TAMPEREEN
AMMATTIKORKEAKOULU

Laser-ympäristömallinnus

Kimmo Kujansuu

Opinnäytetyö
Maaliskuu 2016
Tietotekniikka
Sulautetut järjestelmät ja elektroniikka
Älykkäät koneet

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikka
Sulautetut järjestelmät ja elektroniikka
Älykkäät koneet

KUJANSUU KIMMO
Laser-ympäristömallinnus

Opinnäytetyö 41 sivua, joista liitteitä 1 sivua
Maaliskuu 2015

Tämän opinnäytetyön tavoitteena oli skannata ja luoda 3D-malli ympäröivästä maastosta. Työ tehtiin erään yrityksen tarpeeseen. Tässä tutkielmassa käytettiin 2D-laserskanneria ja IMU-anturia, joita testattiin käytännössä ja tutustuttiin niiden toimintaan liittyvään teoriaan. Tavoitteena oli toteuttaa 3D-maastomallinnus edullisten ja erillisien anturien yhdistelmällä. Työn haasteena oli luoda mahdollisimman tarkka 3D-malli ympäröivästä maastosta.

Työ tehtiin ohjelmoimalla käyttäen C++ -ohjelmointikieltä Windows-ympäristössä. 2D-laserskannerin lähettämästä datavirrasta eroteltiin tarvittavat luvut ja tiedot sopiviin muuttujiin tiedon jatkokäsittelyä ja laskentaa varten. Työssä tehtiin mittausalgoritmi, jonka tehtävänä oli ohjeistaa käyttäjää onnistuneeseen 3D-skannaus-mittaustulokseen. Työn lopuksi saatiin piirrettyä työssä tehdyllä 3D-skannausohjelmistolla onnistuneesti 2D-syvyyskuva skannatusta huoneistosta.

3D-mallin skannausresoluutiota rajoittavia tekijöitä ovat IMU-anturin mittaamien pystykallistuskulmien tarkkuus, sekä 2D-laserskannerin vaakaskannaustarkkuus. Suhteellisen luotettavaksi IMU-anturin pystykallistuskulmatarkkuudeksi saatiin testattua noin 0,3-astetta, mikä sopi tähän työhön mainiosti, koska työssä käytetyn 2D-laserskannerin vaakatarkkuus oli myös noin 0,3-astetta. Työn alussa oli ajatus siitä, että 2D-laserskanneri olisi ollut hyvä asentaa työkoneen puomiin, koska silloin skannaus voitaisiin tehdä puomia kääntämällä. Työn edetessä tämä ajatus alkoi vaikuttaa huonolta ratkaisulta, koska työkoneen puomia voi olla vaikeata kääntää tarvittavan hitaasti ja tasaisesti. 2D-laserskannerin hitaaseen ja tasaiseen kääntämiseen voisi olla parempi käyttää askelmoottoria. Tätä työtä voitaisiin jatkokehittää niin, että skannatuista 3D-vektoreista tehtäisiin 3D-malli, jota voitaisiin tutkia ja pyöritellä tietokoneella. Lisäksi näitä paikallaan tehtyjä 3D-skannauksia tehtäisiin enemmän ja yhdistettäisiin useammat 3D-mallit isommaksi 3D-skannauskokonaisuudeksi.

Asiasanat: 3D-malli, 2D-laserskanneri, ympäristömallinnus, kallistuskulman mittaus

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Information Technology
Embedded Systems
Intelligent Machines

KUJANSUU KIMMO
Laser environment modelling

Bachelor's thesis 41 pages, appendices 1 pages
March 2015

The main goal of this thesis was to scan and make a 3D model from a surrounding environment. The project was made for one company and to their need. There was used a 2D laser scanner and an IMU during this work. The sensors were tested in practice and the theory behind them was researched. The aim for the project was to make a 3D environment model by using a few discrete and affordable sensors. The challenge of the project was to scan the environment to make a 3D-model that is as accurate as possible.

All code for this thesis was made in C++ programming language on the Windows platform. A data stream sent by the 2D laser scanner was parsed into separate variables in the code. A scanning algorithm was programmed that assists user to achieve a successful 3D scan by tilting the 2D laser scanner manually. In the end of the project a 2D depth picture was created from a scanned 3D model using the program that was made during the thesis.

There are few things that limits the final scanning resolution. One thing is the precision of the IMU and the other thing is the resolution of the 2D laser scanner. The resolution of the IMU that was tested to be stable and reliable was around 0,2 to 0,3 degrees. This was good enough for this project because the best resolution that is available on the 2D-laser scanner is about 0,33 degrees. At the start of the project there was assumption that a best place for the attachment of the 2D laser scanner would be on the boom of the excavator. The idea was to use this boom to rotate the 2D laser scanner without the need for an additional components and/or parts. This idea turned to be a bad idea because the boom of the excavator rotated too quickly. In order to make a successful scan at such a fine resolution the scanner needs to be rotated at slower speed to achieve more tolerable results. That is why a stepper motor could be a better choice for tilting the 2D laser scanner. This project can be improved so that the already scanned 3D vectors, which are used to draw a 2D depth image, are built into some 3D model format which can be opened and viewed in a 3D modelling program. In addition some multiple stationary 3D scans could be made and merged together to form a larger 3D model of an environment or an area.

Key words: 3D model, 2D laser scanner, environment modelling, tilt angle measurement

SISÄLLYS

JOHDANTO 6
1 2D-LASERSKANNERI 7
1.1 Toimintaperiaate 7
1.2 Laser-mittauksen ominaisuuksia 9
1.3 Käyttöturvallisuus 9
1.4 Mittaustulosten keruu 9
1.5 Verkkoyhteydet ja yhteyksien ohjelmointi datan keräämiseksi 10
2 PYSTYKULMIEN MITTAUS 15
2.1 Mittausperiaate 15
2.2 Kiihtyvyysanturi 15
2.3 Gyroskooppi 18
2.4 Magnetometri 19
2.5 Anturidatojen käsittely ja niiden fuusio 21
2.6 Fixed point -luvut 26
2.7 Kvaternio-formaatti ja tämän muunnos Euler-kulmiksi 28
2.8 CAN-väylä 29
2.9 3D-mittausalgoritmi 32
3 TULOKSET 34
4 POHDINTA 36
LÄHTEET 38
LIITTEET 41
Liite 1. Skannattu kuva suurennettuna 41

LYHENTEET JA TERMIT

A/D-muunnin	Muunnin, joka muuntaa analogisen jännitteen binääriluvuksi (Analog to Digital)
AMR	Vastus, jonka arvo muuttuu magneettikentän vaikutuksesta (Anisotropic Magneto Resistance)
Asynkroninen	Dataa voi kulkea vaihteleva määrä ennalta määrittämättöminä ajanhetkinä
DHCP	Jakaa dynaamisesti verkon konfiguraatioparametrit, kuten esimerkiksi IP-osoitteet (Dynamic Host Configuration Protocol)
DOF	Määrittää vapaasti muuttuvien suureiden määrän, joita käytetään yhtenäisen lopputuloksen laskentaan (Degrees of freedom)
DSP	Digitaalinen signaalin käsittely (Digital Signal Processing)
IMU	3D-kallistuskulmien mittauslaite (Inertial Measurement Unit)
IP	Verkko-osoite (Internet Protocol)
Kvaternio	Yksi matemaattinen tapa ilmaista kappaleen asento avaruudessa
MAC	Verkkolaitteen yksilöllinen ID-numero (Media Access Control)
MEMS	Mikroskooppisen kokoinen sähkömekaaninen laite (Microelectromechanical systems)
OSI-malli	Kuvaa tiedonsiirtoprotokollien yhdistelmän seitsemässä kerroksessa (Open Systems Interconnection model)
Pitch-kulma	Euler-kulma joka kuvaa kappaleen kääntymistä 3D-koordinaatistossa Y-akselin ympäri
Roll-kulma	Euler-kulma joka kuvaa kappaleen kääntymistä 3D-koordinaatistossa X-akselin ympäri
TCP	Internet-tiedonsiirto-protokolla (Transmission Control Protocol)
Yaw-kulma	Euler-kulma joka kuvaa kappaleen kääntymistä 3D-koordinaatistossa Z-akselin ympäri

JOHDANTO

3D-ympäristömallinnusta on tehty jo vuosia, mutta yhä saatavilla olevat 3D-laserskannerit ovat melko kalliita. Tämän työn tarkoituksena on tutkia, kuinka 3D-ympäristömallinnus voitaisiin toteuttaa edullisemmin yhdistelmällä halvempia antureita.

Tämä tutkielma sai alkunsa erään yrityksen pyynnöstä. Heillä on tarve skannata ja luoda 3D-malli ympäröivästä maastosta. Maastomallin avulla voidaan parantaa ja tehostaa työkoneen työjälkeä entisestään.

Tässä tutkielmassa käytetään hyväksi 2D-laserskanneria ja IMU-anturia, joita testataan käytännössä, ja tutustutaan niihin liittyvään teoriaan. Työssä perehdytään myös hieman TCP/IP-protokollaan, koska laserskannerin mittaama data siirretään kyseisellä protokollalla. Lisäksi tutustutaan hieman CAN-väylän toimintaan, koska IMU-anturi hyödyntää tätä väylää.

Työ tehdään ohjelmoimalla C++ -ohjelma, jolla kerätään 2D-laserskannerin mitatut etäisyydet ja IMU-anturin mittaamat kallistuskulmat. Kerättyjen tietojen perusteella on tarkoitus laskea ohjelmassa tarvittavat arvot ja piirtää mustavalkoinen syvyyskuva 2D-tasossa skannaustuloksen hahmottamista varten. Ohjelma tallentaa kaikki skannatut 3D-vektorit jokaisesta sektorin pisteestä 3D-mallin myöhempää rakentamista varten, mutta 3D-mallin luontia tässä työssä ei kuitenkaan tarkastella, vaan keskitytään valokuvamaisen tietokoneella piirretyn kuvan tekemiseen. Tässä työssä tavoitteena on toteuttaa yhden alueen 3D-skannaus kerrallaan.

1 2D-LASERSKANNERI

1.1 Toimintaperiaate

Laservaloa voidaan käyttää etäisyyksien mittaamiseen. Etäisyyksien mittaus perustuu pulssimaisen laservalon lähettämiseen, ja sen takaisin heijastumisen havaitsemiseen. Takaisin heijastuminen mitataan valodiodin avulla. Etäisyys lasketaan valopulssin lentoajasta, toisin sanoen siitä, kun laservalo lähtee laserdiodista ja saapuu takaisin valodiodiin. (Amann, Lescure, Myllylä & Rioux 2000, 12; TiM5xx manuaali 2015, 6.)

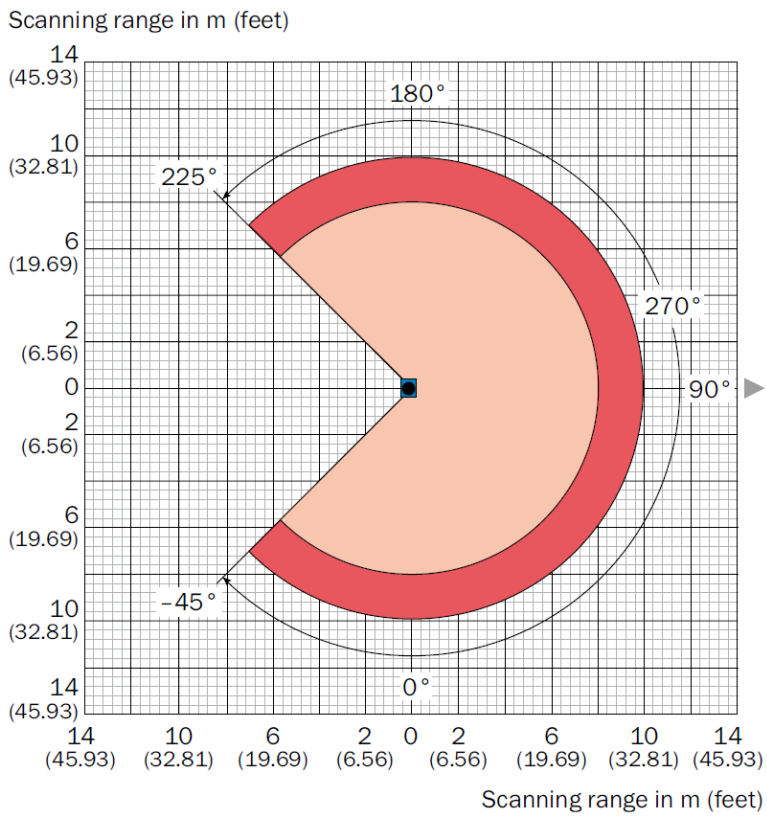
Mitattu etäisyys lasketaan kaavalla

$$s = c * {}^{t_{rt}}/_{2}, \qquad [1]$$

jossa s on etäisyys metreinä, c on valon nopeus (n. $3*10^8$ m/s) ja t_{rt} on valon edestakainen kulkuaika. Mikäli mittaustarkkuutena halutaan käyttää 1 mm, niin huomioitavaa on, että laserpulssin aikaintervalli pitää olla vähintään 6,7 ps. Esimerkkinä on laskettu pienin mitattava etäisyys 6,7 ps laserpulssin aikaintervallilla. (Amann, Lescure, Myllylä & Rioux 2000, 12; TiM5xx manuaali 2015, 6.)

$$s = 3 * 10^8 \frac{\mathrm{m}}{\mathrm{s}} * {}^{6{,}7 * 10^{-12}\ s}/_{2} = 0{,}001\ \mathrm{m} \qquad [2]$$

2D-lasermittauksessa käytetään pyörivää peiliä, joka taittaa laserpulssit 270-asteen alueelle. Nämä pulssit mitataan valodiodin avulla 0,33-asteen välein, jolloin saadaan kaikkiaan 811 kappaletta mitattuja etäisyyksiä. Työssä käytettävä 2D-laser kykenee mittaamaan 15 Hz taajuudella. Kuvassa 1 on havainnollistettu skannausaluetta. (Amann, Lescure, Myllylä & Rioux 2000, 12; TiM5xx manuaali 2015, 6.)



KUVA 1. Laserskannausalue (TiM5xx käyttöohje 2015)

2D-laserskannerilla mitataan 2D-vektoreita 270 kappaletta 1-asteen resoluutiolla käyttäen sisään rakennettua mediaanisuodinta tai 811 kappaletta 0,33-asteen resoluutiolla ilman tätä suodinta. Tämä mediaanisuodin voidaan asettaa päälle tai pois päältä valmistajan tarjoaman ohjelmiston avulla.

1.2 Laser-mittauksen ominaisuuksia

Laservalolla mitattaessa etäisyyksiä on otettava huomioon eri materiaalien valon heijastuvuuskyky. Taulukossa 1 on esitelty tämän työn kannalta tärkeimpien materiaalien kykyä heijastaa laservaloa.

TAULUKKO 1. Laservalon heijastuvuuksia eri materiaaleilla (Reflectivity Of… 2016, 3)

Materiaali	Heijastuvuus (%)
Betoni (sileä)	24
Asfaltti (pienikivinen)	17
Lumi	80 - 90

1.3 Käyttöturvallisuus

Kun käytetään laservaloja ihmisten lähettyvillä, niin niistä ei saa koitua haittaa silmille. Työssä käytetyssä 2D-laserskannerissa on 850 nm infrapunalaser, joka on silmäturvallinen luokan 1 laser perustuen standardiin EN 60825-1 (2007-10). (2D laser scanners TiM5xx / TiM56x / Indoor 2016.)

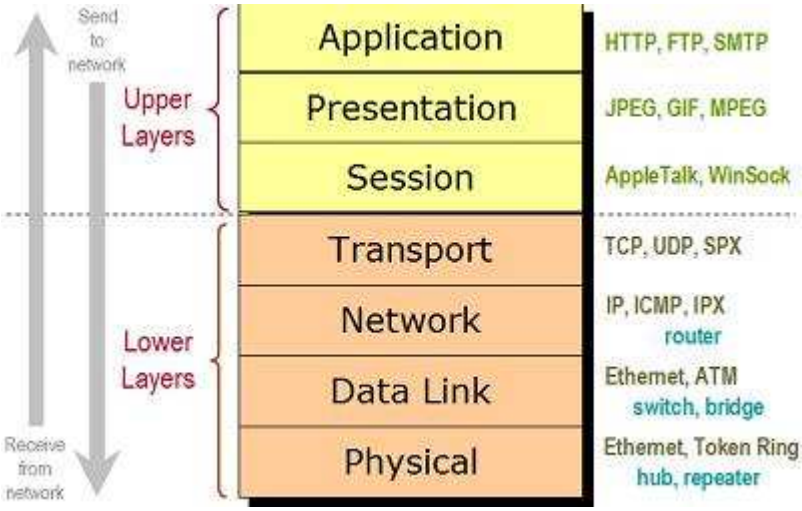
Luokan 1 laserin suurin sallittu valoteho on 0,39 mW. Tähän samaan luokkaan kuuluvat muun muassa CD- ja DVD-soittimet. Kirjoittavat CD- ja DVD-soittimet eivät kuulu tähän luokkaan. (Laserluokat 2015.)

1.4 Mittaustulosten keruu

2D-laserskanneri kytketään Ethernet-kaapelilla (RJ-45), joko verkkokytkimeen, reitittimeen tai PC:n verkkokorttiin, ja sen IP-asetukset asetetaan valmistajan tarjoamalla ohjelmistolla. Konfigurointi-ohjelma osaa automaattisesti määrätä 2D-laserskannerille IP-osoitteen ja aliverkonpeitteen, jotka kyseinen ohjelma tallentaa 2D-laserskannerin muistiin. Verkkoon liittyvät asiat käsitellään seuraavassa luvussa tarkemmin. 2D-laserskanneri lähettää TCP-protokollalla ASCII-muotoista dataa, joka on tarkemmin määritelty 2D-laserskannerin datakirjassa.

1.5 Verkkoyhteydet ja yhteyksien ohjelmointi datan keräämiseksi

2D-laserskannerin mittaustulokset lähetetään Ethernet-liitynnän kautta TCP/IP-protokollan avulla. Ennen kun syvennytään TCP/IP-protokollaan ja Ethernet:iin, tarkastellaan aluksi hieman OSI-mallia. OSI-malli on kehitetty referenssiksi havainnollistamaan monimutkaisempia tiedonsiirtojärjestelmiä, jotta niitä voidaan ymmärtää helpommin. OSI-mallin ideana on helpottaa internet-tiedonsiirtojärjestelmän suunnittelua, sekä kyseisen järjestelmän toiminnan havainnollistamista. Ethernet on OSI-mallin fyysinen- ja linkkikerros. TCP on OSI-mallin kuljetinkerros. Edellä mainitut asiat on havainnollistettu kuvassa 2.



KUVA 2. OSI-malli (Mitchell 2015)

Fyysinen kerros vastaa datan lähettämisestä, ja sen vastaanottamisesta. Tämä kerros huolehtii tarvittavasta johtokoodauksesta ja signaloinnista. TCP toimii kuljetuskerroksella, jonka tarkoituksena on sallia tietojenvälitys lähde- ja kohdekoneiden välillä. TCP on yhteydellinen yhteys, joka tarjoaa luotettavan datan siirron, sekä hävinneiden pakettien uudelleen lähettämisen. TCP tarjoaa myös luotettavan datavirran lähetyksen. TCP-yhteys toimii asiakas/palvelin- periaatteella. Ethernet hoitaa datan kuljetuksen, mutta se ei vastaa hävinneistä paketeista.

IP-osoitteista on hyvä tietää, että palvelimen ja asiakkaan on oltava samassa osoiteavaruudessa, jotta ne voivat kommunikoida keskenään. IP-osoite on laitekohtainen ja kahdella laitteella ei saa olla samaa IP-osoitetta. Aliverkonpeite määrittää aliverkon osoiteavaruuden, jossa voi olla vaihteleva määrä verkkolaitteita riippuen peitteen suuruudesta. Aliver-

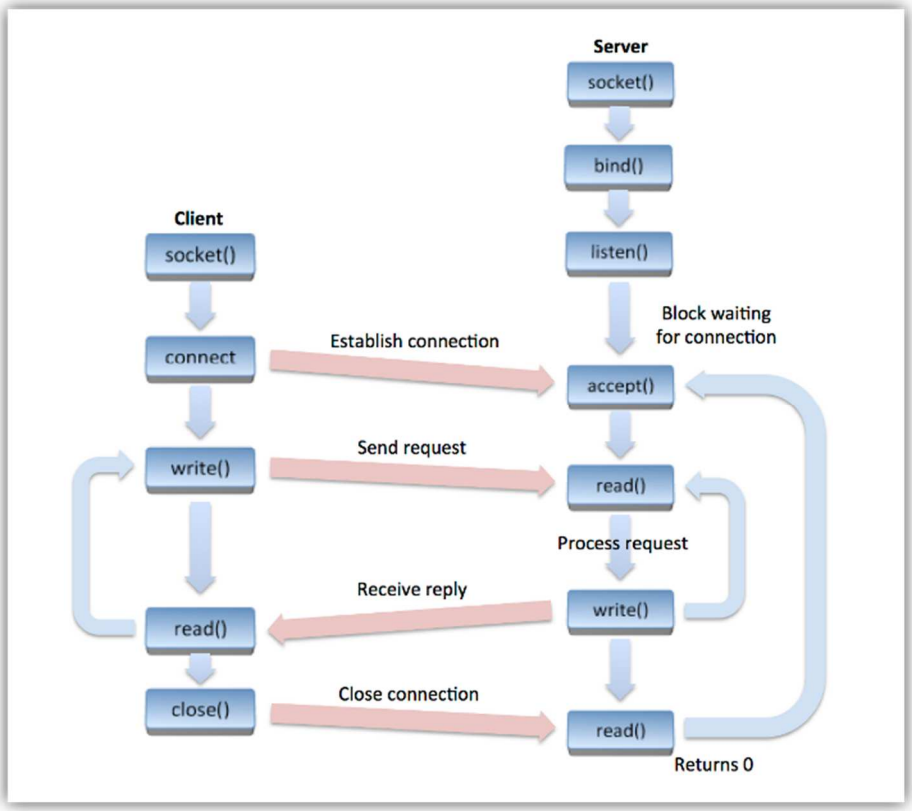
konpeite ilmoitetaan usein desimaalinumerosarjana pisteillä erotettuna, joka voi olla esimerkiksi 255.255.255.0. Tämä on yleinen C-luokan aliverkon peite. Toinen merkintätapa kyseiselle aliverkolle on /24, joka tarkoittaa, että aliverkonpeitteen numerosarjassa on vasemmalta oikealle luettaessa 24 MSB-bittiä 1-tilassa. Numerosarja heksadesimaaleina ilmoitettuna on näin ollen FF.FF.FF.00, mikä binäärimuodossa vastaa bittisarjaa 11111111.11111111.11111111.00000000. Binäärisessä muodossa kaikki 0-tilaa ilmaisevat bitit ovat käytettävissä IP-osoitteen määrittämiseksi.

Esimerkiksi tällä aliverkonpeitteellä IP-osoitteet x.x.x.0 – x.x.x.255 ovat samassa aliverkossa. Pisteillä erotetut x-kirjaimet edustavat jotakin verkko-osoitteen alkuosaa. Osoitealue lasketaan aliverkonpeitteen 0-tilaa ilmaisevista biteistä, kun ne ovat aluksi 0-tilassa, ja päättyvät siihen, kun ne asetetaan 1-tilaan. Tällöin binääriluvut ovat väliltä 00000000 – 1111111. Desimaaleina ilmaistuna nämä luvut ovat väliltä 0 – 255. Esimerkiksi osoitteet 192.168.0.0 – 192.168.0.255 ovat samassa osoiteavaruudessa. Tässä verkossa voi siis olla yhteensä 252 eri laitetta.

Tässä tapauksessa osoite 192.168.0.0 on varattu verkon osoitteelle ja osoite 192.168.0.1 on varattu reitittimelle, sekä osoite 192.168.0.255 on varattu broadcast-lähetteille. Muut osoitteet ovat vapaasti käytettävissä. Mikäli laitteet kytketään lähiverkkoon, niin reitittimessä on useimmiten käytössä DHCP-palvelin. DHCP-palvelimen tehtävänä on jakaa lähiverkkoon kytketyille verkkolaitteille yksilölliset IP-osoitteet hallitusti. DHCP-palvelin voi järjestää IP-osoitteet eri laitteille siten, että aiemmin verkkolaitteille määrätyt osoitteet pysyvät samoina jonkin määräajan, vaikka nämä laitteet eivät näkyisi aktiivisina verkossa.

DHCP-palvelin tunnistaa eri laitteet niiden yksilöllisen MAC-osoitteen perusteella. Mikäli jokin laite ei esiinny aktiivisena lähiverkossa määräaikaan mennessä, sen osoite vapautetaan ja voidaan määrätä uudelle verkossa esiintyvälle laitteelle. IP-osoitteet voidaan määrittää myös staattisiksi, mutta silloin käyttäjän on itse pidettävä kirjaa IP-osoitteista, jotta samaa osoitetta ei määrätä useammalle laitteelle.

Seuraavaksi käydään hieman läpi TCP-ohjelmoinnin perustoimintoja. Kuvassa 3 on esitetty TCP-yhteyden muodostus palvelimen ja asiakkaan välille.



KUVA 3. TCP-yhteyden muodostuskaavio (Hargrave 2013)

TCP-yhteyden muodostamiseksi palvelin/isäntä-ohjelma kutsuu socket-funktion, jonka avulla luodaan asiakkaita kuunteleva liityntä (socket). Seuraavaksi määritetään kuunneltavan asiakkaan IP-osoite ja TCP-portti, miltä yhteyspyyntöä odotetaan. Tämän jälkeen-kutsutaan bind-funktio. Sitten kutsutaan listen-funktio, joka jää estämään ohjelman suorituksen etenemisen, kunnes accept-funktio kutsutaan asiakkaan pyytämän yhteyden hyväksymiseksi. Tässä vaiheessa palvelin-ohjelma on luku- ja kirjoitusmoodissa. (Hargrave 2013; Kozierok 2015.)

TCP-yhteyden muodostamiseksi asiakas puolestaan yhdistyy palvelimeen kutsumalla asiakkaan omaa socket-funktiota ja sitten connect-funktiota. Asiakas käyttää liitynnän (socket) IP-osoitetta ja TCP-porttia, jotka ovat samat, mitkä ovat käytössä palvelimen bind-funktiota kutsuttaessa. Palvelimen päässä accept-funktio palauttaa muodostetun yhteyden (connection socket) kuvauksen (descriptor), kun asiakkaan yhteyspyyntö on vastaanotettu. Tämän jälkeen, kun yhteys on muodostettu asiakkaan ja palvelimen välille, palvelinohjelman suoritus jää odottamaan asiakkaan pyyntöjä. (Hargrave 2013; Kozierok 2015.)

Asiakas kutsuu write-funktiota lähettääkseen pyynnön palvelimelle ja jää sitten odottamaan palvelimen vastausta read-funktiota kutsumalla. Kun palvelin on suorittanut valmiiksi asiakkaan pyytämän tehtävän, se lähettää vastauksen asiakkaalle. Asiakkaan ja palvelimen vuorovaikutus toistuu, kunnes asiakas on saanut haluamansa palvelut toteutetuiksi. Tämän jälkeen asiakas sulkee yhteyden. Palvelin havaitsee yhteyden lopetuksen siten, että palvelimen read-funktio palauttaa arvon 0. Palvelin sulkee tällöin myös yhteyden ja valmistautuu muodostamaan uuden yhteyden seuraavan asiakkaan kanssa. (Hargrave 2013; Kozierok 2015.)

Huomioitavaa on, että usein monet asiakas/palvelin-yhteydet käsitellään rinnakkaisina ohjelmien suorituksina (multiple threads), mutta tässä opinnäytetyössä on käytetty sarjamuotoista ohjelmasuoritusta, koska yhteys muodostetaan vain yhden asiakkaan kanssa. (Hargrave 2013; Kozierok 2015.)

Seuraavaksi on esitelty muutamia matalan tason yhteydenmuodostus-ohjelmaan liittyviä asioita. TCP-puskuriin kirjoitetaan matalan tason ohjelmalla asynkronisesti asiakkaan lähettämä data, ja read-funktiolla TCP-puskurista luetaan dataa ylemmän tason ohjelman muistiin. TCP-vastaanottopuskurissa on matalan tason ohjelmassa sisäinen luentaosoitin. Luentaosoitin muistaa kohdan, josta datan lukua jatketaan silloin, kun read-funktiota kutsutaan seuraavan kerran uudelleen. TCP-puskuria ei voi tyhjentää millään komennolla, vaan se tyhjenee sitä mukaa, kun sieltä luetaan dataa. Tämä asia on otettava huomioon ohjelman suunnittelussa.

Esimerkiksi, mikäli 2D-laserskanneri on datavirran lähetysmoodissa, ja halutaan lukea 2D-laserskannerin lähettämää skannausdataa tietystä vertikaalisesta skannauskulmasta. Tällöin TCP-vastaanottopuskuriin tallentuu kyseisen skannauskulman dataa. Tämän skannatun kulman data on luettavissa aivan normaalisti ylemmän tason ohjelmassa. Seuraavaksi muutetaan skannauskulmaa ja mitataan uutta 2D-laserskannerin lähettämää dataa. Tämä data menee myös TCP-vastaanottopuskuriin moitteettomasti. Nyt kun yritetään lukea tämä uusi data TCP-vastaanottopuskurista, niin saadaankin luettua vain vanhan skannauskulman arvoja. Tämä johtuu siitä, että skannausdatavirtaa on mennyt TCP-vastaanottopuskuriin todella paljon ensimmäistä kulmaa mitattaessa ja seuraavan skannauskulman datan alkamiskohtaa TCP-puskurista on vaikea määrittää. Tässä tapauksessa 2D-laserskanneri on asetettava lähettämään skannausdataa asiakkaan pyynnöstä vain kerran.

Tällöin TCP-vastaanottopuskuriin ei tulvi turhaa skannausdataa, ja näin voidaan varmistua, että viimeisin skannattu data on heti saatavilla TCP-vastaanottopuskurista skannauksen jälkeen.

2 PYSTYKULMIEN MITTAUS

2.1 Mittausperiaate

Tämän työn tavoitteena on tehdä laitteisto, jolla voidaan skannata ja mallintaa maastoa 3D-tasossa käyttämällä 2D-laserskanneria. Pelkkä 2D-laserskannerilla mitattu tieto ei riitä 3D-mallinnukseen, vaan tarvitaan myös tietoa siitä, mikä on pystykallistuskulma skannauksen aikana. Tämä pystykallistuskulma voidaan mitata käyttämällä kolmen eri anturien mittaamien datojen fuusiota. Tämä fuusio tehdään kiihtyvyysanturin, gyroskoopin ja magnetometrin mittaamista datoista, joiden kaikkien parhaat ominaisuudet yhdistetään. Teoriassa gyroskoopista saadaan pelkästään tarvittavat tiedot kappaleen asennon määrittämiseksi 3D-avaruudessa, mutta käytännössä tämä ei onnistu aivan niin helposti, koska mittavirheet kasvavat ajan funktiona datan integroinnin seurauksena.

Kiihtyvyysanturilla voidaan myös pelkästään mitata kallistuskulmia hyödyntäen maan vetovoimaa, mutta mittausdatan luotettavuus menetetään heti, mikäli tämä anturi joutuu kiihtyvään liikkeeseen eli anturi tärähtää tai värähtelee mekaanisesti. Myös anturin siirtyminen paikasta toiseen aiheuttaa häiriötä kallistuskulmien mittaukseen. Kuten aiemmin todettiin, niin kiihtyvyysanturilla kallistuskulmien mittaamiseksi voidaan hyödyntää maan vetovoimaa, joka on noin 9,81 m/s^2 eli 1 g. Tästä saadaan luotettava referenssivektori kallistuskulmien määrittämiseksi.

Magnetometri toimii kuten kompassi, ja sen avulla voidaan mitata maapallon magneettikenttää hyödyntäen eri ilmansuuntia. Magnetometri antaa myös luotettavan referenssivektorin, joka osoittaa pohjoiseen. Näiden edellä mainittujen kolmen eri anturien fuusiota kutsutaan lyhenteellä IMU.

2.2 Kiihtyvyysanturi

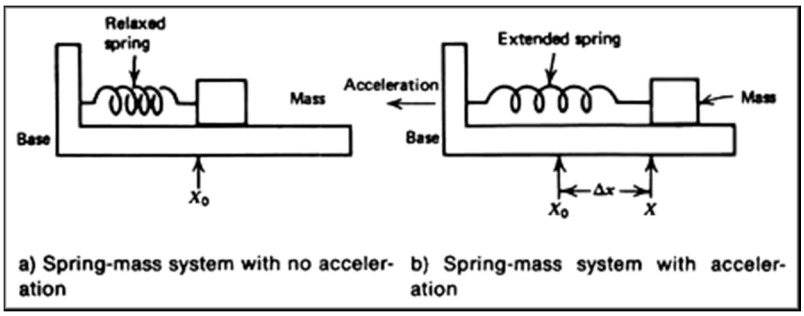
Kiihtyvyys voidaan määritellä Newtonin lain mukaan siten, että massan m ollessa kiihtyvässä liikkeessä a, voiman F on vaikutettava kyseiseen massaan. Tämä suhde on todistettavissa kaavalla (Understanding Acceleration and … 2013)

$$F=ma \qquad [3]$$

Hooken lain mukaan jousi, jonka jousivakio on *k*, venytetään sen tasapainoasemasta *Dx* matkan verran. Tällöin jouseen on kohdistuttava voima *F*. Tämä on todistettavissa kaavalla (Understanding Acceleration and … 2013)

$$F=kDx \qquad [4]$$

Kuvan 4a-kohdassa massa on vapaasti liikkuvalla alustalla, ja se on kiinnitetty alustaan lepotilassa olevalla jousella. Kuvan 4b-kohdassa alusta on kiihtyvässä liikkeessä vasemmalle päin, jolloin jousi on venynyt sen verran, että tämä aiheuttaa voiman massaan, joka saa massan kiihtyvään liikkeeseen.



KUVA 4. Jousi-massa-systeemi-kiihtyvyysanturi (Understanding Acceleration and … 2013)

Tilanne voidaan esittää yhdistämällä Newtonin ja Hooken lait kaavalla (Understanding Acceleration and … 2013)

$$ma=kDx, \qquad [5]$$

jossa *k* on jousivakio [N/m], *Dx* on jousen venymä [m], *m* on massa [kg] ja *a* on kiihtyvyys [m/s^2].

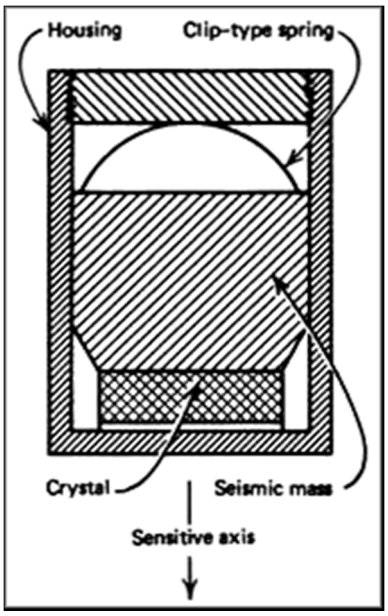
Edellä mainittu kaava voidaan muuttaa muotoon

$$a = \frac{k}{m}\Delta x \qquad [6]$$

kaava todistaa, että mitattavaksi suureeksi riittää vain jousen poikkeama Δx tasapainoasemasta. Mikäli kiihtyvyys on negatiivinen, niin jousi painuu kasaan sen sijaan, että se

venyisi. Tällöin edellä mainittu kaava on myös voimassa. Jousi-massa-systeemi on nykyäänkin perustana monissa kiihtyvyysanturien suunnitteluissa. Massalla, joka muuntaa kiihtyvyyden jousen poikkeamaan tasapainoasemasta, viitataan testimassaan tai seismiseen massaan (seismic mass). Tällä tavalla voidaan ymmärtää, että kiihtyvyyden mittaus perustuu yksinkertaistettuna jousen poikkeamaan tasapainoasemasta. Monien anturien suunnitteluissa käytetään eri menetelmiä tämän poikkeaman mittaamiseksi. (Understanding Acceleration and … 2013)

Nykyään kiihtyvyyden mittaamiseen käytetään muun muassa piezosähköistä ilmiötä hyväksi. Tällainen menetelmä perustuu tietynlaiseen kiteen ominaisuuteen, joka muodostaa jännitteen sen yli, mikäli tämä joutuu paineen kohteeksi. Toisin sanoen kide joutuu mekaanisesti puristuksiin tai venytykseen. Tämän kiihtyvyysanturin perusidea on esitetty kuvassa 5 (Understanding Acceleration and … 2013)



KUVA 5. Piezosähköinen kiihtyvyysanturi (Understanding Acceleration and … 2013)

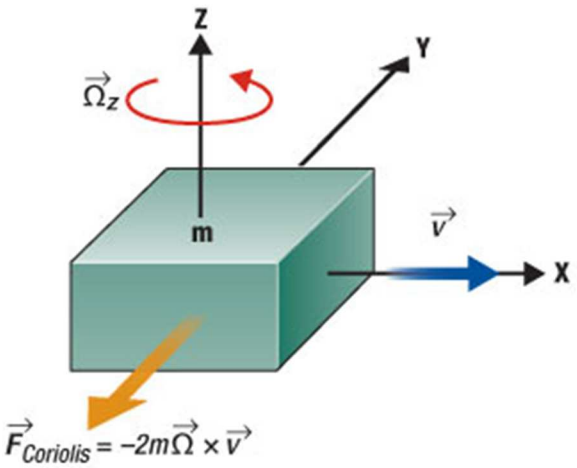
Piezosähköiseen kiteeseen on kiinnitetty testimassa tai toiselta nimeltään seisminen massa. Kun systeemi joutuu kiihtyvään liikkeeseen, jousi aiheuttaa stressiä piezosähköiseen kiteeseen voimalla $F=ma$, joka puolestaan aiheuttaa jännitteen kiteen yli. Kiihtyvyys voidaan mitata tämän jännitteen avulla. Kide on erittäin suuri-impedanssinen jännitelähde, jolloin sen yli vaikuttavan jännitteen mittaamiseksi tarvitaan suuri-impedanssisen sisäänmenon omaava ja pienikohinainen ilmaisin, sekä vahvistin. Ulostulojännitteet ovat

kiteillä usein millivolttien luokkaa. Piezosähköisten kiihtyvyysanturien luonnollinen taajuus voi olla 5 kHz, joten niitä voidaan käyttää värähtelyiden ja tärähdyksien mittaamiseen. (Understanding Acceleration and … 2013)

2.3 Gyroskooppi

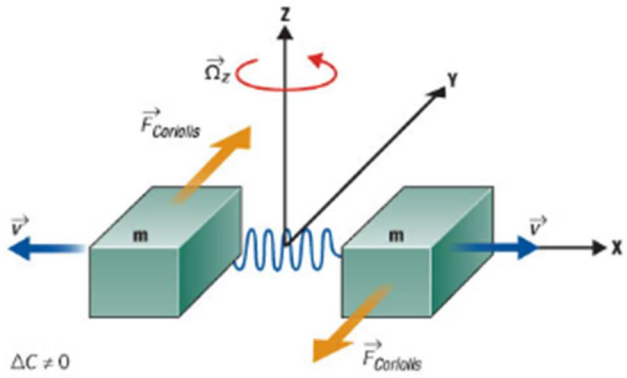
Gyroskooppi perustuu pyörivän hyrrän pyörimisliikkeeseen ja tämän kykyyn pyrkiä vastustamaan olotilansa muutoksia. Gyroskooppia käytetään mittaamaan kulman muuttumisnopeutta. Kääntymistä mitataan yleensä X-, Y- tai Z-akseliin verrattuna. (Esfandyari, De Nuccio & Xu 2010.)

Gyroskooppeja voidaan tehdä erilaisin keinoin, mutta tässä yhteydessä puhutaan vain MEMS-antureista. MEMS-gyroskooppi perustuu Coriolis-ilmiöön, jonka avulla kulman muutosta mitataan. Kuvassa 6 on esitetty Coriolis-ilmiön periaate. (Esfandyari, De Nuccio & Xu 2010.)



KUVA 6. Coriolis-ilmiö (Esfandyari, De Nuccio & Xu 2010)

Kun massa *m* liikkuu vektorin *v* suuntaan ja on tasaisesti pyörivässä liikkeessä Z-akselin ympäri kulmanopeudella Ω, tällöin massaan kohdistuu voima keltaisen nuolen suuntaan ($F_{Coriolis}$). Tämä on Coriolis-voima, josta aiheutuu massan fyysinen siirtyminen. Massan siirtyminen havaitaan kapasitiivisesti mittaavan rakenteen avulla. Monissa MEMS-gyroskoopeissa käytetään tuning fork -konfiguraatiota, joka tarkoittaa, että kaksi massaa värähtelevät ja liikkuvat jatkuvasti vastakkaisiin suuntiin (kuva 7). (Esfandyari, De Nuccio & Xu 2010.)



KUVA 7. Tuning fork -gyroskooppi (Esfandyari, De Nuccio & Xu 2010)

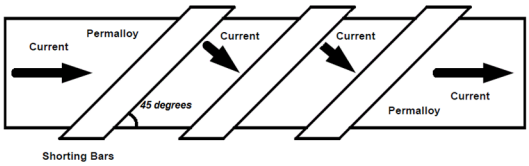
Kun systeemiin lisätään kulmanopeus Ω Z-akselin ympäri, Coriolis-voima vaikuttaa molemmilla massoilla vastakkaisiin suuntiin. Tämä aiheuttaa kapasitanssin muutoksen. Kapasitanssin muutos on kulmanopeuteen Ω verrannollinen, ja se muunnetaan analogisessa anturissa jännitteeksi tai digitaalisessa anturissa binääriluvuksi A/D-muuntimen avulla. Kun massoihin vaikuttaa lineaarinen kiihtyvyys, ne liikkuvat samaan suuntaan, jolloin kapasitanssin muutosta ei havaita. Tämä todistaa sen, että MEMS-gyroskooppi ei ole herkkä lineaariselle kiihtyvyydelle. Lineaarista kiihtyvyyttä aiheuttaa esimerkiksi kallistus, tärähdys tai värähtely. Huomioitavaa on, että lämpötilan muutos ja lämpökohina aiheuttaa mitattuun gyroskooppidataan häiriöitä, vaikka gyroskooppi olisi vakaasti paikallaan. Häiriöt aiheuttavat virhettä lopullisien kulmien mittauksissa, koska mitatut kulmien muutosnopeudet integroidaan. Tätä asiaa on käsitelty tarkemmin kohdassa anturidatan käsittely. (Esfandyari, De Nuccio & Xu 2010; Geen & Krakauer 2003.)

2.4 Magnetometri

Magneettikenttiä voidaan mitata monellakin eri tapaa, mutta tässä opinnäytetyössä tutustutaan tarkemmin vain AMR-anturiin, ja sen toimintaperiaatteeseen. Anisotrooppinen magnetoresistiivinen anturi on valmistettu permalloy nimisestä aineesta (NiFe), joka sisältää noin 80 % nikkeliä ja 20 % rautaa. Permalloy on samankaltaista ainetta, kuin myymetalli. Näillä aineilla on suuri magneettinen permeabiliteetti, eli kyky varastoida magneettikentän energiaa. (What is Permalloy? 2016.)

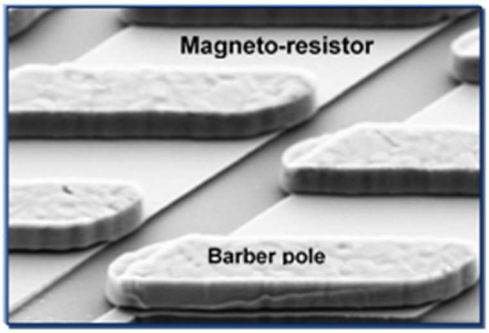
AMR-anturi on valmistettu ohuesta filmin palasesta permalloy-metallia, joka on asetettu pii-alustalle resistiiviseksi nauhaksi. Tämän nauhan päälle on asetettu bias-oikosulkupa-

laset, jotka pakottavat virran kulkusuunnan 45-asteen kulmaan permalloy-nauhaan nähden. Virta kulkee lyhintä reittiä 45-asteen kulmassa bias-palasesta toiseen. Kuvassa 8 on esitetty AMR-vastuksen periaate. (Mason 2003; Magneto resistor 2016.)



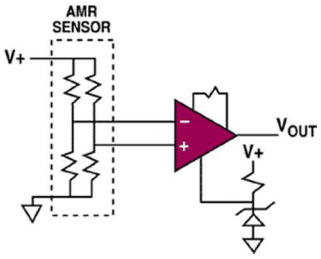
KUVA 8. AMR-vastuksen periaatteellinen kuva (Magneto resistor 2016)

Kuvassa 9 on esitetty mikroskooppikuva AMR-vastuksesta.



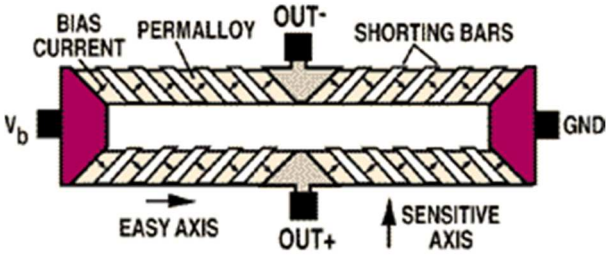
KUVA 9. AMR-vastuksen mikroskooppikuva (Anisotropic Magneto Resistance 2016)

AMR-vastuksen altistuessa magneettikentälle, sen magneettiset alueet heilahtavat ympäri ja sähköinen resistanssi muuttuu noin 2 – 3 %. Tyypillisesti 4 kappaletta tällaisia magnetoresistiivisiä vastuksia on asetettu Wheatstonen silta -konfiguraatioon, kuten kuvassa 10 on esitetty. (Mason 2003; Magneto resistor 2016.)



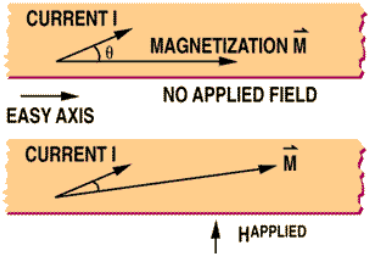
KUVA 10. Magnetoresistiiviset vastukset Wheatstonen silta -kytkennässä (Caruso, Bratland, Smith & Schneider 2009)

Kuvassa 11 on esitetty kokonaisuudessaan AMR-vastus Wheatstonen silta -kytkennässä. Kuvassa näkyy mitattavan magneettikentän akseli (sensitive axis) suhteessa AMR-vastuksiin nähden. (Magneto resistor 2016.)



KUVA 11. 4 kpl AMR-vastuksia Wheatstonen silta -kytkennässä (Caruso, Bratland, Smith & Schneider 2009)

Kuvassa 12 on esitettynä vektorimuodossa virran ja magneettikentän väliset suhteet.



KUVA 12. Virran ja magneettikentän suunnat vektoreina esitettynä (Caruso, Bratland, Smith & Schneider 2009)

Kuvan 12 ylemmässä kohdassa on esitetty tilanne, jolloin AMR-vastukseen ei vaikuta ulkoista magneettikenttää ja alemassa kohdassa AMR-vastukseen vaikuttaa magneettikenttä ($H^{APPLIED}$). Permalloy-metallissa on magnetoitumavektori M, johon vaikuttaa mitattava ulkoinen magneettikenttä. Permalloy-filmin resistanssi muuttuu magnetoitumavektorin M, ja sen läpi kulkevan virran välisen kulman funktiona. Tämä resistanssin muutos on magnetoresistiivinen efekti. (Magneto resistor 2016.)

2.5 Anturidatojen käsittely ja niiden fuusio

Anturifuusio on pelkkä hieno nimitys mitattujen datojen virheiden korjauksille, ja sitä se nimenomaan onkin (Winer 2016). IMU koostuu yleensä 3 eri anturista. Nämä anturit ovat kiihtyvyysanturi, gyroskooppi ja magnetometri. Kyseiset anturit yhdessä muodostavat 9-DOF-mittauksen.

Tässä työssä tarkoituksena on mitata laserskannerin asento 3D-koordinaatistossa. Pelkästään gyroskoopin avulla voitaisiin teoriassa mitata tämä asento. Käytännössä kuitenkin pienetkin mittausepätarkkuudet, sekä gyroskoopin kohina kasvattavat mittausvirheitä ajan kanssa niin suuriksi, että tarkkaa asentoa on vaikeata mitata tai peräti asennon tarkasta määrittämisestä tulee täysin mahdotonta.

Gyroskoopin avulla mitataan kappaleen kääntymiskulmanopeutta. Kappaleen asento 3D-koordinaatistossa määritetään kääntymiskulmanopeusdatojen keräämisellä tasaisella intervallilla mahdollisimman lyhytkestoisen ajan verran ja mittaustulokset integroidaan, eli summataan lopuksi yhteen.

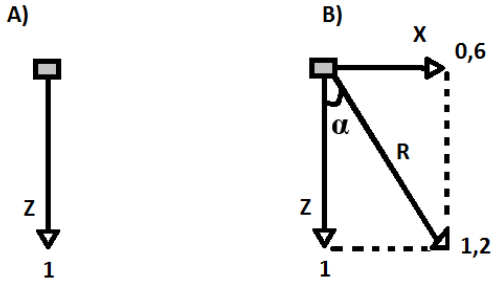
Esimerkki: Oletetaan tilanne, että gyroskooppi on levossa tasaisella alustalla, jolloin sen mittaamaksi arvoksi saadaan vaikka 0,01 °/s. Odotetaan, että kuluu vaikka 5 minuuttia eli 300 s. Tällöin integroitujen arvojen perusteella näyttäisi siltä, että kulma olisi kääntynyt peräti 3-astetta. Tämä voidaan laskea

$$300\ \text{s} * 0{,}01\ °/\text{s} = 3\ ° \qquad [7]$$

Gyroskooppi on ollut levossa paikallaan, mutta integroitu tulos osoittaa, että tämä olisikin kääntynyt 3-astetta 5 minuutin aikana. Tätä kutsutaan mitatun datan ryömimiseksi. Tämä kulman ryömiminen on syy siihen, miksi pelkkää gyroskooppia ei voida käyttää yksinään luotettavuutta vaativissa mittauksissa. Kappaleen asennon määrittämiseen tarvitaan lisäksi myös muita antureita, kuten esimerkiksi kiihtyvyysanturia. Yksi gyroskoopin ominaisuus on myös seuraavanlainen. Kun anturia käännetään vaikkapa 0-asteen lepotilakulmasta 50-asteen kulmaan ja takaisin, niin lopulliseksi kulman lukemaksi ei aina tule 0-astetta. Tällöin kulma voi vaihdella enemmänkin kuin 10-astetta. Tämäkin johtuu mittausepätarkkuuksista, ja niiden vaikutuksista mitattujen lukujen integroinnin tulokseen ajan funktiona.

Kiihtyvyysanturin mittaamasta datasta voidaan laskea kallistuskulmat käyttäen hyväksi maapallon vetovoimaa. Kiihtyvyysanturin ollessa levossa ja täysin suorassa tasaisella alustalla, siihen vaikuttaa vain maan vetovoima (9,81 m/s^2). Tämä voidaan ajatella yksikkövektoriksi, joka osoittaa positiivisen Z-akselin suuntaan, ja vektorin suuruus on tällöin 1. Tätä vektoria voidaan käyttää hyväksi referenssinä kallistuskulmien määrittämiseen.

Käytännössä kappaleen kallistuskulmia ei voida myöskään määrittää tarkkaan pelkästään kiihtyvyysanturilla. Tämä johtuu siitä, että kiihtyvyysanturi on todella kohinainen, ja se on herkkä erilaisille mekaaniselle liikkeelle, sekä värinälle. Tämä voidaan havainnollistaa kuvan 13 avulla.



KUVA 13. Kiihtyvyysanturin tarkastelu vektorien avulla

Kuvan 13 A-kohdassa kiihtyvyysanturi on lepotilassa, jolloin resultanttivektori osoittaa täysin alaspäin Z-akselin suuntaisesti. Kuvan 13 B-kohdassa anturia liikutetaan oikealle siten, että X-vektorin suuruudeksi muodostuu 0,6. B-kohdan tilanteessa resultanttivektori R muodostuu, kun Z- ja X-vektorit lasketaan yhteen. Tässä tapauksessa resultanttivektori voidaan laskea käyttäen Pythagoraan lausetta

$$R = \sqrt{Z^2 + X^2} \qquad [8]$$

sijoittamalla kaavaan arvot saadaan

$$R = \sqrt{1^2 + 0{,}6^2} = 1{,}167 \approx 1{,}2 \qquad [9]$$

Kuvan 13 B-kohdassa vektori on kulmassa α, joka voidaan laskea kaavalla

$$\alpha = \tan^{-1}\left(\frac{X}{Z}\right) \qquad [10]$$

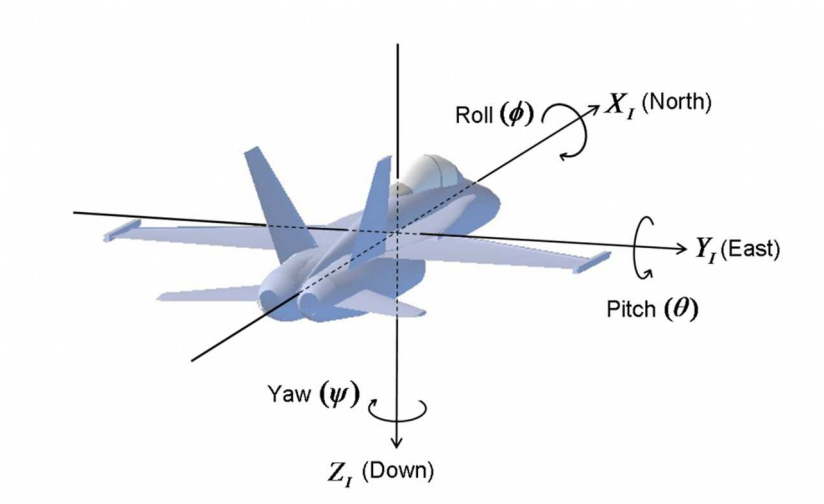
sijoittamalla luvut kaavaan saadaan

$$\alpha = \tan^{-1}\left(\frac{0{,}6}{1}\right) = 31° \qquad [11]$$

resultanttivektori osoittaakin nyt oikealle 31-asteen kulmassa Z-akseliin verrattuna. Resultanttivektorin avulla voidaan siis määrittää hetki, jolloin kiihtyvyysanturi lähtee liikkeelle. Toisin sanoen se osoittaa jotain muuta arvoa, kuin 1 positiivisen Z-akselin suuntaan. Tätä tietoa voidaan käyttää anturifuusiossa siten, että mitataan kiihtyvyysanturin mittaamia kulmia maan gravitaatiovektoria referenssinä käyttäen. Tällöin saadaan melko luotettavaa tietoa, koska gravitaatiovektori ei muutu ja on luotettava mittausreferenssi. Heti kun resultanttivektori indikoi, että kiihtyvyysanturi ei pysy paikallaan, niin voidaan vaihtaa kallistuskulmien määritys gyroskoopin mittaamaa dataa hyödyntämällä. Esimerkissä oli esitettynä, kuinka kiihtyvyysanturin ja gyroskoopin parhaita ominaisuuksia voidaan hyödyntää yhdessä kappaleen asennon määrittämiseksi 2D-avaruudessa. Tämä voidaan myös toteuttaa 3D-avaruudessa, jolloin resultanttivektori saadaan lasketuksi kaavalla

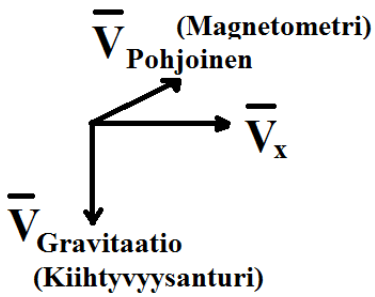
$$R = \sqrt{X^2 + Y^2 + Z^2} \qquad [12]$$

Kallistuskulmat kannattaa siis mitata paikallaan olevalta kiihtyvyysanturilta, ja nämä kulmat tallennetaan gyroskoopin mittaamien arvojen integroinnin lähtötiedoiksi jatkuvasti jokaisella mittaushetkellä. Näin saadaan toteutettua melko yksinkertaisesti luotettava kappaleen asennon määritys. Toisin sanoen kiihtyvyysanturin mittaamiin arvoihin aletaan summaamaan gyroskoopin mittausarvoja vain silloin, kun tiedetään, että kiihtyvyysanturi ei ole levossa. Edellä mainituilla tavoilla voidaan määrittää esimerkiksi Eulerin kulmia käyttäen vain Roll- ja Pitch-kulmat, mutta ei Yaw-kulmaa (kuva 14). Yaw-kulman määrittämiseen tarvitaan lisäksi magnetometri.



KUVA 14. Euler-kulmat (Understanding Euler Angles 2016)

Magnetometrillä mitataan magneettikenttiä, ja sen avulla voidaan saada selville eri ilmansuunnat. Anturifuusioon saadaan siis vielä lisää luotettavuutta ja tarkkuutta, kun voidaan mitata vektori, joka näyttää aina pohjoiseen. Tämä vektori yhdessä gravitaatiovektorin kanssa antaa jo todella hyvät referenssit kappaleen asennon määrittämiseksi. Kuvassa 15 on havainnollistettu näitä vektoreita



KUVA 15. Eri anturien mittaamat vektorit fuusion muodostuksessa

Kuvan 15 tuntematon V_x-vektori voidaan määrittää $V_{Pohjoinen}$ ja $V_{Gravitaatio}$ välisenä vektorien ristitulona. Olkoon laskuesimerkin vuoksi vektori $V_{Gravitaatio}$ A, olkoon vektori $V_{Pohjoinen}$ B ja olkoon vektori V_x C. Vektorit puretaan kompnenetteihin x,y ja z ja tällöin tuntematon vektori V_x voidaan määrittää kaavoilla (Cross Product 2014.)

$$\begin{aligned} C_x &= A_Y B_z - A_z B_y \\ C_y &= A_z B_x - A_x B_z \\ C_z &= A_x B_y - A_y B_x \end{aligned} \qquad [13]$$

Aiemmin oli esiteltynä yksinkertaistettu algoritmi anturidatan fuusioon, mutta tämä voidaan myös tehdä käyttämällä Kalman-suodinta tai komplementaari-suodinta. Komplementaari-suodin on melko yksinkertainen toteuttaa. Yksinkertaisimmassa muodossaan se voidaan esittää kaavalla. (Van de Maele 2013).

$$angle_{new} = 0{,}98 * (angle_{old} + gyrData * dt) + 0{,}02 * accData, \qquad [14]$$

jossa $angle_{old}$ on edellinen mittaustulos, gyrData on gyroskoopilta mitattu kulmanopeus ja accData on kiihtyvyysanturilta mitattu kiihtyvyysdata. Gyroskoopilta saatu data integroidaan, jokaisella aikaintervallilla nykyisen kulman arvon lisäksi. Tämän jälkeen tulos yhdistetään kiihtyvyysanturilta alipäästösuodatettuun dataan. Kaavassa luku 0,98 antaa

98 % painoarvon gyroskoopilta mitattuun ja integroituun arvoon, sekä luku 0,02 antaa 2 % painoarvon kiihtyvyysanturilta mitattuun dataan. Toisin sanoen gyroskoopilta mitattuihin arvoihin luotetaan tässä tapauksessa enemmän. Painoarvoja voidaan muuttaa, mutta niiden kokonaisprosenttiarvo pitää olla 100 %. Kalman-suodin on parempi, kuin komplementaari-suodin, mutta se on monimutkaisempi. Se on hyvä arvioimaan kappaleen sijaintia 3D-avaruudessa. (Van de Maele 2013).

2.6 Fixed point -luvut

Työssä käytetyn IMU:n lähettämät datat ovat Q16.16 fixed point -muodossa. Monissa nykypäivän tietokoneissa on natiivi liukulaskennan (floating point) tuki sisäänrakennettuna, mutta liukulukujen käyttö laskennassa ei ole ainoa tapa lukujen murto-osien laskentaan. Fixed point -lukuja käytetään useimmiten DSP-laskennassa, sekä tietokonepelien laskennassa, koska se on nopeampaa liukulukulaskentaan verrattuna. Fixed point -laskenta on epätarkempaa kuin liukulukulaskenta, ja näin ollen pyöristysvirheitä syntyy enemmän. Pyöristysvirheet eivät kuitenkaan aina haittaa sovelluksesta riippuen, mikäli laskennan nopeudella on korkeampi prioriteetti kuin tarkkuudella.

Tässä projektissa tietokone hoitaa laskennat ja laskennan tarkkuudella on korkeampi prioriteetti kuin sen nopeudella. Tästä syystä fixed point -luvut halutaan muuntaa liukuluvuiksi. Ennen lukujen muuntamista on hyvä hetken perehtyä fixed point -lukuihin. Fixed point -luvuissa on määriteltävä ennen laskentaa, kuinka monta bittiä halutaan varata kokonaislukujen esittämiseen, sekä kuinka monta bittiä esittää murto-osaa eli mantissaa. Fixed point -luvun esitystapaa Cheung (2016) on havainnollistanut kuvassa 12.

```
Decimal number:     123.45
                    ^^^ ^^
                    ||| ||
                    ||| |+---  weight = 1/100
                    ||| +----  weight = 1/10
                    ||+------- weight = 1
                    |+-------- weight = 10
                    +--------- weight = 100
```

KUVA 16. Fixed point -desimaaliluku ja painoarvot (Cheung 2016)

Kuvan 16 esimerkissä on 5-numeroinen desimaaliluku, jossa on varattu 3 numeroa esittämään kokonaislukua (luku 123), ja 2 numeroa (luku 0.45) ovat varattuja murto-osan esitykseen. Kuvassa 16 on havainnollistettuna, että luvun 4 painoarvo on 1/10 (4*(1/10)=0,4), ja luvun 5 painoarvo on 1/100 (5*(1/100)=0,05). Samalla periaatteella

toimii fixed point -luvut binäärisesti laskettaessa. Kuvassa 17 Cheung 2016 on esittänyt vastaavasti fixed point -binäärilukua, ja sen painoarvoja.

```
Binary number:     101.01
                   ^^^ ^^
                   ||| ||
                   ||| |+---  weight = 1/4
                   ||| +----  weight = 1/2
                   ||+------- weight = 1
                   |+-------- weight = 2
                   +--------- weight = 4

The value represented by   101.01  is:

                1*(4) + 0*(2) + 1*(1) + 0*(1/2) + 1*(1/4) =  5 1/4 = 5.25 (decimal)
```

KUVA 17. Fixed point -binääriluku ja painoarvot (Cheung 2016)

Binääriluku on 2-kantainen luku ja siksi painoarvoissa potenssien kantalukuna on 2. Esimerkiksi kuvan 17 binääriluku desimaaliksi laskettuna pisteen vasemmalta puolelta on

$$1*2^0+0*2^1+1*2^2=1+0+4=5 \qquad [15]$$

ja pisteen oikealta puolelta laskettuna

$$0*2^{-1}+1*2^{-2}=0+0{,}25=0{,}25 \qquad [16]$$

Lopullinen desimaaliluku saadaan, kun nämä luvut lasketaan yhteen

$$5 + 0{,}25 = 5{,}25 \qquad [17]$$

Tässä työssä Fixed point -luvut halutaan muntaa liukuluvuiksi, eli floating point -muotoon, jotta laskenta onnistuu mahdollisimman yksinkertaisesti. Tämä onnistuu melko yksinkertaisella tavalla. Kuvassa 14 on esitettynä eräs melko helppo tapa muunnoksen toteuttamiseksi.(How to convert … 2011.)

```
float DataReader::Q16dot16toFloat(uint32_t val)
{
    float value = float((int16_t)(val >> 16)); // whole part
    value += float((val & 0x0000FFFF)) / 65536; // fractional part

return value;
}
```

KUVA 18. Q16.16 fixed point -liukuluku -muunnos

Val-muuttujan sisältämästä fixed point -luvusta tallennetaan value-muuttujaan 16-bittinen kokonaislukua esittävä osa. Sen jälkeen val-muuttujasta erotetaan bittimaskin 0x0000FFFF avulla mantissa-osa käyttäen hyväksi bittitason AND-operaatiota. Jäljelle jäänyt mantissa-osa jaetaan luvulla 65536 (2^{16}), jolloin pelkästään mantissa-osa saadaan muunnettua liukuluvuksi. Tämä jakoluku on valittu siten, että muuttujan 16 LSB-bittiä esittää mantissa-osaa. Mikäli mantissa-osa esitettäisiin 8 LSB-bitillä, niin silloin jaettaisiin luvulla 256 (2^8). Tämän jälkeen summataan value-muuttujassa valmiiksi olevaan kokonaislukuun liukuluvuksi muunnettu mantissa-osa. Tällä tavalla fixed point -luku muunnetaan kätevästi liukuluku-muotoon. (How to convert … 2011.)

2.7 Kvaternio-formaatti ja tämän muunnos Euler-kulmiksi

Käytössä ollut IMU lähettää anturien fuusiodataa kvaternio-muodossa ja laser-maastomallinnus-C++ -ohjelmassa haluttiin käyttää Euler-kulmia hyväksi. Euler-kulmien käytössä on vaarana tapahtua ”gimbal lock”, joka tarkoittaa sitä, että kahden eri akselin ympäri laskettavat pyörimiskulmat joutuvat samalle akselistolle. Tällöin kaksi samassa tasossa olevaa akselistoa pyörittävät kohdetta samalla tavalla, mikä on epätoivottavaa. Yleensä tämä tapahtuu, mikäli jokin pyörimisakseli kääntyy 90-astetta. (Barbic 2012, 7.)

Kvaternio:ssa gimbal lock -ilmiötä ei tapahdu. Kvaternion avulla voidaan esittää ja laskea kappaleen kääntymistä, sekä asentoa 3D-avaruudessa. Kvaternio voidaan esittää järjestettynä parina (Bobic, N. 1998; Van Oosten, J. 2012.)

$$q = [s, v], \qquad [18]$$

jossa s on skalaari ja v on imaginäärivektori. Imaginäärivektori voidaan edelleen esittää seuraavalla tavalla

$$v = (x, y, z) \qquad [19]$$

Kvaternio voidaan myös esittää muodossa

$$q = [s, xi, yj, zk] \ , \qquad [20]$$

jossa s on skalaari (reaaliluku), sekä xi, yj ja zk ovat imaginäärilukuja.

Työssä käytetyn anturin lähettämä kvaternio on muotoa

$$q = [x, y, z, w] \qquad [21]$$

Tässä esitystavassa w on skalaari, sekä x, y ja z ovat edelleen imaginäärilukuja. Jälkimmäisestä kvaternion esitystavasta saadaan muunnettua Euler-kulmat radiaaneina laskemalla seuraavalla kaavalla (Baker, M. J. 2016)

$$\begin{bmatrix} \varphi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \mathrm{atan2}(2(wx + yz), 1 - 2(x^2 + y^2)) \\ \arcsin(2(wy - zx)) \\ \mathrm{atan2}(2(wz + xy), 1 - 2(y^2 + z^2)) \end{bmatrix}, \qquad [22]$$

jossa φ on kulma z-akselin ympäri (yaw), θ on kulma y-akselin ympäri (pitch) ja ψ on kulma x-akselin ympäri (roll). (Baker, M. J. 2016.)

Tämä kvaternion muunnos Euler-kulmiksi on ohjelmoitu C++ -kielellä kuvan 19 mukaisesti.

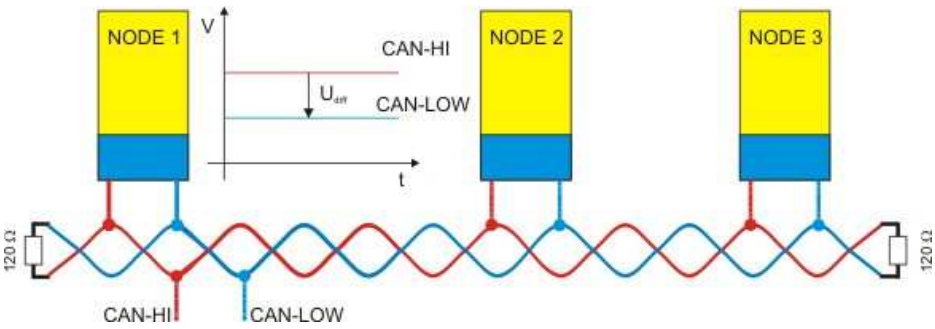
```
// kaavat antavat tuloksen radiaaneina, ja ne muutetaan asteiksi kertomalla luvulla 180/PI
float angleZ = (float) (atan2(2 * (w*x + y*z), 1 - 2 * (pow(x, 2) + pow(y, 2))) * 180 / M_PI);
float angleY = (float) (asin(2 * (w*y - z*x)) * 180 / M_PI);
float angleX = (float) (atan2(2 * (w*z + x*y), 1 - 2 * (pow(y, 2) + pow(z, 2))) * 180 / M_PI);

// tallennetaan eulerin kulmat sopiviin muuttujiin
_eulerAngles.X = angleX; // Roll-kulma
_eulerAngles.Y = angleY; // Pitch-kulma
_eulerAngles.Z = angleZ; // Yaw-kulma
```

KUVA 19. Kvaternio muunnos Euler-kulmiksi

2.8 CAN-väylä

Työssä käytetyt anturit kommunikoivat CAN-väylän avulla. CAN-väylä on lähetyksiin ja ilmoituksiin perustuva väylä. Tämä tarkoittaa, että kaikki laitteet kuulevat toisten laitteiden lähettämät viestit. Seuraavan sivun kuvassa 20 on esitetty CAN-väylän topologia.



KUVA 20. CAN-väylän topologia (BECKHOFF Automation CANopen: Mounting and Wiring 2016)

CAN-väylä on käytännössä kierretty parikaapeli, jossa ei ole jännitteen 0 voltin tasoa, vaan viestit kulkevat differentiaalijännitteinä. Liittimen nimitykset CAN-HI ja CAN-LOW saavat nimityksensä näistä differentiaalijännitetasoista, kuten kuvassa 20 on esitetty. Kaapelien päät pitää sovittaa 120 Ω päätevastuksilla, jotta vältytään signaalien heijastuksilta parikaapelissa. Huomioitavaa on myös se, että laitteet pitää kytkeä mahdollisimman lyhyillä johtimilla CAN-väylään.

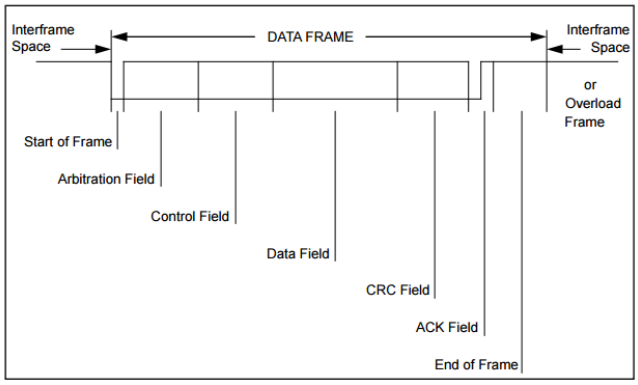
CAN-väylän maksimipituus riippuu väylässä käytettävästä baudinopeudesta. Rajoittavana tekijänä on signaalin etenemisnopeus. Väylään liitettyjen laitteiden pitää saada vastaanotettua viestit melkein samanaikaisesti ja väylään kirjoituksen sovittelu (arbitration) pitää saada kaikille laitteille samanaikaiseksi. Taulukossa 2 on esitetty CAN-väylän pituus suhteessa baudinopeuteen.

TAULUKKO 2. CAN-väylän pituus suhteessa baudinopeuteen (BECKHOFF Automation CANopen: Mounting and Wiring. 2016)

Baudinopeus	CAN-väylän pituus
1 Mbits/s	< 20 m
500 kBits/s	< 100 m
250 kBits/s	< 250 m
125 kBits/s	< 500 m
50 kBits/s	< 1000 m
20 kBits/s	< 2500 m
10 kBits/s	< 5000 m

Tässä työssä käytettiin baudinopeutena 250 kBits/s. Jokaisella laitteella on paikallinen viestisuodin, jonka avulla ne kuuntelevat vain viestejä, joista ne ovat kiinnostuneet. Laitteet ovat kytkettyjä väylään bittitasoisella loogisella AND-konfiguraatiolla. Mikäli siis edes yksi laite lähettää väylään 0-tilan, se dominoi tällöin väylää ja muut laitteet eivät saa kirjoittamaansa 1-tilaa esille samanaikaisesti. CAN-standardissa on määritelty 4 eri viestityyppiä. Viesteillä on omat prioriteetit ja nämä prioriteetit on sisällytetty viestin sovittelukenttään (arbitration field). (CAN Specification Version 2.0 1991.)

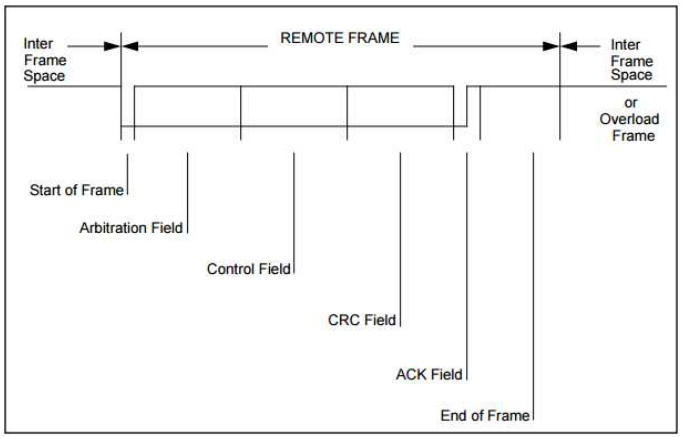
Viesteissä käytetään bittikohtaista väylän kilpavarausmenettelyä, jossa hyödynnetään bittitason AND-operaatiota. Yksinkertaistettuna viesti, jolla on eniten 0 bittejä (dominoivia bittejä) viestin sovittelukentän (arbitration field) MSB-bitteinä, on suurimman prioriteetin omaava viesti. Viestien prioriteetti voidaan havainnollistaa seuraavalla tavalla. Kaksi laitetta, laite A ja laite B, alkaa kirjoittamaan väylään samanaikaisesti. Laite A lähettää viestin A, joka olisi vaikka 0011 ja vastaavasti laite B lähettää viestin B, joka voisi olla esimerkiksi 0001. Molempien laitteiden lähettämissä viesteissä on useampi 0-tila sovittelukentän MSB-bitteinä, mutta lopulta nämä viestit eroavatkin niin, että viestissä A on 1-tila ja viestissä B on 0-tila (3. bitti vasemmalta). Tällöin laite A, jonka viestissä esiintyy nyt 1-tila, huomaakin, että laitteen B lähettämässä viestissä esiintyy 0-tila. Laite A huomioi tämän ja lopettaa lähetyksensä välittömästi, jolloin laite B saa jatkaa lähettämään viestinsä loppuun. 1-tila on siis väistyvä bitti (recessive) ja 0-tila on dominoiva bitti (dominant). CAN-viestin datakehys on esitetty kuvassa 21. (CAN Specification Version 2.0 1991.)



KUVA 21. CAN-viestin datakehys (CAN Specification Version 2.0 1991)

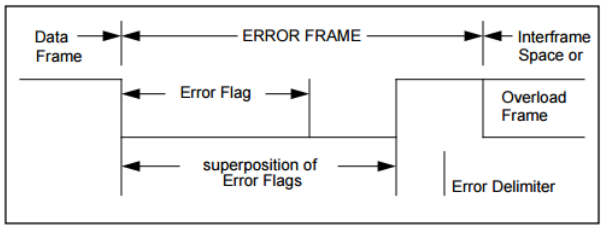
CAN-väylässä käytetään lyhyitä viestejä, jotka voivat maksimissaan olla 94-bittiä pitkiä. Viesteissä ei ole tarkkaan määritelty niiden osoitteita, vaan nämä osoitetiedot määräytyvät viestin sisällön perusteella. Laite, joka on asettunut vastaanottavaksi tietylle datalle,

voi lähettää vastaanotetun datan lähettäjälle Remote Frame -viestin (kuva 22). (CAN Specification Version 2.0 1991).



KUVA 22. Remote Frame (CAN Specification Version 2.0 1991)

Error frame -kehys sisältää kaksi erilaista kenttää. Nämä kentät ovat kerrostettuja. Ensimmäisessä kentässä on yhdistettynä kaikkien laitteiden lähettämät virheliput ja tämän kentän jälkeinen on virheiden erotinmerkki. Kuvassa 23 on esitetty Error frame -kehys. (CAN Specification Version 2.0 1991.)



KUVA 23. Error frame -kehys (CAN Specification Version 2.0 1991)

2.9 3D-mittausalgoritmi

Kun halutaan mitata 3D-ympäristömallia 2D-laserskannerilla, niin pystykallistuskulmat pitää olla tasaisissa suhteissa toisiinsa nähden. Lisäksi pystykulman resoluutioksi tämän työn edetessä valikoitui 0,2 – 0,33 -astetta, koska se on lähes sama, kuin 2D-laserskannerin vaakaresoluutio. Lisäksi se on tarpeeksi stabiili kulman muutos, joka pystyttiin IMU:lla mittaamaan. Tällä resoluutiolla skannaamiseen tarvitaan menetelmää, jonka avulla voidaan olla varma, että nykyiseen skannattuun kulmaan verrattuna seuraava skannattava kulma on juuri 0,2 – 0,33 asteen verran isompi tai pienempi riippuen siitä kumpaan suuntaan skannausta lähdetään tekemään. Työssä päätettiin, että skannaus tehdään

alhaalta ylöspäin. Tämä asia on havainnollistettu kuvassa 24. Skannaukseen käytettiin apuna koululta löytyvää minikaivuria, koska tämä oli helposti siirrettävissä eri kohteisiin maastomallin luonnin yhteydessä. Skannausta tehtiin aluksi myös käsin 2D-laserskanneria ja IMU:a kääntämällä, mutta kaivuri tarjosi vakaamman skannauksen.

A) Alkukulma **B) Loppukulma**

KUVA 24. 2D-laserskannerin kääntö pystysuunnassa kaivurin puomia hyväksi käyttäen

Kuvassa 24 skannauslaser on esitetty punaisella viivalla ja skannerin kääntövarsi on esitettynä mustalla viivalla. Pyöreä musta ympyrä valkoisella täytöllä esittää laserin kääntymispistettä.

Puomi asetetaan kuvan 24 esittämään alkukulmaan ja aletaan mittaamaan 2D-laserskannerin mittaamia vaakapyyhkäisyjä tasavälisillä pystykulmilla puomia ylöspäin kääntäen kohti loppukulmaa. Pystykulmamittausten tasaisuuden varmistamiseksi ohjelmoitiin algoritmi, joka opastaa käyttäjää onnistuneeseen skannaustulokseen. Lopuksi skannausdatasta lasketaan piirrettävät pisteet 2D-tasoon ja piirretään syvyyskuva, joka tallennetaan PNG-tiedostoksi kuvan katselua varten.

3 TULOKSET

Työtä aloittaessa 2D-laserskannerin resoluutio oli 1-aste, joka tarkoittaa, että 270-asteen skannauspyyhkäisyllä saataisiin vain 270 mittaustulosta. Tämä ei ollut tarpeeksi tarkka resoluutio, koska osa skannattavista esineistä ei piirtynyt kunnolla tehtyyn 2D-syvyyskuvaan. Oletettua oli, että tarkkuutta saataisiin paremmaksi, mikäli pyyhkäisyresoluutioksi asetetaan 0,33-astetta. Tällöin saadaan mitattua yhteensä 811 kpl eri etäisyyslukemaa. Tarkkuutta ei käytännössä kuitenkaan aivan näin helposti saatu lisättyä. Alun perin 2D-laserskannerissa oli mediaanisuodin päällä, joka yhdisti 3 vierekkäistä skannaustulosta lopullisen skannauksen keskiarvostamiseksi. Ilman tätä suodinta skannaustuloksiin tuli useita häiriöitä.

Esimerkiksi muutamien pisteiden etäisyyksien mittaustuloksiksi saatiin yli 60 m, vaikka laserskanneri ei datalehden mukaan kyennyt mittaamaan 10 metriä pidemmälle. Näitä harhamittauksia esiintyi jonkin verran. Tämä johti siihen, että tehtiin oma mittaustulosten keskiarvoistus skannaamalla 3 kertaa kaikki 811 pistettä 0,33-asteen resoluutiolla, ja niistä laskettiin keskiarvo. Tämä ratkaisu ei poistanut kuitenkaan alkuperäistä ongelmaa, vaikka yritettiin laskea pisteiden keskiarvot peräti 10 peräkkäisistä skannauksista. Lopuksi päädyttiin siihen ratkaisuun, että yli 10 m harhamittaustulokset nollattiin ja jätettiin huomiotta.

Kuvassa 25 (kuva suurennettuna liitteessä 1) on esitetty mustavalkoisena 3D-ympäristömallinnus 2D-syvyyskuvana huoneiston sisältä, sekä referenssivalokuva skannatusta ympäristöstä. 2D-syvyyskuvassa vaaleammat alueet ovat lähempänä ja tummemmat alueet ovat kauempana. Referenssikuva on otettu eri aikaan kuin laserskannaus, siksi tuolit ovat kuvissa hieman eri asennoissa. Epäonnistuneita skannauksia tehtiin kaiken kaikkiaan yli 50 kpl ennen viimeistä onnistunutta skannausta.

KUVA 25. Laserskannattu ympäristö (vasemmalla) ja referenssi valokuva skannatusta ympäristöstä (oikealla)

4 POHDINTA

Työn tavoitteena oli toteuttaa 3D-maastomalli yhdistämällä 2D-laserskannerin ja IMU-anturin mittaustulokset. 3D-skannaus tehtiin puoliautomaattisesti tarkoittaen sitä, että skanneria ja IMU:a piti kääntää käsin käyttäjää ohjaavan automaattisen mittausalgoritmin avulla. Työn tavoite saavutettiin kiitettävästi.

Työ tehtiin käyttäen C++ -ohjelmointikieltä Windows-ympäristössä. Työssä ohjelmoitiin TCP/IP client- ja server-ohjelmat, sekä USB-CAN-adapteriohjelma käyttäen hyväksi adapterin valmistajan CAN-ohjelmointikirjaston dll-tiedostoja ja funktioita. Työssä piti muuntaa ja tehdä dll-tiedostoista lib-tiedostot. Tämä tehtiin siksi, koska Microsoft Visual Studio 2013, jolla ohjelmoinnit tehtiin, ei kyennyt linkittämään CAN-USB-kirjaston dll-tiedostoja mukaan ohjelmakoodiin ilman näitä puuttuvia lib-tiedostoja. TCP-datavirrasta parsittiin 2D-laserskannerin lähettämät tiedot sopiviin muuttujiin tiedon käsittelyä ja laskentaa varten. Työssä tutustuttiin myös hieman avoimen lähdekoodin libpng-ohjelmointikirjastoon ja ohjelmoitiin sen ympärille omat C++ -kuvankäsittelyluokat, sekä funktiot. Nämä tehtiin siksi, koska haluttiin piirtää tietokoneella 2D-syvyyskuva skannatusta ympäristöstä jälkitarkastelua varten. Vaakaskannaus-mittauksia tehtiin ohjaavan algoritmin avulla 0,3-asteen välein vertikaalisesti vaihtelevalla sektorin koolla. Aluksi skannattiin 90-asteen pystysektori, mutta myöhemmin tämä sektori pienennettiin, koska haluttiin jäljitellä matkapuhelimen kameralla otettua kuvaa. Lopuksi valittiin 50-asteinen pystysektori pystyskannausalueeksi.

Haasteena tässä työssä oli saada tarpeeksi tarkka ja luotettava mittausohjelma, jolla 3D-maastomallinnus voidaan toteuttaa. Haasteena oli myös piirtää 3D-skannatuista vektoreista valokuvan näköinen tietokoneella piirretty kuva. Skannauksia tehtiin yhteensä yli 50 kpl. Ohjelmointiin toi lisähaastetta se, että dll-tiedostot eivät toimineet sellaisenaan, ja niistä piti itse luoda Visual Studio 2013 tarvitsemat lib-tiedostot. Tämä tehtiin siksi, että CAN-USB-adapterin koodi saatiin kääntymään. Haasteena oli myös tallentaa ja pilkkoa tarvittavat luvut muuttujiin 2D-laserskannerin lähettämästä datavirrasta.

Työssä tutustuttiin myös laser-etäisyysmittauksen, kiihtyvyysanturin, gyroskoopin ja magnetometrin teorioihin, sekä niiden käytännön toimintaan. Tämän lisäksi tutustuttiin muutamiin matemaattisiin laskuihin, kuten esimerkiksi kvaternion muunnos Euler-kulmiksi ja laskettiin kahden vektorin ristitulo kolmannen tuntemattoman vektorin määrittämiseksi. Nämä asiat käsiteltiin vasta lopuksi, koska IMU:ssa datan fuusio oli valmiiksi tehtynä, ja tämä anturiyhdistelmä lähetti valmista dataa kvaternio-muodossa. Toisin sanoen kallistuskulmien mittaukset olivat lähtökohtaisesti melkein heti käyttövalmiita tämän työn alusta alkaen.

Työssä testattiin 2D-laserskannerin parasta sijoituspaikkaa kaivuukoneessa. Sen lisäksi testattiin 3D-laserskannausta mahdollisimman optimaalisen kuvan piirtämiseksi kerättyjen datojen avulla. Työssä testattiin myös IMU:n herkkyyttä, ja sen määrittämää minimikallistuskulmaa, jota voidaan mitata luotettavasti.

Parannettavaa tässä työssä voisi olla, että 2D-laserskannerin pystykallistus tehtäisiin erillisen askelmoottorin avulla, koska kaivurin puomin nopeutta ei voida hallita tarpeeksi hyvin. 2D-laserskannerin käyttö 3D-skannaukseen on erittäin hidasta, mikäli halutaan skannata isompaa maastoaluetta. Tässä työssä keskityttiin vain paikallaan yhden alueen skannaukseen. Työtä voitaisiin jatkaa niin, että tehtäisiin Kalman-suotimen avulla maastopaikannus arvio ja laskettaisiin, sekä rakennettaisiin yksittäisistä paikallaan skannatuista datoista isomman alueen 3D-malli.

LÄHTEET

2D laser scanners TiM5xx / TiM56x / Indoor. 2016 .Technical details. Saksa: SICK AG. Luettu 8.3.2016.
https://www.sick.com/de/en/detection-and-ranging-solutions/2d-laser-scanners/tim5xx/tim561-2050101/p/p369446

Amann, M. C., Lescure, T. B. M., Myllylä, R. & Rioux, M. 2000. Laser Pulse Time-of-Flight Distance Measurement. Laser ranging: a critical review of usual techniques for distance measurement. Luettu 7.3.2016.
http://www.utdallas.edu/~aiken/LASERCLASS/laserprinlipels.pdf

Anisotropic Magneto Resistance. 2016. AMR MEMS Sensors. Lab4MEMS. Luettu 15.3.2016.
http://www.lab4mems.upb.ro/technology/amr/

Baker, M. J. 2016. Maths - Conversion Quaternion to Euler. EuclideanSpace - Mathematics and Computing. Luettu 14.3.2016.
http://www.euclideanspace.com/maths/geometry/rotations/conversions/quaternionToEuler/

Barbic, J. 2012.Quaternions and Rotations. CSCI 520 Computer Animation and Simulation. University of Southern California. Luettu 14.3.2016.
http://run.usc.edu/cs520-s12/quaternions/quaternions-cs520.pdf

BECKHOFF Automation CANopen: Mounting and Wiring. 2016. Beckhoff Automation GmbH & Co. KG. Luettu 22.3.2016.
http://infosys.beckhoff.com/english.php?content=../content/1033/cx1500_fb_510/html/cx1500_x510_cable.htm&id=

Bobic, N. 1998. Rotating Objects Using Quaternions. UBM Tech. Luettu 14.3.2016.
http://www.gamasutra.com/view/feature/131686/rotating_objects_using_quaternions.php

CAN Specification Version 2.0. 1991. Bosch GmbH. Luettu 22.3.2016.
http://www.kvaser.com/software/7330130980914/V1/can2spec.pdf

Caruso, M. J., Bratland, T., Smith, C. H. & Schneider, R. 2009. Honeywell Inc & Nonvolatile Electronics Inc. Luettu 15.3.2016.
http://archives.sensorsmag.com/articles/0399/0399_18/main.shtml

Cheung, S. Y. 2016. CS255 Computer Organization & Architecture I. Fixed point numbers. Atlanta: Emory university. Luettu 14.3.2016.
http://www.mathcs.emory.edu/~cheung/Courses/255/Syllabus/5-repr/fixed.html

Cross Product. 2014. MathsIsFun.com. Luettu 24.3.2016.
https://www.mathsisfun.com/algebra/vectors-cross-product.html

Esfandyari, J., De Nuccio, R. & Xu, G. 2010. Introduction to MEMS gyroscopes. STMicroelectronics. Luettu 14.3.2016.
http://electroiq.com/blog/2010/11/introduction-to-mems-gyroscopes/

How to convert packed integer (16.16) fixed-point to float? 2011. Stack Overflow. Luettu 10.2.2016.
http://stackoverflow.com/questions/8638792/how-to-convert-packed-integer-16-16-fixed-point-to-float

Geen, J. & Krakauer, D. 2003. New iMEMS® Angular-Rate-Sensing Gyroscope. Micromachined Products Division. Analog Devices. Luettu 14.3.2016.
http://www.analog.com/library/analogDialogue/archives/37-03/gyro.html

Hargrave, V. 2013. TCP/IP Network Programming Design Patterns in C++. Luettu 10.3.2016. http://vichargrave.com/network-programming-design-patterns-in-c/

Kozierok, C. M. 2015. The TCP/IP Guide. Luettu 10.3.2016.
http://www.tcpipguide.com/free/t_TheOpenSystemInterconnectionOSIReferenceModel.htm

Laserluokat. 2015. Säteilyturvakeskus. Luettu 8.3.2016.
http://www.stuk.fi/aiheet/laserit/laserluokat

Magneto resistor. 2016. Resistor Guide. Luettu 15.3.2016.
http://www.resistorguide.com/magneto-resistor/

Mason, H. 2003. Basic Introduction to the use of Magnetoresistive Sensors. Application Note 37. Zetex Semiconductors. Luettu 15.3.2016.
http://www.diodes.com/_files/product_app_note_pdfs/zetex/an37.pdf

Mitchell, B. 2015. The Seven Layers of the OSI Model Illustrated. Luettu 10.3.2016.
http://compnetworking.about.com/od/osimodel/tp/The-Seven-Layers-of-the-OSI-Model-Illustrated.htm

Reflectivity Of Various Surfaces / Materials. 2016. Application Note AN-G1004. RIEGL Laser Measurement Systems GmbH. Luettu 8.3.2016.
http://www.riegl.com/uploads/tx_pxpriegldownloads/General-Information-Distance-meter.pdf

TiM5xx käyttöohje. 2015. Operating Instructions. Saksa: SICK AG. Luettu 7.3.2016.
https://www.mysick.com/saqqara/im0053143.pdf

TiM5xx manuaali. 2015. Technical Information. Saksa: SICK AG. Luettu 7.3.2016.
https://www.sick.com/media/dox/3/33/133/Technical_information_TiM55x_TiM56x_TiM57x_Ranging_Laser_Scanner_en_IM0053133.PDF

Understanding Acceleration and Choosing an Accelerometer. 2013. White Papers. National Instruments Corporation. Luettu 11.3.2016.
http://www.ni.com/white-paper/3807/en/

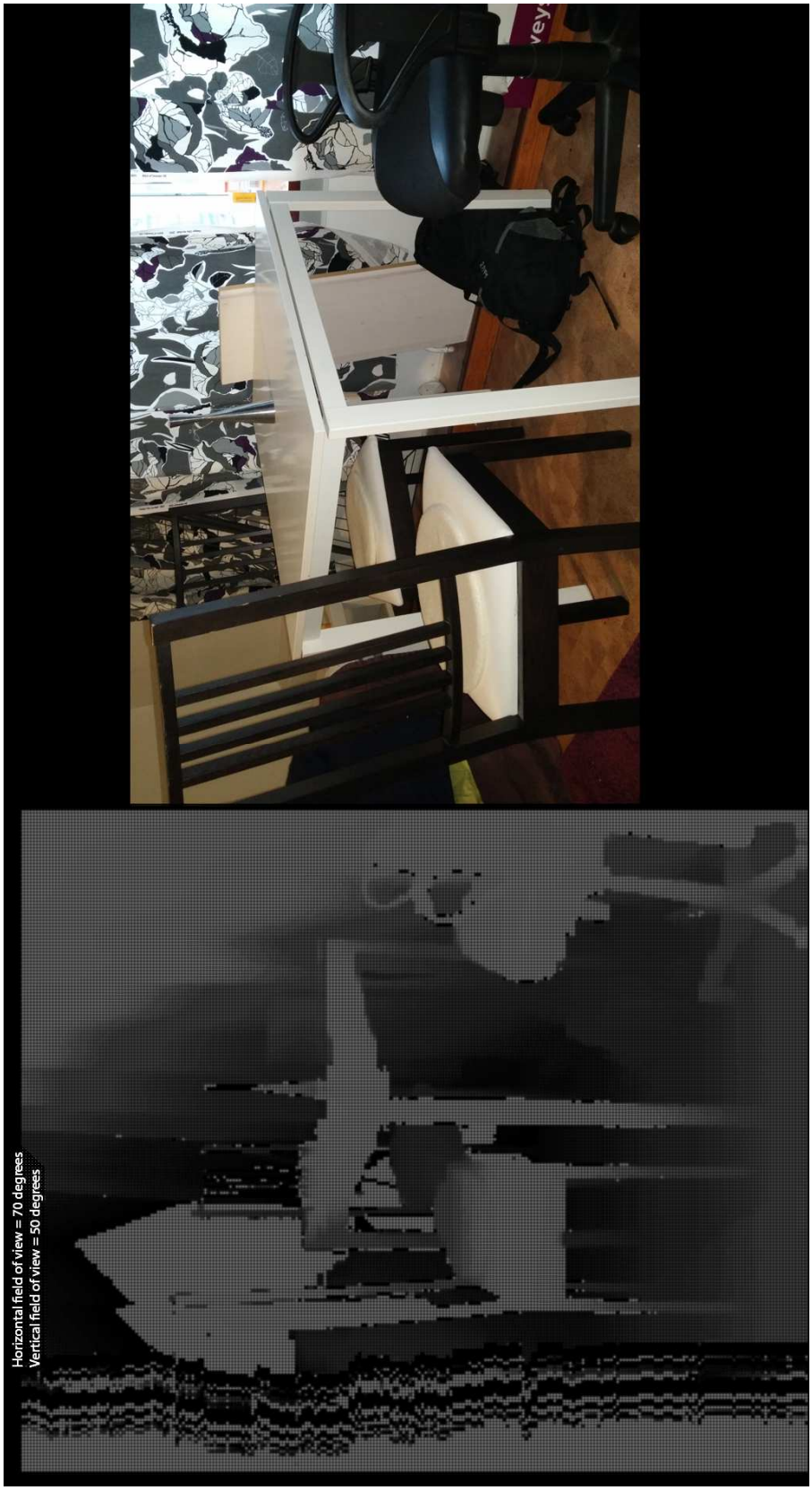
Understanding Euler Angles. 2016. CHRobotics LLC. Luettu 11.3.2016.
http://www.chrobotics.com/library/understanding-euler-angles

Van de Maele, P-J. 2013. Reading a IMU Without Kalman: The Complementary Filter. Luettu 25.3.2016.
http://www.pieter-jan.com/node/11

Van Oosten, J. 2012. Understanding Quaternions. 3D Game Engine Programming. Luettu 14.3.2016.
http://www.3dgep.com/understanding-quaternions/

What is Permalloy?. 2016. WiseGEEK. Conjecture Corporation. Luettu 15.3.2016.
http://www.wisegeek.com/what-is-permalloy.htm

Winer, K. 2016. Affordable 9 DoF Sensor Fusion. Github.com. Luettu 22.3.2016.
https://github.com/kriswiner/MPU-6050/wiki/Affordable-9-DoF-Sensor-Fusion

LIITTEET

Liite 1. Skannattu kuva suurennettuna