 1 LSB = 200 m/sec ÷ 2^{15} = ~6.104 mm/sec

Table 32. Y_DELTVEL_OUT (Page 0, Base Address = 0x52)

Bits	Description
[15:0]	Y-axis delta velocity data; twos complement, ±200 m/sec range, 0 m/sec = 0x0000 1 LSB = 200 m/sec ÷ 2^{15} = ~6.104 mm/sec

Table 33. Z_DELTVEL_OUT (Page 0, Base Address = 0x56)

Bits	Description
[15:0]	Z-axis delta velocity data; twos complement, ±200 m/sec range, 0 m/sec = 0x0000 1 LSB = 200 m/sec ÷ 2^{15} = ~6.104 mm/sec

Table 34. x_DELTVEL_OUT, Data Format Examples

Velocity (m/sec)	Decimal	Hex	Binary
$+200 \times (2^{15} - 1)/2^{15}$	+32,767	0x7FFF	0111 1111 1111 1111
$+400/2^{15}$	+2	0x0002	0000 0000 0000 0010
$+200/2^{15}$	+1	0x0001	0000 0000 0000 0001
0	0	0x0000	0000 0000 0000 0000
$-200/2^{15}$	−1	0xFFFF	1111 1111 1111 1111
$-400/2^{15}$	−2	0xFFFE	1111 1111 1111 1110
−200	−32,768	0x8000	1000 0000 0000 0000

The x_DELTVEL_LOW registers (see Table 35, Table 36 and Table 37) provide additional resolution bits for the delta-velocity measurement and combine with the x_DELTVEL_OUT registers to provide a 32-bit, twos complement number. The MSBs in the x_DELTVEL _LOW registers have a weight of ~3.052 mm/sec (200 m/sec ÷ 2^{16}), and each subsequent bit carries a weight of ½ of the previous one.

Table 35. X_DELTVEL_LOW (Page 0, Base Address = 0x4C)

Bits	Description
[15:0]	X-axis delta velocity data; additional resolution bits

Table 36. Y_DELTVEL_LOW (Page 0, Base Address = 0x50)

Bits	Description
[15:0]	Y-axis delta velocity data; additional resolution bits

Table 37. Z_DELTVEL_LOW (Page 0, Base Address = 0x54)

Bits	Description
[15:0]	Z-axis delta velocity data; additional resolution bits

MAGNETOMETERS

The registers that use the x_MAGN_OUT format are the primary registers for the magnetometer measurements. When processing data from these registers, use a 16-bit, twos complement data format. Table 38, Table 39, and Table 40 provide each register's numerical format, and Table 41 provides x_MAGN_OUT digital coding examples.

Table 38. X_MAGN_OUT (Page 0, Base Address = 0x28)

Bits	Description
[15:0]	X-axis magnetometer data; twos complement, ±3.2767 gauss range, 0 gauss = 0x0000, 1 LSB = 0.1 mgauss

Table 39. Y_MAGN_OUT (Page 0, Base Address = 0x2A)

Bits	Description
[15:0]	Y-axis magnetometer data; twos complement, ±3.2767 gauss range, 0 gauss = 0x0000, 1 LSB = 0.1 mgauss

Table 40. Z_MAGN_OUT (Page 0, Base Address = 0x2C)

Bits	Description
[15:0]	Z-axis magnetometer data; twos complement, ±3.2767 gauss range, 0 gauss = 0x0000, 1 LSB = 0.1 mgauss

Table 41. x_MAGN_OUT Data Format Examples

Magnetic Field	Decimal	Hex	Binary
+3.2767 gauss	+32,767	0x7FFF	0111 1111 1111 1111
+0.2 mgauss	+2	0x0002	0000 0000 0000 0010
+0.1 mgauss	+1	0x0001	0000 0000 0000 0001
0 gauss	0	0x0000	0000 0000 0000 0000
−0.1 mgauss	−1	0xFFFF	1111 1111 1111 1111
−0.2 mgauss	−2	0xFFFE	1111 1111 1111 1110
−3.2768 gauss	−32,768	0x8000	1000 0000 0000 0000

BAROMETER

The BAROM_OUT register (see Table 42) and BAROM_LOW register (see Table 44) provide access to the barometric pressure data. These two registers combine to provide a 32-bit, twos complement format. Some applications are able to use BAROM_OUT by itself. For cases where the finer resolution available from BAROM_LOW is valuable, combine them in the same manner as the gyroscopes (see Figure 18). When processing data from the BAROM_OUT register alone, use a 16-bit, twos complement data format. Table 42 provides the numerical format in BAROM_OUT, and Table 43 provides digital coding examples.

Table 42. BAROM_OUT (Page 0, Base Address = 0x30)

Bits	Description
[15:0]	Barometric pressure; twos complement, ±1.31 bar range, 0 bar = 0x0000, 40 µbar/LSB

Table 43. BAROM_OUT Data Format Examples

Pressure (bar)	Decimal	Hex	Binary
$+0.00004 \times (2^{15} - 1)$	+32,767	0x7FFF	0111 1111 1110 1111
+0.00008	+2	0x0002	0000 0000 0000 0010
+0.00004	+1	0x0001	0000 0000 0000 0001
0	0	0x0000	0000 0000 0000 0000
−0.00004	−1	0xFFFF	1111 1111 1111 1111
−0.00008	−2	0xFFFE	1111 1111 1111 1110
-0.00004×2^{15}	−32,768	0x8000	1000 0000 0000 0000

The BAROM_LOW register provides additional resolution for the barometric pressure measurement. The MSB has a weight of 20 µbar, and each subsequent bit carries a weight of ½ of the previous one.

Table 44. BAROM_LOW (Page 0, Base Address = 0x2E)

Bits	Description
[15:0]	Barometric pressure; additional resolution bits

INTERNAL TEMPERATURE

The TEMP_OUT register provides an internal temperature measurement that can be useful for observing relative temperature changes inside of the ADIS16488 (see Table 45). Table 46 provides TEMP_OUT digital coding examples. Note that this temperature reflects a higher temperature than ambient, due to self-heating.

Table 45. TEMP_OUT (Page 0, Base Address = 0x0E)

Bits	Description
[15:0]	Temperature data; twos complement, 0.00565°C per LSB, 25°C = 0x0000

Table 46. TEMP_OUT Data Format Examples

Temperature (°C)	Decimal	Hex	Binary
+85	+10,619	0x297B	0010 1001 0111 1011
+25 + 0.0113	+2	0x0002	0000 0000 0000 0010
+25 + 0.00565	+1	0x0001	0000 0000 0000 0001
+25	0	0x0000	0000 0000 0000 0000
+25 − 0.00565	−1	0xFFFF	1111 1111 1111 1111
+25 − 0.0113	−2	0xFFFE	1111 1111 1111 1110
−40	−11,504	0xD310	1101 0011 0001 0000

STATUS/ALARM INDICATORS

The SYS_E_FLAG register in Table 47 provides the system error flags and new data bits for the magnetometer and barometer outputs. The new data flags are useful for triggering data collection of the magnetometer and barometer (x_MAGN_OUT and BAROM_xxx registers) because they update at a fixed rate that is not dependent on the DEC_RATE setting. Reading the SYS_E_FLAG register clears all of its error flags and returns each bit to a zero value, with the exception of Bit[7]. If SYS_E_FLAG[7] is high, use the software reset (GLOB_CMD[7], see Table 114) to clear this condition and restore normal operation. If any bit in the SYS_E_FLAG register is associated an error condition that remains after reading this register, this bit automatically returns to an alarm value of 1.

Table 47. SYS_E_FLAG (Page 0, Base Address = 0x08)

Bits	Description (Default = 0x0000)
[15]	Watch dog timer flag (1 = timed out)
[14:10]	Not used
9	New data flag, barometer (1 = new, unread data)[1]
8	New data flag, magnetometer (1 = new, unread data)[2]
7	Processing overrun (1 = error)
6	Flash memory update, result of GLOB_CMD[3] = 1 (1 = failed update, 0 = update successful)
5	Inertial self-test failure (1 = DIAG_STS ≠ 0x0000)
4	Sensor overrange (1 = at least one sensor overranged)
3	SPI communication error (1 = error condition, when the number of SCLK pulses is not equal to a multiple of 16)
[2:1]	Not used
0	Alarm status flag (1 = ALM_STS ≠ 0x0000)

[1] This flag restores to zero after reading the contents on BAROM_OUT.
[2] This flag restores to zero after reading one x_MAGN_OUT register.

The DIAG_STS register in Table 48 provides the flags for the internal self-test function, which is from GLOB_CMD[1] (see Table 114). Note that the barometer's flag, DIAG_STS[11], only updates after start-up and reset operations. Note that reading DIAG_STS also resets it to 0x0000.

Table 48. DIAG_STS (Page 0, Base Address = 0x0A)

Bits	Description (Default = 0x0000)
[15:12]	Not used
11	Self-test failure, barometer (1 = failed at start-up)
10	Self-test failure, Z-axis magnetometer (1 = failure)
9	Self-test failure, Y-axis magnetometer (1 = failure)
8	Self-test failure, X-axis magnetometer (1 = failure)
[7:6]	Not used
5	Self-test failure, Z-axis accelerometer (1 = failure)
4	Self-test failure, Y-axis accelerometer (1 = failure)
3	Self-test failure, X-axis accelerometer (1 = failure)
2	Self-test failure, Z-axis gyroscope (1 = failure)
1	Self-test failure, Y-axis gyroscope (1 = failure)
0	Self-test failure, X-axis gyroscope (1 = failure)

The ALM_STS register in Table 49 provides the alarm bits for the programmable alarm levels of each sensor. Note that reading ALM_STS also resets it to 0x0000.

Table 49. ALM_STS (Page 0, Base Address = 0x0C)

Bits	Description (Default = 0x0000)
[15:12]	Not used
11	Barometer alarm flag (1 = alarm is active)
10	Z-axis magnetometer alarm flag (1 = alarm is active)
9	Y-axis magnetometer alarm flag (1 = alarm is active)
8	X-axis magnetometer alarm flag (1 = alarm is active)
[7:6]	Not used
5	Z-axis accelerometer alarm flag (1 = alarm is active)
4	Y-axis accelerometer alarm flag (1 = alarm is active)
3	X-axis accelerometer alarm flag (1 = alarm is active)
2	Z-axis gyroscope alarm flag (1 = alarm is active)
1	Y-axis gyroscope alarm flag (1 = alarm is active)
0	X-axis gyroscope alarm flag (1 = alarm is active)

FIRMWARE REVISION

The FIRM_REV register (see Table 50) provides the firmware revision for the internal processor. Each nibble represents a digit in this revision code. For example, if FIRM_REV = 0x0102, the firmware revision is 1.02.

Table 50. FIRM_REV (Page 3, Base Address = 0x78)

Bits	Description
[15:12]	Binary, revision, 10's digit
[11:8]	Binary, revision, 1's digit
[7:4]	Binary, revision, tenths digit
[3:0]	Binary, revision, hundredths digit

The FIRM_DM register (see Table 51) contains the month and day of the factory configuration date. FIRM_DM[15:12] and FIRM_DM[11:8] contain digits that represent the month of factory configuration. For example, November is the 11th month in a year and represented by FIRM_DM[15:8] = 0x11. FIRM_DM[7:4] and FIRM_DM[3:0] contain digits that represent the day of factory configuration. For example, the 27th day of the month is represented by FIRM_DM[7:0] = 0x27.

Table 51. FIRM_DM (Page 3, Base Address = 0x7A)

Bits	Description
[15:12]	Binary, month 10's digit, range: 0 to 1
[11:8]	Binary, month 1's digit, range: 0 to 9
[7:4]	Binary, day 10's digit, range: 0 to 3
[3:0]	Binary, day 1's digit, range: 0 to 9

The FIRM_Y register (see Table 52) contains the year of the factory configuration date. For example, the year of 2013 is represented by FIRM_Y = 0x2013.

Table 52. FIRM_Y (Page 3, Base Address = 0x7C)

Bits	Description
[15:12]	Binary, year 1000's digit, range: 0 to 9
[11:8]	Binary, year 100's digit, range: 0 to 9
[7:4]	Binary, year 10's digit, range: 0 to 9
[3:0]	Binary, year 1's digit, range: 0 to 9

PRODUCT IDENTIFICATION

The PROD_ID register (see Table 53) contains the binary equivalent of the part number (16,488 = 0x4068), and the SERIAL_NUM register (see Table 54) contains a lot-specific serial number.

Table 53. PROD_ID (Page 0, Base Address = 0x7E)

Bits	Description (Default = 0x4068)
[15:0]	Product identification = 0x4068

Table 54. SERIAL_NUM (Page 4, Base Address = 0x20)

Bits	Description
[15:0]	Lot-specific serial number

DIGITAL SIGNAL PROCESSING

GYROSCOPES/ACCELEROMETERS

Figure 20 provides a signal flow diagram for all of the components and settings that influence the frequency response for the accelerometers and gyroscopes. The sample rate for each accelerometer and gyroscope is 9.84 kHz. Each sensor has its own averaging/decimation filter stage, which reduces the update rate to 2.46 kSPS. When using the external clock option (FNCTIO_CTRL[7:4], see Table 117), the input clock drives a 4-sample burst at a sample rate of 9.84 kSPS, which feeds into the 4x averaging/decimation filter. This results in a data rate that is equal to the input clock frequency.

AVERAGING/DECIMATION FILTER

The DEC_RATE register (see Table 55) provides user control for the final filter stage (see Figure 20), which averages and decimates the accelerometers, gyroscopes, delta angle, and delta velocity data. The output sample rate is equal to 2460/(DEC_RATE + 1). When using the external clock option (FNCTIO_CTRL[7:4], see Table 117), replace the "2460" number in this relationship, with the input clock frequency. For example, turn to Page 3 (DIN = 0x8003), and set DEC_RATE = 0x18 (DIN = 0x8C18, then DIN = 0x8D00) to reduce the output sample rate to 98.4 SPS (2460 ÷ 25).

Table 55. DEC_RATE (Page 3, Base Address = 0x0C)

Bits	Description (Default = 0x0000)
[15:11]	Don't care
[10:0]	Decimation rate, binary format, maximum = 2047 See Figure 20 for impact on sample rate

MAGNETOMETER/BAROMETER

When using the internal sampling clock, the magnetometer output registers (x_MAGN_OUT) update at a rate of 102.5 SPS and the barometer output registers (BAROM_xxx) update at a rate of 51.25 SPS. When using the external clock, the magnetometers update at a rate of 1/24th of the input clock frequency and the barometers update at a rate that is 1/48th of the input clock frequency. The update rates for the magnetometer and barometers do not change with the DEC_RATE register settings. SYS_E_FLAG[9:8] (see Table 47) offers new data bits for these registers and the SEQ_CNT register provides a counter function to help determine when there is new data in the magnetometer and barometer registers. When SEQ_CNT = 0x0001, there is new data in the magnetometer and barometer output registers. The SEQ_CNT register can be useful during initialization to help synchronize read loops for new data in both magnetometer and barometer outputs. When beginning a continuous read loop, read SEQ_CNT, then subtract this value from the maximum value shown (range) in Table 56 to calculate the number of internal sample cycles until both magnetometer and barometer data is new.

Table 56. SEQ_CNT (Page 0, Base Address = 0x06)

Bits	Description
[15:11]	Don't care
[6:0]	Binary counter: range = 1 to 48/(DEC_RATE + 1)

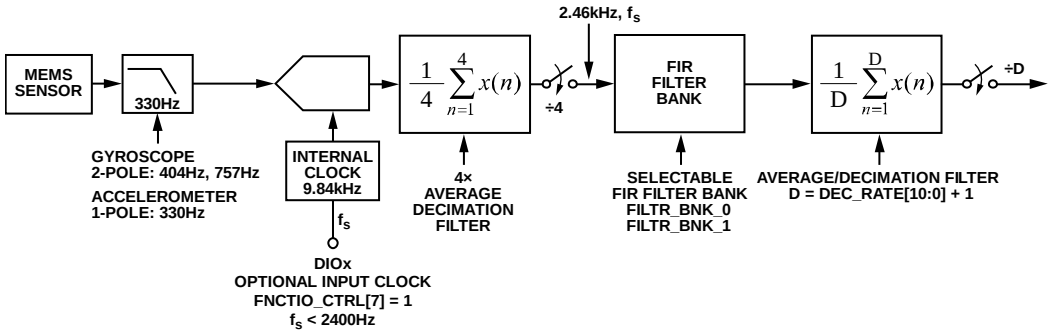


Figure 20. Sampling and Frequency Response Signal Flow

FIR FILTER BANKS

The ADIS16488 provides four configurable, 120-tap FIR filter banks. Each coefficient is 16 bits wide and occupies its own register location with each page. When designing a FIR filter for these banks, use a sample rate of 2.46 kHz and scale the coefficients so that their sum equals 32,768. For filter designs that have less than 120 taps, load the coefficients into the lower portion of the filter and start with Coefficient 1. Make sure that all unused taps are equal to zero, so that they do not add phase delay to the response. The FILTR_BNK_x registers provide three bits per sensor, which configure the filter bank (A, B, C, D) and turn filtering on and off. For example, turn to Page 3 (DIN = 0x8003), then write 0x0057 to FILTR_BNK_0 (DIN = 0x9657, DIN = 0x9700) to set the x-axis gyroscope to use the FIR filter in Bank D, to set the y-axis gyroscope to use the FIR filter in Bank B, and to enable these FIR filters in both x- and y-axis gyroscopes. Note that the filter settings update after writing to the upper byte; therefore, always configure the lower byte first. In cases that require configuration to only the lower byte of either FILTR_BNK_0 or FILTR_BNK_1, complete the process by writing 0x00 to the upper byte.

Table 57. FILTR_BNK_0 (Page 3, Base Address = 0x16)

Bits	Description (Default = 0x0000)
15	Don't care
14	Y-axis accelerometer filter enable (1 = enabled)
[13:12]	Y-axis accelerometer filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
11	X-axis accelerometer filter enable (1 = enabled)
[10:9]	X-axis accelerometer filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
8	Z-axis gyroscope filter enable (1 = enabled)
[7:6]	Z-axis gyroscope filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
5	Y-axis gyroscope filter enable (1 = enabled)
[4:3]	Y-axis gyroscope filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
2	X-axis gyroscope filter enable (1 = enabled)
[1:0]	X-axis gyroscope filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D

Table 58. FILTR_BNK_1 (Page 3, Base Address = 0x18)

Bits	Description (Default = 0x0000)
[15:12]	Don't care
11	Z-axis magnetometer filter enable (1 = enabled)
[10:9]	Z-axis magnetometer filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
8	Y-axis magnetometer filter enable (1 = enabled)
[7:6]	Y-axis magnetometer filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
5	X-axis magnetometer filter enable (1 = enabled)
[4:3]	X-axis magnetometer filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D
2	Z-axis accelerometer filter enable (1 = enabled)
[1:0]	Z-axis accelerometer filter bank selection: 00 = Bank A, 01 = Bank B, 10 = Bank C, 11 = Bank D

Filter Memory Organization

Each filter bank uses two pages of the user register structure. See Table 59, Table 60, Table 61, and Table 62 for the register addresses in each filter bank.

Table 59. Filter Bank A Memory Map

Page	PAGE_ID	Address	Register
5	0x05	0x00	PAGE_ID
5	0x05	0x02 to 0x07	Not used
5	0x05	0x08	FIR_COEF_A000
5	0x05	0x0A	FIR_COEF_A001
5	0x05	0x0C to 0x7C	FIR_COEF_A002 to FIR_COEF_A058
5	0x05	0x7E	FIR_COEF_A059
6	0x06	0x00	PAGE_ID
6	0x06	0x02 to 0x07	Not used
6	0x06	0x08	FIR_COEF_A060
6	0x06	0x0A	FIR_COEF_A061
6	0x06	0x0C to 0x7C	FIR_COEF_A062 to FIR_COEF_A118
6	0x06	0x7E	FIR_COEF_D119

Table 60. Filter Bank B Memory Map

Page	PAGE_ID	Address	Register
7	0x07	0x00	PAGE_ID
7	0x07	0x02 to 0x07	Not used
7	0x07	0x08	FIR_COEF_B000
7	0x07	0x0A	FIR_COEF_B001
7	0x07	0x0C to 0x7C	FIR_COEF_B002 to FIR_COEF_B058
7	0x07	0x7E	FIR_COEF_B059
8	0x08	0x00	PAGE_ID
8	0x08	0x02 to 0x07	Not used
8	0x08	0x08	FIR_COEF_B060
8	0x08	0x0A	FIR_COEF_B061
8	0x08	0x0C to 0x7C	FIR_COEF_B062 to FIR_COEF_B118
8	0x08	0x7E	FIR_COEF_B119

Table 61. Filter Bank C Memory Map

Page	PAGE_ID	Address	Register
9	0x09	0x00	PAGE_ID
9	0x09	0x02 to 0x07	Not used
9	0x09	0x08	FIR_COEF_C000
9	0x09	0x0A	FIR_COEF_C001
9	0x09	0x0C to 0x7C	FIR_COEF_C002 to FIR_COEF_C058
9	0x09	0x7E	FIR_COEF_C059
10	0x0A	0x00	PAGE_ID
10	0x0A	0x02 to 0x07	Not used
10	0x0A	0x08	FIR_COEF_C060
10	0x0A	0x0A	FIR_COEF_C061
10	0x0A	0x0C to 0x7C	FIR_COEF_C062 to FIR_COEF_C118
10	0x0A	0x7E	FIR_COEF_C119

Table 62. Filter Bank D Memory Map

Page	PAGE_ID	Address	Register
11	0x0B	0x00	PAGE_ID
11	0x0B	0x02 to 0x07	Not used
11	0x0B	0x08	FIR_COEF_D000
11	0x0B	0x0A	FIR_COEF_D001
11	0x0B	0x0C to 0x7C	FIR_COEF_D002 to FIR_COEF_D058
11	0x0B	0x7E	FIR_COEF_D059
12	0x0C	0x00	PAGE_ID
12	0x0C	0x02 to 0x07	Not used
12	0x0C	0x08	FIR_COEF_D060
12	0x0C	0x0A	FIR_COEF_D061
12	0x0C	0x0C to 0x7C	FIR_COEF_D062 to FIR_COEF_D118
12	0x0C	0x7E	FIR_COEF_D119

Default Filter Performance

The FIR filter banks have factory-programmed filter designs. They are all low-pass filters that have unity dc gain. Table 63 provides a summary of each filter design, and Figure 21 shows the frequency response characteristics. The phase delay is equal to ½ of the total number of taps.

Table 63. FIR Filter Descriptions, Default Configuration

FIR Filter Bank	Taps	−3 dB Frequency (Hz)
A	120	310
B	120	55
C	32	275
D	32	63

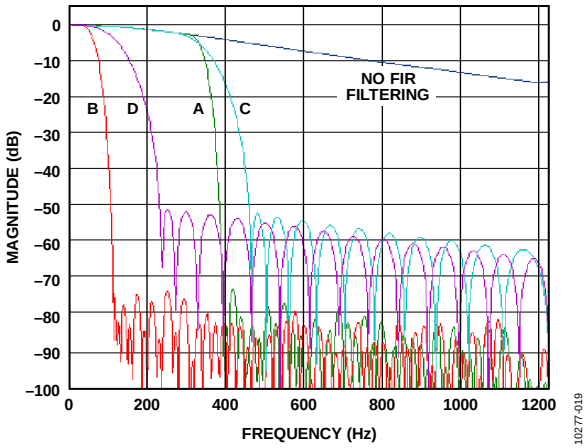


Figure 21. FIR Filter Frequency Response Curves

CALIBRATION

The ADIS16488 factory calibration produces correction formulas for the gyroscopes, accelerometers, magnetometers, and barometers, and then programs them into the flash memory. In addition, there are a series of user-configurable calibration registers, for in-system tuning.

GYROSCOPES

The user-calibration for the gyroscopes includes registers for adjusting bias and sensitivity, as shown in Figure 22.

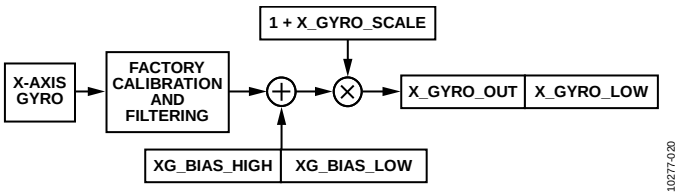


Figure 22. User Calibration Signal Path, Gyroscopes

Manual Bias Correction

The xG_BIAS_HIGH registers (see Table 64, Table 65, and Table 66) and xG_BIAS_LOW registers (see Table 67, Table 68, and Table 69) provide a bias adjustment function for the output of each gyroscope sensor.

Table 64. XG_BIAS_HIGH (Page 2, Base Address = 0x12)

Bits	Description (Default = 0x0000)
[15:0]	X-axis gyroscope offset correction, upper word twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 65. YG_BIAS_HIGH (Page 2, Base Address = 0x16)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis gyroscope offset correction, upper word; twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 66. ZG_BIAS_HIGH (Page 2, Base Address = 0x1A)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis gyroscope offset correction, upper word; twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 67. XG_BIAS_LOW (Page 2, Base Address = 0x10)

Bits	Description (Default = 0x0000)
[15:0]	X-axis gyroscope offset correction, lower word; twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec ÷ 2^{16} = ~0.000000305°/sec

Table 68. YG_BIAS_LOW (Page 2, Base Address = 0x14)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis gyroscope offset correction, lower word; twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec ÷ 2^{16} = ~0.000000305°/sec

Table 69. ZG_BIAS_LOW (Page 2, Base Address = 0x18)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis gyroscope offset correction, lower word twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec ÷ 2^{16} = ~0.000000305°/sec

Bias Null Command

The continuous bias estimator (CBE) accumulates and averages data in a 64-sample FIFO. The average time (t_A) for the bias estimates relies on the sample time base setting in NULL_CNFG[3:0] (see Table 70). Users can load the correction factors of the CBE into the gyroscope offset correction registers (see Table 64, Table 65, Table 66, Table 67, Table 68, and Table 69) using the bias null command in GLOB_CMD[0] (see Table 114). NULL_CNFG[13:8] provide on/off controls for the sensors that update when issuing a bias null command. The factory default configuration for NULL_CNFG enables the bias null command for the gyroscopes, disables the bias null command for the accelerometers, and establishes the average time to ~26.64 seconds.

Table 70. NULL_CNFG (Page 3, Base Address = 0x0E)

Bits	Description (Default = 0x070A)
[15:14]	Not used
13	Z-axis acceleration bias correction enable (1 = enabled)
12	Y-axis acceleration bias correction enable (1 = enabled)
11	X-axis acceleration bias correction enable (1 = enabled)
10	Z-axis gyroscope bias correction enable (1 = enabled)
9	Y-axis gyroscope bias correction enable (1 = enabled)
8	X-axis gyroscope bias correction enable (1 = enabled)
[7:4]	Not used
[3:0]	Time base control (TBC), range: 0 to 13 (default = 10); $t_B = 2^{TBC}/2460$, time base, $t_A = 64 \times t_B$, average time

Turn to Page 3 (DIN = 0x8003) and set GLOB_CMD[0] = 1 (DIN = 0x8201, then DIN = 0x8300) to update the user offset registers with the correction factors of the CBE. Make sure that the inertial platform is stable during the entire average time for optimal bias estimates.

Manual Sensitivity Correction

The x_GYRO_SCALE registers enable sensitivity adjustment (see Table 71, Table 72, and Table 73).

Table 71. X_GYRO_SCALE (Page 2, Base Address = 0x04)

Bits	Description (Default = 0x0000)
[15:0]	X-axis gyroscope scale correction; twos complement, 0x0000 = unity gain, 1 LSB = 1 ÷ 2^{15} = ~0.003052%

Table 72. Y_GYRO_SCALE (Page 2, Base Address = 0x06)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis gyroscope scale correction; twos complement, 0x0000 = unity gain, 1 LSB = 1 ÷ 2^{15} = ~0.003052%

Table 73. Z_GYRO_SCALE (Page 2, Base Address = 0x08)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis gyroscope scale correction; twos complement, 0x0000 = unity gain, 1 LSB = 1 ÷ 2^{15} = ~0.003052%

Linear Acceleration on Effect on Gyroscope Bias

MEMS gyroscopes typically have a bias response to linear acceleration that is normal to their axis of rotation. The ADIS16488 offers an optional compensation function for this effect. The factory-default setting (0x00C0) for the CONFIG register enables this function. To turn it off, turn to Page 3 (DIN = 0x8003) and set CONFIG[7] = 0 (DIN = 0x8A20, DIN = 0x8B00). Note that this also keeps the point of percussion alignment function on.

Table 74. CONFIG (Page 3, Base Address = 0x0A)

Bits	Description (Default = 0x00C0)
[15:8]	Not used
7	Linear-*g* compensation for gyroscopes (1 = enabled)
6	Point of percussion alignment (1 = enabled)
[5:2]	Not used
1	Real-time clock, daylight savings time (1: enabled, 0: disabled)
0	Real-time clock control (1: relative/elapsed timer mode, 0: calendar mode)

ACCELEROMETERS

The user-calibration for the accelerometers includes registers for adjusting bias and sensitivity, as shown in Figure 23.

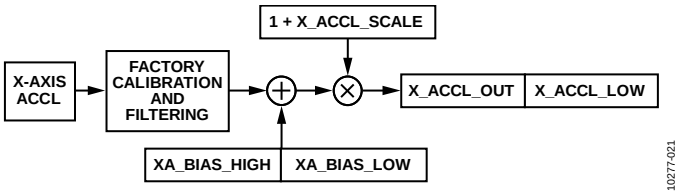


Figure 23. User Calibration Signal Path, Gyroscopes

Manual Bias Correction

The xA_BIAS_HIGH (see Table 75, Table 76, and Table 77) and xA_BIAS_LOW (see Table 78, Table 79, and Table 80) registers provide a bias adjustment function for the output of each accelerometer sensor. The xA_BIAS_HIGH registers use the same format as x_ACCL_OUT registers. The xA_BIAS_LOW registers use the same format as x_ACCL_LOW registers.

Table 75. XA_BIAS_HIGH (Page 2, Base Address = 0x1E)

Bits	Description (Default = 0x0000)
[15:0]	X-axis accelerometer offset correction, high word, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 76. YA_BIAS_HIGH (Page 2, Base Address = 0x22)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis accelerometer offset correction, high word, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 77. ZA_BIAS_HIGH (Page 2, Base Address = 0x26)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis accelerometer offset correction, high word, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 78. XA_BIAS_LOW (Page 2, Base Address = 0x1C)

Bits	Description (Default = 0x0000)
[15:0]	X-axis accelerometer offset correction, low word, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g* ÷ 2^{16} = ~0.0000122 m*g*

Table 79. YA_BIAS_LOW (Page 2, Base Address = 0x20)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis accelerometer offset correction, low word, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g* ÷ 2^{16} = ~0.0000122 m*g*

Table 80. ZA_BIAS_LOW (Page 2, Base Address = 0x24)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis accelerometer offset correction, low word;, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g* ÷ 2^{16} = ~0.0000122 m*g*

Manual Sensitivity Correction

The x_ACCL_SCALE registers enable sensitivity adjustment (see Table 81, Table 82, and Table 83).

Table 81. X_ACCL_SCALE (Page 2, Base Address = 0x0A)

Bits	Description (Default = 0x0000)
[15:0]	X-axis accelerometer scale correction, Twos complement, 0x0000 = unity gain, 1 LSB = 1 ÷ 2^{15} = ~0.003052%

Table 82. Y_ACCL_SCALE (Page 2, Base Address = 0x0C)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis accelerometer scale correction, Twos complement, 0x0000 = unity gain, 1 LSB = 1 ÷ 2^{15} = ~0.003052%

Table 83. Z_ACCL_SCALE (Page 2, Base Address = 0x0E)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis accelerometer scale correction, Twos complement, 0x0000 = unity gain, 1 LSB = 1 ÷ 2^{15} = ~0.003052%

MAGNETOMETERS

The user calibration registers enable both hard-iron and soft-iron correction, as shown in the following relationship:

$$\begin{bmatrix} M_{XC} \\ M_{YC} \\ M_{ZC} \end{bmatrix} = \begin{bmatrix} 1+S_{11} & S_{12} & S_{13} \\ S_{21} & 1+S_{22} & S_{23} \\ S_{31} & S_{32} & 1+S_{33} \end{bmatrix} \times \left(\begin{bmatrix} M_X \\ M_Y \\ M_Z \end{bmatrix} + \begin{bmatrix} H_X \\ H_Y \\ H_Z \end{bmatrix} \right)$$

The M_X, M_Y, and M_Z variables represent the magnetometer data, prior to application of the user correction formula. The M_{XC}, M_{YC}, and M_{ZC} represent the magnetometer data, after the application of the user correction formula.

Hard-Iron Correction

Table 84, Table 85, and Table 86 describe the register format for the hard-iron correction factors: H_X, H_Y, and H_Z. These registers use a twos complement format. Table 87 provides some numerical examples for converting the digital codes for these registers into their decimal equivalent.

Table 84. HARD_IRON_X (Page 2, Base Address = 0x28)

Bits	Description (Default = 0x0000)
[15:0]	X-axis magnetometer hard-iron correction factor, H_X Twos complement, ±3.2767 gauss range, 0.1 mgauss/LSB, 0 gauss = 0x0000 (see Table 87)

Table 85. HARD_IRON_Y (Page 2, Base Address = 0x2A)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis magnetometer hard-iron correction factor, H_Y Twos complement, ±3.2767 gauss range, 0.1 mgauss/LSB, 0 gauss = 0x0000 (see Table 87)

Table 86. HARD_IRON_Z (Page 2, Base Address = 0x2C)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis magnetometer hard-iron correction factor, H_z Twos complement, ±3.2767 gauss range, 0.1 mgauss/LSB, 0 gauss = 0x0000 (see Table 87)

Table 87. x_MAGN_OUT Data Format Examples

Magnetic Field	Decimal	Hex	Binary
+3.2767 gauss	+32,767	0x7FFF	0111 1111 1111 1111
+0.2 mgauss	+2	0x0002	0000 0000 0000 0010
+0.1 mgauss	+1	0x0001	0000 0000 0000 0001
0 gauss	0	0x0000	0000 0000 0000 0000
−0.1 mgauss	−1	0xFFFF	1111 1111 1111 1111
−0.2 mgauss	−2	0xFFFE	1111 1111 1111 1110
−3.2768 gauss	−32,768	0x8000	1000 0000 0000 0000

Soft-Iron Correction Matrix

The soft-iron correction matrix contains correction factors for both sensitivity (S_{11}, S_{22}, S_{33}) and alignment (S_{12}, S_{13}, S_{21}, S_{23}, S_{31}, S_{32}). The registers that represent each soft-iron correction factor are in Table 88 (S_{11}), Table 89 (S_{12}), Table 90 (S_{13}), Table 91 (S_{21}), Table 92 (S_{22}), Table 93 (S_{23}), Table 94 (S_{31}), Table 95 (S_{32}), and Table 96 (S_{33}). Table 97 offers some numerical examples for converting between the digital codes and their effect on the magnetometer output, in terms of percent-change.

Table 88. SOFT_IRON_S11 (Page 2, Base Address = 0x2E)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{11} Twos complement format, see Table 97 for examples

Table 89. SOFT_IRON_S12 (Page 2, Base Address = 0x30)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{12} Twos complement format, see Table 97 for examples

Table 90. SOFT_IRON_S13 (Page 2, Base Address = 0x32)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{13} Twos complement format, see Table 97 for examples

Table 91. SOFT_IRON_S21 (Page 2, Base Address = 0x34)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{21} Twos complement format, see Table 97 for examples

Table 92. SOFT_IRON_S22 (Page 2, Base Address = 0x36)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{22} Twos complement format, see Table 97 for examples

Table 93. SOFT_IRON_S23 (Page 2, Base Address = 0x38)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{23} Twos complement format, see Table 97 for examples

Table 94. SOFT_IRON_S31 (Page 2, Base Address = 0x3A)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{31} Twos complement format, see Table 97 for examples

Table 95. SOFT_IRON_S32 (Page 2, Base Address = 0x3C)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{32} Twos complement format, see Table 97 for examples

Table 96. SOFT_IRON_S33 (Page 2, Base Address = 0x3E)

Bits	Description (Default = 0x0000)
[15:0]	Magnetometer soft-iron correction factor, S_{33} Twos complement format, see Table 97 for examples

Table 97. Soft Iron Correction, Numerical Examples

Delta (%)	Decimal	Hex	Binary
$+100 - 1/2^{16}$	+32,767	0x7FFF	0111 1111 1111 1111
$+200/2^{15}$	+2	0x0002	0000 0000 0000 0010
$+100/2^{15}$	+1	0x0001	0000 0000 0000 0001
0	0	0x0000	0000 0000 0000 0000
$-100/2^{15}$	−1	0xFFFF	1111 1111 1111 1111
$-200/2^{15}$	−2	0xFFFE	1111 1111 1111 1110
−100	−32,768	0x8000	1000 0000 0000 0000

BAROMETERS

The BR_BIAS_HIGH register (see Table 98) and BR_BIAS_LOW register (Table 99) provide an offset control function and use the same format as the output registers, BAROM_OUT and BAROM_LOW.

Table 98. BR_BIAS_HIGH (Page 2, Base Address = 0x42)

Bits	Description (Default = 0x0000)
[15:0]	Barometric pressure bias correction factor, high word Twos complement, ±1.3 bar measurement range, 0 bar = 0x0000, 1 LSB = 40 µbar

Table 99. BR_BIAS_LOW (Page 2, Base Address = 0x40)

Bits	Description (Default = 0x0000)
[15:0]	Barometric pressure bias correction factor, low word Twos complement, ±1.3 bar measurement range, 0 bar = 0x0000, 1 LSB = 40 µbar ÷ 2^{16} = ~0.00061 µbar

RESTORING FACTORY CALIBRATION

Turn to Page 3 (DIN = 0x8003) and set GLOB_CMD[6] = 1 (DIN = 0x8240, DIN = 0x8300) to execute the factory calibration restore function. This function resets each user calibration register to zero, resets all sensor data to 0, and automatically updates the flash memory within 72 ms. See Table 114 for more information on GLOB_CMD.

POINT OF PERCUSSION ALIGNMENT

CONFIG[6] offers a point of percussion alignment function that maps the accelerometer sensors to the corner of the package identified in Figure 24. To activate this feature, turn to Page 3 (DIN = 0x8003), then set CONFIG[6] = 1 (DIN = 0x8A40, DIN = 0x8B00). See Table 74 for more information on the CONFIG register.

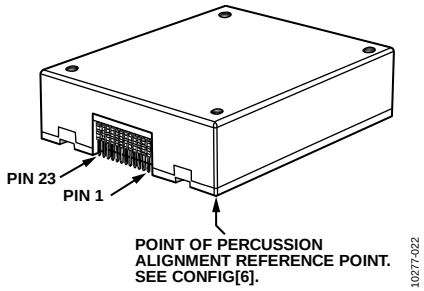


Figure 24. Point of Percussion Reference Point

ALARMS

Each sensor has an independent alarm function that provides controls for alarm magnitude, polarity, and enabling a dynamic rate-of-change option. The ALM_STS register (see Table 49) contains the alarm output flags and the FNCTIO_CTRL register (see Table 117) provides an option for configuring one of the digital I/O lines as an alarm indicator.

STATIC ALARM USE

The static alarm setting compares each sensor's output with the trigger settings in the xx_ALM_MAGN registers (see Table 100, Table 101, Table 102, Table 103, Table 104, Table 105, Table 106, Table 107, Table 108, and Table 109) of that sensor. The polarity controls for each alarm are in the ALM_CNFG_x registers (see Table 110, Table 111, Table 112). The polarity establishes whether greater than or less than produces an alarm condition. The comparison between the xx_ALM_MAGN value and the output data only applies to the upper word or 16 bits of the output data.

DYNAMIC ALARM USE

The dynamic alarm setting provides the option of comparing the change in each sensor's output over a period of 48.7 ms with that sensor's xx_ALM_MAGN register.

Table 100. XG_ALM_MAGN (Page 3, Base Address = 0x28)

Bits	Description (Default = 0x0000)
[15:0]	X-axis gyroscope alarm threshold settings, Twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 101. YG_ALM_MAGN (Page 3, Base Address = 0x2A)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis gyroscope alarm threshold settings, Twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 102. ZG_ALM_MAGN (Page 3, Base Address = 0x2C)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis gyroscope alarm threshold settings, Twos complement, 0°/sec = 0x0000, 1 LSB = 0.02°/sec

Table 103. XA_ALM_MAGN (Page 3, Base Address = 0x2E)

Bits	Description (Default = 0x0000)
[15:0]	X-axis accelerometer alarm threshold settings, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 104. YA_ALM_MAGN (Page 3, Base Address = 0x30)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis accelerometer alarm threshold settings, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 105. ZA_ALM_MAGN (Page 3, Base Address = 0x32)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis accelerometer alarm threshold settings, Twos complement, 0 *g* = 0x0000, 1 LSB = 0.8 m*g*

Table 106. XM_ALM_MAGN (Page 3, Base Address = 0x34)

Bits	Description (Default = 0x0000)
[15:0]	X-axis magnetometer alarm threshold settings, Twos complement, 0 gauss = 0x0000, 1 LSB = 0.1 mgauss

Table 107. YM_ALM_MAGN (Page 3, Base Address = 0x36)

Bits	Description (Default = 0x0000)
[15:0]	Y-axis magnetometer alarm threshold settings, Twos complement, 0 gauss = 0x0000, 1 LSB = 0.1 mgauss

Table 108. ZM_ALM_MAGN (Page 3, Base Address = 0x38)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis magnetometer alarm threshold settings, Twos complement, 0 gauss = 0x0000, 1 LSB = 0.1 mgauss

Table 109. BR_ALM_MAGN (Page 3, Base Address = 0x3A)

Bits	Description (Default = 0x0000)
[15:0]	Z-axis barometer alarm threshold settings, Twos complement, 0 bar = 0x0000, 1 LSB = 40 µbar

Table 110. ALM_CNFG_0 (Page 3, Base Address = 0x20)

Bits	Description (Default = 0x0000)
15	X-axis accelerometer alarm (1 = enabled)
14	Not used
13	X-axis accelerometer alarm polarity (1 = greater than)
12	X-axis accelerometer dynamic enable (1 = enabled)
11	Z-axis gyroscope alarm (1 = enabled)
10	Not used
9	Z-axis gyroscope alarm polarity (1 = greater than)
8	Z-axis gyroscope dynamic enable (1 = enabled)
7	Y-axis gyroscope alarm (1 = enabled)
6	Not used
5	Y-axis gyroscope alarm polarity (1 = greater than)
4	Y-axis gyroscope dynamic enable (1 = enabled)
3	X-axis gyroscope alarm (1 = enabled)
2	Not used
1	X-axis gyroscope alarm polarity (1 = greater than)
0	X-axis gyroscope dynamic enable (1 = enabled)

Table 111. ALM_CNFG_1 (Page 3, Base Address = 0x22)

Bits	Description (Default = 0x0000)
15	Y-axis magnetometer alarm (1 = enabled)
14	Not used
13	Y-axis magnetometer alarm polarity (1 = greater than)
12	Y-axis magnetometer dynamic enable (1 = enabled)
11	X-axis magnetometer (1 = enabled)
10	Not used
9	X-axis magnetometer alarm polarity (1 = greater than)
8	X-axis magnetometer dynamic enable (1 = enabled)
7	Z-axis accelerometer alarm (1 = enabled)
6	Not used
5	Z-axis accelerometer alarm polarity (1 = greater than)
4	Z-axis accelerometer dynamic enable (1 = enabled)
3	Y-axis accelerometer alarm (1 = enabled)
2	Not used
1	Y-axis accelerometer alarm polarity (1 = greater than)
0	Y-axis accelerometer dynamic enable (1 = enabled)

Table 112. ALM_CNFG_2 (Page 3, Base Address = 0x24)

Bits	Description (Default = 0x0000)
[15:8]	Not used
7	Barometer alarm (1 = enabled)
6	Not used
5	Barometer alarm polarity (1 = greater than)
4	Barometer dynamic enable (1 = enabled)
3	Z-axis magnetometer alarm (1 = enabled)
2	Not used
1	Z-axis magnetometer alarm polarity (1 = greater than)
0	Z-axis magnetometer dynamic enable (1 = enabled)

Alarm Example

Table 113 offers an alarm configuration example, which sets the Z-axis gyroscope alarm to trip when Z_GYRO_OUT > 131.1°/sec (0x199B).

Table 113. Alarm Configuration Example

DIN	Description
0xAC9B	Set ZG_ALM_MAGN[7:0] = 0x9B
0xAD19	Set ZG_ALM_MAGN[15:8] = 0x19
0xA000	Set ALM_CNFG_0[7:0] = 0x00
0xA103	Set ALM_CNFG_0[15:8] = 0x03

SYSTEM CONTROLS

The ADIS16488 provides a number of system-level controls for managing its operation, which include reset, self-test, calibration, memory management, and I/O configuration.

GLOBAL COMMANDS

The GLOB_CMD register (see Table 114) provides trigger bits for several operations. Write 1 to the appropriate bit in GLOB_CMD to start a function. After the function completes, the bit restores to 0.

Table 114. GLOB_CMD (Page 3, Base Address = 0x02)

Bits	Description	Execution Time
[15:8]	Not used	Not applicable
7	Software reset	120 ms
6	Factory calibration restore	75 ms
[5:4]	Not used	Not applicable
3	Flash memory update	375 ms
2	Flash memory test	50 ms
1	Self-test	12 ms
0	Bias null	See Table 70

Software Reset

Turn to Page 3 (DIN = 0x8003) and then set GLOB_CMD[7] = 1 (DIN = 0x8280, DIN = 0x8300) to reset the operation, which removes all data, initializes all registers from their flash settings, and starts data collection. This function provides a firmware alternative to the $\overline{\text{RST}}$ line (see Table 5, Pin 8).

Automatic Self-Test

Turn to Page 3 (DIN = 0x8003) and then set GLOB_CMD[1] = 1 (DIN = 0x8202, then DIN = 0x8300) to run an automatic self-test routine, which executes the following steps:

1. Measure output on each sensor.
2. Activate self-test on each sensor.
3. Measure output on each sensor.
4. Deactivate the self-test on each sensor.
5. Calculate the difference with self-test on and off.
6. Compare the difference with internal pass/fail criteria.
7. Report the pass/fail results for each sensor in DIAG_STS.

After waiting 12 ms for this test to complete, turn to Page 0 (DIN = 0x8000) and read DIAG_STS using DIN = 0x0A00. Note that using an external clock can extend this time. When using an external clock of 100 Hz, this time extends to 35 ms. Note that 100 Hz is too slow for optimal sensor performance.

MEMORY MANAGEMENT

The data retention of the flash memory depends on temperature and the number of write cycles. Figure 25 characterizes the dependence on temperature, and the FLSHCNT_LOW and FLSHCNT_HIGH registers (see Table 115 and Table 116) provide a running count of flash write cycles. The flash updates every time GLOB_CMD[6], GLOB_CMD[3], or GLOB_CMD[0] is set to 1.

Table 115. FLSHCNT_LOW (Page 2, Base Address = 0x7C)

Bits	Description
[15:0]	Binary counter; number of flash updates, lower word

Table 116. FLSHCNT_HIGH (Page 2, Base Address = 0x7E)

Bits	Description
[15:0]	Binary counter; number of flash updates, upper word

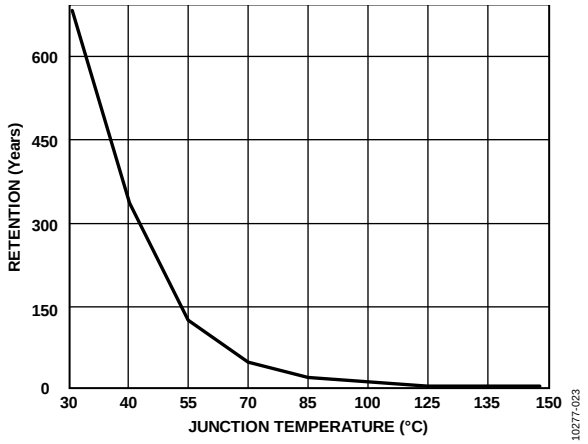


Figure 25. Flash Memory Retention

Flash Memory Test

Turn to Page 3 (DIN = 0x8003), and then set GLOB_CMD[2] = 1 (DIN = 0x8204, DIN = 0x8300) to run a checksum test of the internal flash memory, which compares a factory-programmed value with the current sum of the same memory locations. The result of this test loads into SYS_E_FLAG[6]. Turn to Page 0 (DIN = 0x8000) and use DIN = 0x0800 to read SYS_E_FLAG.

GENERAL-PURPOSE I/O

There are four general-purpose I/O lines: DIO1, DIO2, DIO3, and DIO4. The FNCTIO_CTRL register controls the basic function of each I/O line, which provides a number of useful functions. Each I/O line will only support one function at a time. In cases where a single line has two different assignments, the enable bit for the lower-priority function will automatically reset to zero and be disabled. The priority is (1) data-ready, (2) sync clock input, (3) alarm indicator, and (4) general-purpose, where 1 identifies the highest priority and 4 indicates the lowest priority.

Table 117. FNCTIO_CTRL (Page 3, Base Address = 0x06)

Bits	Description (Default = 0x000D)
[15:12]	Not used
11	Alarm indicator: 1 = enabled, 0 = disabled
10	Alarm indicator polarity: 1 = positive, 0 = negative
[9:8]	Alarm indicator line selection: 00 = DIO1, 01 = DIO2, 10 = DIO3, 11 = DIO4
7	Sync clock input enable: 1 = enabled, 0 = disabled
6	Sync clock input polarity: 1 = rising edge, 0 = falling edge
[5:4]	Sync clock input line selection: 00 = DIO1, 01 = DIO2, 10 = DIO3, 11 = DIO4
3	Data-ready enable: 1 = enabled, 0 = disabled
2	Data-ready polarity: 1 = positive, 0 = negative
[1:0]	Data-ready line selection: 00 = DIO1, 01 = DIO2, 10 = DIO3, 11 = DIO4

Data-Ready Indicator

FNCTIO_CTRL[3:0] provide some configuration options for using one of the DIOx lines as a data-ready indicator signal, which can drive a processor's interrupt control line. The factory default assigns DIO2 as a positive polarity, data-ready signal. Use the following sequence to change this assignment to DIO1 with a negative polarity: turn to Page 3 (DIN = 0x8003) and set FNCTIO_CTRL[3:0] = 1000 (DIN = 0x8608, then DIN = 0x8700). The timing jitter on the data-ready signal is ±1.4 µs.

Input Sync/Clock Control

FNCTIO_CTRL[7:4] provide some configuration options for using one of the DIOx lines as an input synchronization signal for sampling inertial sensor data. For example, use the following sequence to establish DIO4 as a positive polarity, input clock pin and keep the factory default setting for the data-ready function: turn to Page 3 (DIN = 0x8003) and set FNCTIO_CTRL[7:0] = 0xFD (DIN = 0x86FD, then DIN = 0x8700). Note that this command also disables the internal sampling clock, and no data sampling takes place without the input clock signal. When selecting a clock input frequency, consider the 330 Hz sensor bandwidth, because under sampling the sensors can degrade noise and stability performance.

General-Purpose I/O Control

When FNCTIO_CTRL does not configure a DIOx pin, GPIO_CTRL provides register controls for general-purpose use of the pin. GPIO_CTRL[3:0] provides input/output assignment controls for each line. When the DIOx lines are inputs, monitor their level by reading GPIO_CTRL[7:4]. When the DIOx lines are used as outputs, set their level by writing to GPIO_CTRL[7:4]. For example, use the following sequence to set DIO1 and DIO3 as high and low output lines, respectively, and set DIO2 and DIO4 as input lines. Turn to Page 3 (DIN = 0x8003) and set GPIO_CTRL[7:0] = 0x15 (DIN = 0x8815, then DIN = 0x8900).

Table 118. GPIO_CTRL (Page 3, Base Address = 0x08)

Bits	Description (Default = 0x00X0)[1]
[15:8]	Don't care
7	General-Purpose I/O Line 4 (DIO4) data level
6	General-Purpose I/O Line 3 (DIO3) data level
5	General-Purpose I/O Line 2 (DIO2) data level
4	General-Purpose I/O Line 1 (DIO1) data level
3	General-Purpose I/O Line 4 (DIO4) direction control (1 = output, 0 = input)
2	General-Purpose I/O Line 3 (DIO3) direction control (1 = output, 0 = input)
1	General-Purpose I/O Line 2 (DIO2) direction control (1 = output, 0 = input)
0	General-Purpose I/O Line 1 (DIO1) direction control (1 = output, 0 = input)

[1] GPIO_CTRL[7:4] bits reflect the logic levels on DIOx pins and do not have a default setting.

POWER MANAGEMENT

The SLP_CNT register (see Table 119) provides controls for both power-down mode and sleep modes. The trade-off between power-down mode and sleep mode is between idle power and recovery time. Power-down mode offers the best idle power consumption but requires the most time to recover. Also, all volatile settings are lost during power-down but are preserved during sleep mode.

For timed sleep mode, turn to Page 3 (DIN = 0x8003), write the amount of sleep time to SLP_CNT[7:0] and then, set SLP_CNT[8] = 1 (DIN = 0x9101) to start the sleep period. For a timed power-down period, change the last command to set SLP_CNT[9] = 1 (DIN = 0x9102). To power down or sleep for an indefinite period, set SLP_CNT[7:0] = 0x00 first, then set either SLP_CNT[8] or SLP_CNT[9] to 1. Note that the command takes effect when the $\overline{CS}$ line goes high. To awaken the device from sleep or power-down mode, use one of the following options to restore normal operation:

- Assert $\overline{CS}$ from high to low.
- Pulse $\overline{RST}$ low, then high again.
- Cycle the power.

For example, set SLP_CNT[7:0] = 0x64 (DIN = 0x9064), then set SLP_CNT[8] = 1 (DIN = 0x9101) to start a sleep period of 100 seconds.

Table 119. SLP_CNT (Page 3, Base Address = 0x10)

Bits	Description
[15:10]	Not used
9	Power-down mode
8	Normal sleep mode
[7:0]	Programmable time bits; 1 sec/LSB; 0x00 = indefinite

If the sleep mode and power-down mode bits are both set high, the normal sleep mode bit (SLP_CNT[8]) takes precedence.

General-Purpose Registers

The USER_SCR_x registers (see Table 120, Table 121, Table 122, and Table 123) provide four 16-bit registers for storing data.

Table 120. USER_SCR_1 (Page 2, Base Address = 0x74)

Bits	Description
[15:0]	User-defined

Table 121. USER_SCR_2 (Page 2, Base Address = 0x76)

Bits	Description
[15:0]	User-defined

Table 122. USER_SCR_3 (Page 2, Base Address = 0x78)

Bits	Description
[15:0]	User-defined

Table 123. USER_SCR_4 (Page 2, Base Address = 0x7A)

Bits	Description
[15:0]	User-defined

Real-Time Clock Configuration/Data

The VDDRTC power supply pin (see Table 5, Pin 23) provides a separate supply for the real-time clock (RTC) function. This enables the RTC to keep track of time, even when the main supply (VDD) is off. Configure the RTC function by selecting one of two modes in CONFIG[0] (see Table 74). The real-time clock data is available in the TIME_MS_OUT register (see Table 124), TIME_DH_OUT register (see Table 125), and TIME_YM_OUT register (see Table 126). When using the elapsed timer mode, the time data registers start at 0x0000 when the device starts up (or resets) and begin keeping time in a manner that is similar to a stopwatch. When using the clock/calendar mode, write the current time to the real-time registers in the following sequence: seconds (TIME_MS_OUT[5:0]), minutes (TIME_MS_OUT[13:8]), hours (TIME_DH_OUT[5:0]), day (TIME_DH_OUT[12:8]), month (TIME_YM_OUT[3:0]), and year (TIME_YM_OUT[14:8]). The updates to the timer do not become active until a successful write to the TIME_YM_OUT[14:8] byte.

The real-time clock registers reflect the newly updated values only after the next seconds tick of the clock that follows the write to TIME_YM_OUT[14:8] (year). Writing to TIME_YM_OUT[14:8] activates all timing values; therefore, always write to this location last when updating the timer, even if the year information does not require updating.

Write the current time to each time data register after setting CONFIG[0] = 1 (DIN = 0x8003, DIN = 0x8A01). Note that CONFIG[1] provides a bit for managing daylight savings time. After the CONFIG and TIME_xx_OUT registers are configured, set GLOB_CMD[3] = 1 (DIN = 0x8003, DIN = 0x8204, DIN = 0x8300) to back up these settings in flash, and use a separate 3.3 V source to supply power to the VDDRTC function. Note that access to time data in the TIME_xx_OUT registers requires normal operation (VDD = 3.3 V and full startup), but the timer function only requires that VDDRTC = 3.3 V when the rest of the ADIS16488 is turned off.

Table 124. TIME_MS_OUT (Page 0, Base Address = 0x78)

Bits	Description
[15:14]	Not used
[13:8]	Minutes, binary data, range = 0 to 59
[7:6]	Not used
[5:0]	Seconds, binary data, range = 0 to 59

Table 125. TIME_DH_OUT (Page 0, Base Address = 0x7A)

Bits	Description
[15:13]	Not used
[12:8]	Day, binary data, range = 1 to 31
[7:6]	Not used
[5:0]	Hours, binary data, range = 0 to 23

Table 126. TIME_YM_OUT (Page 0, Base Address = 0x7C)

Bits	Description
[15]	Not used
[14:8]	Year, binary data, range = 0 to 99, relative to 2000 A.D.
[7:4]	Not used
[3:0]	Month, binary data, range = 1 to 12

APPLICATIONS INFORMATION

MOUNTING TIPS

For best performance, follow these rules when installing the ADIS16488 into a system.

1. Eliminate opportunity for translational force (x-axis and y-axis direction, per Figure 6) application on the electrical connector.
2. Isolate mounting force to the four corners, on the part of the package surface that surrounds the mounting holes.
3. Use uniform mounting forces on all four corners. The suggested torque setting is 40 inch-ounces (0.285 N-m).

These three rules help prevent nonuniform force profiles, which can warp the package and introduce bias errors in the sensors. Figure 26 provides an example that leverages washers to set the package off the mounting surface and uses 2.85 mm pass-through holes and backside washers/nuts for attachment. Figure 27 and Figure 28 provide some details for mounting hole and connector alignment pin drill locations. For more information on mounting the ADIS16488, see the AN-1295 Application Note, *Mechanical Design Tips for ADIS16375, ADIS16480, ADIS16485, and ADIS16488*.

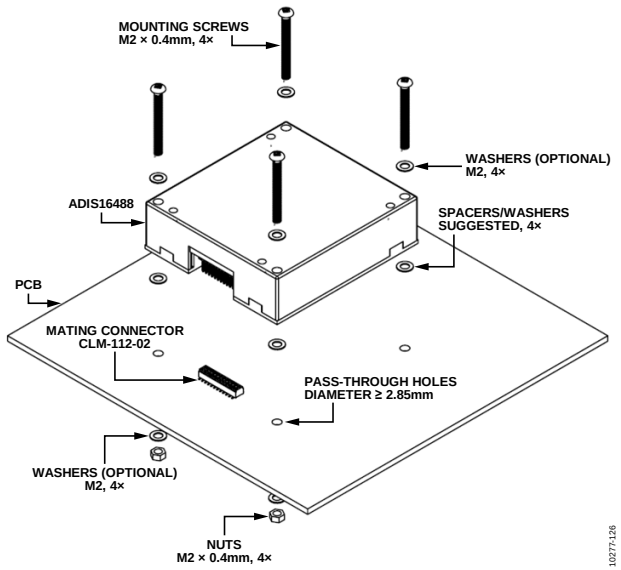


Figure 26. Mounting Example

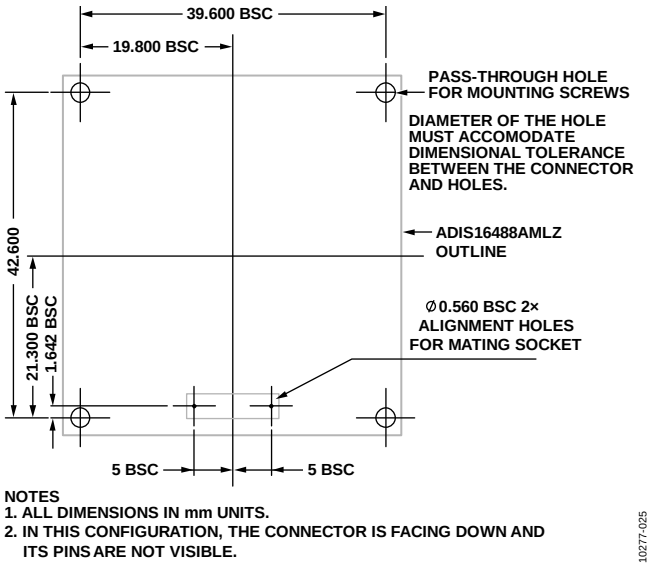


Figure 27. Suggested PCB Layout Pattern, Connector Down

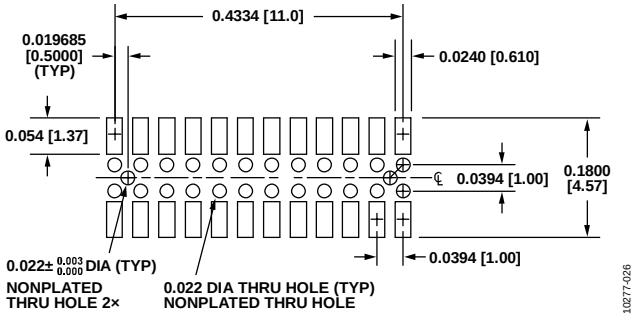


Figure 28. Suggested Layout and Mechanical Design When Using Samtec P/N CLM-112-02-G-D-A for the Mating Connector

EVALUATION TOOLS

Breakout Board, ADIS16IMU1/PCBZ

The ADIS16IMU1/PCBZ (sold separately) provides a breakout board function for the ADIS16488, which means that it provides access to the ADIS16488 through larger connectors that support standard 1 mm ribbon cabling. It also provides four mounting holes for attachment of the ADIS16488 to the breakout board. For more information on the ADIS16IMU1/PCBZ, see http://www.analog.com/en/evaluation/eval-adis16imu1/eb.html.

PC-Based Evaluation, EVAL-ADIS

The EVAL-ADIS system supports PC-based evaluation of the ADIS16488. For more information on the EVAL-ADIS system, see http://www.analog.com/EVAL-ADIS.

POWER SUPPLY CONSIDERATIONS

The ADIS16488 has approximately ~24 µF of capacitance across the VDD and GND pins. While this capacitor bank provides a large amount of localized filtering, it also presents an opportunity for excessive charging current when the VDD voltage ramps too quickly. Use the following relationship to help determine the appropriate VDD voltage profile, with respect to any current limit functions that can cause the power supply to lose regulation and potentially introduce un-safe conditions for the ADIS16488.

$$i(t) = C\,{}^{dV}\!/_{dt}$$

In addition to managing the initial voltage ramp, take note of the transient current demand that the ADIS16488 requires during its start-up/self-initialization process. Once VDD reaches 2.85 V, the ADIS16488 begins its start-up process. Figure 29 offers a broad perspective that communicates when to expect the spikes in current, while Figure 30 provide more detail on the current/time behavior during the peak transient condition, which typically occurs approximately 350 ms after VDD reaches 2.85 V. In Figure 30, notice that the peak current approaches 600 mA and the transient condition lasts for approximately 1.75 ms.

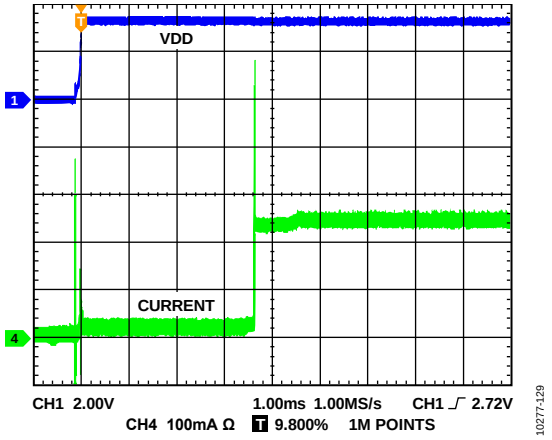


Figure 29. Transient Current Demand, Start-up

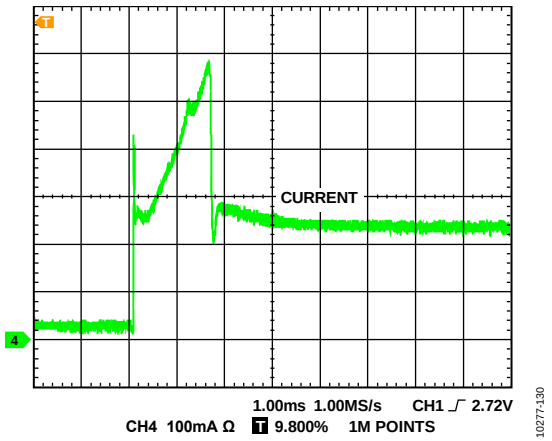


Figure 30. Transient Current Demand, Peak Demand

OUTLINE DIMENSIONS

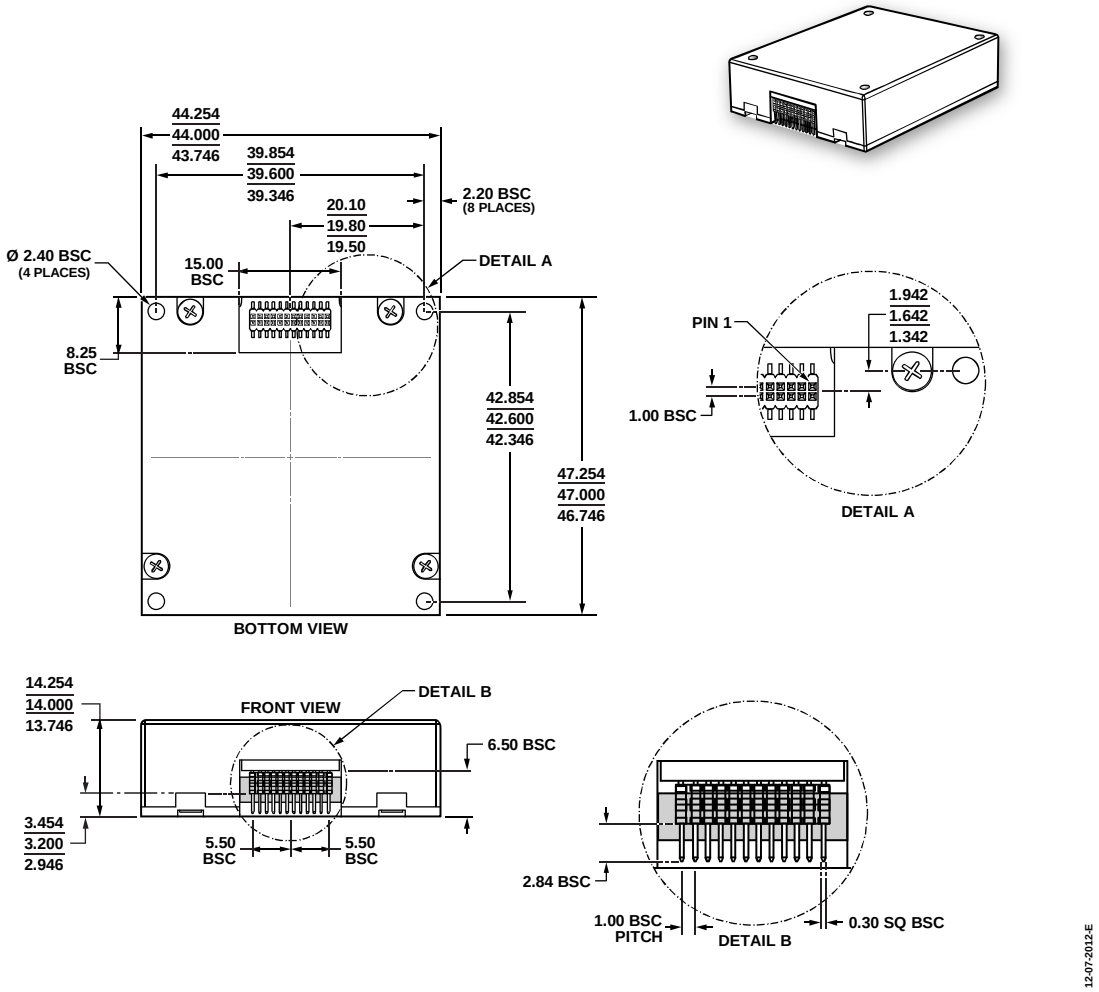


Figure 31. 24-Lead Module with Connector Interface [MODULE]
(ML-24-6)
Dimensions shown in millimeters

ORDERING GUIDE

Model[1]	Temperature Range	Package Description	Package Option
ADIS16488AMLZ	−40°C to +85°C	24-Lead Module with Connector Interface [MODULE]	ML-24-6

[1] Z = RoHS Compliant Part.

NOTES

©2011–2014 Analog Devices, Inc. All rights reserved. Trademarks and registered trademarks are the property of their respective owners.
D10277-0-5/14(G)