TARTU ÜLIKOOL
LOODUS- JA TEHNOLOOGIATEADUSKOND
TEHNOLOOGIAINSTITUUT

Mark Laane

TARKVARA LOOMINE KEHAASENDI JÄLGIMISE SÜSTEEMILE

Bakalaureusetöö (12 EAP)

Juhendaja:
B.Sc. Artur Abels

Tartu 2015

Sisukord

Sissejuhatus **4**

1. Metoodika kirjeldus **5**

1.1. Kehaasendi jälgimise põhimõte 5

1.2. Ülevaade suuna jälgimisest 5

1.2.1. Kasutatavad sensorid 6

1.2.2. Suuna esitamise viisid 6

1.2.3. Sebastian Madgwicki suuna filter 8

2. Riistvaraline lahendus **10**

2.1. Süsteemi struktuur 10

2.2. Klientseade 11

2.3. Koordinaatorseade 12

2.4. Seadmete alamosad 13

2.4.1. Inertsiaalandur ST Microelectronics LSM9DS0 13

2.4.2. Raadiomoodul Nordic semiconductor nRF24L01 13

3. Tarkvara **15**

3.1. Tarkvara arvutis 15

3.1.1. HardwareManager 16

3.1.2. SerialManager 17

3.1.3. SkeletonScene 18

3.1.4. ImuPlotter 20

3.2. Seadmete püsivara 20

3.2.1. Klientseadme püsivara 20

3.2.2. Koordinaatorseadme püsivara 22

4. Magnetvälja moonutuste korrigeerimine **23**

4.1. Moonutused 23

4.2. Korrigeerimise metoodika 24

4.3. Korrigeerimise tarkvara 25

Kokkuvõte **27**

Summary 28

Allikad 30

Sissejuhatus

Virtuaalreaalsus on tehnoloogia, mis võimaldab selle kasutajal kogeda ning mõjutada arvuti poolt simuleeritud tehiskeskkondi. Selle jaoks loodud süsteemide eesmärk on viia tehismaailm kasutajani võimalikult tõetruult ning pakkuda võimalusi selle simulatsiooniga interakteerumiseks. Kasutaja kogeb viibimist selles maailmas ning parimal juhul unustab pärismaailma olemasolu. [1] Sellise kogemuse saamiseks on loodud mitmeid erinevaid seadmeid. Spetsiaalsed peakomplektid [2], kindad [3] ja jooksurajad [4] võimaldavad kasutajal tehisruumi tajuda ning seda ka oma tegevustega mõjutada. Oculus Rift on peakomplekt, mis võimaldab virtuaalses ruumis ringi vaadata. Virtuaalne ruum kuvatakse stereoskoopilise pildina, mis muutub vastavalt kasutaja peaasendile. Oculus Rift on hetkel veel arendusjärgus, kuid selle prototüüpi on võimalik kõigil osta. Tavakasutajale mõeldud lõppversiooni lubatakse 2016. aasta esimeseks pooleks. [2]

Käesoleva töö ajendiks on soov muuta Oculus Riftiga kogetu veelgi tõetruumaks, lisades peaasendi jälgimisele ka kehaasendi jälgimise. See annab võimaluse näha enda keha liikumist virtuaalreaalsuses ning kasutada loomulikke liigutusi virtuaalmaailmas liikumiseks ning selle mõjutamiseks. Käesoleva bakalaureusetöö eesmärgiks on luua tarkvara juhtmevabale kehaasendi jälgimise süsteemile. Töö tulemusena valmib süsteem, mis suudab kujutada kasutaja kehaasendit arvutis oleval inimkeha mudelil. Süsteem koosneb moodulitest, mida kasutaja saab kinnitada jälgitavate kehaosade külge ning ühest koordinaatorseadmest, mis ühendub arvuti külge. Keha küljes olevad moodulid jälgivad pidevalt oma suunda ruumis kasutades selleks kiirendusandurit, güroskoopi ja magnetomeetrit. Moodulid edastavad oma suuna läbi koordinaatorseadme arvutile ning arvuti kuvab kasutaja asendit inimkeha mudelil reaalajas.

Bakalaureusetöö koosneb neljast osast. Esimeses osas kirjeldatakse keha asendi ja kehaosade suuna jälgimise viisi ning kasutatavaid sensoreid. Lisaks antakse ülevaade suuna jälgimisel kasutatavast algoritmist ning matemaatilisest taustast. Teises osas antakse ülevaade loodud süsteemi ülesehitusest ja kasutatavast riistvarast. Selles osas kirjeldatakse süsteemis kasutatavaid seadmeid ning kirjeldatakse ka nende ülesehitust ja parameetreid. Kolmandas osas kirjeldatakse süsteemi jaoks loodud tarkvara. Selles osas vaadeldakse nii arvutile loodud tarkvara, mille abil visualiseeritakse ja kontrollitakse seadmete tööd kui ka seadmetele loodud püsivara. Neljandas osas kirjeldatakse magnetvälja moonutuste allikaid ja nende korrigeerimise võimalusi. Lisaks kirjeldatakse antud süsteemis kasutatud korrigeerimist ning selle jaoks loodud tarkvara.

1. Metoodika kirjeldus

1.1. Kehaasendi jälgimise põhimõte

Iga kehaosa asend on täielikult määratud, kui on teada selle asukoht ning suund. Kehaosa asukoha määrab ära kehaosa, mille külge see kinnitub. Seega piisab kogu keha asendi kindlakstegemiseks vaid kehaosade suundade jälgimisest. Antud töö puhul on välja valitud 11 tähtsamat kehaosa, mille suunda jälgitakse: rindkere, õlavarred, küünarvarred, labakäed, reied ja sääred.

Töö käigus valmis süsteem, mis jälgib valitud kehaosade suundasid ning kuvab arvuti ekraanil keha mudelit, mille kehaosad on samas asendis, kui süsteemi kasutajal. Kehaosade suundade jälgimiseks kinnitatakse kasutaja iga jälgitava kehaosa külge spetsiaalne juhtmevaba moodul. Igas moodulis on inertsiaal- ja magnetsensorid, mille abil jälgitakse mooduli ning seega ka vastava kehaosa suunda.

Arvutis kuvatav keha mudel koosneb määratud pikkusega osadest, mis kinnituvad üksteise külge. 11 valitud kehaosa on võimalik pöörata vabalt valitud suundadesse, pöörates selleks vastavat moodulit. Keha keskpunkt on fikseeritud koordinaatide algpunkti. Iga kehaosa suund on määratud ainult vastava mooduli suuna järgi. Kui kõik moodulid on kinnitatud kasutaja vastavate kehaosade külge, jäljendab arvutis kuvatav keha mudel kasutaja kehaasendit.

Kirjeldatud viisil keha asendi jälgimine ei garanteeri jälgitavate kehaosade asukoha täpset jäljendamist. Näiteks kui keha mudeli küünarluud on märkimisväärselt pikemad kui kasutaja omad, siis käte kokkupanemisel liiguvad mudelis käed üksteisest läbi. Selleks, et ka kehaosade asukohad ruumis võimalikult hästi kattuksid, peavad mudeli kehaosad olema samades proportsioonides moodulite kandja kehaosadega.

1.2. Ülevaade suuna jälgimisest

Ühe mooduli suunda hinnatakse kolme anduri abil. Güroskoop, kiirendusandur ja magnetomeeter mõõdavad vastavalt mooduli pöörlemist, sellele mõjuvat kiirendust ning magnetvälja mooduli ümber. Anduritest saadud andmeid kasutatakse Sebastian Madgwicki suunafiltris, mille väljundiks on mooduli suund kvaterniooni kujul (täpsemalt töö alapeatükis 1.2.2 ja 1.2.3).

1.2.1. Kasutatavad sensorid

Güroskoop on andur, millega on võimalik mõõta pöörlemise kiirust ja suunda. Integreerides saadud tulemust üle aja, saame tulemuseks selle aja jooksul tehtud pöördenurga. Kui anduri algne suund on teada, saab pöörlemiskiiruse integreerimise teel arvutada sensori uue suuna. Kirjeldatud suuna mõõtmise viisi puuduseks on aga ajas akumuleeruv viga. Arvutatud suuna väärtus muutub integreerimisel üha ebatäpsemaks. Seetõttu pole võimalik kasutada güroskoopi suuna jälgimisel ainsa sensorina. [5]

Kiirendusandur ja magnetomeeter on andurid, mis võimaldavad mõõta vastavalt kiirendust ja magnetvälja. Kui sensorid seisavad paigal ja muud magnetväljad puuduvad, saab nendega mõõta Maa gravitatsiooni ja magnetvälja. Gravitatsioonivektor on suunatud Maa keskpunkti poole ja maa magnetvälja vektor Maa magneetilise lõunapooluse suunas. Nii moodustavad need kaks vektorit taustsüsteemi, mille suhtes saab määrata seadme suuna. Süsteemi liigutamisel liitub aga kiirendusanduri väljundile liigutamisest põhjustatud kiirendus, mis muudab võimatuks gravitatsioonivektori suuna kindlakstegemise. [5] Seega sobib kiirendusanduri ja magnetomeetri kasutamine paigaloleva seadme suuna hindamiseks, kuid ei ole piisav liikuva seadme asendi hindamiseks.

Töös kasutatav Sebastian Madgwicki suunafiltri algoritm kasutab kõigi kolme sensori tulemusi, et hinnata liikuva seadme suunda. Selleks kasutatakse kahes eelnevas lõigus kirjeldatud suuna jälgimise meetodi kooslust. Lõpliku suunahinnangu andmisel arvestatakse kummagi meetodi abil saadud hinnangut. Seetõttu võimaldab Sebastian Madgwicki algoritm hinnata liikuva seadme suunda ka pikema aja jooksul. [5]

1.2.2. Suuna esitamise viisid

Koordinaatteljestike või kehade suunda kolmemõõtmelises ruumis saab esitada mitmel viisil. Üheks võimaluseks on kasutada Euleri nurkasid, kus jälgitava keha suund määratakse ära kolme nurga abil. Teisendus ühest koordinaatteljestikust teise jagatakse kolmeks alampöördeks ümber koordinaattelgede. Euleri nurkade abil esitatud suunda on intuitiivselt lihtne mõista, kuid sellel esitusviisil on mitmeid matemaatilisi probleeme. Esiteks leidub suunasid, kus kasutatavates valemitesse tekivad singulaarsused. Teiseks, väikeste suunamuutuste integreerimisel on Euleri nurgad ebatäpsemad kui kvaternioonid. [6]

Teine võimalus on kasutada suuna esitamiseks kvaternioone, millel puuduvad Euleri nurkadele omased probleemid. Käesolevas töös kasutatav suunafiltri algoritm kasutab ja annab tulemuse just kvaterniooni kujul. Kvaternioon on matemaatiline konstruktsioon, mida saab esitada kui nelikut

$$\boldsymbol{q} = [q_0 \;\; q_1 \;\; q_2 \;\; q_3],$$

kus q_0 q_1 q_2 ja q_3 on reaalarvud [7]. Kvaterniooni kaaskompleks $\overline{\boldsymbol{q}}$, moodul $\|\boldsymbol{q}\|$ ja pöördkvaternioon $\boldsymbol{q}^{-1}$ on defineeritud järgnevalt:

$$\overline{\boldsymbol{q}} = [q_0 \; -q_1 \;\; -q_2 \;\; -q_3]$$

$$\|\boldsymbol{q}\| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2}$$

$$\boldsymbol{q}^{-1} = \overline{\boldsymbol{q}}.$$

Keha suuna esitamiseks kasutatakse ühikkvaternioone. Need on kvaternioonid, mille moodul on 1. Kui ühikkvaterniooniga on esitatud pööret taustkoordinaatsüsteemist keha koordinaatsüsteemi, siis selle pöördkvaternioon esitab vastupidist pööret – keha koordinaatsüsteemist taustkoordinaatsüsteemi. [6]

Kvaternioonide kujul esitatud pöördeid saab oma vahel kombineerida kvaternioonkorrutise teel. Kvaternioonkorrutist märgitakse tähisega $\otimes$. Näiteks pöörde $\boldsymbol{q}_A$ ja $\boldsymbol{q}_B$ kombineerimisel saame liitpöörde $\boldsymbol{q}_{AB}$, mis kujutab endas kahe pöörde järjest rakendamist, mida võib defineerida järgnevalt:

$$\boldsymbol{q}_{AB} = \boldsymbol{q}_A \otimes \boldsymbol{q}_B.$$

Kvaternioonkorrutis ei ole kommutatiivne $(\boldsymbol{q}_A \otimes \boldsymbol{q}_B \neq \boldsymbol{q}_B \otimes \boldsymbol{q}_A)$, kinnitades fakti, et kahe pöörde erinevas järjekorras rakendamine ei anna sama tulemust $(\boldsymbol{q}_{AB} \neq \boldsymbol{q}_{BA})$. [5]

Kvaternioontuletis $\dot{\boldsymbol{q}}$ kirjeldab suuna muutumise kiirust ning on seotud nurkkiirusega järgnevalt:

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix},$$

kus $\boldsymbol{q}$ on suund ning ω_x, ω_y ja ω_z nurkkiirused lokaalse koordinaatsüsteemi kolme telje ümber. [6]

1.2.3. Sebastian Madgwicki suuna filter

Sebastian Madgwicki filter kasutab suuna määramiseks kiirendusanduri, güroskoobi ja magnetomeetri andmeid. Filtri väljundiks on seadme suund kvaterniooni kujul. Algoritm kasutab kahte erinevat meetodit suuna määramiseks ning kombineerib saadud tulemusi lõpliku tulemuse saamiseks. Järgnevalt antakse ülevaade kummastki meetodist. Seejärel kirjeldatakse kummagi meetodi tulemuste kombineerimist suuna hindamisel.

1.2.3.1. Suund pöörlemiskiirusest

Güroskoobi abil mõõdetakse seadme pöörlemist ümber kolme sensori telje – x, y ja z. Tulemuseks on pöörlemiskiirused vastavalt ω_x , ω_y ja ω_z . Kvaternioontuletist $\dot{\boldsymbol{q}}$, mis kirjeldab maa koordinaatsüsteemi pöörlemise kiirust seadme koordinaatsüsteemi suhtes, saab avaldada järgnevalt:

$$s_\omega = \begin{bmatrix} 0 \\ \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}$$

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \otimes s_\omega.$$

Kui algne suund $\boldsymbol{q}_{t-1}$ on teada, saab kvaterniooni tuletist $\dot{\boldsymbol{q}}_t$ ajahetkel t kasutada suuna leidmiseks ajahetkel t. Selleks tuleb numbriliselt integreerida kvaterniooni tuletist läbi aja nagu on näidatud järgnevas valemis:

$$\boldsymbol{q}_t = \boldsymbol{q}_{t-1} + \dot{\boldsymbol{q}}_t \Delta t.$$

Selles valemis tähistab $\boldsymbol{q}_{t-1}$ eelmist asendihinnangut, $\boldsymbol{q}_t$ uut asendihinnangut ning Δt diskreetimisperioodi. [5]

1.2.3.2. Asend magnetvälja ja kiirendusvektorist

Maa magnetväli ja gravitatsioon moodustavad kaks vektorit, mille suund ja tugevus on Maa koordinaatsüsteemis teada. Mõõtes seadmele mõjuvat gravitatsiooni ja magnetvälja on võimalik määrata üheselt seadme koordinaatsüsteemi asend Maa koordinaatsüsteemi suhtes. [5]

Kiirendusandur mõõdab maa gravitatsioonivälja poolt tekitatud raskuskiirendust ning seadme liigutamisest põhjustatud kiirenduse tugevust ning suunda. Magnetomeeter mõõdab seda ümbritseva magnetvälja tugevust ning suunda. See hõlmab endas nii maa magnetvälja kui ka

lähedal asuvate esemete magnetväljasid ning kohalikke magnetvälja moonutusi. Asendi määramisel eeldatakse, et magnetomeeter ja kiirendusandur mõõdavad ainult maa magnetvälja ja gravitatsiooni. [5]

Asendi leidmiseks lahendatakse optimiseerimisprobleem, kus eesmärgiks on leida asend, mille puhul eeldatavad gravitatsiooni ja magnetvälja vektorid kattuvad võimalikult hästi sensorite abil mõõdetud gravitatsiooni ja magnetvälja vektoritega. Selleks minimiseeritakse funktsiooni, mis kirjeldab mõõdetud ja eeldatud vektorite erinevust. Iga uue mõõtetulemuse saamisel liigutakse otsinguruumis konstantse sammu võrra minimiseeritava funktsiooni gradiendile vastupidises suunas, lokaalse miinimumi poole. [5]

1.2.3.3. Kahe meetodi kombineerimine

Seadme koordinaatsüsteemi asendi hinnang maa koordinaatsüsteemi suhtes arvutatakse kahe eelneva meetodi tulemuste kombineerimisel. Kummalegi tulemusele määratakse kaal. Asendihinnang on määratud järgneva valemiga, kus $\boldsymbol{q}_t$ on lõplik asendi hinnang, $\boldsymbol{q}_{\nabla,t}$ magnetvälja ja kiirendusvektorist saadud asendihinnang, $\boldsymbol{q}_{\omega,t}$ pöörlemiskiirusest saadud asendihinnang ning γ on kaal:

$$\boldsymbol{q}_t = \gamma \boldsymbol{q}_{\nabla,t} + (1-\gamma)\boldsymbol{q}_{\omega,t}\,, \qquad 0 \leq \gamma \leq 1$$

1.2.3.4. Filtri parameetrite määramine

Filtri rakendamisel tuleb seadistada kaks parameetrit βja ϑ, millest kumbi kirjeldab erinevat tüüpi güroskoobi vigasid. β kirjeldab vigasid, mille keskväärtus on 0. Sellisteks on näiteks kvantiseerimisviga ning sensori müra. ϑ kirjeldab vigu, mille keskväärtus ei ole null ja mis põhjustavad güroskoobi nullväärtuse kõrvalekallet. Nende väärtused tuleb määrata vastavalt järgnevatele valemitele, kus $\widetilde{\omega}_\beta$ on iga telje hinnanguline vigade suurus, mille keskväärtus on 0 ja $\widetilde{\omega}_\vartheta$ *on* iga telje hinnanguline nullväärtuse kõrvalekalle:

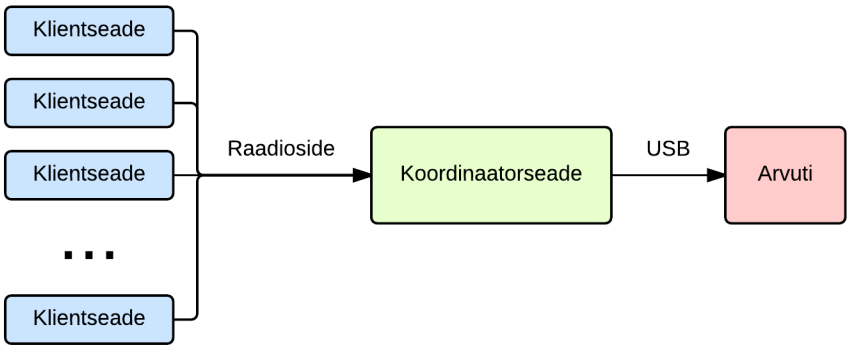
$$\beta = \sqrt{\frac{3}{4}\widetilde{\omega}_\beta}$$

$$\vartheta = \sqrt{\frac{3}{4}\widetilde{\omega}_\vartheta}$$

2. Riistvaraline lahendus

2.1. Süsteemi struktuur

Süsteem koosneb arvutist, selle külge ühendatud koordinaatorseadmest ning juhtmevabadest klientseadmetest, mis kinnitatakse kasutaja keha külge. Süsteemi abil jälgitakse 11 kehaosa suunda. Selleks kasutatakse 8 klientseadet. Klientseadmeid on kahte tüüp: 6 on ühe andurikomplektiga ning 2 kahe andurikomplektiga (Joonis 2.2). Rindkere, õlavarte, reite ja säärte klientseadmed sisaldavad endas ühte andurikomplekti. Kummagi käe jaoks on üks seade, mis koosneb kahest osast. Selle klientseadme kummaski pooles on üks andurikomplekt. Üks kinnitub küünarvarre ja teine labakäe külge. Joonis 2.1 kujutab süsteemi ülesehitust ning suunainfo liikumist läbi süsteemi. Klientseadmed saadavad andmed oma suuna kohta raadioside teel koordinaatorseadmele. Koordinaatorseade edastab need üle USB siini arvutile.



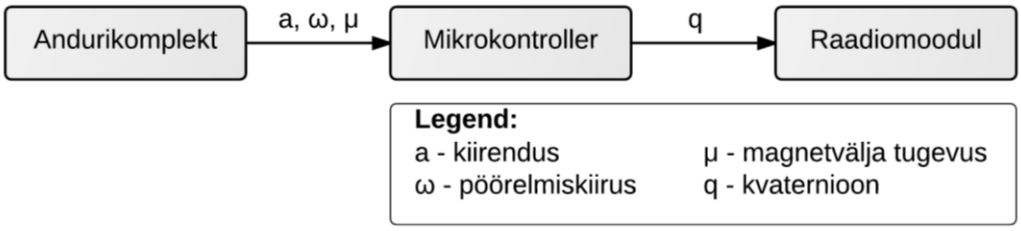
Joonis 2.1 Süsteemi ülesehitus ja andmete liikumine läbi selle.

Joonis 2.2 Kahest osast koosnev klientseade küünarvarre ja labakäe suuna jälgimiseks.

2.2. Klientseade

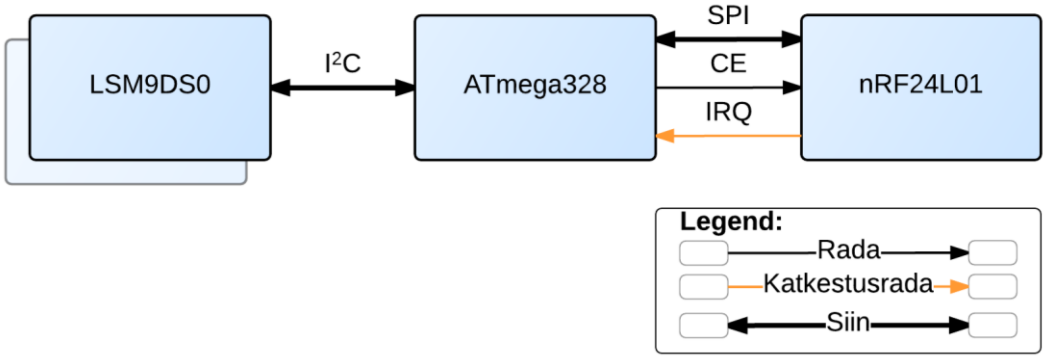
Klientseade on iseseisev seade, mis jälgib pidevalt oma suunda ruumis. Kasutaja mugavuse eesmärgil puuduvad keha külge kinnitatavatel moodulitel välised juhtmed. Iga moodul omab oma energiaallikat ning andmeid oma suuna või mõõtetulemuste kohta edastab see koordinaatorseadmele raadioside teel. Väliste juhtmete puudumine muudab lihtsamaks kogu süsteemi kasutaja külge kinnitamise ning lubab kasutajal piiramatult liikuda.

Klientseade koosneb ühest või kahest andurikomplektist, ühest mikrokontrollerist, raadiomoodulist ja energiaallikast. Joonisel 2.3 on kujutatud klientseadme kolm tähtsamat alamosa ning andmete liikumist läbi nende. Andurikomplekt sisaldab endas güroskoopi, kiirendusandurit ja magnetomeetrit. Nende sensoritega mõõdetakse kolmes teljes vastavalt seadme pöörlemiskiirust, seadmele mõjuvat kiirendust ning ümbritseva magnetvälja tugevust. Mikrokontrolleris arvutatakse saadud mõõtetulemustest klientseadme suund kaustades Sebastian Madgwicki suunafiltri algoritmi. Kvaterniooni kujul olev suund edastatakse raadiomooduli abil koordinaatorseadmele.



Joonis 2.3 Andmete liikumine klientseadmes

Ülevaade klientseadmes kasutatavatest mikrokiipidest ning andmevahetuseks kasutatavatest siinidest on antud joonisel 2.4. Kogu klientseadme tööd koordineerib Arduino Pro Mini arendusmoodulil asuv ATmega328 mikrokontroller, mis töötab taktsagedusel 8Mhz. Sensoreid sisaldav süsteemikiip (ingl. k. *system on a chip*) LSM9DS0 koos selle tööks vajalike väliste

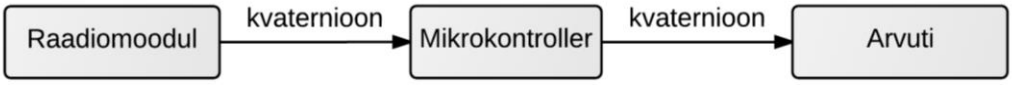


Joonis 2.4 Siinid ja liinid klientseadmes

komponentidega asub firma SparkFun poolt loodud moodulil. LSM9DS0 sisaldab endas kolmeteljelist güroskoopi, kiirendusandurit ning magnetomeetrit. [8] Andmevahetus andurite süsteemikiibi LSM9DS0 ja mikrokontrolleri ATmega328 vahel toimub üle I^2C (ingl. k *Inter-Integrated Circuit*) siini. Raadioühenduse loomiseks koordinaatorseadmega kasutatakse nRF24L01 raadiomoodulit. Raadiomooduli ja ATmega328 vaheline andmevahetus toimub üle SPI (ingl. k. *Serial Peripheral Interface*) siini. Lisaks on mikrokontrolleri ja raadiomooduli vahel veel CE (ingl. k. *Chip Enable*) liin, millega saab mikrokontroller juhtida raadiomooduli tööd ning katkestusliin, mille kaudu saab raadiomoodul genereerida katkestuse mikrokontrolleril. Kogu süsteemi varustab energiaga 400mAh liitiumpolümeeraku.

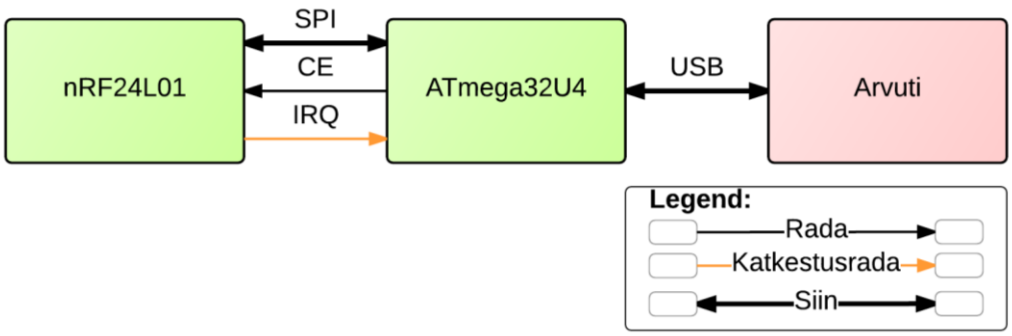
2.3. Koordinaatorseade

Koordinaatorseadme tööks on klientseadmetes oleva info edastamine arvutile. Selleks saadab koordinaatorseade perioodiliselt päringuid klientseadmetele ning edastab saadud vastused arvutile. Joonisel 2.5 on kujutatud koordinaatorseadme kahte tähtsamat alamosa ja arvutit ning andmete liikumist arvutisse läbi nende. Raadiomooduli abil võtab mikrokontroller vastu klientseadmete poolt saadetud suunaandmed ning edastab need arvutile üle USB liidese. Peamiseks edastatavaks infoks on klientseadmete suunad kvaternioonide kujul. Samas on võimalik edastada arvutisse ka näiteks sensorite toorandmeid.



Joonis 2.5 Andmete liikumine arvutisse läbi koordinaatorseadme alamosade.

Ülevaade koordinaatorseadmes kasutatavatest mikrokiipidest ning andmevahetuseks kasutatavatest siinidest on antud joonisel 2.6 . Lisaks on joonisel näha ühendus arvutiga. Kogu koordinaatorseadme tööd juhib SparkFun Pro Micro arendusmoodulil olev ATmega32u4 mikrokontroller, mis töötab taktsagedusel 8Mhz. Sellel mikrokontrolleril on USB liides, mis võimaldab seda lihtsasti ühendada arvutiga [9]. Ühendus raadiomooduliga on identne klientseadmes kasutatud ühendusega. Koordinaatorseade saab toitepinge USB siini kaudu arvutist.



Joonis 2.6 Koordinaatorseadme ülesehitus ja ühendus arvutiga

2.4. Seadmete alamosad

2.4.1. Inertsiaalandur ST Microelectronics LSM9DS0

LSM9DS0 on mikrokiipmoodul, mis sisaldab endas kolmeteljelist kiirendusandurit, magnetväljasensorit ja güroskoopi. Mooduliga on võimalik suhelda nii I2C kui ka SPI liidese kaudu. Mooduli igal sensoril on võimalik valida mõõtevahemikku. Kiirendusanduri mõõtevahemik võib olla ±2, ±4, ±6, ±8, või ±16g, magnetomeetril ±2, ±4, ±8 või ±12 gaussi ning güroskoobil ±245, ±500 või ±2000 kraadi sekundis. [8]

Iga sensori iga kanali väljundiks on 16-bitine täisarv. Seadme kasutamisel peavad mõõdetavad väärtused jääma alati valitud mõõtevahemikku. Vastasel juhul sensori väljund küllastub ning ei vasta mõõdetavale väärtusele. Samas on kasulik hoida mõõtevahemik võimalikult kitsas, et oleks võimalik mõõta ka väiksemaid muutusi. Käesolevas töös valiti kiirendusanduri mõõtevahemikuks ±8g. Selleks simuleeriti seadmega tehtavaid järsemaid liigutusi ning valiti võimalikult kitsas mõõtevahemik, kus sensori väljund ei küllastunud. Samamoodi valiti güroskoobi mõõtevahemik, kuid seekord simuleeriti seadmega tehtavaid järsemaid pöördeid. Maa pinnal oleva Maa magnetvälja tugevus jääb 0.22 ja 0.67 gaussi vahele [10]. Seetõttu valiti magnetomeetri minimaalne mõõtevahemik ±2 gaussi.

2.4.2. Raadiomoodul Nordic semiconductor nRF24L01

nRF24L01 on madala energiatarbega raadiomoodul, mis kasutab andmevahetuseks litsentseerimata 2.4Ghz raadiosagedusriba (ISM – *Industrial, Scientific and Medical Band*). Moodulil on mikrokontrolleriga suhtlemiseks SPI liides, mille kaudu toimub nii andmevahetus kui

ka mooduli seadistamine. Moodul toetab andmevahetuskiiruseid 250kbps, 1Mbps ja 2Mbps. Antud töös kasutatakse andmevahetuskiirust 1Mbps, kuna sel juhul on vastuvõtja tundlikkus 3db parem, kui 2Mbps puhul, andes süsteemi kasutajal suurema liikumisvabaduse. [11]

nRF24L01 raadiomoodul kasutab ShockBurst paketiformaati, mille abil on võimalik saata kuni 32 baiti kasulikku infot. Üks raadiopakett koosneb preambulist, vastuvõtva raadio aadressist, kasulikust infost ning tsükkelkoodkontrollist (CRC - cyclic redundancy check). Saatmisel paneb raadiomoodul kokku paketi ning edastab selle raadiosaatja abil. Vastuvõtmisel jälgitakse raadiokanalit ning korrektse paketi leidmisel asetatakse selle sisu RX puhvrisse, kui sihtaadress kattub raadiomooduli aadressiga. Saadetavat ja vastuvõetavat infot hoitakse kolmes 32 baidises vastuvõtu (RX - receive) ja kolmes saatmise (TX – transmit) puhvris. See lubab raadiomoodulil vastu võtta pakette ka siis, kui mikrokontroller ei ole veel eelmist välja lugenud. Saatmisel võimaldab see mikrokontrolleril panna ootele paketi, kui eelmine ei ole veel saadetud. [11]

Raadiomoodulil on katkestusnõudeliin, mis aktiviseerub kui mõni kolmest sisemisest katkestusallikast on aktiivne. Aktiviseerumisel tõmmatakse katkestusliin madalaks. Katkestusallikateks on paketi edukas saatmine, vastuvõtmine ning kordussaatmiste piirarvu ületamine. Kõik katkestusallikad on eraldi maskeeritavad. [11]

3. Tarkvara

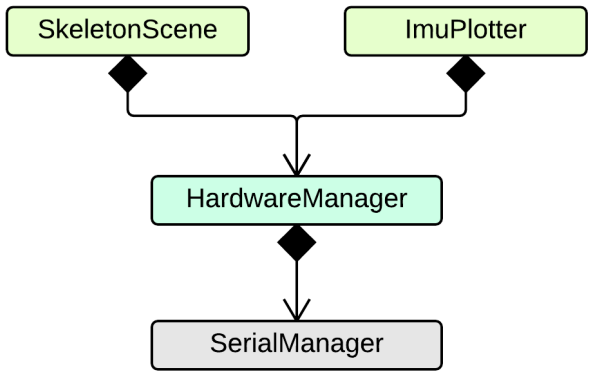
Loodud tarkvara saab jaotada kaheks: arvutis olev tarkvara ja seadmete mikrokontrollerites olev püsivara. Arvutis olev tarkvara on kirjutatud programmeerimiskeeles Python. See programmeerimiskeel valiti oma lihtsuse ja paindlikkuse pärast, mis võimaldab kiirelt luua suuremaid süsteeme. Seadmete püsivara on kirjutatud programmeerimiskeeles C++.

3.1. Tarkvara arvutis

Arvutis olev tarkvara hõlmab endas kolme programmi. Esimene neist täidab töö põhieesmärki - programm kuvab ekraanil inimkeha mudelit, mille 11 valitud kehaosa järgivad reaalajas inimkeha külge kinnitatud klientseadmete asendeid. Teine programm võimaldab jälgida sensorite andmeid graafikutel, et veenduda sensorite korrektses töös ja kalibratsioonis. Graafikutelt on võimalik jälgida nii sensoritest tulevaid toorandmeid kui ka asendifiltri sisendiks olevaid kalibreeritud andmeid. See aitab tuvastada ja siluda riistvara, tarkvara ja kalibreerimise vigu. Kolmas programm võimaldab kalibreerida klientseadmete magnetomeetreid. Kalibreerimistarkvara kirjeldatakse peatükis 4.3.

Python toetab objektorienteeritud programmeerimise paradigmat. See võimaldab kapseldada koodi klassidesse, mille objektid suhtlevad omavahel vaid avalikke meetodeid kasutades. Järgnevalt on kirjeldatud loodud tarkvara tähtsamaid klasse ning nende koostööd erinevates programmides.

Süsteemi keskseks klassiks on HardwareManager, mida teised klassid saavad kasutada riistvaraga suhtlemiseks (Joonis 3.1). Klassil HardwareManager on avalikud meetodid, läbi mille saab kasutada riistvara kogu funktsionaalsust. HardwareManager klass omakorda kasutab klassi SerialManager jadaühenduse haldamiseks. SkeletonScene klass võimaldab kujutada inimkeha erinevates asendites. Selle klassi isendi loomisel kuvatakse aken, kus on inimkeha kolmemõõtmeline mudel mille kehaosi saab liigutada soovitud asenditesse. SkeletonScene kasutab HardwareManager klassi et saada reaalajas asendiinfot seadmetest ja kuvada kasutaja keha

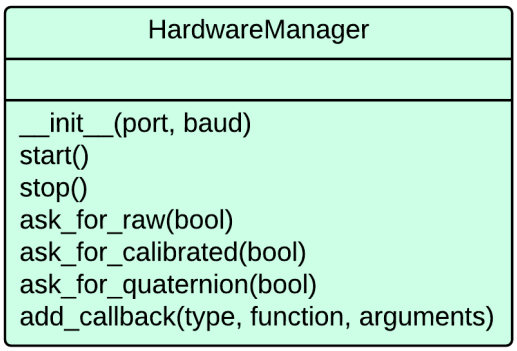


Joonis 3.1 Seosed tähtsamate klasside vahel

jäljendavat kujutist. Klassi ImuPlotter abil saab graafikutel visualiseerida sensoritest tulevaid toor- ja kalibreeritud andmeid. ImuPlotter kasutab samuti riistvarast andmete kättesaamiseks HardwareManager klassi. Lisaks loodi kalibreerimistarkvara, mille abil on võimalik kalibreerida magnetomeetrit ning visualiseerida nii toor, kui kalibreeritud magnetomeetri andmeid kolmemõõtmelisel hajuvusdiagrammil. Kalibreerimistarkvaras kuvatakse HardwareManager klassi abil saadud andmed kolmemõõtmelisel graafikul klassi ScatterPlotter abiga.

3.1.1. HardwareManager

HardwareManager on klass, mis kapseldab endasse kogu riistvaraga suhtlemise loogika. Klassil on avalikud meetodid, millega saab juhtida riistvara tööd ning kasutada kogu selle funktsionaalsust (Joonis 3.2). Meetodid nagu näiteks *ask_quaternion*, *ask_calibrated* ja *ask_raw* instrueerivad koordinaatorseadet küsima klientseadmetelt vastavalt kas kvaternioone, sensorite kalibreeritud andmeid või toorandmeid. HardwareManager klass võimaldab ka vastu võtta riistvara poolt saadetud sõnumeid. Selleks tuleb *add_callback* meetodi abil lisada nimekirja funktsioonid, mille HardwareManager välja kutsub, kui soovitud tüüpi sõnum on saabunud. Klass sõelub (ing. k. *parse*) riistvara poolt tulevad sõnumid ning käivitab järjest kõik *add_callback* meetodi abil lisatud funktsioonid. Käivitatud funktsioonile antakse argumendiks saabunud sõelutud andmed. Selline ehitus lubab mitmel programmiosal üksteisel sõltumatult kasutada riistvarast tulevaid sõnumeid.

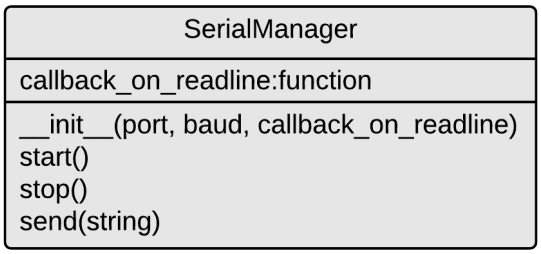


Joonis 3.2 Klassi HardwareManager avalikud meetodid

HardwareManager kasutab jadaühenduse loomiseks ja hoidmiseks SerialManager klassi. HardwareManager klassi isendi loomisel luuakse ka SerialManager klassi isend, mis kapseldab endas jadaühenduse kasutamise detailid.

3.1.2. SerialManager

SerialManager on klass, mis kapseldab endasse jadaühenduse kaudu info vahetamise detailid. Klassil on avalikuks kasutamiseks *start* ja *stop* meetodid, millega vastavalt saab ühenduse luua ja peatada (Joonis 3.3). Lisaks on meetod *send*, mille abil saab saata koordinaatorseadmele andmeid. Andmete vastuvõtmisel koordinaatorseadmest käivitatakse *callback* funktsioon mis määratakse isendi loomisel.



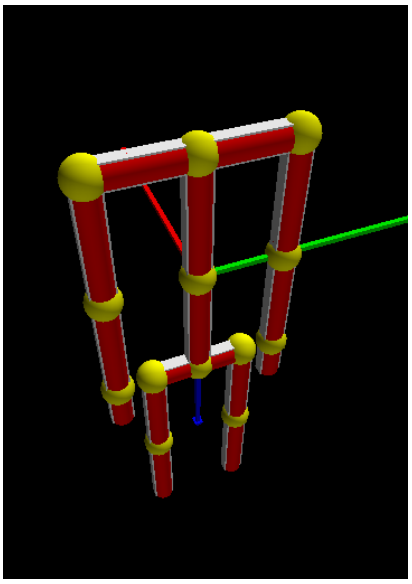
Joonis 3.3 Klassi SerialManager avalikud atribuudid ja meetodid.

SerialManager klassi isendi loomisel luuakse kaks lõime – üks andmete saatmise ja teine vastuvõtmise jaoks. Jadaühenduse kaudu info saatmiseks kasutatav *send* meetod asetab argumendiks oleva andmehulga vaid järjekorda. Jadaporti saadab infot vaid andmete saatmise lõim, mis võtab andmed järjekorrast. Järjekorra kasutamine annab võimaluse mitmel lõimel korraga turvaliselt jadaporti andmeid saata kasutades *send* meetodit. Teine, saatmislõim jälgib jadapordist tulevat infot ning ühe tervikliku rea vastuvõtmise järel kutsub välja *callback* funktsiooni, mis isendi loomisel määrati.

Alternatiivselt võiks kasutada ühte lõime, mis vaheldumisi kontrollib nii sissetuleva info olemasolu, kui välja mineva info olemasolu. Sel juhul kasutataks protsessori ressursse ka siis, kui jadaühendust ei kasutata. Lisaks tuleb piirata selle tsükli kiirust, et mitte kogu aeg kasutada kogu protsessori vaba aega. See aga vähendaks jadaühenduse läbilaskevõimet. Kahe eraldi lõime kasutamine võimaldab kummalgi lõimel jääda ootama, kui jadaühendust ei kasutata. Saatmislõim jääb ootama saatmisjärjekorrast elemendi võtmise juurde ning vastuvõtmislõim jääb ootama rida jadaühendusest. Sellises seisus ei kasuta kumbki lõim protsessori ressursse. Samas võimaldab selline struktuur ka täielikult ära kasutada protsessori kogu vaba ressursi, et tagada jadaühenduse võimalikult suur läbilaskevõime.

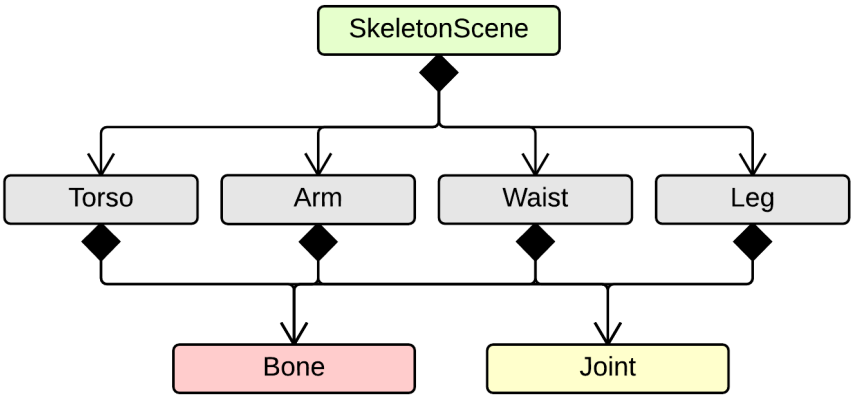
3.1.3. SkeletonScene

Inimese kujutise arvutis kuvamiseks on kasutatud moodulite paketti VPython, mille abil saab luua interaktiivseid kolmemõõtmelisi mudeleid. Selle moodulite paketi abil saab kuvada kolmemõõtmelisi objekte ilma graafika koodi kirjutamiseta. Loodud objekte kuvab eraldi lõimes jooksev moodul Visual. Hiire abil on saab kolmemõõtmelist stseeni pöörata ning nihutada.



Joonis 3.4 Inimese mudeli kujutis programmis skeleton_scene.py.

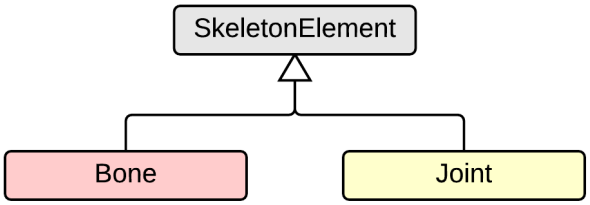
Inimese kujutise kuvamiseks ja süsteemi töö visualiseerimiseks loodi programm skeleton_scene.py. Selle peamine klass on SkeletonScene. SkeletonScene on klass, mis kuvab kolmemõõtmist inimese mudeli kujutist kasutades VPythoni moodulite paketti. Madalaimal tasemel koosneb mudel Bone (edaspidi töös ka „luu“) ja Joint (edaspidi töös ka „liiges“) klassi isenditest. Joonisel 3.4 on näha SkeletonScene klassi abil loodud inimese keha mudel. Luid kuvatakse punaste silindritena, millel on hallid triibud ning liigeseid kollaste keradena. Kuna luid ja liigeseid on keha mudelis palju, on need grupeeritud klassidesse Torso, Arm, Waist ja Leg (Joonis 3.5). SkeletonScene kasutab ühte Torso ja Waist klassi isendit ja kahte Arm ja Leg isendit. Kõik need klassid sisaldavad Bone ja Joint klasse, mille abil vastav kehaosa on kujutatud.



Joonis 3.5 SkeletonScene poolt kasutavad klassid.

3.1.3.1. SkeletonElement

SkeletonElement on baasklass, mis hoiab endas skeleti põhiosade, Bone ja Joint klasside, ühiseid meetodeid ja välju. See klass on mõeldud vaid laiendamiseks Bone ja Joint klasside poolt. Peamiselt kirjeldab see klass endas skeleti ülesehitust, struktuuri. SkeletonElement isenditest saab luua puulaadse struktuuri, kus igal isendil võib olla mitu alamat ning kuni üks ülem. Iga alam kinnitub oma ülema külge. Alamaid saab isendile lisada meetodiga *attach*. SkeletonElement isendi pööramisel või liigutamisel ruumis tuleb uuendada ka kõikide selle külge kinnitatud alamate asukohti. Selleks on meetod *reposition_attachments*, mis uuendab rekursiivselt kõikide alamate asukohad.



Joonis 3.6 SkeletonElement on Bone ja Joint klasside ülemklass.

3.1.3.2. Bone

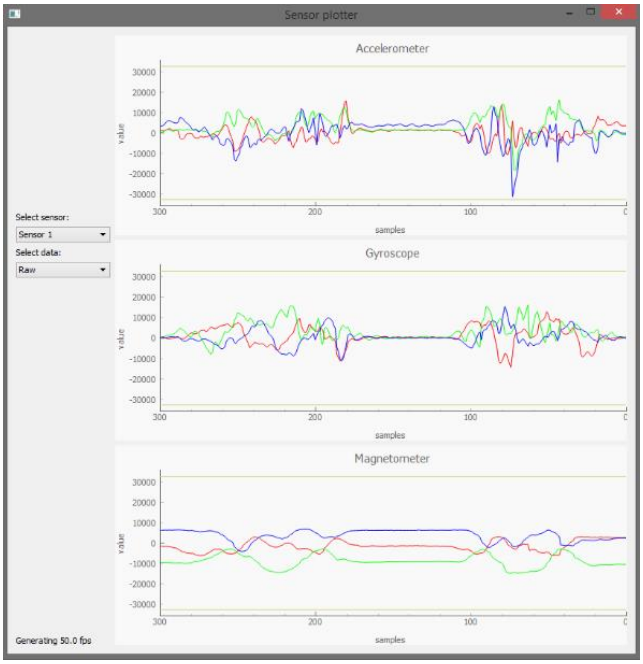
Bone klassi isendid on kuvatava inimkeha kujutise ühtedeks põhilisteks komponentideks. Bone klass on klassi SkeletonElement alamklass, pärides sellega kõik eelnevalt kirjeldatud omadused. Graafiliselt kujutatakse ühte luud punase silindri ja halli ristküliku kooslusena. Luu üks ots on kinnituspunktiks oma ülemaga ning teine ots kinnituspunktiks alamatega. Luud on pööratavad meetodi *set_orientation* abil vabalt valitud asendisse. Seda meetodit kasutatakse siis, kui klientseadmetelt saabub info kehaosa suuna kohta.

3.1.3.3. Joint

Joint klassi isendite põhiline eesmärk on visuaalselt eraldada luud ning täita ära vahed, mis tekivad kahe luu ühenduskohal. Graafiliselt kujutatakse neid kollaste keradena. Kerad luude ühenduskohtades muudavad skeleti struktuuri jälgimise lihtsamaks, tähistades ühe luu lõppu ja teise algust. Lisaks seob see inimese keha mudeli visuaalselt kokku, kattes vahed, mis tekivad kahe luu ühenduskohal. Ka Joint klassi ülemklassiks on SkeletonElement. Seega on ka liigesed puustruktuuri osad – neid saab kinnitada ja nende külge saab teisi elemente kinnitada.

3.1.4. ImuPlotter

Sensorite töö visualiseerimiseks ja kontrollimiseks loodi Pythoni programm imu_plotter.py mis reaalajas kuvab valitud mooduli sensorite väljundid kõigis kolmes teljes (Joonis 3.7). Programmi peamine klass on ImuPlotter. Selle klassi instantsi loomisel kuvatakse kasutajaliides, kus on graafikud iga sensori väljundi kuvamiseks ning rippmenüüd jälgitava sensori ja andmetüübi valimiseks. Klass ImuPlotter kasutab riistvaraga suhtlemiseks HardwareManager klassi. Läbi HardwareManager klassi palutakse koordinaatorseadmel saata kas toor- või kalibreeritud andmeid ning kuvatakse need seejärel graafikutel.



Joonis 3.7 Sensoritest tulevate andmete visualiseerimiseks loodud programm imu_plotter.py

3.2. Seadmete püsivara

Seadmetele kirjutatud püsivara jaguneb kaheks. Kummalgi seadmetüübil, klient- ja koordinaatorseadmel, on oma tarkvara. Klientseadme mikrokontrolleris olev püsivara kogub sensoritest andmeid, kasutab neid oma positsiooni arvutamiseks ning vastab raadiomooduli teel saadud päringutele. Koordinaatorseade saadab klientseadmetele andmepäringuid ning edastab saadud vastused arvutile.

3.2.1. Klientseadme püsivara

Klientseadme peamine eesmärk on oma asendi pidev jälgimine. Selleks tuleb klientseadmel perioodiliselt mõõta kiirendust, pöörlemiskiirust ja magnetvälja inertsiaal- ja magnetandurite abil. Saadud väärtuseid kasutades hindab moodul suunafiltri abil oma asendit. Lisaks suhtleb klientseade koordinaatorseadmega, vastates selle päringutele.

Raadiomooduli ning LSM9DS0 kiibiga suhtlemiseks kasutatakse juba olemasolevaid Arduino platvormile loodud teeke. nRF24L01 raadiomooduliga suhtlemiseks kasutatakse GitHub kasutaja

TMRh20 teeki RF24. See teek on GitHub kasutaja maniacbug poolt loodud teegi RF24 haru. Antud haru on lihtsasti kasutatav ning järgib paremini kiibi tootja dokumentatsiooni juhiseid kui algne teek [12]. LSM9DS0 süsteemikiibiga suhtlemiseks kasutatakse Jim Lindblomi kirjutatud teeki. Teek on samuti mõeldud Arduino platvormile ning selle abil on võimalik suhelda kiviga nii üle I2C kui SPI. Selles on realiseeritud funktsioonid nii sensoritest andmete kättesaamiseks kui ka sensorite mõõtevahemiku ja värskendussageduse muutmiseks.

Andmevahetust klientseadmetega alustab alati koordinaatorseade. Seetõttu on klientseadmes mugav ära kasutada raadiomooduli katkestusnõudeliini, et anda teada mikrokontrollerile saabunud andmepäringust. Selleks seadistatakse raadiomooduli katkestusnõudeliin aktiviseeruma vaid paketi edukal vastuvõtmisel. Teised katkestusallikad maskeeritakse. Raadiomooduli katkestusnõude väljund on ühendatud ühte mikrokontrolleri katkestusnõude sisendisse. Selle katkestusnõudeliini aktiviseerumine käivitab mikrokontrolleris katkestustöötleja, mis võtab raadiomoodulist vastu päringu ning vastab sellele.

Klientseade saab andmepäringule vastata kohe, kui vastuvõetud pakett nõuab vastamist vaid ühelt klientseadmelt. Kui vastuvõetud pakett nõuab vastamist kõigilt klientseadmetelt (näiteks asendipäringu puhul), tuleb ajastada klientseadmete vastuseid nii, et raadiosignaalid üksteisega ajaliselt ei kattuks. Seetõttu peavad klientseadmed vastama järjekorras, viivitades vastusega vastavalt oma järjekorranumbrile. Selleks käivitavad kõik klientseadmed paketi vastuvõtmise järel loenduri, mis genereerib perioodiliselt katkestusi. Iga katkestuse ajal suurendatakse katkestuse loendurit 1 võrra ning kui see on võrdne klientseadme järjekorranumbriga, saadetakse vastus koordinaatorseadmele. Katkestusi genereeritakse iga 680µs tagant, seega 11 klientseadme andmed edastatakse 7.5ms jooksul. Kui katkestuste loendur on jõudnud viimase klientseadme järjekorranumbrini, peatatakse kõikides klientseadmetes loendur ning rohkem katkestusi ei genereerita.

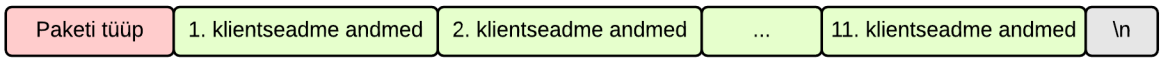
3.2.1.1. Raadioside paketiseerimine

Antud raadiomoodulid võimaldavad seadistada igale raadiomoodulile oma aadressi, et andmepakette oleks võimalik saata ainult ühele vastuvõtjale. Koordinaatorseadmel on tihti vaja saata aga päring korraga kõikidele klientseadmetele. Seetõttu on kõikide klientseadmete raadiomoodulite aadressid identsed.

nRF24L01 raadiomoodul võimaldab saata kuni 32 baidiseid pakette. Et tõlgendada raadiomoodulist saadud 32 baidist andmeüksust lõi töö autor oma protokolli. Esimesed 3 andmeüksuse baiti on päiseks, mis on kõigil andmeüksustel ühesuguse struktuuriga. Päises on kolm ühe baidilist välja: saatja aadress, sensorikomplekti number ja info tüüp. Saatja aadressi ja sensorikomplekti numbrit kasutatakse infot saatva klientseadme tuvastamiseks. Info tüüp määrab ära, kuidas tõlgendada ülejäänud andmeid.

3.2.2. Koordinaatorseadme püsivara

Koordinaatorseadme peamiseks eesmärgiks on klientseadmetes olevate andmete transportimine arvutisse. Selleks teeb koordinaatorseade raadiomooduli abil perioodilisi andmepäringuid klientseadmetele. Klientseadmetelt saadud vastused edastab seade läbi USB siini arvutisse. Läbi USB siini on loodud virtuaalne jadaühendus. Jadaühenduse paketid algavad paketitüübi tähisega ning lõppevad reavahetuse märgiga (Joonis 3.8). Binaarandmed edastatakse üle jadaühenduse teisendatuna kuueteistkümnendsüsteemi. See teisendus tehakse, et vältida reavahetuse märgi sattumist andmetesse, mis märgiks paketi lõppu.

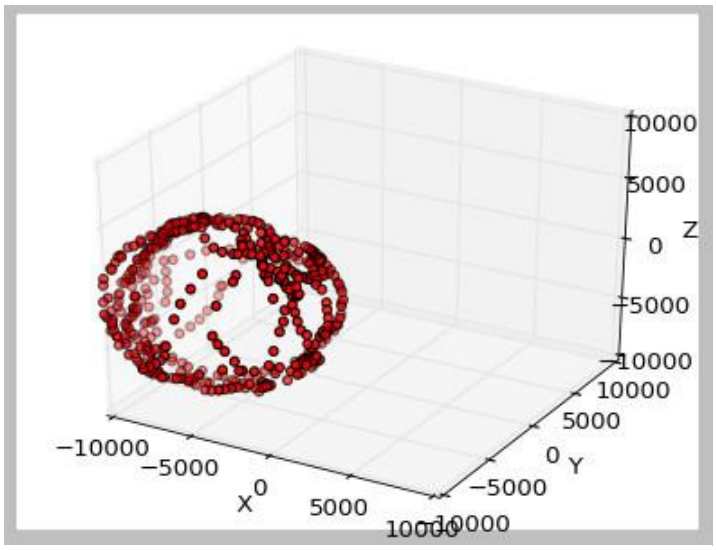


Joonis 3.8 Jadaühenduse teel saadetava paketi struktuur

Iga 13ms tagant teeb koordinaatorseade päringu klientseadmetele. Iga klientseade vastab ühe paketiga, milles olev info asetatakse jadaühenduse paketti klientseadme numbrile vastavale kohale. Kui aeg on teha järgmine päring, saadetakse jadaühenduse pakett arvutisse ning tehakse raadio teel uus päring klientseadmetele.

4. Magnetvälja moonutuste korrigeerimine

Süsteemi katsetamise käigus selgus, et suur osa klientseadmetest ei ole võimelised oma suunda ruumis hindama ilma magnetomeetri tulemuste korrigeerimiseta. Kui magnetomeetri poolt mõõdetud magnetväljas puuduksid moonutused, oleks seadme pöörlemisel mõõdetud magnetvälja tugevuse vektor alati ühe pikkusega. Saadud tulemused moodustaksid kolmemõõtmelisel graafikul sfääri, mille keskpunkt asub koordinaatteljestiku nullpunktis ning mille raadis on võrdne välise magnetvälja tugevusega B. Seadmesisesed moonutused aga nihutavad selle sfääri keskpunkti ning moonutavad selle kuju. Seetõttu on iga magnetomeetrit kasutava seadme tähtsaks osaks on selle kalibreerimine. [13] Joonis 4.1 kujutab ühe kalibreerimata klientseadme magnetomeetri toorandmeid. Joonisel on näha andmete tugevat nihet x ja y telje negatiivses suunas, mis on põhjustatud klientseadmes kasutatud ferromagneetilistest materjalidest.



Joonis 4.1 Kalibreerimata magnetomeetri toorandmed kujutatuna kolmemõõtmelisel hajuvusdiagrammil

4.1. Moonutused

Moonutusi sensorit ümbritsevas magnetväljas on erinevaid. Iga moonutus avaldub toorandmetes erinevalt. Järgnevalt on kirjeldatud erinevate moonutuste põhjuseid ning mõjusid mõõtetulemustele.

Esimest tüüpi (ingl. k *hard-iron*) moonutused on põhjustatud seadmes kasutatud ferromagneetilistest materjalidest. Seadmes olevad magneetunud materjalid tekitavad sensori ümber konstantse tugevusega välja, mis ajas ei muutu. See konstante väli liitub maa magnetväljaga. Selle tulemusel on kõik maa magnetvälja mõõtetulemused konstantse väärtuse võrra nihkes. [13]

Teist tüüpi (ingl. k *soft-iron*) moonutused on põhjustatud magnetiliselt pehmetest materjalidest. Selle moonutuse mõjul muutub magnetomeetri väljund sfäärist kaldega ellipsi kujuliseks. [13]

Lisaks magnetvälja moonutustele mõjutab tulemuste täpsust ka sensori telgede tundlikkuse erinevus. Ideaalsel juhul on magnetvälja sensori iga kanal võrdse tundlikkusega, kuid reaalsuses on need erinevad. Selle vea tulemusena moodustub sensori pööramisel mõõtetulemustest sfääri asemel ellipsoid. [13]

Antud töös oli suurimaks moonutuste allikaks klientseadmete sees kasutatud ferromagneetilised materjalid. Trükkplaadi, kruvide ja aku poolt loodud magnetvälja tugevus oli mitmel juhul isegi suurem, kui Maa magnetvälja tugevus. Selline juht on näha ka joonisel 4.1 , kus sensori pööramisel ei saavutata y-telje suunas kunagi positiivseid väärtuseid. Kalibreerimata tulemuste kasutamisel jäi asendifiltri väljund alati mingisse kindlasse suunavahemikku. Pärast kalibreerimist see efekt kadus.

Klientseadmete esimene prototüüp kasutas energiaallikana AAA patareisid, kuid need mõjutasid oluliselt sensorit ümbritsevat magnetvälja. Sensori suhtes alati ühes asendis oleva patarei magnetvälja moonutusi sai kalibreerimisega oluliselt vähendada. Samas ilmnes, et moonutused ei jää samasuguseks, kui patareid pöörata pesas ümber oma telje. See tähendab, et igal patarei vahetusel tuleb seade uuesti kalibreerida. Selle töö vältimiseks asendati klientseadmete AAA patareid liitiumpolümeerakudega, mida ei pea laadimiseks seadmest eemaldama. Lisaks on liitiumpolümeeraku mõju magnetväljale väiksem kui AAA patareil.

4.2. Korrigeerimise metoodika

Antud töös modelleeriti ja korrigeeriti ferromagneetilistest materjalidest põhjustatud nihet ning sensori telgede tundlikkuse viga. Et teisendada sensorist saadud mõõtetulemus korrigeeritud väärtuseks kasutati järgevat valemit, kus M_x, M_y ja M_z on mõõtetulemused ja B_x, B_y ja B_z, korrigeeritud väärtused.

$$\begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix} \begin{bmatrix} M_x - h_x \\ M_y - h_y \\ M_z - h_z \end{bmatrix}$$

Kokku on valemis kuus otsitavat parameetrit. Kolm parameetrit, h_x, h_y ja h_z, kirjeldavad andmete nihet nullnivoo suhtes ning ülejäänud kolm, s_x, s_y ja s_z telgede tundlikkust.

Magnetomeetri kalibreerimiseks kogutakse sensori abil toorandmeid ning kasutatakse neid kuue parameetri leidmiseks. Eesmärgiks on leida parameetrid, mis kõige paremini teisendavad iga

andmepunkti (h_x, h_y, h_z) punktiks ühiksfääril (B_x, B_y ja B_z), mille keskpunkt asub koordinaatide algpunktis. Parameetrid leitakse, kasutades vähimruutude meetodit.

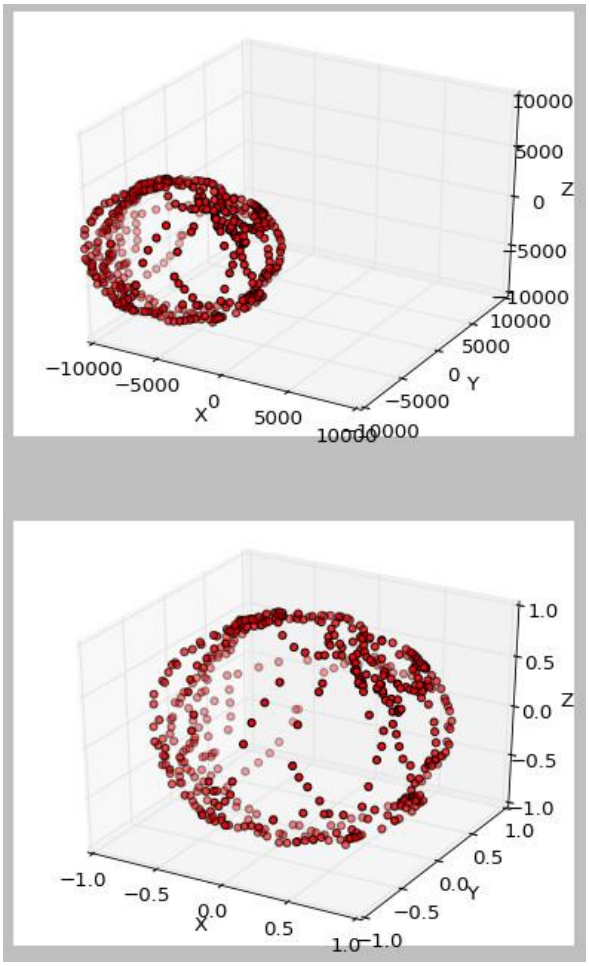
Kui on teada kõik kuus parameetrit, saab eelnevalt kirjeldatud valemi abil kujutada iga mõõtetulemuse korrigeeritud väärtuseks. Saadud korrigeeritud väärtused on lähendid ühikvektorile ning moodustavad koos ühiksfääri, mille keskpunkt asub koordinaatide algpunktis.

4.3. Korrigeerimise tarkvara

Magnetomeetri kalibreerimiseks on loodud Pythoni programm mis kasutab tulemuste visualiseerimiseks Matplotlib teeki. Klassi ScatterPlotter loomisel kuvatakse kaks kolmemõõtmelist hajuvusdiagrammi (Joonis 4.2). Ühel kuvatakse toorandmeid ja teisel kalibreeritud andmeid. Andmed kuvatakse punktidena kolmemõõtmelises ruumis. Töötava magnetomeetri suvalisel pööramisel tekib toorandmete graafikule ellipsoidne punktiparv. Teisel graafikul kuvatakse need samad punktid pärast nihutamist ja skaleerimist nii, et punktiparv moodustaks sfääri raadiusega 1, mille keskpunkt asub koordinaatide algpunktis.

Kalibreerimiseks tuleb seadet keerutada nii, et saadud tulemused kataksid ühtlaselt kogu mõõtepiirkonna, et mudelisse sobitamine oleks võimalikult täpne. Selleks tuleb seadet pöörata ühtlasel kiirusel ümber erinevate telgede, et seade satuks võimalikult erinevatesse asenditesse. Selle tulemusena tekib esimesele graafikule ellipsoidne punktide kogum. Kalibreeritud andmete graafikule tekib õnnestunud kalibreeringu leidmisel sfääriline punktide kogum, mille keskpunkt on koordinaatide alguspunktis ning mille raadius on igal teljel 1. Sel juhul võib kopeerida programmi väljundist kalibreerimistulemused ning asetada need lähtekoodi.

Klientseadme magnetomeetri kalibreeringu kontrollimiseks, pärast konstantide laadimist klientseadmesse, kasutati mõõtetulemuste visualiseerimise programmi imu_plotter.py. Programm seadistati kuvama valitud klientseadme kalibreeritud mõõtetulemusi. Klientseadmega tehti juhuslikke pöördeid ning jälgiti kalibreeritud magnetomeetri tulemusi. Magnetomeeter loeti korrektselt kalibreerituks kui kõikide telgede amplituudid olid lähedased ühele nii positiivses kui negatiivses suunas.



Joonis 4.2 Kalibreerimisprogrammis magnetometer_calibration.py kuvatud kalibreerimata ja kalibreeritud andmed.

Kokkuvõte

Virtuaalreaalsus on tehnoloogia, mis võimaldab selle kasutajal kogeda arvutite abil simuleeritud tehiskeskkondi. Peakomplektid nagu Oculus Rift lubavad kasutajal virtuaalses ruumis ringi vaadata, kuvades stereoskoopilist pilti, mis muutub vastavalt kasutaja peaasendile. Käesoleva töö eesmärgiks oli luua tarkvara juhtmevabale kehaasendi jälgimise süsteemile. Sellise süsteemi olemasolu võimaldab luua virtuaalreaalsussüsteeme, kus kasutaja näeb tehismaailmas oma keha ning saab loomulike liigutuste abil sellega interakteeruda

Keha asendi tuvastamiseks jälgitakse 11 kehaosa suunda ruumis. Iga jälgitava kehaosa külge saab kasutaja kinnitada juhtmevaba mooduli, mis jälgib oma suunda. Moodul sisaldab kiirendusandurit, güroskoopi ja magnetomeetrit ning kasutab suuna hindamiseks Sebastian Madgwicki suunafiltri algoritmi. Töö käigus valmis tarkvara juhtmevabadele moodulitele, arvutisse infot edastavale koordinaatorseadmele ning arvutile. Lisaks kirjeldatakse töös magnetvälja moonutuste allikaid, nende korrigeerimise võimalusi ning korrigeerimise jaoks loodud tarkvara.

Töö tulemusena valmis süsteem, mis jälgib kasutaja kehaasendit ning kuvab selle arvutis oleval inimkeha mudelil. Lisaks valmis tarkvara andurite väljundite graafiliseks esitamiseks ning magnetomeetri kalibreerimiseks. Käesolevaks hetkeks on töö tulemust kasutatud ka virtuaalreaalsussüsteemis, kus kasutaja saab virtuaalreaalsuses näha liikumas enda käsi, jalgu ja keha Oculus Rifti abil (Joonis 4.3 ja Joonis 4.4). Edasi tasub kindlasti uurida, kuidas kasutada jalgade suunainfot nii, et kasutaja oleks võimalus süsteemi abil virtuaalses ruumis ka ringi liikuda.

Joonis 4.3 Kasutaja vaatab virtuaalreaalsuses oma käsi.

Summary

Software development for body tracking system

Virtual reality is a type of technology that utilizes computers to simulate environments that users can experience and interact with. Virtual reality headsets allow users to look around in virtual space by displaying a stereoscopic image of the virtual world that changes according to the users current head orientation. Adding body tracking would allow users to see their bodies in the virtual world, thus providing better immersion. The aim of the thesis was to develop a software for a real-time body tracking system. This system could be used along with the existing virtual reality headsets allowing users to interact with simulation using intuitive natural movements.

The body tracking was implemented by tracking the orientation of 11 body parts. Every body part being tracked has a wireless module attached to it which contains a number of sensors. The module contains an accelerometer, a gyroscope and a magnetometer and uses a Sebastian Madgwicks orientation filter to accurately estimate orientation based on sensor measurements. The orientations are sent wirelessly to a computer through a coordinator module. The computer uses the orientations to visualize the body posture. A software was developed for wireless tracking modules, for the coordinating module attached to a computer and for a computer processing and visualizing the body posture. Additionally the sources of magnetic distortions and errors were investigated, after which methods to correct these distortions were proposed and implemented on the system.

As a result of this thesis, a system was created that can successfully track the body posture and visualize it on the human model displayed on the computer screen. Now, the system is also used in a virtual reality system that allows users to see their hands, feet and body in the virtual world using the Oculus Rift headset (Figures 4.3 and 4.4). Further improvements could be made to have the system recognize the movement of legs so that the user can experience walking in the virtual world.

Joonis 4.4 Kasutaja vaatab virtuaalreaalsuses oma jalga.

Allikad

[1] Y. A. G. V. Boas, „Overview of Virtual Reality Technologies,“ University of Southampton, Southampton, United Kingdom, 2013.

[2] Oculus, „Oculus Rift,“ [Võrgumaterjal]. URL: https://www.oculus.com/. [Kasutatud 11 05 2015].

[3] Control VR, „Control VR,“ [Võrgumaterjal]. URL: http://controlvr.com/. [Kasutatud 11 05 2015].

[4] Virtuix, „Virtuix Omni,“ [Võrgumaterjal]. URL: http://www.virtuix.com/. [Kasutatud 11 05 2015].

[5] S. Madgwick, "An efficient orientation filter for inertial and inertial/magnetic sensor arrays," *Report x-io and University of Bristol (UK),* 2010.

[6] J. Diebel, „Representing Attitude: Euler Angles, Unit Quaternions, and Rotation,“ Stanford University, California, 2006.

[7] J. B. Kuipers, „Quaternions and rotation sequences,“ *Princeton university press,* kd. 66, pp. 127-143, 1999.

[8] STMicroelectronics, „LSM9DS0 iNEMO inertial module:3D accelerometer, 3D gyroscope, 3D magnetometer,“ 8 2013. [Võrgumaterjal]. URL: http://www.st.com/st-web-ui/static/active/en/resource/technical/ document/datasheet/DM00087365.pdf. [Kasutatud 2 4 2015].

[9] Atmel Corporation, „ATmega16U4/ATmega32U4 Datasheet,“ 09 2014. [Võrgumaterjal]. URL: http://www.atmel.com/Images/ Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf. [Kasutatud 2015 05 10].

[10] S. Maus, S. McLean, M. Nair, C. Rollins, S. Macmillan, B. Hamilton ja A. Thomson, „The US/UK World Magnetic Model for 2010-2015,,“ NOAA Technical Report NESDIS/NGDC, 2010.

[11] Nordic Semiconductor, „nRF24L01 - 2.4GHz RF,“ 7 2007. [Võrgumaterjal]. URL: http://www.nordicsemi.com/eng/content/download/2730/34105/file/nRF24L01_Product_Specification_v2_0.pdf. [Kasutatud 2 4 2015].

[12] TMRh20, „Optimized High Speed NRF24L01+ Driver Class Documenation,“ [Võrgumaterjal]. URL: http://tmrh20.github.io/RF24/. [Kasutatud 29 04 2015].

[13] STMicroelectronics, „AN3192: Using LSM303DLH for a tilt compensated electronic compass,“ 2010.

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Mark Laane,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

Tarkvara loomine kehaasendi jälgimise süsteemile,

mille juhendaja on Artur Abels,

1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu alates **01.06.2020** kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **22.05.2015**