TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND

Elektroenergeetika ja mehhatroonika instituut

SAMAAEGNE LOKALISEERIMINE JA KAARDI KOOSTAMINE MOBIILSETEL ROBOTITEL

SIMULTANEOUS LOCALIZATION AND MAPPING IN MOBILE ROBOTS

BAKALAUREUSETÖÖ

Üliõpilane : Georg Reintam

Üliõplaskood : MAHB142506

Juhenadaja : Mart Tamre

Tallinn 2018

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.

Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

“.....” ..... 201.....

Autor: . . . .
/ allkiri /

Töö vastab bakalaureusetöö/magistritööle esitatud nõuetele

“.....” ..... 201.....

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud

“.....” .....201... .

Kaitsmiskomisjoni esimees ....
/ nimi ja allkiri /

TTÜ Instituudi nimetus

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Georg Reintam 142506 MAHB

Õppekava, peaeriala: MAHB02/13 Mehhatroonika

Juhendaja(d): Mart Tamre, mart.tamre@ttu.ee , 620 3202

Lõputöö teema: SLAM algoritmide võrdlemine

(eesti keeles) *Pealkiri peab olema võimalikult lühike ja konkreetne, ent samas andma lugejale selge ettekujutuse uurimisprobleemi põhiolemusest*

(inglise keeles) Comparing different SLAM algorithms

Lõputöö põhieesmärgid:

1. Ülevaade valdkonna hetkeseisundist
2. Valitud algoritmide põhjalik arusaam
3. Simuleeritud keskkonnal algoritmide võimekuste erinevuste võrdlemine

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Tutvumine kirjandusega	25.03
2.	Simulatsiooni ja mudelite simuleerimine/leidmine	25.04
3.	Andmete analüüs ja töö vormistamine	14.05

Töö keel: Eesti **Lõputöö esitamise tähtaeg:** 25.05.2018a

Üliõpilane: Georg Reintam ..... "12" 03 2018.a

/allkiri/

Juhendaja: Mart Tamre ..... "12." 03 2018.a

/allkiri/

Sisukord

1 SISSEJUHATUS

Robotid vajavad juhiseid (mis on ette antud), et navigeerida punktist A punkti B. Üks võimalus seda saavutada on kasutada roboti positsiooni leidmiseks välise sensorsüsteemi abi. Selle tarbeks võib siseruumides kasutada liikumise tuvastamise süsteemi (motion capture system), mis kasutab visuaalseid sensoreid keskkonna kohal, et edastada robotile informatsioon hetkelisest asukohast. Välistingimustes kõige levinum viis positsioneerimiseks on kasutada satelliitnavigatsioon süsteemi (GPS). Selliseid lahendusi on mugav kasutada, kuna esimese puhul võib poosi anda kuni millimeetri - ja teisel puhul sentimeetri täpsusega. Mis saab siis kui nende vaheline ühendus ära kaob? Levi puudumise tõttu eksib robot ära ja ei suuda tööd lõpuni korrektselt viia. Roboti kasutamise eeldused on tema rakendamine nii tihti kui võimalik ja inimesest sõltumatu, kuid kui ei tööta siis kui tarvis, mis kasu robotis on?

Navigeerimine on robotite korrapärase funktsioneerimise ja töö eesmärgi saavutamise alustala. Selle õnnestumise tarbeks peab robot tajuma ennast ümbritsevat keskkonda, tajumisest saadud informatsiooni tõlgendamine ja selle abil teekonna planeerimine. Süsteemi disainides on programmeerijal eesmärk kaasata inim-operaatorit võimalikult vähe ja kui võimalik siis saavutada täielik autonoomsus. Mobiilsetes robotites on selle üheks eelduseks toodud see, et robot peab kasutama ainult enda küljes olevaid sensoreid ja arvutamisvõimsust. Samuti ei tohi ka kasutada välist abi (näiteks GPS).

Käesolev töö püüab esmalt leida vastuse kuidas robot suudab töötada kasutades ainult enda küljes asetsevaid sensoreid. Teisena disainida töötav süsteem väli- ja teenindusrobotite tarbeks, kasutades kirjanduses välja pakutud meetodeid.

Töö esimeses osas vaadatakse autonoomse roboti mõistet ja iseloomu ning sellest lähtudes kuidas üldiselt süsteem peab olema ülesse ehitatud. Teine osa kitsendab fookust ja uurib mis on navigeerimine ja kuidas leida vastus küsimusele „kus ma olen“. Kolmas osa toob välja kirjanduse poolt pakutud lahendused teises osas tekkinud probleemidele. Viimases, neljandas osas, vastan sissejuhatuses tekkinud probleemile, et kuidas navigeerida ilma a posteriori kaardi või väliste sensorite abita. Samuti on välja pakutud süsteem raamistik väli- ja teenindusrobotite positsiooni leidmiseks ilma süsteemivälise abita.

2 AUTONOOMNE MOBIILNE ROBOT

Enne robotsüsteemi disainimist peab inseneril olema võimalikult täpne arusaam, mis on töö tingimused ja eesmärgid. Samuti tuleb selgeks teha, kui suur inimese roll on toimiva süsteemi juhtimisel, sest see määrab ära disaini raskuse – lihtsa protseduurilise ülesande täitmiseks ei vaja robot intelligentsi. Vastupidi on juhul, kui üheselt määratud lahendust ei ole, ning töö täideviimine sünnib süsteemisisese arutlemise käigus.
Selles peatükis püüan leida vastuse mis tähendab autonoomsus ning kuidas seda saavutada mobiilsel robotil. Selle jaoks tuleb ära määrata kui kõrgel tasemel autonoomsust mingi kindel ülesanne vajab. Kuidas neid määrata?

2.1 Autonoomsus

Roboti jaoks üheselt on raske ära määrata autonoomsuse tähendust, vaid tuleb vaadata vastavalt roboti liigile ning töö ülesandele(robot tolmuimeja võib töötada kas operaatori otsesel juhtimisel või suudab iseseisvalt planeerida enese liikumist). Autonoomsuse täielikuks tagamiseks peab täitma järgnevad eeldused:

- Inimelemendi puudumine süsteemis (*control loop*) ehk toimib sõltumatult.
- Robot peab kasutama ainult enda küljes olevaid seadmeid - iseseisev.
- Suutma vastu hakata juhuslikkudele probleemidele ehk autopiloodi olemasolu.
- Loob oma võimed (isereguleeriv)
- Ei sõltu operaatorist ellujäämise puhul (iseseisev)
- Genereerib endale eesmärgid (autonoomne)
- Loob oma eksistentsi – autopoieetiline (elussüsteemide eneseloomise protsess).

Kui eelneva loetelu kolmas, kuues ja seitsmes punkt kuuluvad täna veel teadusulmesse, siis ülejäänu saavutamine on teoreetiliselt jõukohane. Täna autonoomsete robotite all mõeldakse ennekõike seda, et süsteem suudab ülesande täita iseseisvalt ning süsteem on mõjutatud ja samas mõjutab ise reaalselt maailma otse (directly), ilma inimest segamatta. Tänu sellele väljund mõjutab järgnevat sisendit, süsteemi juhib töökeskkonna sees toimuv ning tekib kinnine ring (*closed loop*).
Roboti süsteemi eesmärk on täita temale määratud ülesandeid. Säärane ülesanne võib olla: satelliidina maakerast pildi tegemine, kuukulgurina kuul ringi navigeerida ja edastada informatsiooni maale, kodune tolmuimeja, tööstusrobot mis keevitab või joodab. Valdkondi on palju, kus tänapäeval kasutatakse robotit. Iga eelnev robot on disainitud tema ülesande järgi ja iga töö ei vaja intelligentsi. Nagu

jooniselt 1 näeb, siis on igal robotil omad nõuete raskusastmed töö korrapärase tegemise tarbeks. Autonoomsuse nõudmist mingile robotile saab leida läbi 8 iseloomustava valdkonna:

- Kui muutuv on keskkond?
- Kui mitmekesised on ülesanded?
- Kui rikkas on keskkonna tähenduslikkus?
- Keskkonna ning ülesannete dünaamilisus.
- Ei sõltu operaatorist ellujäämise puhul (iseseisev)
- Ümbruskonna mõõtmine (*partial observability*)? Kui terve objekt ei ole sensorile nähtav. Vastavas süsteemis talletakse mõõtmed mällu ja mida uuendatakse uue info tulekul.
- Kui tähtis on koostöö teiste agentidega(inimeste/robotitega)
- Kui suurt autonoomsust robot kokkuvõttes vajab ülesande valikus ja selle täitmiseks.

	(i) Environment Variability	*(ii)* Task Diversity	*(iii)* Semantics	*(iv)* Dynamics	*(v)* Partial Observability	*(vi)* Cost & Criticality	*(vii)* Interaction & Cooperation	*(viii)* Level of Autonomy
Manufacturing robots	*l*	*l*	*l*	*h*	*l*	*m*	*l*	*l*
Industrial mobile robots	*m*	*m*	*m*	*m*	*m*	*m*	*h*	*l*
Exploration & rescue robots	*h*	*m*	*l*	*m*	*h*	*h*	*h*	*h*
Service and domestic robots	*h*	*h*	*h*	*l*	*h*	*l*	*h*	*m-h*
Autonomous spacecraft	*l*	*l*	*l*	*l*	*h*	*l*	*l*	*m*
Autonomous aerial vehicles	*m*	*m*	*m*	*h*	*h*	*h*	*m*	*m-h*
Autonomous cars	*m*	*l*	*l*	*h*	*m*	*h*	*m*	*l*

Joonis 1: Robti autonoomsuse analüüs[4]

Intelligentne masin peab mõtlema analüütiliselt. Sõnaraamatus on robot defineeritud kui „Robot on masin, mis viib automaatselt täide rida – programmi poolt antud, käske“. Taoline robot ei mõtle iseseisvalt, vaid seda programmi või süsteemi disainides peab disainer a priori sisestama sobilikud käsud. Kuid sellise roboti töötamine on piiratud, sest on programmeeritud kindlaks määratud töökeskkonnas ja ei suuda leida optimaalset lahendust probleemile.

Tööstuses kasutatavad robotites on kõige levinumaks vormiks robot käsi, mille ülesanne on näiteks värvimine, keevitamine ja sarnaseid korduvaid ülesandeid. Masin on õlast kinnitatud kindlasse asukohta tööliinil (statsionaarne) ja sellisel õnnestub liikuda kiirelt ja täpselt. Selliseid masinaid nimetatakse manipulatsioonrobotiteks. Sellistel robotitel puudub mobiilsus. Tänu sellele on sellese kategooriasse kuuluvatel robotitel limiteeritud liikumisulatus, mis sõltub sellest, kuhu see on parasjagu kinnitatud. Teine robotite liigendus on mobiilne robot, mille peamine eeldus eelmise ees on võimalus liikuda mööda töökeskkonda. Kuna autonoomsuses on eesmärgiks imiteerida inimest, seetõttu peab olema võimalikult suur liikumisulatus, siis robot ei tohi olla statsionaarne, vaid mobiilne.

2.2 Mobiilsus

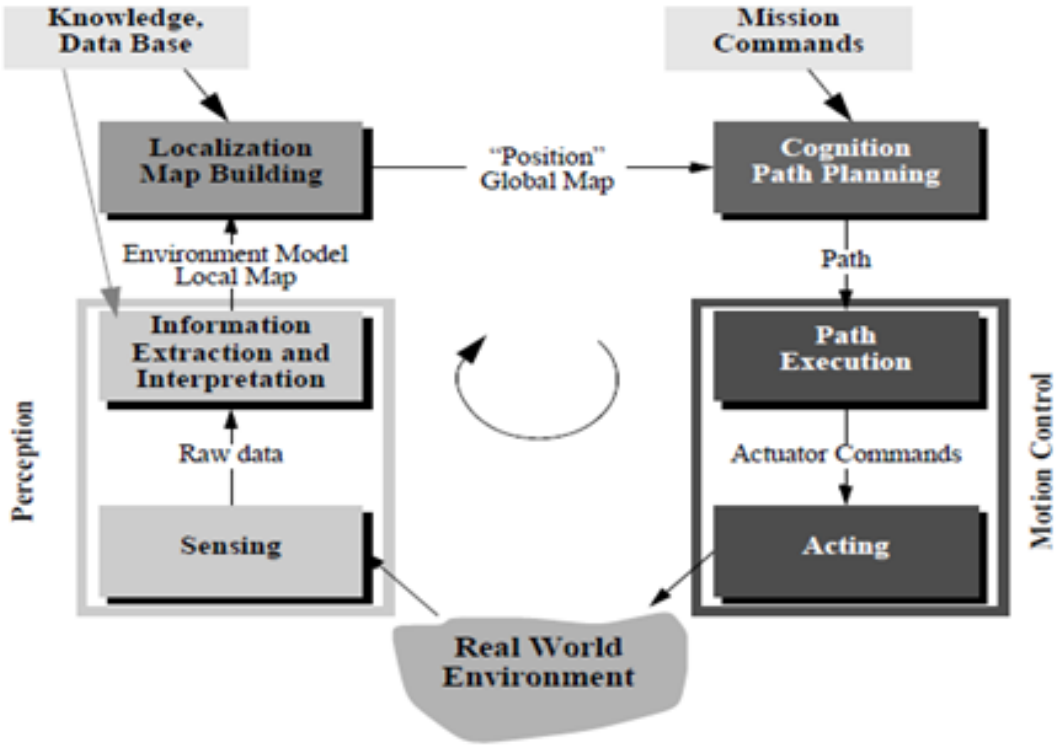
Mobiilsus on võime liikuda piiramatult läbi keskkonna. Selle tarbeks on mobiilsetel robotitel tarvis liikumis mehhanismi (*locomotion mechanism*). Võimalusi on mitmeid ja sobiliku valiku tegemine on disaini protsessis väga tähtis otsus. Valikus on kõndivad, hüppavad, jooksvad, lendavad ja veerevad. Kuid enamus moodused on mehaanilised raskesti teostatavad (humanoid robot kahel jalal) siis kasutatakse kas ratastel veerevaid või propelleri abil lendavaid mehhanisme praktilise töö puhul. Manipulatsioon robotitel on keha fikseeritud kuid liigutab seal paiknevaid objekte. Kahe roboti vahet on hea illustreerida mingi objekti võtmisega laualt. Fikseeritud manipulaator robot peab ära tuvastama objekti positsiooni keskkonnas, välja arvutama trajektoori ja sooritama ülesande. Mobiilsed robotid omakorda peavad otsustama kust kohast on objektile kõige parem läheneda ja alles siis tegeleda kuidas objekt ülesse tõsta. Vedamispõhised (*locomotion*) robotitel on ümbritsev keskkond süsteemis fikseeritud ja nemad ise on liikuvad. See teeb mobiilsete robotite kontrollimise omakorda raskemaks, kuna statsionaarne roboti ülesanded asuvad kindlaksmääratud keskkonnas mis on üldiselt süsteemile teada, siis mobiilsed robotid tegutsevad tundmatus keskkonnas.
Mobiilne robot on autonoomne siis, kui on võimeline navigeerima tundmatus keskkonnas ilma a priori teadmisteta. Süsteem peab selgusele jõudma: mida teha järgmisena? Kuidas saavutada soovitud roboti olek, mis on tarvilik ülesande täitmiseks? Milline peab süsteem olema, et täita käsku nagu näiteks:

- Kindla ajaga jõuda punkt A punkti B.
- Täita ülesanne kindlaks määratud energiakuluga
- Kaardistada keskkond ette antud täpsusega
- Valida alternatiivsete teekondade vahel (teekonna optimeerimine)
- Püsida teel ja vältida takistusi

Planeerimine ja kontroll viitavad samale asjale, ehk mida robot teeb, kuid süsteemi erinevatel tasanditel. Võib jaguneda kolmeks tasandiks, kus kõige ülemisel on kaalutlev arutlemine (*Deliberative reasoning*), siis trajektoori planeerimine ja madalama astme kontroll. Ajaliselt on määratud kaugele vaatav (*far time horizon*), lähitulevik, hetkeline. Sensorite kasutamine toimub siis kui plaan ebaõnnestub, monitoorida trajektoori, otse (*directly*) suhtlus keskkonnaga.
Sisemine määramatusega(*Uncertainty*) toimetulek on ühe intelligentse modernse roboti pea ülesanne. Ta teab ainult nii palju kui disainer ette on andnud ja mida sensorid parasjagu annavad. Kuid kuidas toime tulla nende määramatusega, mis tekib ülesannete täitmisel? Millist strateegiat tuleb

lahenduse jaoks rakendada? Kirjandus on ajalooliselt välja pakkunud mitmeid erinevaid lähenemisviise, kuidas selle määramatusega toime tulla

- Mudeli põhine (*hierarchial/deliberative*) – Eeldab täieliku teadmist keskkonnast. Süsteem toimib näe-mõtle-tegutse ringina.
- Käitumispõhine (*Reactive/behavior based*) – Süsteem eeldab, et ei tea keskkonnast midagi ning kasutab hetkelist sensorite andmeid navigeerimiseks. Süsteem eeldab samuti täitmiseks kindla protseduurilise lahenduse olemasolu. Töötab põhimõttel näe – tegutse.
- Hübriid – kasutab mõlema eelneva omadusi, kus planeerib ideaalse maailma puhul ja reageerib siis kui tekib töö sooritamisel tõrge (*run-time error*). Töötab põhimõttel näe, vajadusel mõtle, tegutse.



Joonis 2: Näe-mõtle-tegutse ring [10]

Kuna autonoomsuse tingimuse täitmisel ei saa robot eeldada, et on olemas kindel protseduuriline lahendus (näe-tegutse). Süsteemi disainides peab ei pea muretsema programmi sisesele arutluskäigule, vaid tegutseb kindla plaani järgi. Teised kaks valikut, mudeli põhine ning hübriidi liiguvad näe-mõtle-tegutse ringi järgi. Need vastavad oluliselt paremini mobiilse autonoomse kriteeriumite-

le.Näe-mõtle-tegutse süsteemi toimimiseks on kirjanduses eeldatud, et tarvis on nelja kindla süsteemiosa korrapärast koostööd ja mida on näha joonisel 2.

- Tajumine (*perception*) – sensorite abil keskkonna tajumine ja informatsiooni töötlemine mõtteka informatsiooni tarbeks.
- Lokaliseerimine – roboti poosi määramine keskkonnas
- Tunnetamine (*cognition*) – Eesmärgi saavutamise strateegia leidmine
- Liikumise kontroll (*motion control*) – sobiliku trajektoori arvutamine

Seda moodi annab mudelipõhisele süsteemile autonoomsuse „mõtle“ osa. Selle süsteemi mõtlemise osa koosneb ära toodud neljast osast. Sealt kõige olulisem on lokaliseerimine. Tähtis on ta selle pärast, et tajumisest tulnud informatsiooni abil leiab end ülesse ja tunnetamine ning liikumise kontroll kasutab oma arvutustes tema poolt antud muutujaid. Töö kvaliteedi määrab ära lokaliseerimise täpsus.

2.3 Kokkuvõte

Uuriti autonoomsuse ja roboti olemust, kus alguses vaatasime neid mõisteid eraldi ja lõpuks koos. Autonoomsuse täielikuks tagamiseks sai välja toodud seitse tingimust, Nende tingimuste eesmärk oli saavutada inimese-sarnane intelligents. Kuid robootika eesmärk ei ole inimese kehastamine maal, vaid kindla eesmärgistatud ülesande lahendamine. Seetõttu kui rääkida autonoomsest mobiilsest robotist, siis mõeldakse kui süsteemi, mis suudab operaatori poolt dikteeritud rida käskude täitmist iseseisvalt king kasutades enda küljes olevaid sensoreid keskkonna jälgimiseks. Kuna iga roboti ülesanne erineb teistest, siis sai ära toodud tabel, mille abil autonoomsuse taset määrata. Peatükki lõpus tuli selgusele, et navigeerimine on mobiilsete robotite korrapärase funktsioneerimise ja töö eesmärgi saavutamise alustala, mille täpsusest järelduvad teiste süsteemiosade tulemus.

3 NAVIGEERIMINE

Autonoomne navigeerimine nõuab mobiilse roboti süsteemilt palju. Selle õnnestumiseks (õnnestumiseks mõeldakse siin kohal algolekust lõppolekusse jõudmine) peavad tajumise (*perception*), *lokaliseerimise* (localization), tunnetamise (*cognition*) ja liikumise juhtimine (*motion control*) moodulid omavahel õnnestunult suhtlema. Tajumise moodul peab sensorite abil keskkonda tajuma ja andmetest välja filtreerima tähendusliku informatsiooni; lokaliseerimise ülesanne on positsiooni(x,y) määramine töökeskkonnas; tunnetamine tegeleb planeerimisega; ja liikumise juhtimine tegeleb täiturite käsutamisega.

3.1 Kus ma olen?

Leonard ja Durrant-Whyte on võtnud navigeerimis probleemi kokku kolme küsimusega: „Kus ma olen?“, „Kuhu ma lähen?“, „Kuidas ma sinna jõuan?“. Nendele küsimustele peab leidma järjekorras vastused, seetõttu ei saa tegeleda ennem sihtkohale jõudmise probleemiga, kui on selgeks tehtud oma hetkeline asukoht. Eelnevalt mainitud nelja mooduli eesmärk on leida nendele küsimustele iseseisvalt vastused.

Joonis 2 pool ära toodud süsteemist on lähiajal uuritud kõige enam lokaliseerimise moodulit ja mis aitab leida vastuse esimesele küsimusele „Kus ma olen?“. Selle huvi põhjus on tingitud eelkõige kahest põhjusest, et asuda teiste küsimuste juurde, peab esimene küsimus olema lõplikult lahendatud ning teiseks, et täna töötavate robotite nõudlus positsioneerimise järgi. Teine põhjus on aktuaalne selle tõttu, et tajumisest tulnud informatsiooni abil leiab robot end ülesse ja tunnetamine ning liikumise juhtimine kasutab oma arvutustes tema poolt antud muutujaid. Nende muutujate täpsus määrab ära lokaliseerimise täpsuse millest omakorda sõltub töö kvaliteet. Seega on tähtis sellele küsimusele vastata.

Kõigepealt tuleb vastata käsimusele kas lokaliseerimist on üldse vaja. Et minna punktist A punkti B ei ole ilmtingimata vaja kaarti. Autonoomse roboti jaoks on tähtis, et ei sõidaks takistustesse sisse ning tunneb ära kuna on jõudnud sihtkohta. Käitumis-põhised (behavior based) süsteemid eeldavad kindlat protseduurilise lahenduse olemasolu ja ei vaja kaarti ega sellel lokaliseerimist. Need kasutavad navigeerimiseks „vasakut seina järgiv“, „joont järgiv“ või mõnda muud sarnast lahendust. Selliste eelis on süsteemi lihtsus mille tõttu saab ülesannet täita kiirelt. Selliste süsteemide negatiivsed osad on, et robot on programmeeritud ainult spetsiaalseks, kindlaks määratud töökeskkonnas liikuma ja kui see keskkond on suur, siis sobivat süsteemi on keeruline leida. Kui isegi siis leitakse sobilik süsteem (mis enamjaolt koosneb mitmetest ala-süsteemidest, mis töötavad samaaegselt (*active/passive behavior*), ning on hästi optimeeritud, siis nende koostöötamises on latentsus(*latency*), mis teeb töö täitmise kohmakaks. Kõige suurem küsimus sellise lähenemise juures on see, et kuidas teada auto-

noomselt (ilma väliste abideta), et eesmärk on saavutatud?

Teine lähenemine lokaliseerimises kasutab navigeerimiseks kaarti – mis on esitatud keskkonna mudelina. Kaardipõhisel lähenemisel lokaliseerimisel kasutab süsteem sensoritelt saadud informatsiooni, koostab nende põhjal enda sisese kaardi (*internal representation*) ja uuendab oma arvamust (belief) positsiooni keskkonna suhtes. Selle meetodi eelis eelmise ees on suuresti selles, et tagab robotile töövõime erinevates keskkondades (kaart, kui roboti ja operaatori vaheline suhtlusmeedium). Kui roboti enda sisene kaart samas erineb reaalsest maailmast, siis robot ei pruugi ülesannet korrapäraselt sooritada.

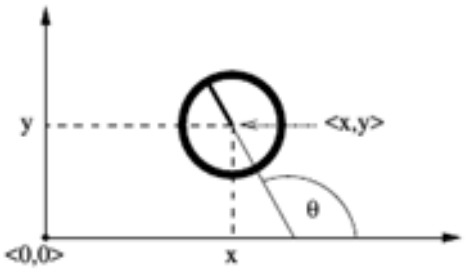
Tänu sellele arutlusele võib väita, et autonoomsel mobiilse roboti navigeerimise õnnestumiseks on tarvis lokaliseerida roboti asukoht keskkonnas. Leidsime, et võib saada ka ilma hakkama, aga sellisel juhul on roboti võimekus tugevalt piiratud.

3.2 Lokaliseerimine

Lokaliseerimise probleemi võib nimetada koordinaatide teisendamiseks (*coordinate transformation*). Kaart on iseloomustatud globaalses koordinaat süsteemis, mis ei ole otseselt seotud roboti poosiga. Lokaliseerimise protsess ongi leida kooskõla kaardi koordinaat süsteemi ja roboti lokaalse koordinaat süsteemi vahel. Selline informatsioonide omavaheline suhtlemine aitab robotil suhestada informatsioon kahe koordinaadi vahel.

Kinemaatika aitab kirjeldada kontrollmeetmete (*control actions*) roboti konfiguratsioonile. Jäiga keha konfiguratsioon on enamjaolt kirjeldatud 6 muutjuga – 3 dimensioonilise süsteemi koordinaadid ning 3 euleri nurka (roll, pitch yaw). Mobiilsetel robotite (siinkohal on ainult mõeldud tasastel pindadel liikuvad) poos (pose) on kolme muutujasse kokku pandud. Joonis 3 on näha, et koosneb kahe dimensioonilisest koordinaat-teljestikul ja orientatsioonist. Saame anda robot poosi vektor kujul (x, y, θ). Poos ilma orientatsioonita kutsutakse asukohaks (location).

Mobiilne robot võib lokaliseerida mitmeti. Kõige levinum viis lokaliseerida on kasutada väliseid sen-



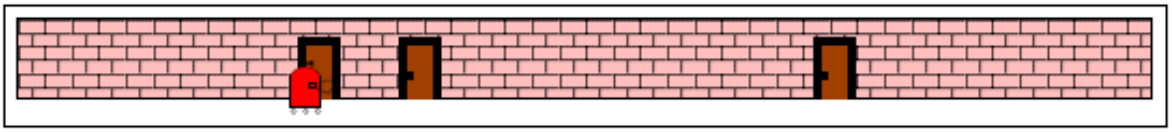
Joonis 3: Roboti poos koordinaat süsteemis [10]

soreid , tähiseid (*landmarks*) või „majakaid“ (*beacons*). Mehitamata õhusõidukid kasutavad positsioneerimiseks lisaks teistele abistavatele sensoritele (inertsisensor, visuaalne sensor jms.) veel satelliitnavigatsiooni süsteemi (GPS). Kuid see ei ole alati käepärast. Kui selline süsteem satub olukorda kus signaal on häiritud, siis on navigeerimine väga tugevalt häiritud. Sellisel juhul jääb lokaliseerimine ainuüksi abistavate sensoritele hoolde. Kuna nendel sensoritel võib kohata palju müra ning muid andmeid rikkuvaid iseärasusi, siis tekib kõrvale kalle (*drift*) ja süsteem on eksinud. GPS põhine oleku muutumine on väga paljudes ülesannetes ka ebapraktiline. Enamus mobiilsed robotid peavad navigeerima dünaamilistes keskkondades ning väga suurt täpsust nõudvates ülesannetes.
Lokaliseerimine ulatub aga oluliselt kaugemale kui teada oma asukohta maa pinnal. Robot mis peab suhtlema inimestega peab ära määrama suhtelise positsiooni inimese suhtes. Sellest koorub välja lokaliseerimise tähendus – koostada kaart ning ära määrata roboti poos selle kaardi suhtes.
Tänapäeval kasutatakse veel *Motion Capture Systemsit*, kus paigutatakse kaamerad keskkonda ja need omakorda edastavad informatsiooni robotitele. Selliste süsteemide puhul robot tugineb lokaliseerimisel välisele abile. Sellisel juhul robotid lokaliseerivad passiivselt ja majakad on aktiivsed. Süsteem vajab peale roboti ka keskkonna disainimist, kus tuleb üles seada vastav robotiväline süsteem. Tähiste järgi lokaliseerimisel paigaldatakse keskkonda kunstlikud tähised (*artificial landmarks*). Selline süsteemi koosneb kahest etapist. Kui tähis on sensoritele nähtav, siis robot suudab täpselt lokaliseerida ja uuendada oma positsiooni ilma kuhjunud mürata. Teises faasis, kus robot ei näe tähiseid, siis positsiooni ebamäärasus tõuseb ja langeb uuesti kuni uus, ära tuntav tähis on nähtav. Sellisel juhul kasutab robot pimenavigatsiooni (*dead reckoning*). Tähistepõhise navigeerimise kasutamist raskendab asjaolu, et korrapäraseks toimimiseks peab palju tähiseid keskkonda tehislikult rakendama. Mobiilsed robotid, mis soovivad navigeerida autonoomselt on huvitatud kaardipõhisest lokaliseerimisest, kus puuduvad välised sensorid või tähised ja tuleb ära kasutada robotitel kasutatavaid sensoreid. Samuti tähendab see seda, et roboti tarkvara peab asuma süsteemi sees (arvutamis võimsust on tarvis palju, kuna kaardi koostamine on väga koormav).

3.3 Tõenäosuslik kaardipõhine lokaliseerimine

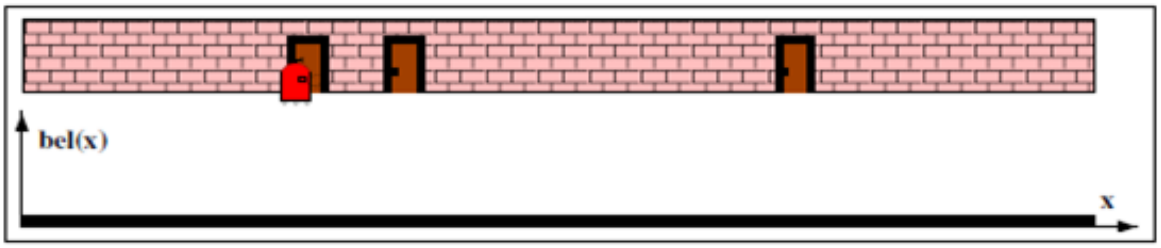
Mobiilse lokaliseerimise või positsiooni hindamise (*estimation*) peaprobleemiks on määrata ära roboti poos antud keskkonna mudeli suhtes. Seda nimetatakse positsiooni hindamiseks (*position estimation*). Probleemi teeb raskeks see, et robot ei saa positsiooni leida otse tänu robotil asuvate sensorite, seda tänu mõõdetes esinevatele mürale. Samuti ei ole olemas üksikut sensorit, mis annab poosi leidmiseks vajaliku informatsiooni . Tarvis on vaja mitut sensorit (juhul kui ei kasuta visuaalset sensorit – kaamerat) ning peab positsiooni leidmisesse arvestama ka aja.
Oletame, et meil on robot mis suudab ära tuvastada koridoris uksed ja tema ülesandeks on määrata

ära seal positsioon. Robotile on antud ühe dimensiooniline kaart (antud näite puhul koridor 3 uksega (uksi võib nimetada omadusteks(*features*) kaardil)). Joonisel on antud kaart näidatud. Robot alustab



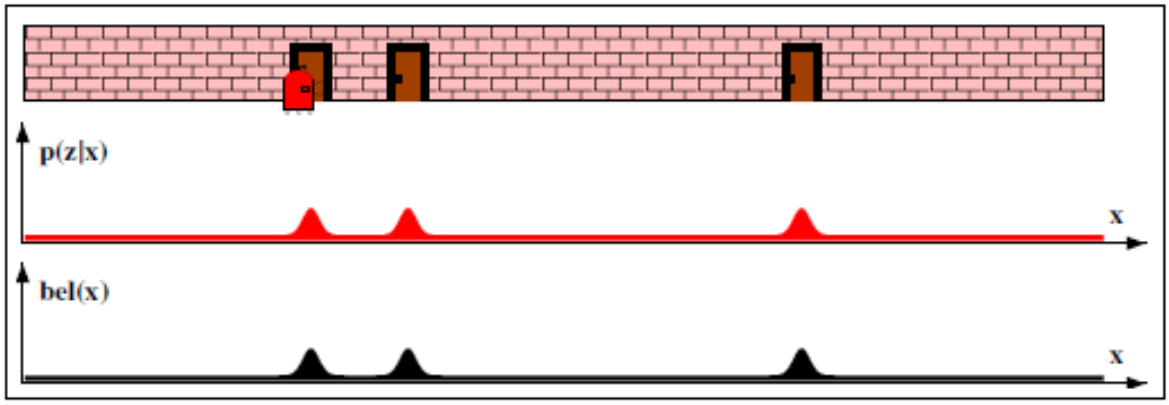
Joonis 4: Robot ja kaart [11]

liikumist. Kuna ta hakkas just ennast lokaliseerima, siis jaotus on ühtlane. Kui robot liigub edasi, siis samal ajal sensoritega jälgib seina. Kui robot jõuab ukse juurde, ning võtab mõõdu, siis roboti arvami-



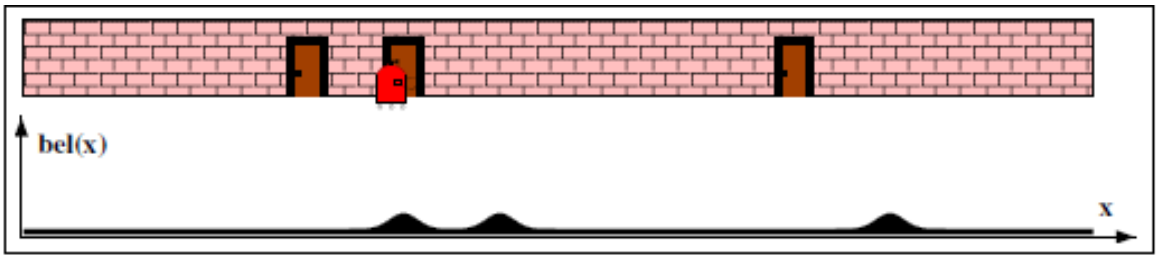
Joonis 5: Robot pole veel ust näinud [11]

ne (*belief*) on jaotatud võrdselt. Kuna ta jõuab esimese ukse juurde, siis süsteem ei milline täpsemalt see neist kolmest on seega kõikidel on võrdne tõenäosus. $p(z|x)$ on z tinglik tõenäousus läbi selle, et me teame x. Robot liigub edasi, ning arvamus liigub vastavalt temaga. Arvamuse ebamäärasus aga

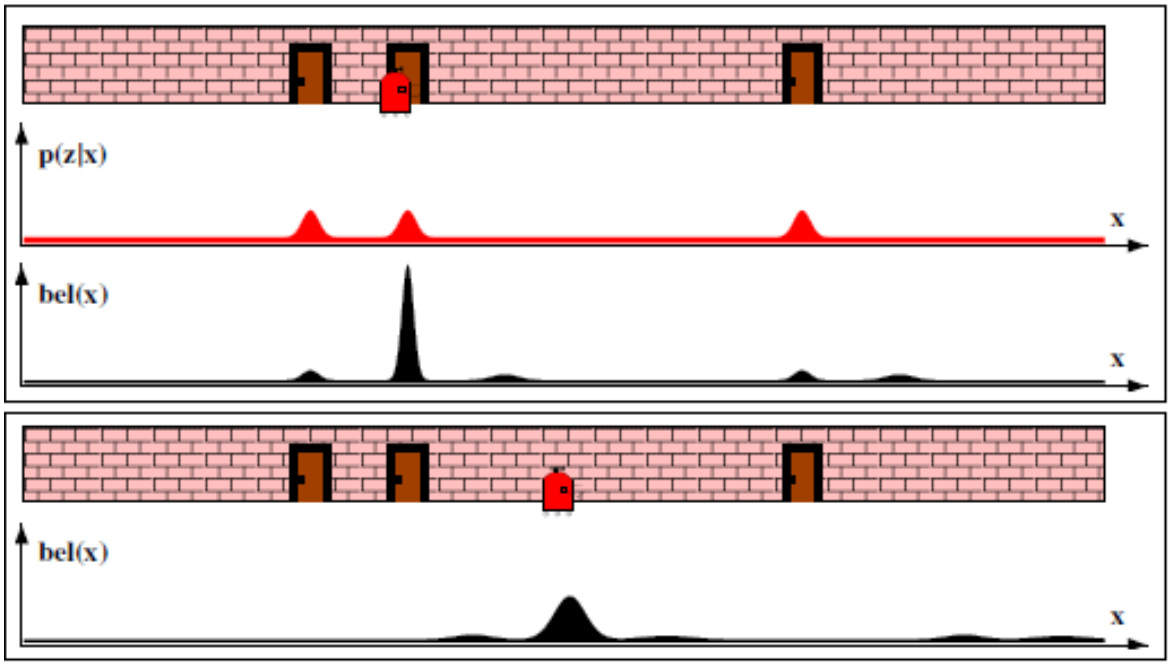


Joonis 6: Robot näeb esimest ust [11]

suureneb kuna odomeetri näit on väga ebamäärane (müra tõttu) ning uut ust ei ole veel tuvastatud Liikudes edasi robot näeb ust. Tänu sellele roboti arvamus oma positsioonist unimodaalne ning robot on ära lokaliseeritud kuna teiste uste tõenäosused on väiksemad ning süsteem arvestab kõige suurema tõenäosusega asukohta.



Joonis 7: Robot kahe ukse vahel [11]



Joonis 8: Robot tuvastas teise ukse ning roboti lokaliseerimine õnnestunud [11]

3.4 Kokkuvõte

Navigeerimise süsteem peab oskama vastata kolmele küsimusele: „Kus ma olen?“, „kuhu ma lähen?“, „Kuidas ma sinna jõuan?“. Kõige esimesena peab ära vastama esimesele küsimusele, et saaks asuda teiste kallale. Sellele küsimusele leiab vastuse läbi positsioneerimise.

4 SAMAAEGNE LOKALISEERIMINE JA KAARDI KOOSTAMINE

Tekkinud on kana-ja-muna probleem – kaarti on vaja lokaliseerimiseks ja poosi leidmiseks on vaja kaarti. Selle jaoks on vaja hinnata (*estimate*) roboti poosi ja tähiste (olenevalt süsteemist) paiknemist samaaegselt. Probleemi lahendamiseks on välja pakutud SLAM (simultaneous localization and mapping) – kaardi koostamine keskkonnast ja samaaegselt asukoha määramine koostataval kaardil. Esitame probleemi tõenäosuse vaatepunktist.

4.1 Probleem

Probleemi kirjeldamiseks tuleb defineerida

- $x_t = (X_x, X_y, X_o)^T$ on kolme dimensiooniline olekuvektor (*state vector*), mis iseloomustab roboti poosi ajal t.
    - $X_{o:t} = \{x_o, x_1, ..., x_t\} = \{X_{0:t-1}, x_t\}$ - Roboti asukoha ajalugu
- z_t– sensori mõõt ajal t võetud i tähisest
    - $Z_{0:t} = \{z_1, z_2, ..., z_t\} = \{Z_{0:t-1}, z_t\}$ kõik tähiste mõõted
- u_t – kontroll andmed (control data)
    - $U_{o:t} = \{u_1, u_2, ..., u_t\} = \{U_{o:t-1}, u_t\}$ - Roboti kontrollsisendite ajalugu
- m_i – keskkonna kaart
    - $m = \{m_1, m_2, ..., m_t\}$ – kõik tähised

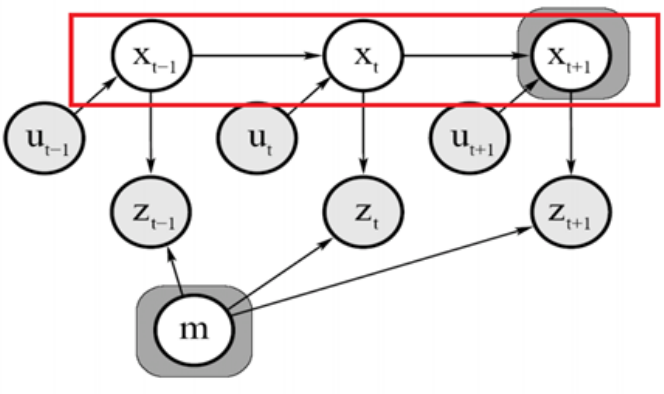
Kirjanduses on esile toodud kaks probleemi: täis (*full*) ja *online* SLAM. *Online* SLAM probleemi lahendamiseks on tarvis välja arvutada hetkelise positsioon x_t üle kaardi m.

$$p(x_t, m|Z_t, U_t) \tag{1}$$

Kui aga täis SLAM lahendamiseks on tarvilik välja arvutada posterior üle terve käidud tee, alates x_o.

$$p(X_t, m|Z_t, U_t) \tag{2}$$

See probleem püüab arvutada roboti olek x_t ja kaart m. Need on omakorda sõltuvuses eelnevatest mõõtudest z_t ja kontroll andmetest u_t, mis meil on teada mudelite kaudu. Roboti süsteem



Joonis 9: Online SLAM graafiline mudel ning punase ruuduga on tähistatud full SLAMI graafiline lahend [11]

peab lisaks veel olema varustatu mõõte ja liikumis mudeliga (*motion model*). Liikumis mudel seob odomeetri või kiiruse (velocity mõõtedu_t roboti asukohaga x_{t-1} ja x_t. Mõõte mudel omakorda seob sensori mõõted z_t kokku kaardi m ja roboti asukoha x_t. Mudelid on antud x_t tõenäosusjaotusena eeldades, et robot alustas teada olevast asukohast x_{t-1} ja odomeetri admetest u_t.

$$P(x_t|x_{t-1}, u_t) \tag{3}$$

$$P(z_t|x_t, m) \tag{4}$$

Need mudelid on mitte-deterministlikud, mis tähendab ükskõik mida me arvutame, ei ole absoluutne või üheselt määratud. See omadus kehtib terve SLAM probleemi kohta, mille lahendus ei ole absoluutne.

4.2 Kolm paradigmat

Kirjanduses on välja toodud kolm paradigmat lahendamaks eelnevalt esitatu probleemi lahendamiseks ning ülejäänud on tuletatud nendest või on nende vahelised hübriidid.

4.2.1 Kalmani filtrid

Kalmani filter kuulub Gaussi filtrite hulka ja mudelid peavad olema lineaarsed või lineariseeritud. Laiendatud Kalmani filter kasutab ühte oleku vektorit (*state vector*) ennustamaks roboti asukohta ja tähiste(*features*) asukohta keskkonnas, mille juures on müra kovariatsioonimaatriks. See maatriks esitab määramatust (*uncertainty*) koos roboti ja tähiste oleku hinnangute(estimation) vaheline korrelatsioon. Roboti liikumise vältel töö keskkonnast võetud mõõtete abil laiendatud Kalmani filter uuendab oleku vektorit ja kovariatsioonimaatriksit. Iga uue tähise leidmisel lisab süsteem uue oleku olemasolevasse oleku vektorisse, nii et kovaratsioonimaatriks kasvab kvartaalselt. Selline süsteem

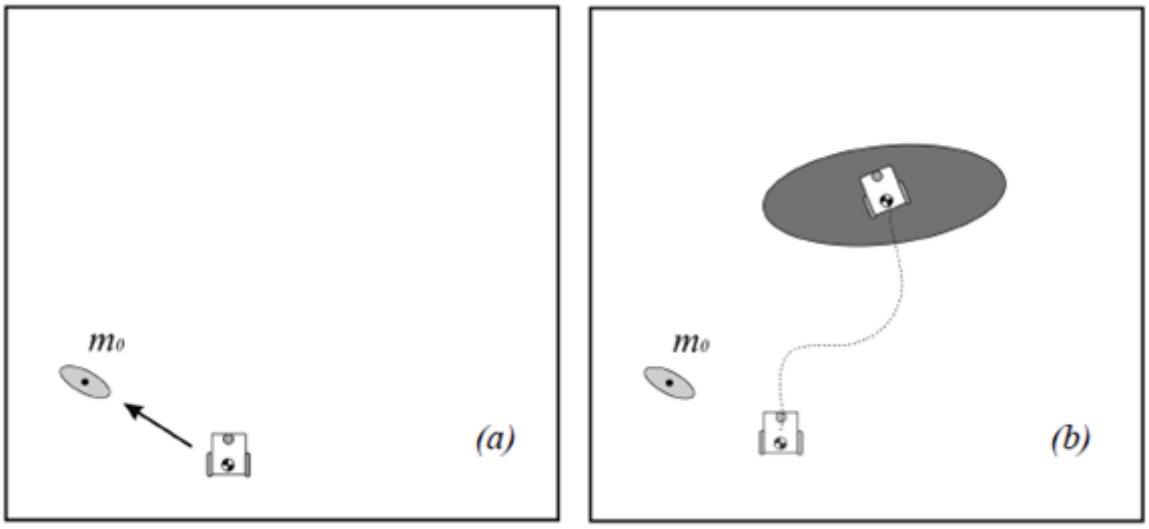
eeldab, et keskkonnast leitud tähised saab meetriliselt esitada ja roboti positsioon ning tähiste asukohad oleksid esitatud ruumiliselt.
EKF algoritm esitab roboti hinnangu(*estimate*) üle mitmemõõtelise Gaussi jaoutusena;

$$p(x_t, m|Z_t, U_t) = N(\mu_t, \Sigma_t) \tag{5}$$

Vektor $_t$ sisaldab parimat hinnangut(keskväärtus) omaenda positsiooni x_t ja tähiste asukoha üle. Vektori suurus on 3+2N, kus kolm muutujat seletab roboti asukohta ja 2N muutujat N tähise jaoks kaardil. Maatriks Σ_t on roboti hinnanguline müra vektoris μ_t. Vaatame kuidas toimib laiendatud kalmani filtri põhjal samaaegne kaardistamine ning lokaliseerimine. Roboti alustamis positsioonil eeldame, et positsiooni ebamäärasus on 0. Iga sammu tagant peab süsteem:

- Ennustama(*predict*) - Kuidas robot on liikunud
- Mõõtma(*Measure*)
- Uuendama(*Update*) - Roboti enda sisene väliskeskkonna üles tähendus

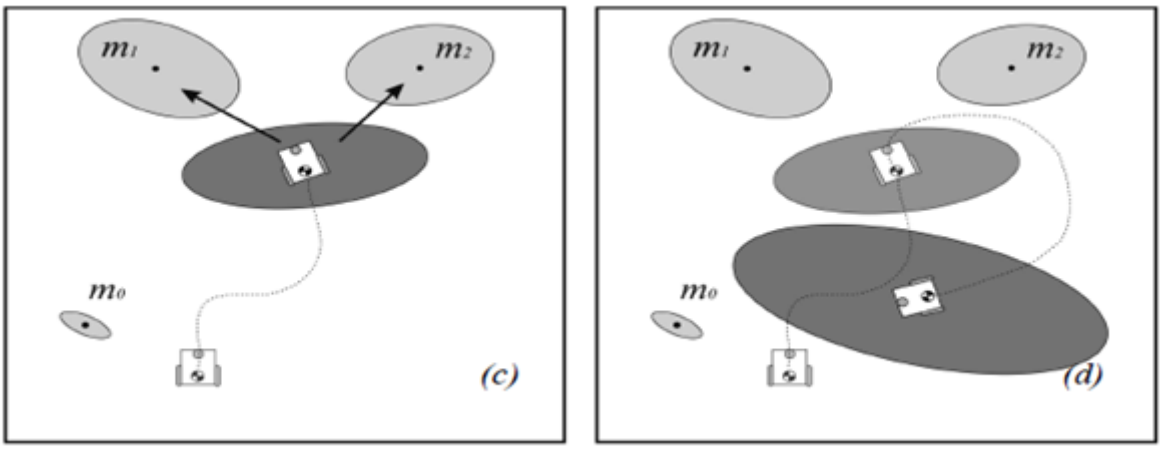
Robot leiab tähise ja kaardistab selle ebamääraselt kaardile. Ebamäärasus tuleneb sensori mürast, mida põhjustab mõõte mudel. Süsteem uuendab ennast. Robot liigub edasi. Seda tehes positsiooni ebamäärasus suureneb tänu odomeetritelt esineva mürale, täpsemini oleku mudel. Süsteem peab siin ennustama oma asukohta. Robot liigub uuesti edasi. Seda tehes positsiooni ebamäärasus suure-



Joonis 10: Robot näeb esimest tähist (a) ning liigub edasi (b) [10]

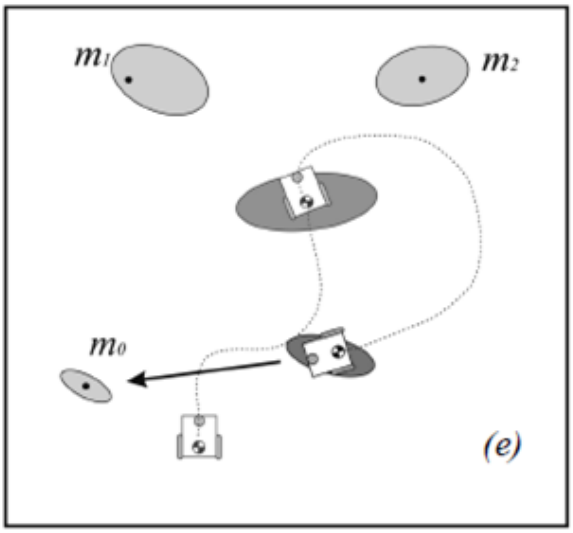
neb tänu odomeetritelt esineva mürale, täpsemini oleku mudel. Süsteem peab siin ennustama oma asukohta. Edasi liikudes tuvastab kahte tähist mida süsteem peab mõõtma esmakordselt. Need tähised tuleb nüüd kaardistada. Kaardile tekivad nad ebamääraselt, mis on nüüd tingitud mõõtmis müra

ja roboti poosi ebamäärasusest. Pärast kahe tähise mõõtmist robot liigub uuesti ja positsiooni ebamäärasus kasvab ehk süsteem ennustab oma kohta. Robot näeb uuesti esimest tähist ja tänu millele



Joonis 11: (c) näeb kahte uut tähist ning (b) liigub edasi [10]

asukoha ebamäärasus väheneb. Selle põhjuseks on see, et roboti algfaasis eeldasime, et positsiooni ebamäärasus on 0. Süsteemil saab ring täis (loop closure). Kuna filtrite abil lahendatakse online



Joonis 12: Näeb esimest tähist [10]

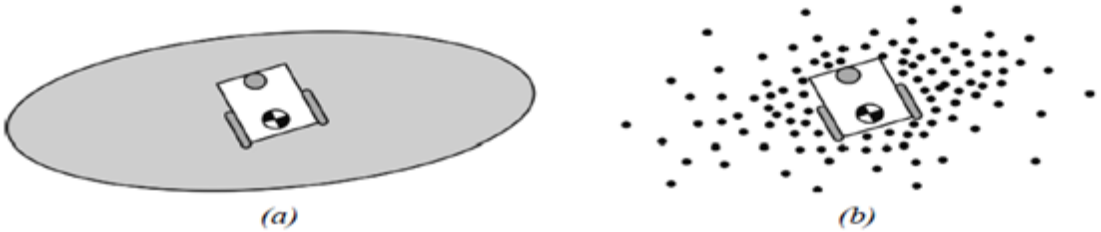
slam probleemi, siis pärast igat positsiooni muutumist kustutame eelnevad mõõtmed (tähiste asukoht eelneva positsiooni suhtes). Küll aga summeerime eelneva informatsiooni uue positsiooni suhtes kasutades oleku vektorit ja sellega seonduv kovaratsiooni vektor.

4.2.2 Osakeste filter

Kui Kalmani filtrite pea eelduseks oli see, et saavad ainult kasutada mudeleid, millel on normaaljaotus. Kui ei ole, siis laiendatud kalamani filteri algoritm püüab seda algul lineariseerida. Teine võimalus

on kasutada osakeste filtrit (*Particle filter*). Kui laiendatud Kalmani filtri põhine SLAM esitas tõenäosuse jaotuse roboti asukoha kohta parameetriliselt, mis on kahe dimensiooniline normaal jaotus 13. Osakeste filter esitab tõenäosuse jaotuse punktide kogumina mis on suvaliselt suvaliselt jaotatud ning punktid on normaaljaotuse keskel tihedalt ja hõreneb sellest kaugenedes. Iga osake on oleku hüpotees.

Osakese filter järgib kolme sammulist protseduuri. Esmalt võetakseproovid(*Samples*). Nendel on kin-



Joonis 13: (a) Kalmani filtri jaotus ning (b) osakeste jaotus [10]

del oma väärtus(sisaldab sensorite väärtusi). Kui kalmaan filtri põhine algoritm märgib ära ühe kindla Landmarki, siis osakeste filter on ette antud proovide arv N, mis on võetud keskkonnast. Kui robot liigub siis ideaalis peab ta ära tundma endale sobiva proovi. Teine samm (kui esimene on proovide võtmine) ongi roboti hetkelisele positsioonile leida sobivad variandid. Sobivale proovile sellisel juhul antakse numbriline väärtus - mida sobivam, seda suurem *(importance weights)*. Kui see protsess on tehtud siis jäävad valikusse ainult suurema väärtusega proovid ja väiksemad kukuvad valikust välja.Kolmanda sammuna on ette nähtud uute proovide võtmine. Uued proovid ei võeta suvalise jaotusega ruumis, vaid raskemate, allesjäänud proovide lähedusse (asendades vanad proovid).

4.2.3 Graafika põhised süsteemid

Kolmas meetod lahendamaks SLAM probleemi on kasutada graafika põhist algoritmi. Kui filtrite põhised (Kalmani ja osakeste filter) meetodid lahendasid online SLAM probleemi, siis graafika põhised algoritmid tegelevad *full* SLAM'iga. Graaf on struktuur, mida kasutatakse objektide vaheliste seoste kujutamiseks. Neid objekte nimetatakse graafi tippudeks (nodes/vertex) ning seoseid graafi servadeks(edge). SLAMis esindavad tipud roboti positsiooni mingil teatud ajahetkel mis on saadud positsiooni ja mõõtetulemustega ja tippude vahelised seosed, mis on esitatud servadega, postisoonide vahelist ruumilist seost. Servad tekivad mõõte tulemuste abil või roboti enese liikumise toimingust. Tundes ära varem mõõdetud alasi, tekivad seosed (mis tekivad odomeetri abil) mitte järjestikuliste pooside vahel. Kui graafika on lõplikult koostatud, siis tuleb määrata ära kõige tõenäosuslikum kaart pärast mida saab koostada kaardi. Taoline süsteem koosneb kahest osast, graafika ehitamisest, mida nimetatakse fron-endiks, mis tegeleb värskete andmete (*raw data*) töötlemisega. *Back-Endiks* on

graafika optimiseerimine, kus kõige levinum lahendamis viis on kasutada Gauss-newton vea miniseerimine.

4.2.4 Nägemispõhine

Välis maailma tunnetab ainult läbi kaamera. See teeb selle eriliseks. Teistel on liikumis ja mõõte mudelid eraldi välja arvestatud.

Tähised on äratuntavad elemendi struktuurid keskkonnas. Tavaliselt on need keskkonnast mõõdu teel saadud ja matemaatiliselt iseloomustatav. Head tähised on kergesti nähtavad ja lihtsalt eristatavad keskkonnas. Enamjaolt jagatakse tähised kahte valdkonda. Esimene on alam-astme tähised (*low-level features(geometric primitives*))need on jooned, punktid, nurgad, ääred, ringidalam-astme tähised on algandmete abstraktsioonid, see tõttu edastab vähem andmeid kuid see eest on kõrge tähiste eristusvõime. Seda kasutades loodetakse et ebamäärane ja tähtsusetu informatsioon filtreeritakse välja. Teised on kõrg-astme tähised (*objects*) nagu uksed, toolid, lauad. Kõrg-astme tähised pakuvad algandmete maksimaalset abstraktsiooni, eesmärgiks vähendada andmete suurust võimalikult palju kuid samal ajal pakkuda ka maksimaalset tähiste eristusvõimet. Suur risk on seotud sellega, et abstraktsiooni protsessis, võib filtreerimise käigus kaotsi minna tarvilik informatsioon.

Visuaalsel SLAM probleem on efektiivne video andmetöötlus, et kätte saade suurest hulgast andmetes kõige tähtsam. Raskus ongi pikalt toimiva SLAM saavutamisel, kuna aja kasvades kasvab ka arvutuse maht, mis lõpuks koormab liialt pardal olevat arvutit. Visual SLAM eelis on see, et Data association on peaaegu täiuslik võrreldes teiste sensoritega.

5 DROONIDE NAVIGEERIMINE ILMA GPS ÜHENDUSETA

Täna püüavad mitmed suurfirmad (DJI, Parrot) pakkuda klientidele mõistliku hinnaga droone. Neid disainides on eesmärgiks olnud see, et ükskõik mis sõiduoskusega inimene on võimeline sellega lendama. Seda saab tagada assisteerides süsteemisiseselt operaatorit, et temale jääks ainult lõpp sihtkoha määramine või videopildi vahendusel juhtkangiga (*joystick*) juhtimine. Süsteem assisteerib operaatorit eelkõige selles osas, et hoiab drooni õhus stabiilsena. Operaator võib soovi korral lihtsalt kontrolleri eemale panna, ning droon hoiab oma positsiooni. Kui ei oleks mingit tarka süsteemi vahel, kukus robot maha.

Siiski need robotid ei ole täiesti autonoomsed. Nagu töö teoreetilises osas järeldus, siis täiesti autonoomne on mobiilne robot sellisel juhul, kui suudab enda töö ülesandega hakkama saad kasutades ainult enda küljes olevaid sensoreid ja arvutuslikku võimsust. Sensorid ei tohi ka sõltuda välistest süsteemidest, nagu müüdavad droonid enamjaolt kasutavad GPS. Kuidas peab süsteemi arhitektuur välja nägema, et operaator dikteerib ainult algpositsiooni suhtes sihtkoha (sõida (x,y) asukohta) ning robot saavutab selle iseseisvalt?

Näite süsteemiks oleme võtnud DJI PHANTOM 4 drooni. Alguses vaatame kuidas algne süsteem välja näeb ning kuidas funktsioonid paraneksid kui kasutada mõnda iseseisvalt kaarti koostavat algoritmi ning süsteemi. Seejärel modifitseerime olemasolevat süsteemi, et selle külge saaks hakata rakendama mõnda *state-of-a-art* süsteeme ning algoritme.

5.1 DJI Phantom 4 süsteem

Täieliku autonoomsuse kasutamine mõne kaarti ehitava süsteemi, rikastaks DJI Phantom 4 Pro funktsionaalsust keskkondades, kus GPS ühendust ei ole või levi on halb . Samuti suudaks pakkuda sarnast lahendust, nagu GPS põhine *waypoints* (GPS abilt tarkvaras sihtkoha ette määramine) sõitmine, ainult siseruumides. Veel suudaks:

- Kaart aitab suurema täpsusega teekonna planeerimist ning tagaks paremini takistusi vältida tänu detailsemale keskkonna ülevaatele
- 2) Kaart kui eesmärk ise. Parem analüüs keskkonnast.

Praegu roboti operaatorid peavad isegi väljas sõites sõltuma välistingimuste poolt seatud piirangutele. GPS moodulid ei suuda kunagi tagada 100 protsendilist ühendust. Kui signaali levi on nõrk, siis selle lühiajalise puudumisel suudab sensorite fusiooni(*sensor fusion*) algoritmid positsiooni ja olekut kontrollida. Mida pikem on aeg, kus navigeerimisel ei saa GPS kasutad, seda suurem kõrvalekalle (*drift*)

tegelikkusega tekib ning droon eksib ära (kõrvalekalle suureneb ajas kasvavalt). Eksimine lõppeb sellega, et täitur järgivad kontroll süsteemi poolt koostatud juhiseid edasi ning navigeerib tundmatusse ja on lõplikult kadunud või sõitnud vastu takistust.
DJI pakub rakenduses signaali kadumisel või oleku ülevaate kadumisel mõningaid võimalusi päästa olukorda. Arvestatud on ka olukorda, kus kontrolleri ja roboti vaheline side kaob ära. Sellistel juhtudel on olemas *Return To Home* (RTH) nupp. Töötab GPS signaali poolt määratud waypointi juurde tagasi tulemisel kindlaks määratud kõrgusel (operaatori poolt) ning TTS süsteem ei tööta. Ehk kui metsas, linnas või vari katuse all on kokkupõrke oht väga suur.
Drooni juhtimisel on olemas kolme erineva funktsiooniga seadistust;

- Positsiooni režiim (*P-mode*) – Töötab siis, kui GPS signaal on tugev. Selles režiimis kasutab GPS ja takistuse tuvastamise süsteem (obstacle sensing system), et stabiliseerida, vältida takistusi või järgneda liikuvale kehale.

- Spordi režiim (*S-mode*) – Roboti parema manööverdamiseks on kontroller tehtud tundlikumaks. Positsioneerimiseks kasutab GPS ja takistuse tuvastamise süsteem on välja lülitatud.

- *A-mode* - Kui GPS ja takistuse tuvastamise süsteem ei ole saadaval, siis robot kasutab baromeetrit kõrguse hoidmiseks.

Positsiooni režiim kasutab takistuse tuvastamise süsteem (*obstacle sensing system*), et assisteerida operaatorit. Eesmärk on tuvastada teele jäävaid (roboti ees olevaid) takistusi, et juhtimissüsteemile nendest teada anda. Keskkonna tajumiseks kasutab kahte hüdrolokaatorit ja nelja monokulaarset nägemisandurit , mis asuvad masina põhjas. Välitingimustes tagab autonoomse navigeerimise õnnestumise ning kohtades, kus GPS signaal puudub, hoiab ära tajumispiirkonda jäävate takistustega kokkupõrke. Süsteemi töötamise tingimus on veel ka see, et valgus tingimused peavad olema head, ehk kaamera peab eristama keskkonnas tekstuure. Sellised olukorrad on näiteks kui on ühesuguse tekstuuriga pind(kivisein) kus objekte omavahel on raske erista. Samuti kui suurel kiirusel sõita ning kaamera pilt on hägus. Hüdrolokaatorid omakorda ei suuda tagada täpset informatsiooni signaali neelavate/segavate pindade kohal – veepind, paks vaip või dünaamiline keskkond. Probleem TTS on veel see, et sensoritel on väike töö ulatus – töötab kuni 10 meetrini, ning kui on pime, ei tööta üldse. Joonisel 14 on ära toodud kuidas süsteem kasutab ära enda sensoreid. Ülal oli ära toodud kolm süsteemi poolt pakutavad režiimi. Kuna kahel viimasel (*Sport- ja A-mode*) pakub süsteem kõige vähem tuge operaatorile, siis vaatame ainut esimest – P-mode.Kui robot on Positsiooni režiimis, siis droon aitab saavutatud oleku jõudmiseni üsna palju kaasa. Selles režiimis manuaalselt juhtides saab liikuda 3 dimensioonis – x,y,z. *Yaw, Pitch, Roll* juhtimisega tegelev süsteem. Kui aga operaator on kontrolleris sisestanud täpse GPS koordinaadi või isegi ette joonistanud kindla trajektoori, siis suudab sihtpunkti

Joonis 14: Sensorite kasutamie

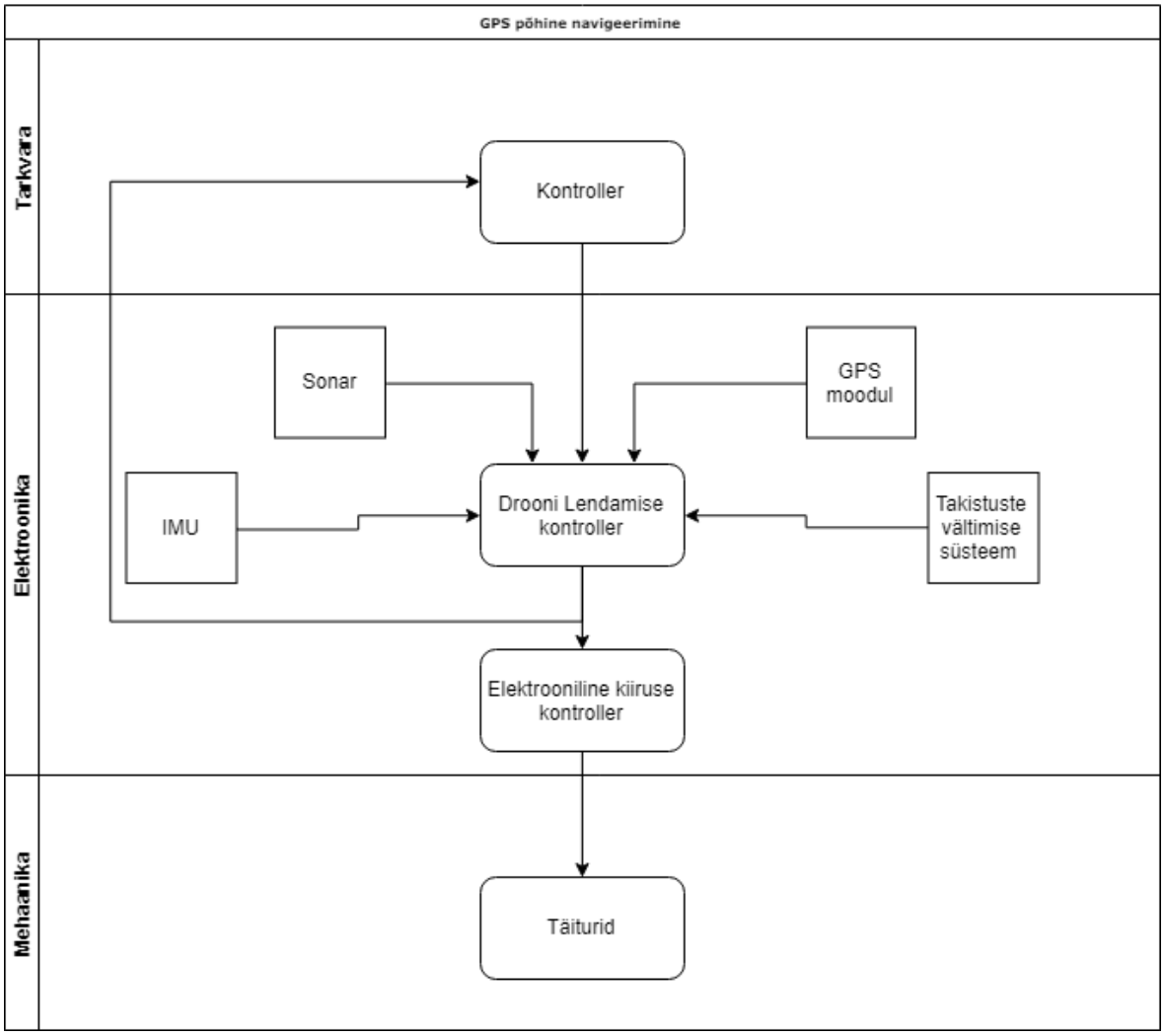
jõuda iseseisvalt. Teel saavutatud oleku suunas aitavad DJI loodud takistuste vältimise süsteem ümber takistuste põigelda.

Joonisel 15 on ära näidatuid süsteemi arhitektuur. Kontrollerist annab süsteem navigatsiooni tarbeks vajaliku informatsiooni Drooni Lendamise kontrollerile (*Drone Flight Controller*) . See on siinjuhul süsteemi aju. Lendamise kontrolleri sisendisse tulevad lisaks mainitule veel parda peal olevad sensorid – IMU, Sonar, nägemisandureid kasutatav takistuste vältimise süsteem. Samuti saabub sisendisse ka GPS signaali vastuvõtja andmed. Väljund suundub kas tagasi kontrollerisse ja Elektroonilise kiiruse kontrollerisse (*Electronic Speed Controller*). Kiiruse kontrolleri ülesanne on kontrollida ning reguleerida elektri mootorite kiirust.

Täieliku autonoomsuse saavutamiseks üheks võimaluseks on, et süsteem kasutab enda koostatud kaarti. See kaardistab alates ajahetkest t1 enda suhtes objektid koostatavasse kaardi mudelisse. Samuti tunneb sellisel juhul oma positsiooni igal ajahetkel, siis on ka eelnenud trajektoor teada. Sellisel juhuks edestaks uus süsteem ka praegust RTH nuppu, kus algpunkti saabumise käsu saabumisel saab planeerida kõige optimaalsema tee tagasi koju - tervelt.

Arvatav põhjus, miks täna täiesti autonoomsed süsteemid ei ole kasutusel, seisneb selle rakendamise raskuses kuna drooni kontroll süsteem peab kontrollima 6 DoF (*degrees of freedom*). Kui seda võrrelda maapinnal tasasel maal ratastel liikuvad robotit, kus saab poosi seletada 3 muutujaga – x,y ning orientatsioon θ, kus oleku hindamine on saavutanud kasutamiseks piisava täpsuse[1] . Teine ja veel suurem probleem on keskkonna tuvastamine enda ümber kaameraga.

Kuna droon on väga manööverdamis võimeline ning kiire, siis kaamera hägususe tõttu on tähiste/de-



Joonis 15: GPS põhine navigeerimine

tailide tuvastamine raskendatud. Kaamera tööd veel raskendab keskkond kus peab liikuma – dünaamiline ning kiiresti vahelduv.

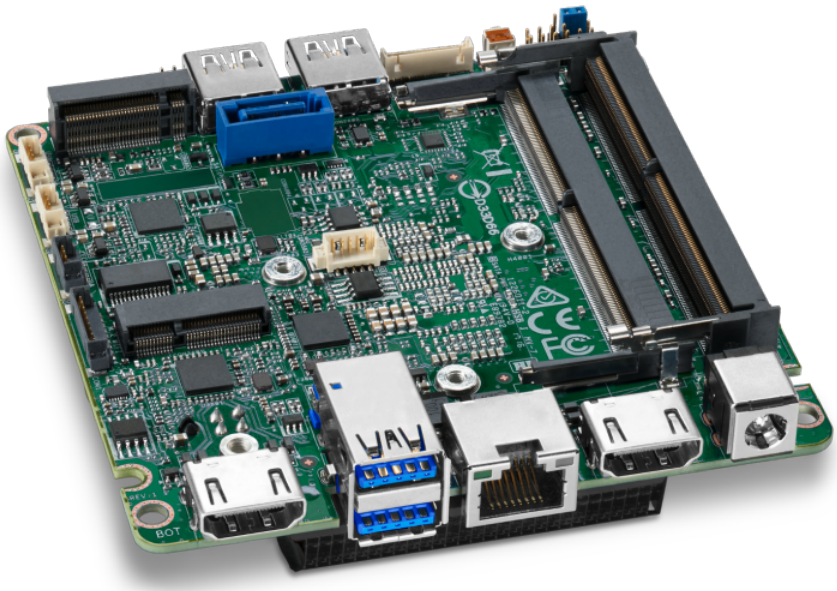
5.2 Autonoomne navigeerimine ilma GPS moodulita

Muuta DJI Phantom robot autonoomseks mobiilseks robotiks, tuleb muuta süsteemi – GPS moodul on ebavajalik ning positsioneerimine peab toimuma iseseisvalt. Vajaliku iseseisvuse saavutamiseks tuleb mõningal määral muuta nii riistvara kui süsteemi arhitektuuri. Kuna navigeerimine tundmatus keskkonnas on väga keeruline, sest peale selle, et süsteemil peab olema hea oleku ennustus ja kontroll, peab robot ehitama täpset kaarti liikumise ajal ning planeerima eduka trajektoori algpunktist sihtpunkti kasutama reaal-ajas koostatavat kaarti ning uuendama trajektoori koos kaardi uuenemisega, kus sensorid seda võimaldavad.

Esmalt tuleb roboti külge panema arvuti, et arvutusliku võimsust suurendada kaardi koostamiseks ning roboti oleku ning trajektoori arvutamises. Selleks otstarbeks pakub näiteks Intel koostatava süs-

teemi jaoks suure võimsuse ja väikeste dimensiooniga arvutit. Protsessori ning GPU valikul peab lähtuma sellest, mida suurem võimsus, seda efektiivsem sõidu ajal see on. Kuna silma peab peal hoidma ka hindadel, siis osutuks süsteemis valituks Intel i3-7100U protsessor.

Sensoreid on meil tarvis oleku hindamiseks ja kaardistamiseks. Oleku hindamiseks enamjaolt kasu-



Joonis 16: Intel NUC plaat NUC713DNBE

tatakse droonides kas nägemisandurite või LIDAR (*Light Detection and Ranging*) sensorite põhiseks. Kirjanduses on esile toodud, et nägemisandurite kasutamine on efektiivsem ennustamaks roboti olekut.

Kaardistamise jaoks sensori valik on probleemne. Kuna nägemispõhiste andurite järgi kaardistamine ei ole kas piisavalt täpne ja suurt arvutusvõimsust nõudev mille tõttu ei ole suuteline jooksma reaalajas. Paremini kõlbab selleks LIDAR sensor ning sellel juhul ei saa mitte ilma selleta hakkama, kuna süsteem vajab ümbruskonna kaardistamiseks 3D informatsiooni. Kuna LIDAR sensorite nõudmised on väga suured (täpsus), siis hinnad on samuti kõrged. Selleks sai valitud Hokuyo UST-10LX.

Kui esialgne soov oli valmis ehitada kasutades ainult DJI Phantom 4 peal asuvaid komponente, siis süsteemi disainides ja eesmärki silmas pidades ei ole see võimalik. Joonis 18 on ära toodud internetis leiduvad pakkumised soovitud komponentidele. 3D Lidar sensori puhul oleks võimalik raha säästa tehes ise lidari sensori all ringi käiv servo mootorit kasutav alus. Kuid sellisel juhul oleks sensori müra ning mõõte täpsused väga kaheldava väärtusega. Roboti navigeerimise edukus sõltub kaardi täpsusest, mis omakorda sõltub tugevalt sensori täpsusest. Sellisel juhul süsteemi õnnestumiseks on tarvis kvaliteetset sensorit.

Joonis 17: Hokuyo UST-10LX

Pakkuja	Mudel	Hind
www.robotshop.com	Hokuyo UST-10LX	1437 €
www.amazon.com	NUC713DNBE	230 €
		1667 €

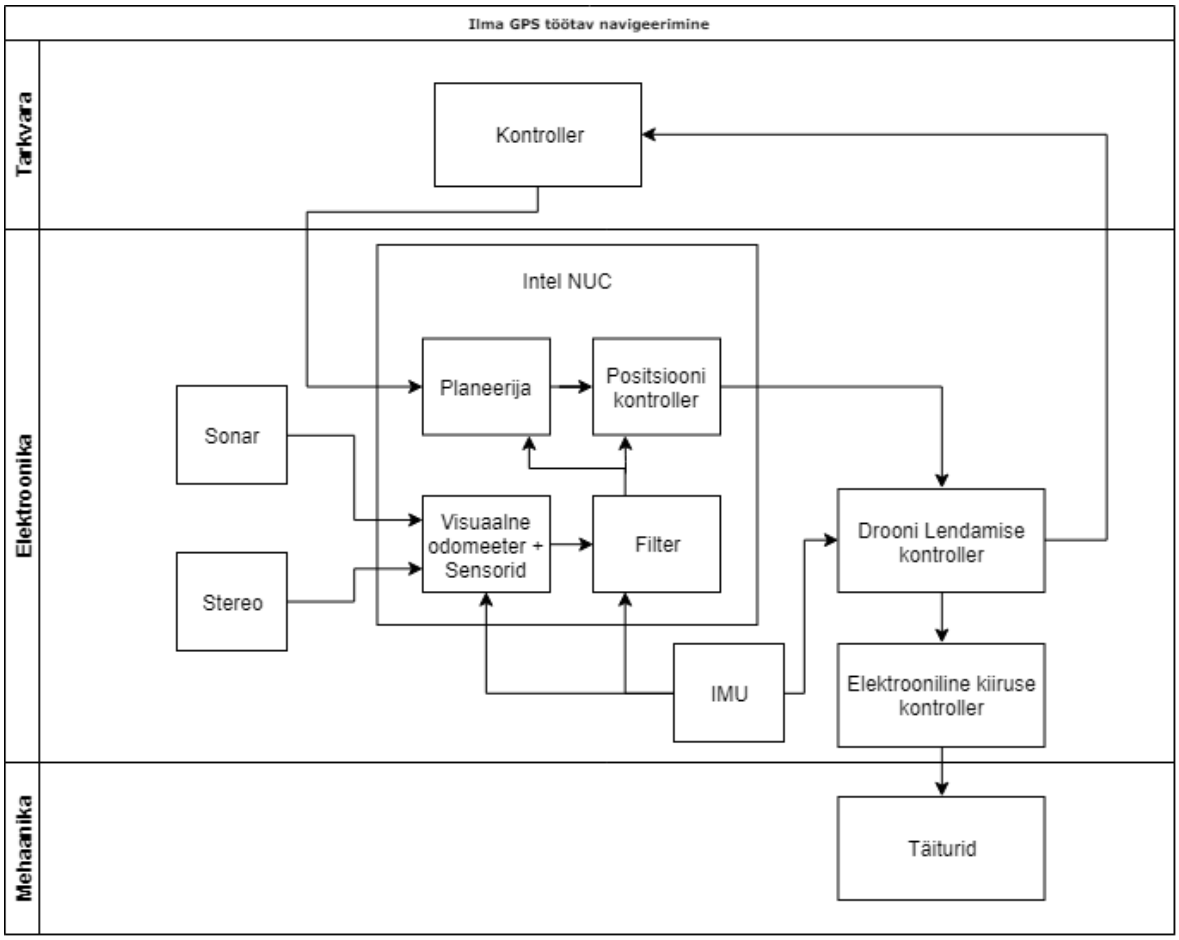
Joonis 18: Uute komponentide hind

5.2.1 Süsteemi arhitektuur

Autonoomse navigeerimise tarbeks on vajalik süsteemi kohandada vastavalt vajadusele. Tarkvara komponendid süsteemis jagunevad nelja kategooriasse[7] hindamine (*estimation*), kontroll, kaardistamine ning planeerimine. Joonis 19 on näha, kuidas uus modifitseeritud süsteem näeb välja.

Kõigepealt on meil vaja hinnata roboti poosi 6 DoF, milleks kasutame odomeetri mudelit. Selle võib kätte saada lihtsalt kasutades andmeid IMUlt, kuid see on ebatäpne ning mida aeg edasi seda rohkem ebatäpsemaks ta läheb ning roboti reaalne asukoht jääb tundmatuks. Selle tarbeks on kirjandusest [9] pakutud välja odomeetri mudel kätte saada läbi nägemissensori, mille abil määratakse ära positsioon ja orientatsioon analüüsides järjestikulisi kaadreid. Oma süsteemis kasutame selle määramiseks Forster [3] poolt pakutud *Semi-direct Visual Odometry* raamistikku. See lähenemine pakub mulle kaadrite vahelise liikumise, kus kõigepealt joondab pildid omavahel omaduste (*Features*) järgi – näiteks sirgjooned või nurgad pildil, ning siis kasutab optimiseerimiseks Bundle adjustmenti. Nagu testidest välja on tulnud, et kiiruse ja efektiivsuse suhe on SVO parem teistest välja pakutud lahendustest.

Odomeetri mudelile lisaks tuleb täpsema oleku saamiseks kasutada teisi pardal olevate sensorite mõõte. Kuna iga sensor annab pooliku ülevaate roboti olekust, siis tuleb neid iga olekut muutva põhjuse tarbeks olema mingi sensori või mudeli mõõt. Kõik need mõõdud omakorda ühendatakse kokku



Joonis 19: Ilma GPS navigeerimise süsteem

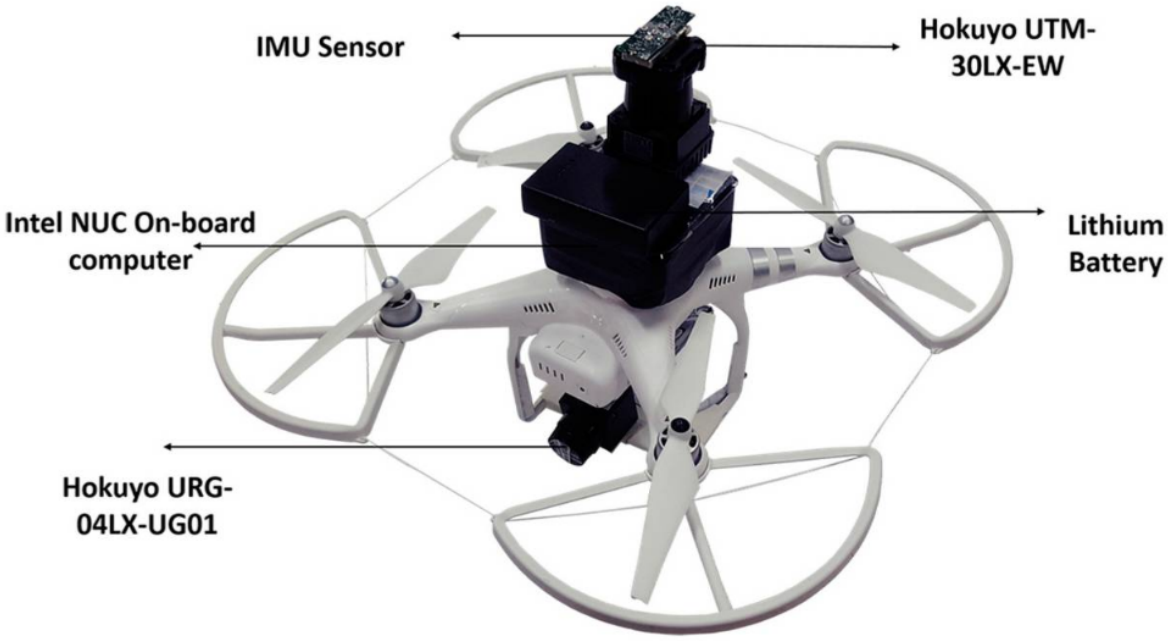
(*Sensor Fusion*) läbi mingi filtri. Meie kasutame selleks Laiendatud Kalmani Filtrit. Oleku vektor meie süsteemi puhul oleks [7] tuginedes:

$$x = [p^T, \dot{p}^T, \theta, \alpha, \gamma, b_a^T, b_w^T]^T \tag{6}$$

Kus p on globaalse kaardil roboti positsioon, $\dot{p}$ globaalse kaardi kiirus. α ,θ , γ on vastavad euleri nurgad. Samuti asub oleku võrrandis IMU sensorite mõõdud b_a ning b_w , mis on vastavalt kiirendusmõõturi ja güroskoobi kõrvalekalle (*bias*). Süsteemi kontrolli tarbeks on jagatud kaheks – esimeses, sisemises (Drooni lendamise kontrolleris) kontrollib nurk- ja orientatsioon kiirust. Väline, ehk süsteem mis asub NUC arvutis, kontrollib positsiooni ning lineaarset roboti kiirust. Kui välimise süsteemi positsiooni kontroller saab käsuks kindla soovitud oleku mõõted (positsoon, kiirus, kiirendus) planeerijalt ning sellele lisaks kasutab filtrilt poolt arvutatud hetke oleku hinnangu, siis arvutab positsiooni kontroller sobiliku kiiruse, orientatsiooni mille saadab edasi Drooni lendamise kontrollerisse. See omakorda arvutab välja vaja minema propellerite tõukejõu (*Thrust*) ning elektroonilise kiiruse kontroller (*Electronic Speed Controller*) juhib nende abil mootoreid.

Eelnevalt seletatud kontroller tegeleb navigatsiooni süsteemi poolt saadetud juhiste järgi. Planeeri-

jasse saadetakse alguses soovitud lõppsihtkoht. Planeerija genereerib sobiliku teekonna kasutades kaarti (mis on koostatud olekust ja sensorite mõõtmetest), ning saadab selle trajektoori generaatorisse mis leiab hetkel teada olevale ümbruskonna järgi trajektoori. Et üleval hoida (koostada ning uuendada) globaalset kaarti keskkonnast, on mobiilse roboti jaoks arvutuslikult liiga raske ning esineb rohkesti müra. Sellisel juhul on mõttekam kasutada väiksemat, lokaalset kaarti, mis asub LIDARI mõõte ajahetkel keskkonnas. Lokaalsest kaardist tuleneva informatsiooni järgi saab koostada globaalse kaardi, mis kasutab lokaalse kaardi andmeid, et koostada enda kaarti.



Joonis 20: Eeldatav roboti väljanägemine [5]

Joonisel 20 on ära toodud Ajay Kumar[5] poolt ehitatud robot, kus on eelneva aasta mudelile, DJI phantom 3 peale rakendatud sarnane süsteem ja nende uurimisest võib järeldada, et ka käesoleva töö pakutud drooni keha suudab lisaraskusi kanda.

5.3 Süsteemi kokkuvõte

Disaini eesmärk oli muuta DJI Phantom 4 droon täiesti autonoomseks ja iseseisvalt navigeeritavaks. Senine lahendus kasutab navigeerimiseks kontakti kontrolleriga ning vajab GPS ühendust. Praegune süsteem kasutab oma oleku leidmiseks olekuvektori GPS tulevat signaali. Nagu töö teoreetilises osas seletatud, on mobiilne robot autonoomne alles siis, kui saab ilma väliste sensori ning operaatori abita navigeeritud. Välja näidatud süsteem kasutab nüüd oleku vektoris ainult endast sõltuvate sensorite mõõdud. Sellise omaduse saavutamiseks pakuti välja üldine süsteemi arhitektuur, mis võtab arvesse olemasolevaid komponente ning vajadusel lisada juurde uusi. Süsteem kasutab põhiliselt kahte alam-

süsteemi osa, mis koosneb sise- ja välissüsteemis. Esimese peaülessane on kasutada välissüsteemist arvutatavad parameetreid roboti orientatsiooni ning nurkkiiruse kontrollimiseks. Välissüsteemi ülesanne on hoolitseda täpse roboti oleku hindamisega (state estimation), Koostada kaart ning kasutada neid trajektoori leidmisel. Lõppsihtkoht on kontrolleri poolt süsteemi sisse määratud ja lõppeesmärk on sinna edukalt jõuda.

6 KOKKUVÕTE

Lõputöö eesmärgiks oli leida süsteemne lahendus autonoomsele robotile. Esimeses peatükkis oli uurimise all autonoomsus ja roboti olemus, kus algus vaadati neid mõisteid eraldi ja lõpuks vaadati neid koos. Autonoomsuse täielikuks tagamiseks sai välja toodud seitse tingimust. Nende tingimuste eesmärk oli saavutada inimese-sarnane intelligents. Kuid robootika eesmärk ei ole inimese kehastamine maal, vaid kindla eesmärgistatud ülesande lahendamisega. Seetõttu kui rääkida autonoomsest mobiilsest robotist, siis mõeldakse kui süsteemi, mis suudab operaatori poolt dikteeritud rida käskude täitmist iseseisvalt ning kasutades enda küljes olevaid sensoreid keskkonna jälgimiseks. Kuna iga ülesanne on teistest erinev, siis sai ära toodud tabel, mille abil autonoomsuse taset määrata. Peatükki lõpus tuli selgusele, et navigeerimine on mobiilsete robotite korrapärase funktsioneerimise ja töö eesmärgi saavutamise alustala.

Teises peatükkis lahati navigeerimise süsteemi tervikuna. Selle õnnestumiseks peavad tajumise, lokaliseerimise, tunnetamise ja liikumise juhtimise moodulid omavahel õnnestunult suhtlema. Ideaalmaailmas peab sellistest moodulitest koosnev süsteem oskama järjest vastata kolmele küsimusele: „Kus ma olen?“, „kuhu ma lähen?“, „kuidas ma sinna jõuan?“. Kõik eelnevad ei ole võrdse kaaluga, kus tähtsuse määrab ära küsimuse järjekord ja edasi arutati esimesele küsimusele vastuse andmist. Leida lahendus küsimusele „Kus ma olen?“, tuleb süsteemil selgeks teha enda poos. Matemaatiliselt saab anda roboti poosi vektor kujul tasapinnalisel robotil (x, y, θ), mis koosneb kahe dimensioonilest koordinaat-teljestiku koordinaadist ja orientatsioonist. Kuid sellest ainuüksi ei piisa, vaid samuti on tarvis keskkonna mudelit(kaarti), et poosiga midagi peale hakata. Kolmandas peatükkis toodi kirjanduse abil lahendus eelnevale probleemile - SLAM (*simultaneous localization and mapping*) – kaardi koostamine keskkonnast ja samaaegselt asukoha määramine koostataval kaardil. Selle lahendamiseks on välja pakutud 3 paradigmat: Kalmani filtri-, osakese filtri-, graafika põhine.

Viimases peatükkis sai eelnevalt seletatud informatsiooni abil tehtud süsteemi kavand autonoomse väli- ja teenindusrobotite tarbeks. Valiti vastavalt tingimustele tajurid ja süsteemi omadused. Vaatluse all oli DJI PHANTOM droon, mis lokaliseerib ennast töö ajal GPS abil. Eesmärk oli ära näidata, kuidas sellised välised positsioneerimis süsteemid asendada ära ainult roboti peal olevate sensorite ja arvutus võimsusega - et saavutada autonoomne robot. Üks viis seda saavutada on roboti siseselt ehitada enda ümbruskonnast kaart ning robotit positsioneerida selle suhtes. Süsteemi ideaalsel toimimisel on robotil enda asukoha üle alati teadmine olemas.

7 SUMMARY

The aim of the thesis was to find a systematic solution for an autonomous robot. In the first chapter, the study was about autonomy and the nature of the robot, where at the beginning I looked at these concepts separately and finally looked at them together. Seven conditions were issued to ensure full autonomy. The purpose of these conditions were to achieve a robot with similar intelligence as humans. But the purpose of robotics is not the embodiment of man in the earth, but the solution of a determined task. Therefore, when speaking of an autonomous mobile robot, it is meant as a system that can independently handle the commands of the use the on-board sensors to monitor the environment. Since each task is different from each other, the table below provides the basis for determining the degree of autonomy. At the end of the chapter, I explained that navigation is the cornerstone for the functioning of mobile robots and the purpose of the work. The second chapter explores the navigation system as a whole. In order to succeed, the modules of perception, localization, perception and movement must communicate with each other successfully. In an ideal world, a system consisting of such modules must consistently answer three questions: ”Where am I?”, ”Where do I go?”, ”How do I get there?”. All the previous ones are not of equal weight, the importance of which determines the order of the question and chapter discusses the answer to the first question. Finding a solution to the question ”Where am I?”, system has to find a way to find its position. Mathematically, a robot's posture vector can be given in the form of a (for a plane robot) (x, y, θ), which consists of the coordinate and orientation of two dimensional coordinate system. However, this alone is not enough - environmental model (map) is also needed. In the third chapter, the literature was used to solve the previously discussed problem - SLAM simultaneous localization and mapping. Three paradigms have been proposed to solve this: Kalman filter, particle filter, graphic based.
In the last chapter, DJI PHANTOM 4 robot system was discussed. Robot current system architecture enables to get robot position relative to the surrouning enviroment with the help of external sensor system and also external computing power. With the help of the literature a solution framwork was proposed to navigate fully autonomously – finding the robot pose using only on-board capabilities.

8 KASUTATUD KIRJANDUSE LOETELU

[1] C. Cadena et al. “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age”. *IEEE Transactions on Robotics* 32.6 (2016), l. 1309–1332.

[2] Howie M Choset. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[3] C. Forster et al. “SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems”. *IEEE Transactions on Robotics* 33.2 (2017), l. 249–265.

[4] Félix Ingrand ja Malik Ghallab. “Deliberation for autonomous robots: A survey”. *Artificial Intelligence* 247 (juuni 2017), pp.10–44.

[5] G. Ajay Kumar et al. “A LiDAR and IMU Integrated Indoor Navigation System for UAVs and Its Application in Real-Time Pipeline Classification”. Teoksessa: *Sensors*. 2017.

[6] Steven M LaValle. *Planning algorithms*. Cambridge university press, 2006.

[7] Kartik Mohta et al. “Fast, autonomous flight in GPS-denied and cluttered environments”. *Journal of Field Robotics* 35.1 (2018), l. 101–120.

[8] Gerasimos G Rigatos. “Extended Kalman and Particle Filtering for sensor fusion in motion control of mobile robots”. *Mathematics and computers in simulation* 81.3 (2010), l. 590–607.

[9] D. Scaramuzza ja F. Fraundorfer. “Visual Odometry [Tutorial]”. *IEEE Robotics Automation Magazine* 18.4 (2011), l. 80–92.

[10] Roland Siegwart ja Illah R. Nourbakhsh. *Introduction to Autonomous Mobile Robots*. Scituate, MA, USA: Bradford Company, 2004.

[11] Sebastian Thrun, Wolfram Burgard ja Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.