Kandidatavhandling

Kandidatprogrammet i Datavetenskap

Samtidig lokalisering och kartläggning med datorseende

Sebastian Sergelius

12.12.2020

Matematisk-naturvetenskapliga fakulteten

Helsingfors universitet

Handledare

Doktor Jeremias Berg

Examinatorer

Docent Patrik Floréen

Kontaktinformation

PB 68 (Pehr Kalms gata 5)
00014 Helsingfors universitet, Finland

E-postadress: info@cs.helsinki.fi
URL: http://www.cs.helsinki.fi

HELSINGIN YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Study programme
Matematisk-naturvetenskapliga fakulteten	Kandidatprogrammet i Datavetenskap

Tekijä — Författare — Author

Sebastian Sergelius

Työn nimi — Arbetets titel — Title

Samtidig lokalisering och kartläggning med datorseende

Ohjaajat — Handledare — Supervisors

Doktor Jeremias Berg

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Kandidatavhandling	12.12.2020	20 sidor

Tiivistelmä — Referat — Abstract

En drönare, det vill säga en flygande robot, har använts till att skaffa visuell information från katastrofområden och till att förhindra industriolyckor. För att vara användbara måste drönare kunna navigera autonomt i miljöer. Navigering är ett forskningsområde som har utvecklats under de senaste årtionden. För att en drönare skulle kunna navigera autonomt måste den vara medveten om sitt läge, sin omgivning, och hastighet. En viktig del i autonom navigering för drönare är att den kan samtidigt kartlägga omgivningen och lokalisera sig själv i den.

Visionsbaserad navigering med kameror är ett lovande forskningsområde inom robotnavigering. Med hjälp av datorer som används i visionsbaserad navigering går det att kartlägga omgivningen och lokalisera sig själv i den. Kameror används för att extrahera egenskaper ur bilder och de är inte beroende på en utomstående entitet för att skaffa data om omgivningen. Syftet med denna avhandling är att undersöka visionsbaserad navigering från drönarens perspektiv med fokus på samtidig kartläggning och lokalisering och hur detta problem har angripits.

ACM Computing Classification System (CCS)
Computer systems organization → Embedded and cyber-physical systems
→ Robotics → Robotic autonomy

Avainsanat — Nyckelord — Keywords

drönare, kartläggning, lokalisering, samtidig lokalisering och kartläggning, datorseende

Säilytyspaikka — Förvaringsställe — Where deposited

Helsingfors universitets bibliotek

Muita tietoja — övriga uppgifter — Additional information

Innehåll

1 Inledning

En drönare är ett obemannat luftfordon. För att drönare är mer versatila än robotar som inte flyger så finns det särskilda användningsfall för dem. De kan övervaka markföroreningar eller växternas behov av vatten och näring samt förhindra industriolyckor (Motlagh m. fl., 2017). Drönare har också använts vid katastrofområden, som vid jordbävningen nära Japan för att mäta strålningsvärden i Fukushima och att skaffa visuell information om katastrofområdet samt i räddning av invandrare vid medelhavet. Inom logistikbranschen är man intresserad om autonoma drönaren som skulle leverera paket för kunder, till exempel 86 % av Amazons paket väger mindre än 2.26 kg och de har tänkt på att leverera paket inom 30 minuter för människor som bor nära Amazons lager (Charlie Rose, CBS News, u. å.).

Drönare kan styras med hjälp av visionsbaserad-, tröghets- eller satellitnavigation (Lu m. fl., 2018). För att en drönare skulle kunna navigera självständigt måste den vara medveten om sitt läge, sin omgivning och sin hastighet. Andra faktorer som borde beaktas då en drönare flyger självständigt är hur de behandlar data från sina inbyggda sensorer för att kartlägga miljön och hitta sin egen position i denna miljö samt hur de planerar en väg som inte kolliderar med hinder.

Ett av de centrala forskningsfälten inom robotnavigering är navigering med hjälp av datorseende (Desouza och Kak, 2002). För att robotar skulle kunna navigera autonomt måste de kunna samtidigt kartlägga sin omgivning och lokalisera sig själv i omgivningen. Detta kallas för SLAM (Simultaneous Localization and Mapping). SLAM är ett problem som har fått uppmärksamhet av det vetenskapliga samfundet sedan 80-talet och har sätts som ett rön om det löses. Utmaningar i utvecklingen av kartläggnings- och lokaliseringsalgoritmer är att konstruera algoritmer som fungerar i dynamiska och stora miljöer (Durrant-Whyte och Bailey, 2006). Sannolikhetsberäkning är en möjlighet för att lösa SLAM-problemet och flesta av de algoritmer som har löst problemet baserar sig på sannolikhetsberäkning (Thrun m. fl., 2005).

Avhandlingen är en översikt om SLAM-problemet och vilka metoder det finns för att kartlägga miljön och lokalisera sig själv med hjälp av datorseende. Frågor som behandlas är: Vad är det fundamentala svagheten med de estimerande SLAM-lösningarna? Vilka SLAM-metoder skulle kunna användas eller har använts med drönare och vad begränsar användningen av dessa? Varför har visionsbaserad navigation fått så mycket uppmärksamhet än andra navigeringsmetoder?

Resten av avhandlingen är strukturerad enligt följande: I kapitel 2 diskuteras bakgrund bakom visionsbaserad navigering. Vad är navigering och vilka sensorer kan användas i visionsbaserad navigering samt hur extraheras data ur bilder. Kapitel 2 kommer också att diskutera lokalisering och kartläggning. Kapitel 3 diskuterar om samtidig lokalisering och kartläggning och hur SLAM kan grupperas i olika kategorier enligt hurudan data som man har. Kapitel 4 sammanfattar avhandlingen.

2 Bakgrund

2.1 Visionsbaserad navigering

Med navigering anser vi att en drönare planerar och utför en rutt från en given startplats till ett givet mål (Lu m. fl., 2018). För att kunna navigera till målet måste drönaren vara medveten om sitt läge, sin miljö och sin hastighet. För att kunna navigera autonomt måste drönaren kunna undvika hinder, planera sin rutt samt ta omvägar vid behov. I visionsbaserad navigering används visuella sensorer för att få bilder som indata. Bilderna behandlas sedan med algoritmer för att få en representation av omgivningen och för att lokalisera drönare i omgivningen.

Visuella sensorer som används i visionsbaserad navigering är monokulära-, stereo-, RGB-D- och fisheye-kameror samt kombinationer av dessa (Lu m. fl., 2018). En monokulärkamera innehåller ett kameraobjektiv och stereokamera har två kameraobjektiv. En fisheye-kamera är en monokulärkamera där man använder ett fisheye-objektiv för att få ett bredare synfält i bilden. Med RGB-D kamera kan man samtidigt ta bilder av omgivningen och uppskatta djuphet med hjälp av infrarödsensorer, oftast används dessa inomhus på grund av att de har svaga infrarödsensorer och kan bara upptäcka djuphet på kort avstånd.

Jämfört med de traditionella navigeringsmetoder, såsom satellitnavigering som använder GPS-signaler (Global Positioning System) för lokalisering, och tröghetsnavigering (Inerital Navigation System) som använder accelerometer och gyroskop, är fördelarna med kameror att man får visuell information, som till exempel färg, textur och former av omgivningen och att kameror är passiva, det vill säga de kan inte bli störda av utomstående signaler eller upptäckas av utomstående entitet. Dessutom kan kameror användas för att navigera i områden där GPS-signaler inte fungerar. I och med att kameror är passiva kan man beräkna drönarens position i omgivningen inombords, förutom det är kamerasensorer billiga att operera och väger lite (Kehoe m. fl., 2006; Lu m. fl., 2018).

Drönare byggs mindre än förr och har därför begränsad batteri- och bärkapacitet. Visionsbaserad navigering med drönare som använder sig av laser- och ultraljudsensorer har uteslutnas på grund av att dessa sensorer förbrukar mer elektricitet och väger mer än kameror (Fraundorfer m. fl., 2012). Med hjälp av visionsbaserad navigering som använder sig av kameror kan man få rikligt med information av omgivningen (Lu m. fl., 2018).

2.2 Bildbearbetning

Med kameror får man sekventiella bilder ur omgivningen som sedan bearbetas med algoritmer för att få en representation av miljön (Desouza och Kak, 2002). En digital bild är konstruerad ur många bildpunkter. Från dessa bildpunkter kan extraheras data, till exempel färg, som används i bildbearbetning. Bildbearbetning betyder att man gör utdrag ur bilder i form av former, kanter, linjer, rörelse, färg eller mönster och bygger en karta av de egenskaper som man extraherat ur bilderna.

För att extrahera landmärken ur bilder, det vill säga former, kanter eller linjer, så finns det olika bilbearbetningsalgoritmer såsom SIFT (Scale-Invariant Feature Transform), HARRIS-hörnupptäckare, SURF (Speeded Up Robust Features), ORB (Oriented FAST and Rotated BRIEF), med mera (Rublee m. fl., 2011; Aulinas m. fl., 2008; Se, D. Lowe m. fl., 2002). SIFT, SURF och ORB prövar hitta egenskaper från bilderna som är invarianta för rotation, det vill säga egenskaper som kan hittas från olika synvinklar (Rublee m. fl., 2011).

Bildbearbetningsalgoritmerna fungerar i allmänhet så att den går igenom bilderna per varje bildpunkt och prövar hitta områden i bilderna som har stora skillnader i färgnyanser. ORB-bildbearbetningsalgoritmen, som diskuteras i avhandlingen, är en kombination av flera algoritmer som söker egenskaper ur bilder och konstruerar landmärken av dessa egenskaper, som sedan kan sparas för senare användning.

Som först så använder ORB en algoritm som går igenom varje bildpunkt och jämför ljusstyrkan av en bildpunkt till bildpunkterna runtomkring. Om skillnaden i ljusstyrkan är stor så använder algoritmen detta område som en egenskap i bilden. För att minska på antalet egenskaper och för att hitta egenskaper som är invarianta för rotation används HARRIS-hörnupptäckningsalgoritmen för att extrahera möjliga hörn ur bilden (Rublee m. fl., 2011). För de egenskaper som hittas med båda algoritmerna används sedan olika algoritmer för att avbilda hur egenskaperna skulle se ut då man roterar eller skalar bilden. Med hjälp av dessa algoritmer går det att spara egenskaperna som landmärken i kartan (Rublee m. fl., 2011). Enligt Rublee m. fl., 2011 är ORB mer energieffektiv bildbearbetningsalgoritm än SIFT för att den kräver mindre beräkning för att extrahera egenskaper och konstruera landmärken ur bilder.

Ett sätt att förbättra extrahering av egenskaper ur bilder är att utjämna bilder (Se, D. Lowe m. fl., 2002). Detta menar att man gör bilder otydligare för att få bort digitalt brus och att bilden har mindre detalj. Då man extraherar egenskaper ur bilder som är otydligare hittar algoritmerna färre egenskaper än om bilden var i sin originalform. Med att extrahera färre egenskaper ur bilderna då en robot navigerar kan algoritmerna beräkna snabbare robotens

läge i omgivningen, på grund av att beräkningstiden är mindre för att matcha alla landmärken man observerat med tidigare observationerna.

Jämfört med andra sensorer, såsom laser- och ultraljud, finns det unika utmaningar då man använder kameror. Förändring i belysning, vädret och årstid samt skuggor är ett problem som måste tas i beaktande med kameror, speciellt då man navigerar utomhus (Desouza och Kak, 2002). Dessa problem har lösts med att bearbeta bildernas färgnyanser, färgmättnad (saturation) och ljusstyrka.

2.3 Lokalisering i omgivningen

Med att en robot lokaliserar sig själv menar vi att den tar reda på sin position i omgivningen (Desouza och Kak, 2002). Från bilderna ur kamerorna i drönaren identifieras landmärken, sedan matchas de observerade landmärken med de som finns i kartan efter vilket drönaren kan estimera sin position i omgivningen. Lokalisering av en drönare är enkelt att beräkna då man har en uppfattning om miljön, till exempel en karta.

Då en drönare navigerar beräknas positionen och orientationen av drönaren till exempel med sannolikhetsberäkning baserat på tid och indata av sensorer (Thrun m. fl., 2005). Efter varje rörelse som drönaren gör måste den estimera sin position i omgivningen. För att beräkna sannolikheten av sin position i omgivningen använder den som data rörelseinformation, och efter att den har rört på sig, observerade landmärken. Genom att observera landmärken efter sin rörelse kan drönaren förstärka sin säkerhet om sin position i omgivningen.

Då man använder sannolikhetsberäkning för lokalisering är robotens läge i omgivningen en kontinuerlig stokastisk sanningsvariabel X, där x representerar ett läge i X. För att beräkna sannolikheten att roboten befinner sig vid x vid tidpunkt t betecknas det som $P(X = x_t)$ eller kortare som $P(x_t)$ (Thrun m. fl., 2005). Variabeln x_t beskriver då robotens läge, som i drönarens fall kan vara till exempel drönarens koordinater och orientering i omgivningen. Rörelseinformation som skaffas vid tidpunkt t betecknas som u_t och landmärken som extraheras från kamerasensorernas bilder med z_t. Lokaliseringsproblemet är att bestämma x_t med sannolikhetsberäkning givet alla föregående lägen $x_{0:t-1}$, landmärkesuppfattningar $z_{1:t-1}$ och rörelseinformationer $u_{1:t}$ samt då vi antar att roboten rör på sig före den skaffar sensorinformationen. Då kan betingade sannolikhetsformeln för robotens läge vid $X = x_t$ skrivas som:

$$P(x_t|x_{0:t-1}, z_{1:t-1}, u_{1:t}) \tag{2.1}$$

Problem uppstår i beräkningen då man navigerar länge för att man tar i beaktande alla

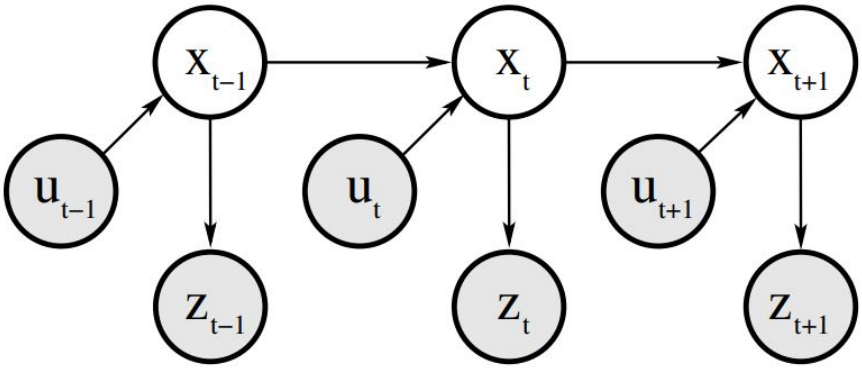
tidigare lägen $x_{0:t-1}$ för att beräkna nästa läge (Thrun m. fl., 2005). För att minska beräkningsbehov i lokaliseringsproblemet antar man Markovegenskapen för variabeln x. Med Markovegenskapen menar man att sannolikheten att roboten befinner sig vid x_t är bara beroende på den tidigare läge, alltså sannolikheten vid x_{t-1}, och inte på händelseförloppet som hände före roboten befinner sig vid x_{t-1}. Då man antar Markovegenskapen så kan man begränsa att x_t är bara beroende på den tidigare position x_{t-1} och rörelseinformation u_t. Figur 2.1 demonstrerar Markovegenskapen. Lokaliseringsproblemet med Markovegenskapen är att räkna ut följande:

$$P(x_t|x_{t-1}, u_t) \tag{2.2}$$

Då vi antar att roboten rör på sig och sedan skaffar sensorinformation samt att x_t har Markovegenskapen kan sannolikhetsfördelningen för landmärkesuppfattningar z_t skrivas:

$$P(z_t|x_{0:t}, z_{1:t-1}, u_{1:t}) = P(z_t|x_t) \tag{2.3}$$

Formeln $P(x_t|x_{t-1}, u_t)$ är rörelseuppfattningssannolikheten och $P(z_t|x_t)$ är landmärkesuppfattningssannolikheten för roboten (Thrun m. fl., 2005).



Figur 2.1: Lokaliseringsproblemet med Markovegenskapen är att bestämma x_t givet x_{t-1} och u_t. En bild på lokaliseringsproblemet som visualiserar Markovegenskapen för variabeln x (Thrun m. fl., 2005).

Robotens läge kan beräknas med Bayes filter som använder sig av rörelse- och landmärkesuppfattningssannolikheten (Thrun m. fl., 2005). Bayes filter är en rekursiv algoritm som beräknar sannolikheten för robotens position baserat på robotens tidigare position, rörelseinformation och observerade landmärken. I Bayes filter algoritmen beräknas $bel(x_t)$, som är

en sannolikhetsfördelning av robotens position i omgivningen.

Algorithm BFA: Bayes Filter Algoritm

`BayesFilter(`$bel(x_{t-1})$`, `u_t`, `z_t`):`

forall x_t **do**

  $spådom = \int P(x_t|x_{t-1}, u_t)bel(x_{t-1})dx_{t-1}$

  $bel(x_t) = \eta P(z_t|x_t)\ spådom$

end

return $bel(x_t)$

Bayes filters algoritmen fungerar så att för varje x_t beräknas två kritiska steg, spådom och korrigering (Thrun m. fl., 2005). Spådomsdelen i algoritmen är delen där vi beräknar integralen för två faktorer: sannolikheten att rörelseinformationen u_t tar oss till x_t och den gamla distributionen av positionen $bel(x_{t-1})$. Korrigeringsdelen är då man multiplicerar spådomsdelen med den sannolikhet att z_t skulle observeras för varje möjliga x_t. I korrideringsdelen används också en normering konstant η för att normera sannolikheten. Bayes filter algoritmen är definierat ovan, se algoritmen 1. Utmaningen i algoritmen är att beräkna integralen i spådomsdelen. För att algoritmen skulle fungera i allmänhet, måste man begränsa robotens läge att vara en ändlig variabel så att integralen över spådomsdelen blir ändlig.

Med hjälp av Bayes regler, lagen om total sannolikhet för kontinuerliga variabler och Markovegenskapen kan det med matematisk induktion bevisas att Bayes filtern är korrekt, då man antar att x_t har Markovegenskapen och att $bel(x_0)$ sannolikhetsfördelning är känd då $t = 0$. Med att sannolikhetsfunktionen är känd menar vi att roboten vet sin position vid början med full säkerhet eller så är $bel(x_0)$ en likformig sannolikhetsfördelning.

$$
\begin{aligned}
bel(x_t) &= P(x_t|z_{1:t}, u_{1:t}) && \text{(BF1)}\\
&= \eta P(z_t|x_t, z_{1:t-1}, u_{1:t})P(x_t|z_{1:t-1}, u_{1:t}) && \text{(BF2)}\\
&= \eta \underline{P(z_t|x_t)} P(x_t|z_{1:t-1}, u_{1:t}) && \text{(BF3)}\\
&= \eta P(z_t|x_t) \underline{\int P(x_t|x_{t-1}, z_{1:t-1}, u_{1:t})P(x_{t-1}|z_{1:t-1}, u_{1:t})dx_{t-1}} && \text{(BF4)}\\
&= \eta P(z_t|x_t) \int \underline{P(x_t|x_{t-1}, u_t)} P(x_{t-1}|z_{1:t-1}, u_{1:t})dx_{t-1} && \text{(BF5)}\\
&= \eta P(z_t|x_t) \int P(x_t|x_{t-1}, u_t)P(x_{t-1}|z_{1:t-1}, \underline{u_{1:t-1}})dx_{t-1} && \text{(BF6)}\\
&= \eta P(z_t|x_t) \int P(x_t|x_{t-1}, u_t)\underline{bel(x_{t-1})}dx_{t-1} && \text{(BF7)}
\end{aligned}
$$

Sannolikhetsfördelningen i Bayes filter är BF1, det vill säga sannolikheten att roboten är vid x_t, som betecknas $bel(x_t)$, är samma som sannolikheten av x_t givet $z_{1:t}$ och $u_{1:t}$. Med Bayes teorem kan detta skrivas om som i ekvation BF2. För att få ekvationen BF3 använder

vi Markovegenskapen för x_t, då kan betingade sannolikheten för z_t skrivas som landmärkesuppfattningssannolikheten, $P(z_t|x_t)$. Med satsen om total sannolikhet för kontinuerliga variabler, som uttrycker sannolikheten för enskilda händelser till en betingade sannolikhet, kan formeln skrivas om till BF4 (Thrun m. fl., 2005, Kapitel 2.2). Vid BF5 och BF6 använder vi Markovegenskapen för x_t, se Markovkedjan i figur 2.1. Till slut får vi BF7, som bevisar med induktion att algoritmen fungerar (Thrun m. fl., 2005).

Bayes filter algoritmen baserar sig starkt på Markovegenskapen att det nuvarande läge är oberoende av tidigare data (Thrun m. fl., 2005). I robotnavigering är det inte dock så lätt att anta detta, för det skulle betyda att allt runtomkring roboten borde vara statiskts. Till exempel om en bil rör på sig då roboten navigerar så då man gör landmärkesuppfattning kommer Bayes filter algoritmen att ge en felaktig estimation. Bayes filter är basis för många andra lokaliseringsalgoritmer, som till exempel Kalman filter eller Markov lokalisation.

I navigering är det svårt att begränsa robotens tillstånd så att integralen i Bayes filter blir en summa över spådomsdelen. Detta gäller endast om roboten navigerar i en begränsad omgivning. En begränsad omgivning kan vara till exempel en våning i en byggnad. För att åstadkomma robotar som navigerar autonomt i stora miljöer måste man approximera. Då möjligheterna av robotens läge är oändlig, till exempel utomhus, beräknar lokaliseringsalgoritmer ett approximativt värde för sannolikhetsfördelningen. Det är i beräkningen av approximativa värdet där olika lokaliseringsalgoritmer skiljer sig (Thrun m. fl., 2005).

Genom att approximera sannolikhetsfördelningen lämnar man bort någon av egenskaperna som är viktigt för robotnavigering, såsom beräkningseffektivitet, noggrannhet i approximerande av robotens läge eller enkel implementering av algoritmen för robotar, för att till exempel förbättra någon annan av egenskaperna (Thrun m. fl., 2005). Om man vill nå beräkningseffektivitet då man approximerar så avstår man till exempel i noggrannhet av robotens läge.

Lokaliseringsproblemet kan delas i global och lokal lokalisering (Desouza och Kak, 2002; Se, D. G. Lowe m. fl., 2005). Med global lokalisering har roboten ingen vetskap om sin position vid början av navigeringen. I lokal lokalisering har roboten en ungefärlig eller exakt vetskap om sin plats vid början av navigeringen som den fått som indata. Då man börjar navigera, alltså när tidpunkt t är noll, så då man använder sig av globala lokaliseringsmetoder är sannolikheten att roboten är vid $bel(x_0)$ en likformig sannolikhetsfördelning över alla lägen i stokastiska variabeln X, och i lokala lokaliseringsmetoder är det till exempel $bel(x_0) = 1$ eller en normalfördelning runt läget x_0 (Thrun m. fl., 2005). Lokala lokaliseringsmetoder strävar för att korrigera fel som uppstår av robotens rörelse. Med globala lokaliseringsmetoder kan roboten klara av verkliga misstag då den estimerar sin position i omgivningen, till exempel

roboten kan klara av en situation där den kidnappas och förs till en annan plats.

2.4 Kartläggning av omgivningen

Med kartläggning av omgivningen menar vi att en robot använder data den har tillgång till för att kartlägga sin omgivning. Detta kan vara till exempel rörelseinformation eller sensorinformation samt kombinationer av dessa (Thrun m. fl., 2005). I lokal lokalisering, där man vet robotens läge och har åtkomst till rörelseinformation, är kartläggning lättare att göra än då robotens position i början är okänd. För att konstruera en karta så måste man ta i beaktande störningar som uppstår av rörelse- och sensorinformation.

En karta om miljön kan representeras i två (2D) eller tre (3D) dimensioner (Lu m. fl., 2018). Att konstruera en 3D-representation av omgivningen kräver det mer beräkningskapacitet än att konstruera en 2D-karta (Thrun m. fl., 2005). För robotar som navigerar på stadig grund är oftast en 2D-representation tillräckligt, men för drönaren behöver man kartlägga i 3D.

För att konstruera 3D-kartor behöver man uppskatta djup. Metoder för att uppskatta djup med datorseende är till exempel beräkning av binokulära skillnaden eller rörelseparallax (Mansour m. fl., 2019). Rörelseparallax betyder att objekt rör på sig snabbare i synfältet då de är närmare observatören och långsammare då de är långt borta. Detta gäller också om observatören rör på sig och objektet står stilla (Mansour m. fl., 2019; Rogers och Graham, 1979). För att räkna binokulära skillnaden för observerade kännetecken behöver man två kameror som är riktade parallellt i samma linje så att deras synfält överlappar, kamerornas distans från varandra är känd och från båda kamerornas bilder utdras samma kännetecken. Från skillnaden mellan kännetecknets horisontala position i båda av kamerornas bilder kan man sedan estimera djup. Med en monokulär kamera kan man uppskatta djup i bilder baserat på rörelseparallax. För detta behöver man veta distansen till ett objekt då navigeringen börjar. Som indata för beräkningen får man bilder och rörelseinformation av roboten. Rörelseinformationen i robotar på stadig grund kan man få till exempel från en hastighetsmätare som är inbyggt i roboten.

Mansour m. fl., 2019 undersökte djupuppskattning med rörelseparallax. Genom att räkna binokulära skillnaden från stereokameror märkte de att stereokameror fungerar bättre då objekten är nära medan monokulära kameror kan uppfatta mer precis distans på långt håll (Mansour m. fl., 2019). Med stereokamera kan djup estimeras bra upp till cirka 10 meters avstånd, och vid 20 meters avstånd så är estimeringarna oanvändbara. Vid 20 meters avstånd måste man växla över till att använda monokulära kameran. Mansour m. fl., 2019 skriver att forskning för att skaffa rörelseinformation från bilderna är viktigt, på grund av att rörel-

seinformation behövs för att lokalisera roboten i omgivningen och kartlägga omgivningen. Genom att uppskatta rörelseinformation från bilder kan man avlägsna hastighetsmätare som drönare inte har tillgång till. Detta ger möjlighet att estimera djup för drönare.

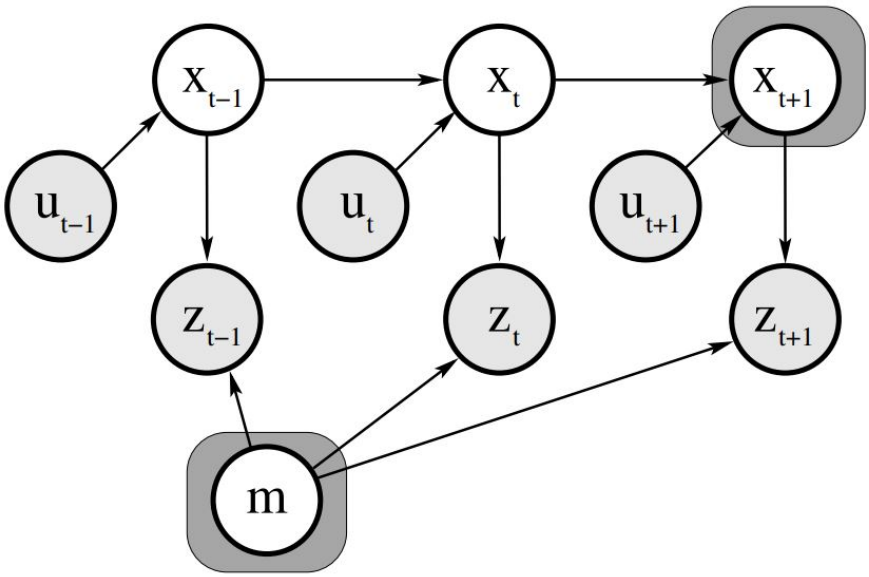
Kartor kan sparas i olika format, såsom datorstödd konstruktion (CAD, Computer-Aided Design), beläggningskarta (Occupancy Grid Map) eller en enkel graf om landmärken och deras sammankopplingar (Desouza och Kak, 2002). En datorstödd konstruktion av miljön kan vara en mycket detaljerad representation av omgivningen. En beläggningskarta är en 2D eller 3D-modell av omgivningen som är sparat i ett rutsystem där rutor som är upptagna är någon objekt i miljön, oftast har dessa rutorna sparat i sig en sannolikhet att där finns någon objekt i vägen (Heng m. fl., 2011; Desouza och Kak, 2002). Kartan kan vara färdigt sparad för en drönare eller så kan miljön kartläggas med hjälp av bilderna från sensorerna då den flyger (Lu m. fl., 2018). Med 3D volymetriska sensorer kan man konstruera en 3D modell och spara denna information i en Octree-struktur. Med strukturen kan data om miljön packas i mindre format utan att tappa möjligheten att uppdatera informationen vid behov. En annan metod som tas upp är med stereovisionsalgoritmer göra en djuphetskarta och behandla data till plana ytor som minskar på missvisning som uppstår med användning av stereovision algoritmer när man bygger upp djuphetskartan.

Oftast har robotar inte en färdig karta som de kan använda för att planera sin rutt eller att navigera (Thrun m. fl., 2005). För att åstadkomma autonoma robotar måste de själv kunna kartlägga sin omgivning. Kartläggning för robotar är ett problem som är svårt utan lokalisering och lokaliseringsproblemet är svårt att lösa utan att man kartlägger, därför är det nyttigt att lösa båda problem samtidigt.

3 Samtidig lokalisering och kartläggning

Samtidig lokalisering och kartläggning (SLAM) är ett av de grundläggande problem inom robotnavigering (Durrant-Whyte och Bailey, 2006). SLAM-problemet definieras så att en robot som inte har tidigare information om sin plats eller omgivning skall samtidigt bygga en karta av omgivningen och lokaliserar sig själv relativt till kartan som den bygger, till exempel med hjälp av datorseende och att identifiera landmärken. Med att beräkna robotens läge menar vi beräkningen av $P(x_t, m|z_{1:t}, u_{1:t})$, det vill säga, sannolikheten att roboten befinner sig vid x_t och kartan m givet alla landmärkesuppfattningar $z_{1:t}$ och rörelseinformation $u_{1:t}$.

Det finns två varianter av SLAM-problemet: Fullständig SLAM och Online SLAM (Thrun m. fl., 2005). Skillnaden med dessa två är hur man beräknar betingade sannolikheten för x_t och m. I Fullständig SLAM beräknas betingade sannolikheten för hela robotens positionskedja $x_{0:t}$, det vill säga $P(x_{0:t}, m|z_{1:t}, u_{1:t})$, medan i Online SLAM använder man bara senaste läge x_t och kartan m, då är formeln $P(x_t, m|z_{1:t}, u_{1:t})$, alltså döljer tidigare rörelse- och landmärkesinformation för att estimera den nya positionen av roboten. Några lösningar för SLAM som baserar sig på sannolikhetsberäkning är Extended Kalman Filter (EKF-SLAM), som är en Online SLAM-lösning, och FastSLAM, som är en Fullständig SLAM-lösning (Durrant-Whyte och Bailey, 2006; Thrun m. fl., 2005).



Figur 3.1: Bild på Online SLAM-problemet där man beräknar sannolikheten av robotens position och bygger kartan från sensorinformation z och rörelseinformation u (Thrun m. fl., 2005).

SLAM-problemet har delproblem som måste lösas för att få robotar att navigera autonomiskt (Aulinas m. fl., 2008). Ett stort problem i flesta visionsbaserade SLAM-lösningar är dataföreningsproblemet. Detta betyder att man identifierar två olika landmärken som en och samma. Dataföreningsproblemet kan uppstå redan vid korta rörelsen av robotar eller då en robot har navigerat och kommer till en plats som den redan har varit i vid, detta kallas för loopstängning (Loop Closure). Med att förena landmärken fel så uppkommer det missvisningar då man estimerar positionen av roboten.

Positionen av roboten i omgivningen och landmärkens position i kartan kan också beräknas med att räkna distansen till landmärken och bevara en matris med landmärkens sannolikhetsposition i omgivningen och landmärkes förhållande till varandra, alltså en korrelationsmatris (Durrant-Whyte och Bailey, 2006; Thrun m. fl., 2005). Genom att bevara en korrelationsmatris så har man den fördelen att när man lokaliserar en robot med stor sannolikhet går det att uppdatera sannolikhetsfördelningen över landmärkets position. Nackdelen med denna metod är att då man observerat stora mängder av landmärken så kommer beräkningsbehovet att växa kvadratiskt enligt antalet observerade landmärken.

Visuell kartläggning och lokalisering kan delas i tre kategorier baserat på data som man har i början av navigeringen. Kategorierna är kartlösa (mapless), kartbaserade (map-based) och kartbyggande (map-building) system (Lu m. fl., 2018).

3.1 Kartlösa system

Kartlösa situationen är oftast den realistiska system då man navigerar (Thrun m. fl., 2005). I system utan karta navigerar drönare bara med hjälp av att observera tydliga egenskaper i miljön (Desouza och Kak, 2002). Dessa kan vara till exempel väggar, dörrar, möbler eller andra landmärken. Metoder för navigering i kartlösa system, som diskuteras i avhandlingen, är optiskt flöde (Optical flow) och spårning av egenskaper (Feature tracking).

3.1.1 Optiskt flöde

Optiskt flöde hänvisar till uppfattad relativ rörelse i sekventiella bilder mellan observatören och till exempel någon fästpunkt i miljön (Kehoe m. fl., 2006). Rörelse uppstår i sekventiella bilder till exempel då man vänder kameran eller om något rör på sig då kameran är stilla. Från relativa rörelsen i sekventiella bilder kan man beräkna flödesvektorer.

För att demonstrera ett exempel hur optiskt flöde kan användas för robotnavigering har Santos-Victor m. fl., 1993 använt optiskt flöde i en robot för att imitera bin (Santos-Victor

m. fl., 1993). De placerade två kameror, som är parallelt riktade ifrån varandra, på varsin sida av en robot och beräknade flödesvektorer från bilderna av båda kamerorna då roboten rör på sig. Om flödesvektorernas beräknade längd var samma på båda sidorna så far roboten rakt framåt, annars om andra sidas flödesvektoren är kortare än den andra så vänder den mot den sida vilkens flödesvektor är kortare. Med att använda optiskt flöde för navigering behöver man inte uppskatta djuphet. Denna navigeringsmetod fungerar dåligt i strukturlösa miljö där det inte finns någon fästpunkt eller landmärke som kan användas för att beräkna flödesvektorerna.

Fastän metoden som Santos-Victor m. fl., 1993 använde tog bara i beaktande horisontala flödesvektorer och demonstrerade att optiskt flöde kan användas i robotnavigering på stadig grund, så har sedan detta användning av optiska flödesmetoder använts i drönare (Chao m. fl., 2013). Nuförtiden används optiskt flöde i drönare för att uppskatta avstånd, hålla sin höjd, undvika hinder, beräkna hastighet och landa på en plattform som rör på sig.

3.1.2 Spårning av egenskaper

Spårning av egenskaper (Feature tracking) används för att skaffa information om objekt, såsom linjer, hörn och olika landmärken som är invariant för rörelse med hjälp av kameror och bildbearbetningsalgoritmer (Lu m. fl., 2018). Med hjälp av dessa landmärken och deras relativa rörelse i sekventiella bilder kan man bestämma robotens position och bygga en karta. Då drönare navigerar i omgivningen, så kommer den troligtvis att se samma landmärken från olika perspektiv, som hjälper drönare att beräkna en bättre sannolikhet för sin position i omgivningen samt uppdatera landmärkes position i kartan. Traditionella SLAM-lösningar, såsom EKF-SLAM och FastSLAM faller under denna kategori (Latif och Saddik, 2019). Dessa SLAM-lösningar fungerar inte då det finns mycket landmärken i kartan för till exempel med EKF-SLAM är problemet beräkningsbehovet som växer kvadratiskt av antalet landmärken.

För att drönaren har begränsad batterikapacitet och bärkraft har Gu m. fl., 2015 föreslagit användning av VOSLAM (Visual Odometry SLAM) som kan kartlägga omgivningen och lokalisera observatören då det finns upp till tusentals landmärken i kartan med en stereokamera som tar 31 bilder per sekund och en krets som konsumerar tio gånger mindre energi än en Intel i7 3770K-processor (Gu m. fl., 2015). Deras VOSLAM algoritm implementeras i en FPGA-krets (Field-Programmable Gate Array) vilkens logik är omprogrammerbar, den använder färre energi än vanliga processorer och kan beräkna parallellt. VOSLAM, med speciell hårdvara, klarar av upp till 30 000 landmärken i kartan, jämfört med EKF-SLAM, som klarar av kring tusen landmärken för att fungera i realtid (Thrun m. fl., 2005; Gu m. fl., 2015).

VOSLAM-lösningen fungerar så att den extraherar landmärken från bilder med bildbear-

betningsalgoritmen SIFT (Gu m. fl., 2015). Med hjälp av att beräkna binokulära skillnaden från bilder, för att uppskatta djuphet, som diskuterades i kapitel Kartläggning av omgivningen, konstruerar de en 3D-representation av dessa landmärken som de sparar i en matris. Med FPGA-kretsen beräknas vektorvinklarna för varje observerade landmärke då roboten navigerar och med hjälp av dessa kan de matcha observerade landmärken med de som de har i kartan för att lokalisera drönaren. Då algoritmen bygger kartan tar de bort avvikande träffar inom landmärken med hjälp av en algoritm, så att inte kartan uppfylls med osäkra landmärkesuppfattningar.

Enligt Gu m. fl., 2015 kan VOSLAM estimera positionen av drönaren, hantera loopstängning och enligt dem är troligtvis en av de mest energieffektiva lösning för SLAM-problemet (Gu m. fl., 2015). Något som deras lösning inte tar i beaktande är SIFT-algoritmens tidskrav, som betyder att beräkningsbehovet växer. Latif och Saddik, 2019 använder VOSLAM i en drönare där de delat upp processen så att indata från stereokameran, bildbearbetning och 3D-representationen bearbetas i en egen processor och uppdatering av kartan, matcha observerade landmärken med kartan, lokalisering och uppdatering av kartan hanteras i FPGA-kretsen (Latif och Saddik, 2019).

3.2 Kartbaserade system

Med kartbaserade system har drönare en färdig vetskap om miljön som kan vara i form av geometriska modeller, beläggningskarta eller förhållande mellan landmärken (Desouza och Kak, 2002). Idén är att då roboten navigerar prövar den hitta till exempel landmärken från bilder som är lika till de landmärken som roboten vet om. När den observerat landmärken och förenat observationerna med kartan som den har så beräknas sannolikheten av roboten position i miljön. Med kartbaserad navigering kan en drönare planera sin rutt i förhand och beräkna omvägar under navigeringen då det behövs (Lu m. fl., 2018).

Kartbaserad lokalisering med datorseende kan delas upp i fyra steg, som är följande (Desouza och Kak, 2002):

- skaffa sensorinformation av kamerorna med bildbearbetning
- upptäcka landmärken från informationen med bildbearbetning
- matcha observationerna med kartan
- beräkna positionen av roboten

Det svåraste steget av dessa är att matcha observationerna med kartan som den har, som är dataföreningsproblemet. Roboten kan inte med full säkerhet veta sin position och då är det svårt att matcha landmärken med kartan (Desouza och Kak, 2002).

Då kartan finns kan man fokusera på lokalisering av roboten (Desouza och Kak, 2002). I global lokalisering måste man lita på att man kan förena observationerna med informationen man har och ta i beaktande osäkerheten med att observationerna kan matcha flera av de landmärken man vet om. Globala lokaliseringsproblemet, där roboten har ingen aning om vart den är i kartan vid början, kan lösas till exempel med Monte Carlo lokalisation eller Markov lokalisation, som båda är en variant av Bayes filter som presenterades i kapitel Lokalisering i omgivningen (Thrun m. fl., 2005).

Monte Carlo lokalisation fungerar så att en robot antar med lika sannolikhet sin position i kartan vid början av navigeringen (Dellaert m. fl., 1999). Då man approximerar sannolikheten av robotens position i Monte Carlo så väljer man slumpmässigt platser vart roboten möjligen skulle befinnas med den rörelseorder som den tar och sedan rör roboten på sig, efter det observerar landmärken, förenar dessa landmärken med kartan som den har och på basis av denna information kan roboten estimera vilken av dessa slumpmässiga positionsval var den bästa.

Markov lokalisationsalgoritmen klarar också av sig globala lokaliseringsproblemet då man har en karta (Thrun m. fl., 2005). Istället för att anta bara en sannolikhet av robotens position i kartan så uppehåller den en sannolikhetsfördelning över hela kartan. Till exempel då roboten börjar navigera så är sannolikhetsfördelningen enhetlig och då den rör på sig och om den observerar ett landmärke så ökar sannolikheten av robotens position för varje plats det finns ett liknande landmärke i kartan. När den rör på sig vidare så kommer den troligtvis att observera mera landmärken och på basis av rörelseinformationen och observerade landmärken kan den vid något skede vara ganska säker av sin position.

I lokal lokalisering där roboten har en vetskap om sin position i början av navigering behöver lokaliseringsalgoritmen hålla koll på rörelseinformationen som roboten utför och på basis av rörelseinformationen beräkna nya positionen (Dellaert m. fl., 1999; Thrun m. fl., 2005). Då roboten rör på sig så minskar sannolikheten av robotens nya position på basis om hur lokaliseringsalgoritmen tar i beaktande felmarginal i rörelsen. Då osäkerheten av sin egen position är för stor så använder den observerade landmärken och förenar dessa med kartan den har för att öka på sannolikheten av sin position.

3.3 Kartbyggande system

Kartbyggande system används då det är svårt att navigera med en existerande karta om omgivningen eller om kartan inte finns, som i katastrofområden (Lu m. fl., 2018). Roboten kan vara medveten eller omedveten om sin position vid början av kartbyggandet (Se, D. G. Lowe m. fl., 2005). Kartor kan byggas i två eller tre dimensioner (Thrun m. fl., 2005). Fördelar med att bygga en karta i 3D och använda denna för lokalisering är att det uppkommer mindre missvisningar då roboten lokaliserar sig själv. Till exempel med 2D-kartor, om en robot navigerar i en korridor och kommer till slutet av korridoren kan det vara svårt för roboten att veta i vilken ända den är om korridoren är symmetrisk. Med 3D så finns det mera data att använda för att lokalisera, men detta betyder att man behöver mera beräkningskapacitet. Kartbyggande systems bildbearbetning kan delas tre kategorier, som är indirekta, direkta och hybrida metoder, som sammanslår indirekta och direkta metoder (Lu m. fl., 2018).

3.3.1 Indirekta metoder

I bildbearbetning som använder indirekta metoder tar man kännetecken ur bilden som är invarianta för rotation, synvinkeländringar och rörelseoskärpa, dessa ges som indata som sedan kan användas för rörelseuppfattning och lokalisering (Lu m. fl., 2018).

Ett sätt att konstruera en karta är att beakta robotens rörelse och synvinkel (Se, D. G. Lowe m. fl., 2005). Se, D. Lowe m. fl., 2002 har gjort dessa samt använt indirekta metoder i sin artikel ”Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks” för att bygga en 3D-karta (Se, D. Lowe m. fl., 2002). Från stereokamerornas bilder, utjämnar de bilderna så att de har mindre detalj i sig och använder SIFT för att extrahera egenskaper ur bilderna. Med denna metod har de kunnat konstruera en 3D-karta av omgivningen baserat på landmärken utan att spara korrelationsmatris mellan landmärken som minskar på behov av beräkning då man lokaliserar roboten.

Denna metod att konstruera kartor och lokalisera roboten har ändå problem då det kommer till loopstängning och uppdatering av kartan (Se, D. G. Lowe m. fl., 2005). Med att använda samma bildbearbetningsmetoder och med att kartlägga delar av områden och senare bygga en stor global karta fick forskarna loopstängning och kart-uppdateringen löst.

Globala kartan uppbyggdes så att de identifiera identiska landmärken från dessa små kartor och kunde från dessa foga ihop kartor som var bredvid varandra till en stor karta. Indirekta metoder fungerar dåligt då det finns få egenskaper att extrahera ur bilder, såsom i strukturlösa miljön (Lu m. fl., 2018).

3.3.2 Direkta metoder

Direkta metoder fungerar bättre i strukturlösa miljön än indirekta metoder (Engel m. fl., 2014). Jämfört med indirekta metoder där man prövar hitta många landmärken ur bilderna så i direkta metoder använder man hela bilden för att hitta geometriska egenskaper. Med hjälp av dessa så kan man konstruera en detaljerad karta med extra processorberäkning (Lu m. fl., 2018). Då man konstruerar en mycket detaljerad karta, till exempel i 3D, går det att använda kartan för något annat än navigering, som till exempel att få information från katastrofområden. Att kartlägga i detalj används bara i speciella fall för drönare, på grund av att det kräver mycket beräkningskapacitet och förbrukar mycket energi.

4 Sammanfattning

Samtidig lokalisering och kartläggning i dynamiska miljön är ett problem som måste lösas för att drönare skulle kunna navigera autonomiskt. Oftast är robotens omgivning oändlig och då man estimerar positionen av roboten måste man approximera. Då man approximerar sannolikheten av robotens position är den sällan fullständigt medveten om sin position.

För att verkligen ha autonoma drönaren, som skulle använda sig av visionsbaserad navigering, måste man kunna kombinera effektiva bildbearbetningsalgoritmer, lokaliseringsalgoritmer och kartläggningsalgoritmer så att dessa algoritmer skulle fungera samtidigt, i realtid samt ge bra uppskattningar av drönarens läge.

En drönare har begränsningar då det kommer till beräkningsbehov på grund av att kommersiella drönare byggs mindre än förr och har begränsad batterikapacitet. Drönare har inte tillgång till samma rörelseinformation som robotar på stadig grund som oftast har inbyggda tröghets- och hastighetsmätare i sig. Därför är det viktigt att det finns forskning inom beräkning av rörelseinformationen från kamerabilder.

Denna avhandling var en ytlig översikt om SLAM-problemet och vad är basis för de SLAM-lösningar som finns idag. De problem som finns idag är att konstruera pålitliga algoritmer i visionsbaserad navigering som kan behandla olika scenario, till exempel att estimera djuphet för att bygga 3D-kartor, positionen av roboten, kartläggning och uppdatering av kartan, dataförening, loopstängning samt ruttplanering och undvika hinder under navigeringen som avhandlingen har inte haft fokus på.

Referenser

Aulinas, J., Petillot, Y., Salvi, J. och Llado, X. (jan. 2008). ”The SLAM problem: a survey”. I: vol. 184, s. 363–371. DOI: 10.3233/978-1-58603-925-7-363.

Chao, H., Gu, Y. och Napolitano, M. (2013). ”A survey of optical flow techniques for UAV navigation applications”. I: *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, s. 710–716. DOI: 10.1109/ICUAS.2013.6564752.

Charlie Rose, CBS News (u. å.). *Amazon's Jeff Bezos looks to the future*. https://www.cbsnews.com/news/amazons-jeff-bezos-looks-to-the-future/, läst 02.11.2020.

Dellaert, F., Fox, D., Burgard, W. och Thrun, S. (1999). ”Monte Carlo localization for mobile robots”. I: *Proceedings of 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*. Vol. 2, 1322–1328 vol.2. DOI: 10.1109/ROBOT.1999.772544.

Desouza, G. N. och Kak, A. C. (2002). ”Vision for mobile robot navigation: a survey”. I: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.2, s. 237–267. DOI: 10.1109/34.982903.

Durrant-Whyte, H. och Bailey, T. (2006). ”Simultaneous localization and mapping: part I”. I: *IEEE Robotics Automation Magazine* 13.2, s. 99–110. DOI: 10.1109/MRA.2006.1638022.

Engel, J., Schöps, T. och Cremers, D. (2014). ”LSD-SLAM: Large-Scale Direct Monocular SLAM”. I: *ECCV*. ISBN: 978-3-319-10604-5. DOI: 10.1007/978-3-319-10605-2_54.

Fraundorfer, F., Heng, L., Honegger, D., Lee, G. H., Meier, L., Tanskanen, P. och Pollefeys, M. (2012). ”Vision-based autonomous mapping and exploration using a quadrotor MAV”. I: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, s. 4557–4564. DOI: 10.1109/IROS.2012.6385934.

Gu, M., Guo, K., Wang, W., Wang, Y. och Yang, H. (2015). ”An FPGA-based real-time simultaneous localization and mapping system”. I: *2015 International Conference on Field Programmable Technology (FPT)*, s. 200–203. DOI: 10.1109/FPT.2015.7393150.

Heng, L., Lee, G. H., Fraundorfer, F. och Pollefeys, M. (2011). ”Real-time photo-realistic 3D mapping for micro aerial vehicles”. I: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, s. 4012–4019. DOI: 10.1109/IROS.2011.6095058.

Kehoe, J., Watkins, A., Causey, R. och Lind, R. (aug. 2006). ”State Estimation Using Optical Flow from Parallax-Weighted Feature Tracking”. I: DOI: 10.2514/6.2006-6721. URL: https://mae.ufl.edu/ricklind/rick_pro/of/GNC06_OF.pdf.

Latif, R. och Saddik, A. (2019). ”SLAM algorithms implementation in a UAV, based on a heterogeneous system: A survey”. I: *2019 4th World Conference on Complex Systems (WCCS)*, s. 1–6. DOI: 10.1109/ICoCS.2019.8930783.

Lu, Y., Xue, Z., Xia, G.-S. och Zhang, L. (2018). ”A survey on vision-based UAV navigation”. I: *Geo-spatial Information Science* 21.1, s. 21–32. DOI: 10.1080/10095020.2017.1420509.

Mansour, M., Davidson, P., Stepanov, O. och Piché, R. (2019). ”Relative Importance of Binocular Disparity and Motion Parallax for Depth Estimation: A Computer Vision Approach”. I: *Remote Sensing* 11.17. DOI: https://doi.org/10.3390/rs11171990.

Motlagh, N. H., Bagaa, M. och Taleb, T. (2017). ”UAV-Based IoT Platform: A Crowd Surveillance Use Case”. I: *IEEE Communications Magazine* 55.2, s. 128–134. DOI: 10.1109/MCOM.2017.1600587CM.

Rogers, B. och Graham, M. (febr. 1979). ”Motion parallax as an independent cue for depth perception”. I: *Perception* 8, s. 125–34. DOI: 10.1068/p080125.

Rublee, E., Rabaud, V., Konolige, K. och Bradski, G. (2011). ”ORB: An efficient alternative to SIFT or SURF”. I: *2011 International Conference on Computer Vision*, s. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

Santos-Victor, J., Sandini, G., Curotto, F. och Garibaldi, S. (1993). ”Divergent stereo for robot navigation: learning from bees”. I: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, s. 434–439. DOI: 10.1109/CVPR.1993.341094.

Se, S., Lowe, D. G. och Little, J. J. (2005). ”Vision-based global localization and mapping for mobile robots”. I: *IEEE Transactions on Robotics* 21.3, s. 364–375. DOI: 10.1109/TRO.2004.839228.

Se, S., Lowe, D. och Little, J. (aug. 2002). ”Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks”. I: *The International Journal of Robotics Research* 21, s. 735–760. DOI: 10.1177/027836402761412467. URL: https://www.cs.ubc.ca/~lowe/papers/ijrr02.pdf.

Thrun, S., Burgard, W. och Fox, D. (2005). *Probabilistic robotics.* Intelligent robotics and autonomous agents. MIT Press, s. 1–647. ISBN: 978-0-262-20162-9.