CHALMERS

ISPRINKLE - EN AUTONOM BLOMVATTNARE

Kandidatarbete inom civilingenjörsprogrammet

ALICE ABRAHAMSSON
LUCAS BÄCKVALL
ANTON JANSSON
JOHAN MARTINSSON
MAX WEDENMARK

Department of electrical engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden, 14 maj 2018

Abstract

The purpose of the project was to produce a prototype for an autonomous, mobile robot which would be able to identify plants with the help of image recognition and maneuver to said plants. The resulting prototype is a starting point for future bachelor thesis projects aimed towards advancements of this prototype.

The requirements for the prototype was, after the completion of this project, that it should be able to locate and identify one plant placed in a home environment without previous knowledge about the environment or the plant's position. The prototype would be able to do this without running into obstacles.

The finished prototype utilizes a LiDAR to collect measurement points live and builds a map that is used to navigate through the unknown environment. Thanks to the live update, the robot detects any obstacles that obstructs the path. To find the plants, Google Cloud Vision (Google CV) is used for image recognition. The result from Google CV is analyzed and by dividing the image into smaller segments it is possible to calculate the angle from the robots driving direction relative to the plant.

The report is written in Swedish.

Begreppslista

Begrepp	Förklaring
AGV	Automated Guided Vehicle, en robot som navigerar med hjälp av markörer, lasrar, magneter eller kameror med mera.
Arduino	Ett hård- och mjukvarusystem för hobbyutveckling av elektronik.
Arduinokort	En mikrokontroller med några få portar och en del analoga och digitala I/O stift.
Google CV	Google Cloud Vision API, bildigenkänningstjänst som använder sig av färdiglärt neuralt nätverk.
I/O stift	Input/Output stift. En input- och outputkontakt som enkelt kan kopplas till ett separat system av elektroniska komponenter.
LiDAR	Light Detection And Ranging. En sensor som mäter avstånd utifrån reflektionstiden för en laser.
PWM	Pulsbreddsmodulering. Används för att reglera strömförsörjning till servomotorer.
Python	Ett objektorienterat programmeringsspråk.
Seriell kommunikation	En typ av digital kommunikation som sänds i serie via en enda dataledning.
SLAM	Simultaneous Localization And Mapping. En navigeringsteknik för autonoma robotar.

Förord

Följande rapport har skrivits som ett moment i kursen Kandidatarbete inom Elektroteknik på Chalmers tekniska högskola under vårterminen 2018. Arbetet har genomförts av grupp EENX15-18-46 som består av fem studenter, Alice Abrahamsson, Lucas Bäckvall, Anton Jansson, Johan Martinsson och Max Wedenmark.

Vi vill börja med att rikta vårt tack till vår projekthandledare Martin Fabian som har varit till stor hjälp under arbetet med handledning och stöttning. Vi vill också tacka vår examinator Knut Åkesson som har assisterat oss.

Vi tackar även handledare och föreläsare ifrån avdelningen för fackspråk och kommunikation för givande handledningstillfällen och föreläsningar.

Vi tackar också alla föreläsare under kursen som har gett givande information som kunnat applicerats under projektets gång. Slutligen vill vi tacka kursansvariga Jonas Fredriksson och Petter Falkman.

Innehåll

1. Inledning

I dagens samhälle är det fler och fler som reser hemifrån under långa perioder vilket leder till att många inte har möjlighet att vattna sina blommor [1]. Vissa människor glömmer helt av att vattna eller är borta under långa perioder medan andra kanske inte har möjlighet, exempelvis på grund av ålder eller sjukdom. Eftersom levnadsmiljön har stor inverkan på människors hälsa [2] kan blommor i inomhusmiljöer förbättra människors psykosociala hälsa.

För att fler ska kunna ha friska blommor hemma kan en autonom blomvattnare vara en lösning, alltså ett system för bevattning som inte kräver mänsklig inverkan. Idag finns det redan en del uppfinningar som satisfierar detta och gör det möjligt att resa bort ett tag utan att blommor vissnar. Problemen med dagens uppfinningar är bland annat att vissa behöver förbrukningsvaror eller underhåll, vissa är inte estetiskt tilltalande och andra kräver omläggning av rörsystem eller specialtillverkade krukor. Problemen gör det besvärligt för den bekväma hemägaren som bara vill ha en smidig lösning att starta en gång, och som inte kräver uppmärksamhet.

Vad som eftersöks är en mobil lösning som kan användas i varje hem utan tidigare kännedom om placering av möbler och växter. Denna lösning går i stora drag att jämföra med beteendet hos robotgräsklippare eller robotdammsugare och skulle kunna kombineras med respektive funktioner för att vidareutveckla en universell hemrobot.

1.1. Syfte

Syftet med projektet är att framställa en prototyp för en autonom, mobil robot som ska identifiera krukväxter och manövrera till dessa med hjälp av bildigenkänning. Projektets resultat ska kunna ligga till grund för efterföljande kandidatarbeten med mål att färdigställa en slutprodukt.

1.2. Avgränsningar

Projektet kommer inte att omfatta en helhetslösning för den autonoma blomvattnaren utan enbart en prototyp som demonstrerar igenkänning av krukväxter inomhus samt rumsnavigering.

1.2.1. Identifiering av krukväxter

Prototypen kommer inte ta hänsyn till krukväxter som står på fönsterbrädor, hyllor eller hänger i amplar. Den kommer alltså endast hitta krukor som befinner sig på marknivå. Prototypen avgränsas till att endast hitta och identifiera en krukväxt per körning.

1.2.2. Miljömässig avgränsning

Projektet kommer bara vara anpassat till plana ytor inomhus och kommer därmed inte ta hänsyn till trappor eller mattor och dylikt. Möjligheten att manövrera höjdskillnader är ett problem som redan har lösningar vilket exempelvis används i patentet *”stair climbing robot”* [3].

1.2.3. Manuell elförsörjning

Prototypen i projektet kommer endast laddas manuellt. Lösningen för automatisk laddning för autonoma robotar är redan välutvecklad och används idag av robotgräsklippare och robotdammsugare.

1.3. Mål

Utvecklingen av prototypen kommer ske med ett flertal mål i beaktning. Dessa mål dimensioneras för att mötas under en mässa för kandidatarbeten från avdelningen Syscon vid Chalmers tekniska högskola. De sex viktigaste målen presenteras nedan. Roboten förväntas autonomt uppfylla samtliga mål efter uppstart.

- Roboten ska kunna navigera i en föränderlig miljö utan tidigare kunskap om området.
- Roboten ska kunna identifiera en krukväxt som befinner sig mellan en halv meter och två meters avstånd från roboten.
- Roboten ska klara av att manövrera fram till en identifierad krukväxt.
- Från att roboten startats ska den kunna hitta och navigera fram till en krukväxt på högst 10 minuter.
- Roboten ska registrera att den lyfts upp och då sluta köra.
- Roboten ska stanna runt 50 cm framför föremål, statiskt placerade såsom dynamiskt, för att undvika kollision och ha plats för att vända.

2. Teori

Följande kapitel ger information kring grundläggande kunskap och begrepp som är essentiella för att kunna tillgodogöra sig rapporten.

2.1. Arduino

Arduino är ett vanligt förekommande mikrokontrollerkort för programmering av mekatroniska system. Mikrokontrollerkort kan ses som hjärnan till robotens styrsystem och används exempelvis för att skicka styrsignaler till motorer och lampor om hur de ska agera. Arduino är en hårdvara med tillhörande öppen källkod baserat på C++, ett programmeringsspråk för allmänt ändamål.

Arduinokortet har Input/Output stift (I/O stift) för att kunna skicka och erhålla information och strömförsörja komponenter. Maximalt kan Arduinon själv leverera 5V men kan även kompletteras med batterier och andra strömkällor för att kunna styra högre spänningar.

2.2. Google Cloud Vision API

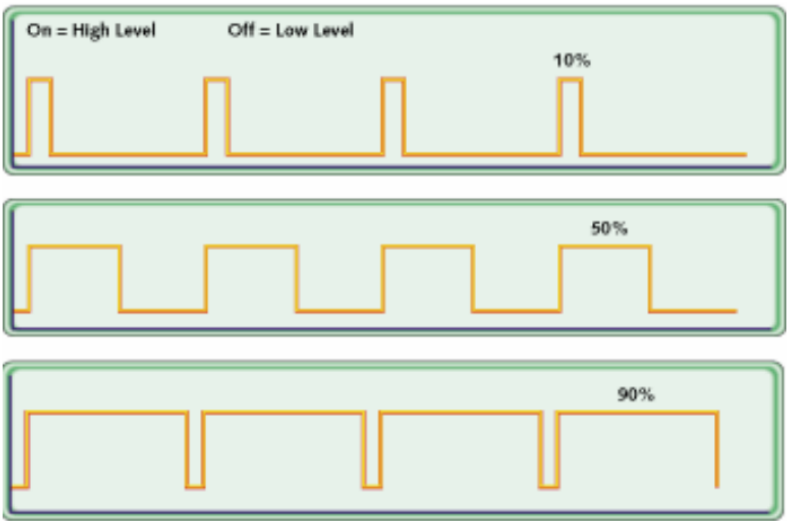
Google Cloud Vision API, hädanefter Google CV, är Googles egna neurala nätverk för bildigenkänning [4]. Google CV möjliggör för användare att ladda upp bilder vilka matchas mot Googles bilddatabas. Det neurala nätverket analyserar möjliga utfall i bilden och returnerar ett svar på ett flertal alternativ på vad bilden sannolikt innehåller.

Google CV returnerar inte alla objekt som finns i bilden utan endast ett urval av toppresultaten som det neurala nätverket tror sig finna. Google CV identifierar lättast objekt som är vanligt att söka efter och som den har fått mycket information om när den började lära sig, till exempel ”face”, ”girl” och ”car”. Andra saker som är avgörande i vad den returnerar är vad som ligger i fokus i bilden och hur stor del av bilden som objektet täcker. I svaret som returneras står det inte var någonstans i bilden som objektet befinner sig utan endast hur stor sannolikhet att det specifika objektet befinner sig någonstans i bilden.

2.3. Pulsbreddsmodulering

PWM är en akronym för pulsbreddsmodulering vilket används för att få en spänning i ett intervall mellan hög och låg spänning. Genom att växla mellan respektive spänningsnivå i perioder enligt figur 1 nedan moduleras spänningen till ett specifikt värde i intervallet.

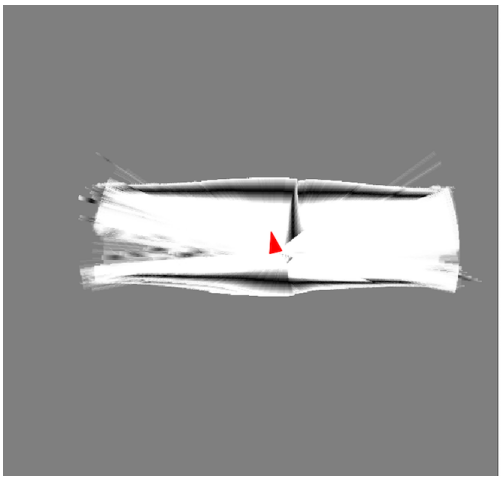
Spänningen som moduleras är genomsnittsspänningen under en period [5]. Figur 1 visar tre diagram för 10%, 50% och 90% modulering vilket motsvarar hög spänning ekvivalent med antal procent av varje period.



Figur 1: Spänningsnivåer vid 10%, 50% och 90% PWM under fyra perioder. [5]

2.4. SLAM

SLAM står för ”Simultaneous Localization And Mapping”. Det är mer ett koncept än en algoritm trots att det vanligtvis kallas för det sistnämnda. SLAM läser kontinuerligt in data från en eller flera sensorer under körning för att kunna rita upp en karta över omgivningen baserat på datan. Kartan uppdateras under körning tillsammans med robotens position. Roboten använder sig aktivt av kartan för navigering och den är inte enbart för att visuellt visa hur omgivningen ser ut. Kartan roboten ritar kan se ut som i figur 2a nedan och motsvaras i verkligheten av det som visas i figur 2b. Vanligtvis kombineras flera metoder för lokalisering eftersom varje metod har olika för- och nackdelar. Vilken kombination som är mest lämplig beror på vilket syfte SLAM har i just det fallet.



(a) De svarta områdena är mätpunkter och den röda pilen markerar robotens position.

(b) Bild över hur korridoren och roboten som är uppritade i kartan bredvid ser ut i verkligheten.

Figur 2: Exempel över en SLAM-karta ser ut beroende på verkligheten

Den lilla röda triangeln i figur 2a symboliserar roboten och är avsmalnande i rörelseriktningen. De svarta linjerna symboliserar väggar eller andra fasta föremål, vita områden är öppna ytor och mörkgråa områden är oidentifierade ytor. Ljusgråa områdena kan uppkomma där inga reflektioner uppfattas, till exempel på grund av för stora avstånd, störningar eller transparenta ytor. Dock är kartan inte alltid helt korrekt utan det kan uppkomma mätfel beroende på vad som används för att läsa av omgivningen och i vilken miljö den körs.

Positionen på kartan uppdateras huvudsakligen på två olika sätt; relativ position eller absolut position. Den enklaste typen av relativ positionering är dödräkning. Detta innebär att rotationen på hjulen mäts för att på så sätt kunna fastställa hur långt roboten har rört sig. Denna metod tar inte hänsyn till faktorer som underlagets friktion och är inte att rekommendera då den ger avvikande resultat. En mer lämplig metod för relativ positionering bygger på att positionen beräknas genom att mäta accelerationen och integrera den två gånger med avseende på tiden roboten kört [6].

Absolut position innebär istället att positionen fastställs exempelvis med hjälp av optiska sensorer såsom en LiDAR vilket är mycket vanligt. Dock finns det vissa nackdelar med en lasermätare. Stora öppna ytor kan vara svåra att lokalisera sig i samt att det i inglasade korridorer kan bli problematiskt på grund av reflektioner.

Det finns ytterligare en metod som ger absolut position och det är GPS. Denna metod har även den vissa svagheter. Den mest uppenbara är problemet med mottagningen. För att kunna få ut en exakt GPS-position måste mottagningen vara mycket god. Detta innebär i

praktiken att GPS inte är lämpligt i inomhusmiljöer eftersom mottagningen många gånger blir relativt svag här. Den här metoden lämpar sig därför bättre för användning utomhus eftersom mottagningen där generellt sett är bättre.

På grund av att omgivningen läses in på ett flertal olika sätt kommer även kartan att behöva ritas på olika sätt. Främst handlar det om ”Feature Maps” eller ”Occupancy Grids”. Det förstnämnda innebär att hela omgivningen inte ritas upp på kartan utan enbart objekt eller detaljer i omgivningen som kan användas som kännetecken eller ”landmarks”. Det här gör att kartan blir relativt svår att läsa för en människa eftersom omgivningen enbart symboliseras med utspridda punkter.

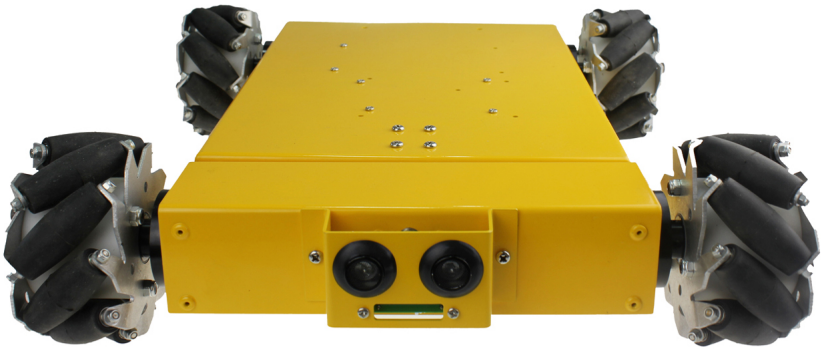
Den andra sorten, ”Occupancy grid”, är mer vad som kan förväntas när ordet karta hörs. Denna karta visar alla väggar och objekt som finns i omgivningen och markerar detta i ett mer eller mindre fint rutnät. Kartan i figur 2a är ett exempel på en sådan karta med ett relativt fint rutnät. Med denna karta kan människor enkelt bilda sig en uppfattning om hur det ser ut i omgivningen men det går inte att skilja på olika objekt i rummet. Exempelvis skulle en låg soffa se likadan ut som en vägg på kartan. Detta skapar lyckligtvis inga problem eftersom roboten kommer att anse att den låga soffan skulle vara ett hinder på samma sätt som en vägg. Problem kan dock uppstå med dynamiska hinder men med en tillräckligt snabb uppdateringsfrekens på kartan undviks detta.

3. Komponenter

Prototypen är uppbyggd av ett flertal olika komponenter som samverkar med varandra. Detta kapitel beskriver de viktigaste komponenterna, deras funktion och syfte.

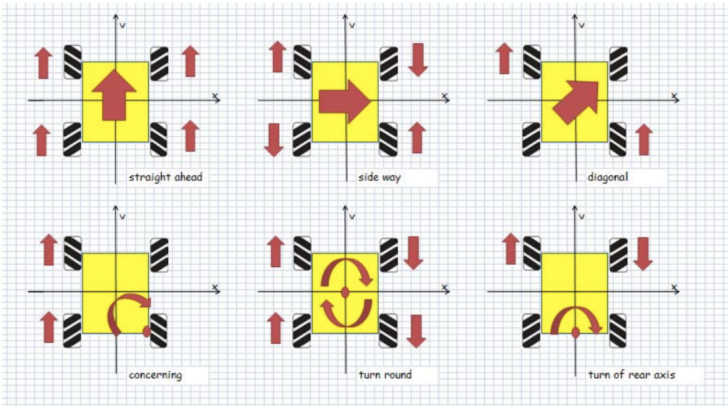
3.1. AGV - Mobile Arduino Robotics Car

Som plattform för hårdvarusystemet används en 4WD Mecanum Wheel Mobile Arduino Robotics Car 10011, hädanefter refererad till som AGV, vilket står för Automated Guided Vehicle. Detta är en mobil robot som drivs med stöd av Arduino. AGV:n visas nedan i figur 3 och har storleken 400x360x100mm.



Figur 3: 4WD Mecanum Wheel Mobile Arduino Robotics Car 10011, frontvy i fågelperspektiv. [7].

AGV:n har färdigmonterade ultraljudssensorer och mecanumhjul vilket möjliggör körning i flera olika riktningar. I figur 4 visas vilka riktningar AGV:n kör i beroende på åt vilka håll mecanumhjulen snurrar.



Figur 4: Rotationsriktningar hos hjulen för att skapa rörelse i olika riktningar. [8]

AGV:n har ett tillhörande programmeringsbibliotek som inkluderar förbestämda funktioner för styrningen av servomotorerna. Dessa funktioner kan alltså utnyttjas för att smidigare använda PWM för att styra motorerna och roboten på önskat sätt [5].

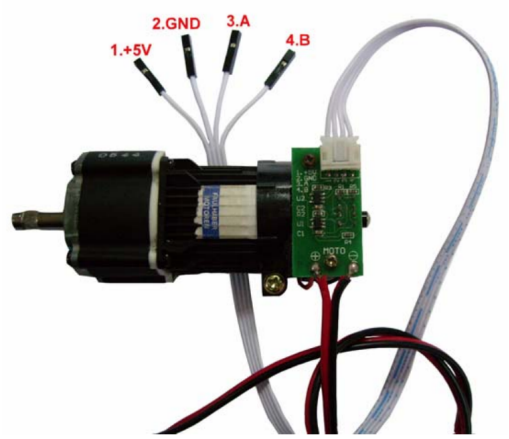
3.1.1. Arduino ATmega328

Till AGV:n medföljer en Arduino ATmega328 vilket är en mikrokontroller. Till mikrokontrollern kommer även en tilläggskort, monterad uppe på Arduinokortet, som syftar till att utöka antalet tillgängliga portar för att kunna driva roboten [9].

3.1.2. Servomotorer

AGV:n har fyra 12V DC servomotorer som driver mecanumhjulen och styrs med hjälp av program via Arduinokortet. De är tillverkade av Faulhaber och har 120 RPM som uthastighet efter en utväxling på 64:1. Dessa motorer är tillverkade sådana att rotorn inte har någon järnkärna vilket leder till att de är mer lämpade för snabb acceleration. Detta eftersom en rotor utan järnkärna väger mindre och har därmed ett lägre tröghetsmoment.

Varje servomotor har två digitala PWM stift (A och B) för pulssignaler som bestämmer hastighet. Två digitala I/O stift (+5V och GND) används för att bestämma rotationsriktningen. Motorn och dess fyra anslutningar visas nedan i figur 5.



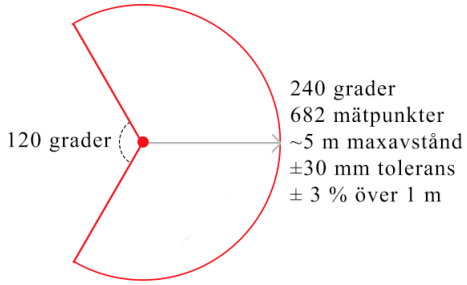
Figur 5: Till AGV:n följer fyra stycken servomotorer. En motor samt dess anslutningar visas ovan [5].

3.2. Batteri

Strömförsörjning till Arduinokortet och roboten kommer från ett uppladdningsbart blybatteri på 7 Ah. Detta är ett encellsbatteri på 12 V. Till batteriet används även en säkring på 30 A för att undvika skador på batteriet och komponenterna vid kortslutning [10]. Batteriet är relativt stort, dess mått är 151x65x97.5 mm och den väger 2.65kg.

3.3. LiDAR

För att samla information från omgivningen används en URG-04LX-UG01 LiDAR sensor från Hokuyo, vilken är en av Hokuyos billigare lösningar [11]. LiDAR:n skickar ut 682 ljuspulser i ett plan i olika riktningar i ett intervall på 240° och kan mäta avstånd på upp till 5 meter. Detta visualiseras nedan i figur 6.



Figur 6: Specifikationer över den LiDAR som används.

LiDAR:n mäter avståndet genom att mäta hur lång tid det tar för en utstrålad laserstråle att reflektera tillbaka från ett objekt till LiDAR:n. Avståndet räknas ut enligt ekvation 1.

$$\text{distance} = \frac{\text{speed of light} * \text{time of flight}}{2} \tag{1}$$

Ett problem med en billigare LiDAR kan vara att lasern inte har tillräckligt hög ljusstyrka. Enligt Hokuyo är LiDAR:n funktionell i en miljö med en ljusstyrka på maximalt 6000 till 10000 lux [12], vilka är ungefärliga värden för ljuset utomhus en molnig dag [13]. Ljusa inomhusmiljöer ligger på cirka 1000-5000 lux [13], vilket betyder att lasern från LiDAR:n inte kommer slås ut av omgivningens ljus. Om ljusstyrkan i miljön är för hög kommer sensorn störas och alltså inte uppfatta laserstrålens reflektion.

LiDAR:n kan inte hantera blanka ytor och fönster, vilket är bristfälligt för bruk i inomhusmiljöer. Dessa ytor kan göra att laserstrålen inte reflekteras tillbaka till sensorn och de misstolkas då för att vara öppna ytor där roboten kan köra. Den LiDAR som används till projektet kan ses nedan i figur 7.

Figur 7: Bild av den LiDAR som används i projektet. Strecket under ”URG” markerar vilken riktning som är 90 grader vänster. Den cylindriska delen är den del där lasern roterar inuti för att mäta avstånd, se det röda området för i vilka vinklar lasern verkar. [14].

3.4. Webbkamera

För bildigenkänning används en webbkamera, *PlaystationEye PS3*. Webbkameran tar bilder som har en upplösning på 640x400 eller 320x200 pixlar beroende på inställning, vilket betyder att bilderna blir relativt lågupplösta [15]. Anledningen till att lågupplösta bilder eftertraktas är att de processas effektivare när bilderna lagras och analyseras både med hänsyn till tid och datautrymme.

3.5. Bärbar dator

En bärbar dator är placerad på roboten för att försörja LiDAR:n med ström, sammankoppla komponenterna samt för att utföra beräkningar. Datorn som används är en ASUS ZenBook UX330CA som innehåller 8 GB Ram-minne, Intel Core m3-7Y30 processor samt använder Ubuntu 17.10, 64 bit som operativsystem [16].

4. Metod

Detta kapitel beskriver förarbetet samt metodiken bakom arbetet. Kapitlet täcker även vissa utmaningar som påträffats och hur dessa har angripits.

4.1. Utveckling av prototypen

I följande avsnitt redovisas arbetsgången för att ta fram koncept för en robot vars funktion var att hitta krukväxter i en okänd miljö. Konceptet togs fram med hjälp av en målgruppsanalys, konceptgenerering samt de ingenjörsmässiga verktygen morfologisk matris och elimineringsmatris.

4.1.1. Målgruppsanalys

Tidigt i arbetet gjordes en målgruppsanalys som låg till grund för alla beslut under framtagandet av mål och konceptval. Målgruppen togs fram genom att reflektera kring vilka som skulle behöva och därmed vore intresserade av en blomvattnarrobot.

En slutgiltig blomvattnarrobot är ämnad för den bekväma hemägaren som inte vill behöva tänka på sina krukväxter vid semesterresor eller andra tillfällen med förlängd frånvaro från hemmet. Till skillnad från andra lösningar ska denna vara helt automatisk och kunna flyttas till nya hem eller platser utan att behöva ytterligare konfigureringar.

4.1.2. Morfologisk matris

Innan konceptgenereringen genomfördes definierades fyra önskvärda delfunktioner.

- **Avsökningsmetod**: Ett sätt för roboten att få information om hur stor omgivningen är.
- **Undvika hinder**: Ett sätt för roboten att få information kring omgivningen för att inte kollidera med hinder.
- **Identifiera kruka**: Ett sätt att säkerställa att roboten hittar krukväxter
- **Energi till kruka**: Vissa idéer krävde någon typ av strömförsörjning till krukan och i andra fall inte.

När delfunktionerna hade tagits fram genererades dellösningar till respektive delfunktion under en workshop. För att kombinera dellösningarna till fullständiga koncept användes en

morfologisk matris 1. De vita cellerna i matrisen representerar lösningarna som togs fram till varje delfunktion.

Tabell 1: Den morfologiska matris som används för att generera olika koncept. Andra kolumnen listar önskade funktioner och de vita cellerna är olika förslag på lösningar. Ett fullständigt koncept är en lösning för varje funktion.

Morfologisk matris		1	2	3	4	5	6	7
A	Avsökningsmetod	Fördefinierad rutt	Följa vägg	Rum-scanning före aktivering	Slumpvis körning	Rum-scanning i realtid		
B	Undvika hinder	IR	Känselspröt	LiDAR	Radar	Ultraljud		
C	Identifiera kruka	Frekvens (Specifik för varje kruka)	Kamera (Streck-kod eller QR-kod på krukan)	Kamera (Färg på kruka)	Kamera (Bildigenkänning)	Position på kruka (Krukan står på en specifik plats i rummet)	Aktiv RFID i kruka	Semiaktiv RFID i kruka
D	Energi i kruka	Batterier	Ej nödvändigt	Sladdar	Solceller			

En fullständig lösning tas fram genom att med hjälp av korsbefruktning generera en kombination av dellösningar till respektive delfunktioner. Ett exempel på en fullständig lösning är A1+B2+C3+D2 vilket resulterar i ett koncept där roboten har en fördefinierad rutt, undviker hinder med hjälp av känselspröt och hittar krukväxter genom att leta efter en specifik färg som bara krukor skulle ha. Den morfologiska matrisen kan på detta sätt generera 700 möjliga lösningar, men många av dessa lösningar är inte realiserbara eller passar inte den tänkta målgruppen och eliminerades direkt med hänsyn till det. Därmed togs endast 11 koncept fram ur matrisen, de 11 koncepten kan ses i bilaga A.

4.1.3. Eliminering och val av koncept

För att bestämma ett slutgiltigt koncept användes en elimineringsmatris med fem olika kriterier, A-E, vilka behövde uppfyllas. Dessa kriterier står listade i tabell 2 nedan.

Tabell 2: Kriterier som har använts för vid konceptval i elimineringsmatrisen.

#	**Kriterium**
A	Löser huvudproblemet (hitta krukväxt)
B	Uppfyller målen
C	Realiserbar
D	Inom kostnadsramen
E	Tillräcklig information finns

Elimineringsmatrisen kan ses i bilaga B där konceptet stryks om det inte uppfyller samtliga kriterier.

Kvar återstod fyra koncept som viktades beroende vad som ansågs vara mest realiserbart och intressant att utföra. Med hänsyn till tillgängliga resurser som kameror, LiDAR och en färdig plattform att bygga roboten på valdes Koncept 10, se tabell 3.

Tabell 3: Den kombination av dellösningar som skapar det valda konceptet.

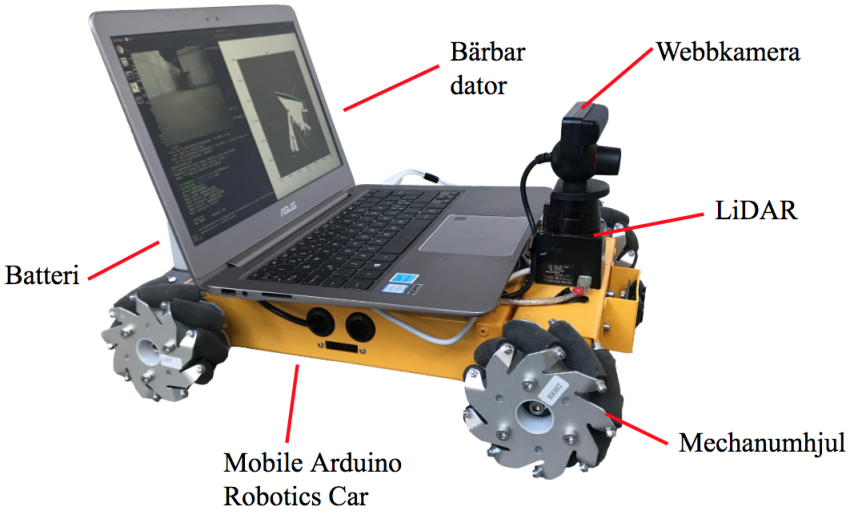
Koncept 10		
A	Avsökningsmetod	Rum-scanning i realtid
B	Undvika hinder	LiDAR
C	Identifiera kruka	Kamera (Bildigenkänning)
D	Energi i kruka	Ej nödvändigt

Det valda konceptet innebar en kombination av dellösningarna; att scanna rummet i realtid, navigera och undvika hinder med hjälp av en LiDAR, samt identifiera krukväxterna med kamera och bildigenkänning.

4.2. Montering av prototyp

De viktigaste komponenterna som användes för att ta fram en prototyp baserat på det valda konceptet finns listade i kapitel 3.

Prototypen baserades på en 4WD Mecanum Wheel Mobile Arduino Robotics Car 10011 som beskrivet i 3.1. På roboten utfördes enklare modifikationer för att anpassa den till projektet. Ultraljudssensorerna togs bort för bättre kabelhantering och längs långsidorna monterades ett datorställ. Längst fram i mitten på roboten monterades LiDAR:n. Ovanpå LiDAR:n fästes en webbkamera och längst bak på roboten ett batteri. Samtliga komponenter var monterade med hänsyn till LiDAR:n för att inget skulle blockera någon av LiDAR:ns mätpunkter. I figur 8 nedan visas roboten och dess komponenter.

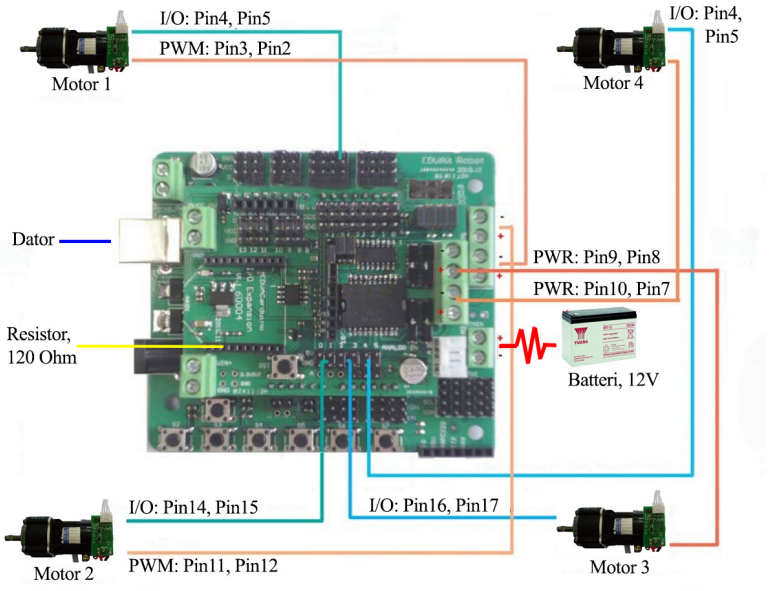


Figur 8: Prototypen med komponenter utmärkta. På datorskärmen skymtar bild från webbkameran och en uppritad karta av omgivningen.

4.2.1. Kopplingsschema för Arduino

I figur 9 nedan visas ett illustrerat kopplingsschema för Arduinokortet. Som tidigare nämnts användes två I/O stift vilka fästes vid de ljusblå och gröna linjerna till kortet, samt två PWM-stift som fästs vid de brandgula linjerna. Datorn som används kopplas seriellt till Arduinokortet via en USB-B-port. Batteriet kopplas till Arduinokortet via en säkring och en strömbrytare i porten från det mörkröda strecket.

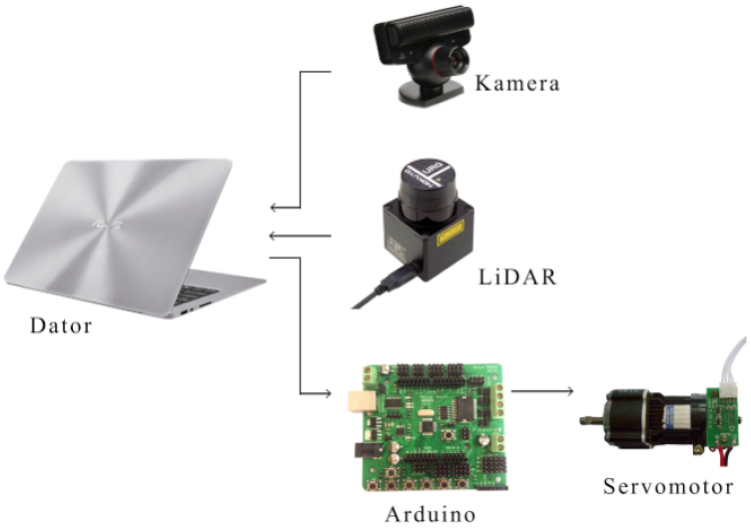
En resistor på 120 Ω placerades mellan 5 V- och Reset-stiften för att möjliggöra den seriella kommunikationen. Utan denna resistor startas programmet om vid varje skickat kommando från datorn, vilket gör det omöjligt att köras.



Figur 9: Schematisk bild av vilka portar servomotorer, datorn, resistorn och batteriet är kopplade till på arduinokortet.

4.3. Mjukvara

All styrning sker via en bärbar dator som utgör en central för all mjukvara på roboten. Att ha styrningen via en dator valdes tidigt i projektet för att det var ett smidigt sätt att få komponenterna att samverka med varandra. Majoriteten av programmeringen skrevs i Python eftersom detta språk var kompatibelt med komponenterna, mjukvaran och datorn som användes. Till datorn var LiDAR:n, webbkameran och Arduinokortet kopplade. Arduinokortet utförde ingen styrlogik själv utan mottog enbart kommandon från datorn och skickade styrsignaler vidare till servomotorerna, se figur 10 för dataflöde mellan komponenterna.



Figur 10: Hur dataflödet för att generera styrsignaler till servomotorerna ser ut.

4.3.1. Servostyrning

Tidigt i arbetet låg fokus på den seriella kommunikationen mellan Python och Arduino. Med hjälp av denna kommunikation kunde alla komponenter inklusive servomotorerna styras via ett huvudprogram skrivet i Python. Arduinokoden formades för att endast mottaga enstaka bytes från huvudprogrammet i Python och översätta detta till kommandon för PWM-styrning. Det går alltså att skicka variabler med värden mellan 0-255 vilket översätts till kommandon i Arduinokoden. Exempelvis om en byte skickas som motsvarar siffran ”1” levererar Arduinokortet den PWM-styrning vilken resulterar i att AGV:n kör framåt.

Till AGV:n fanns ett bibliotek implementerat för att kunna styra AGV:n. Biblioteket hade några inbyggda funktioner, bland annat en PID-regulator vilket reglerar automatiskt PWM-styrningen. I början av projektet användes PID-regulatorn för styrning av servomotorerna, men den gav oönskat resultat och används därför inte i projektet, mer om detta i 6.1.

Istället för att låta PID-regulatorn reglera PWM-styrningen angavs manuella värden för att för att få konsekvent beteende för styrningen.

4.3.2. SLAM implementation - navigering i en okänd miljö

Istället för att spendera resurser på att implementera en SLAM-algoritm från grunden valdes ett existerande bibliotek för Python, BreezySLAM. Detta är ett bibliotek med öppen källkod som valdes eftersom det har stöd för LiDAR:n. BreezySLAM använder data från LiDAR:n för att tillhandahålla enkla funktioner som beräkning av robotens position och körriktning. Eftersom LiDAR:n mäter med en 240° vinkel kan SLAM-algoritmen räkna ut var roboten befinner sig i förhållande till mätpunkterna.

4.3.3. Styralgoritm

Med hjälp av servostyrningen och SLAM-implementationen har de algoritmer för den autonoma styrningen av AGV:n. Dessa algoritmer har iterativt arbetats fram under hela projektets gång.

Under projektets gång insågs snabbt att SLAM-biblioteket hade lite svårt att hänga med när roboten roterade. Därför ändrades styrkoden för att låta roboten rotera med LiDAR:n som mittpunkt.

För att få roboten att åka mot specifika vinklar behövdes en felmarginal då det beräknade vinkelvärdet angavs med många decimaler. Ifrån början användes en vid felmarginal på 15° åt båda hållen eftersom att roboten hade ett dåligt kullager på ett hjul och drog kraftigt åt ena hållet. Felmarginalen sänktes när kullagret byttes ut på det dåliga hjulet.

För att kunna sänka felmarginalen till de 5° som nu används behövde roboten hinna med att märka när den var riktad åt rätt håll i tid. Därför roteras roboten några få grader åt gången för att försäkra sig om att den inte åker förbi angiven vinkel.

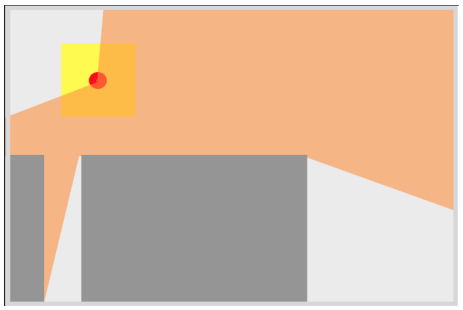
Eftersom roboten ska demonstreras på en specifik känd plats, hålet i studiehallen, har tester utförts där. Hålet är plant, fast med stora glasfönster ut. Dessa övertäcks av stolar vid tester av roboten.

4.3.4. Undvikande av trånga utrymmen

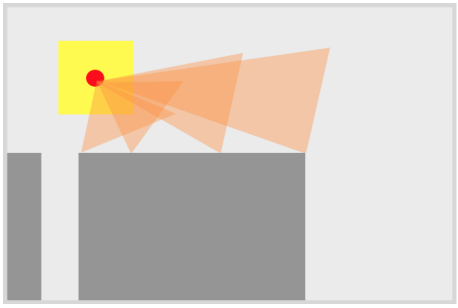
LiDAR:n mäter avståndet var 0.35 grad över 240 grader. Mätpunkterna sparas i en lista som används för att bilda en karta som roboten använder för att navigera i miljön, se de aprikosa området i figur 11a. För teckenförklaring se figur 11d.

Roboten programmerades till att åka mot det längsta avståndet som finns i listan trots att det avståndet kanske bara hittades i en av mätpunkterna i listan. Detta gjorde att roboten fastnade i att försöka åka mot platser som låg bortom smala passager där roboten inte kunde

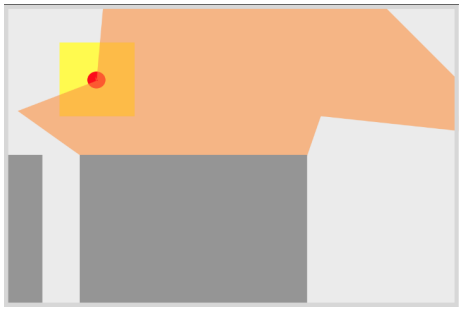
åka igenom, se längst ner i vänster hörn i figur 11a. För att åtgärda detta ändrades listan med mätpunkter. Punkterna analyserades i grupper istället för individuellt, och samtliga punkter i gruppen tilldelades det kortaste värdet i gruppen, se figur 11b. Storleken på grupperna varierades beroende på mätpunkternas avstånd för att garantera en passage som var tillräckligt bred för att roboten skulle kunna passera. Den modifierade listan genererade en ny karta för roboten att köra efter, se det aprikosa området i figur 11c.



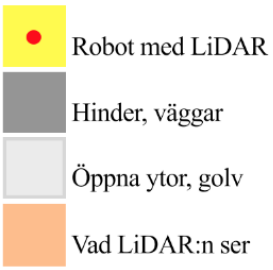
(a) Det aprikosa området är alla de mätpunkter som fås av LiDAR:n.



(b) Mätpunkterna modifieras för att garantera att roboten kan passera.



(c) Det aprikosa området visar den nya kartan som roboten kör efter.



(d) Teckenförklaring för resterande bilder i figuren.

Figur 11: Vad roboten ser när den är stillastående.

4.3.5. Förbehandling av LiDAR-data

För att underlätta för styralgoritmen förarbetas LiDAR-datan för att eliminera störningar och underlätta för styrlogiken. Tanken är att transformera den uppmätta avståndsvektorn till en vektor med längsta avstånd som roboten får plats att åka till, se avsnitt 4.3.4. Eftersom störningar med för korta uppmätta avstånd kan sabotera för algoritmen sätts alla mätvärden som är kortare än 5 cm till 10 meter. Detta eftersom LiDAR:n mäter lite över 5 cm till objekt som ligger mot LiDAR-sensorn.

4.3.6. Bildigenkänning - Identifiera krukväxter

Webbkameran som tar bilder av omgivningen styrs via den bärbara datorn. För att få reda på om det fanns någon krukväxt i miljön analyserades bilden med hjälp av egenskrivna program i Python.

Det första som hände var att programmet initierade en direktsändning av kamerans videoinspelning som fungerade som gränssnitt i utvecklingsfasen. Kameran tar en bild som sparas i jpg-format på en virtuell hårddisk. Innehållet i bilden analyseras sedan med hjälp av Google CV. Att använda en virtuell hårddisk var nödvändigt eftersom Google CV kräver att bilden skickas upp via länk.

Google CV returnerar endast svar för ett fåtal objekt som, med stor säkerhet, finns i bilden vilket leder till två problem. Det första är att det inte går att vara säker på om ett föremål verkligen inte finns i bilden eller om det bara inte är ett toppresultat. Det andra är att det saknas information om var i bilden krukväxten befinner sig. Båda dessa funktioner behöver fungera för att öka tillförlitligheten av identifieringen.

För att identifiera var någonstans i bilden eventuella krukväxter finns delades varje bild upp i mindre segment. Varje segment laddades upp till den virituella hårddisken och analyserades sedan separat.

Empiriskt laborerande med hänsyn till avstånd till krukväxten samt bildens skärpa (se avsnitt 4.4) resulterade i ett rutnät av segment, alltså både horisontell och vertikal uppdelning av bilden. Detta rutnät gör det möjligt att:

- räkna ut åt vilket håll roboten ska åka.
- öka antalet identifierade objekt, alltså minska risken för att roboten missar någon krukväxt.

När segmenten returnerats laddas de upp mot molnet tillsammans med sökord för eftertraktade objekt. De sökord som valdes för att identifiera krukväxter är följande; *plant, flower, flowerpot, herb, houseplant, leaf* och *flora*.

När bilden har analyserats returnerar programmet vilka sorters objekt som påträffats, samt sannolikheten för dessa. Etiketterna jämförs sedan med sökorden och ett svar om bilden har funnit något av sökorden returneras utifrån denna jämförelse.

4.4. Test för bildigenkänning

För att ta reda på hur många delar som bilden behöver delas upp i för att Google CV inte ska missa någon krukväxt gjordes följande test. En kruka placerades på ett bestämt avstånd från kameran i en oordnad miljö för att simulera en hemmiljö. De sju bilder som används i testet ses nedan i figur 12.

(a) Kruka på 2.7 m från kameran.

(b) Kruka på 2.1 m från kameran.

(c) Kruka på 1.7 m från kameran.

(d) Kruka på 1.45 m från kameran.

(e) Kruka på 1.1 m avstånd från kameran.

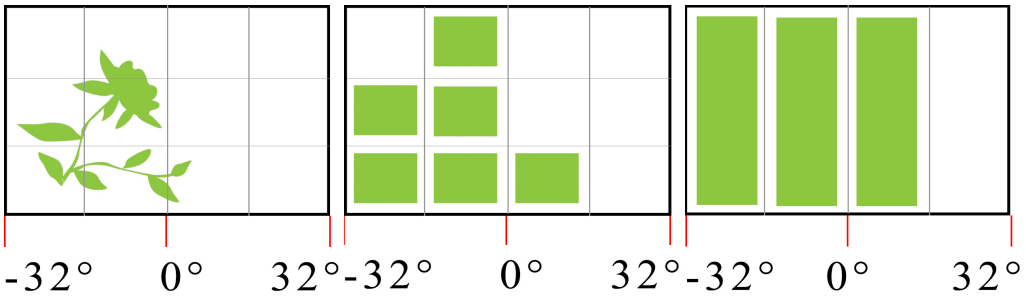
(f) Kruka på 0.7 m från kameran.

(g) Kruka på 0.4 m avstånd från kameran.

Figur 12: Bilder som har använts för att testa bildigenkänning av krukväxter som befinner sig på olika avstånd från kameran.

Varje testbild delades flera gånger upp i fler och därmed mindre segment där varje segment laddades upp och analyserades separat. Resultatet varje uppladdning skulle returnera var huruvida Google CV hittade en krukväxt eller inte, alltså om svaret innehöll något av nyckelorden som nämndes i avsnitt 4.3.6.

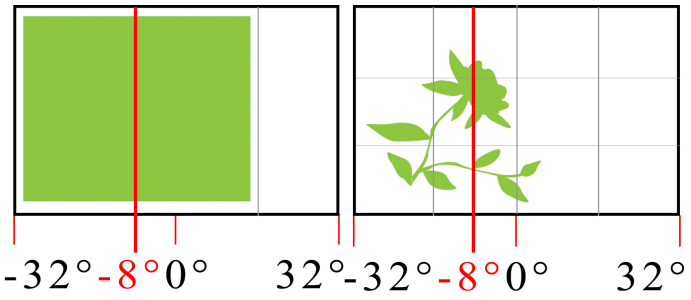
I figur 13a delas ursprungsbilden av en blomma upp i 3x4 segment. Resultatet efter att samtliga segment skickats till Google CV visas i figur 13b där en identifierad blomma symboliseras av en ruta med grön ram. Eftersom roboten inte tar hänsyn till blommor på olika höjd markeras kolonnerna med grönt där krukväxter identifierats. Hur det här kan se ut visas i figur 13c. Kameravinkeln i horisontalplanet är 64° och mitten av bilden sätts till 0° alltså rakt fram för roboten. För att kunna returnera en vinkel till vilken roboten ska rotera läggs de markerade kolonnerna ihop och ett medelvärde beräknas. Det här visas i figur 13d. I detta fall skulle roboten rotera 8° moturs för att sikta in sig på blomman som i figur 13e.



(a) Ett exempel på en bild med en blomma som skickas till Google CV.

(b) Det svar som Google CV ger där de gröna rutorna visar i vilka segment det finns en blomma.

(c) Segmenten gör om till kolonner för att beräkna vilken vinkel roboten ska rotera till.



(d) Medelvärdet av kolonnerna ger målvinkeln för roboten.

(e) Förtydligande för att visa den beräknade vinkeln mot blomman.

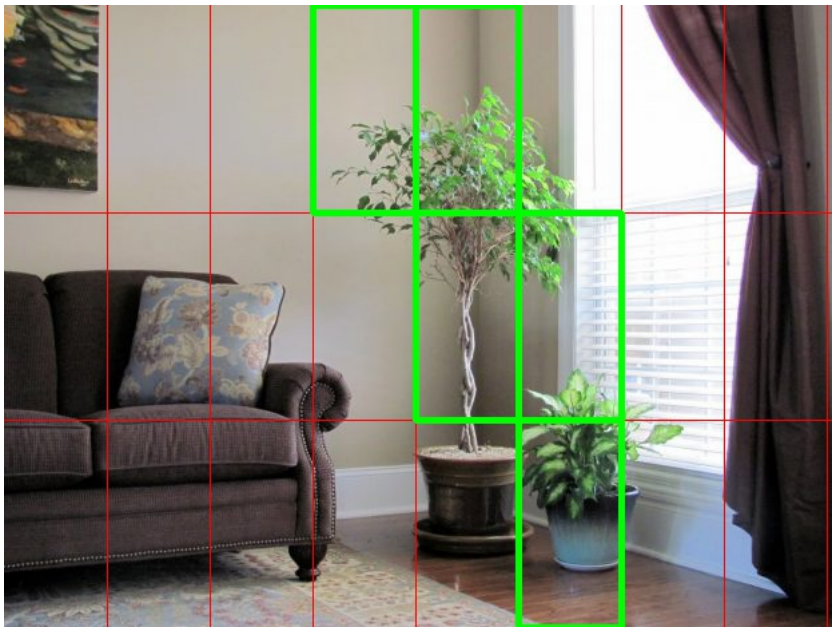
Figur 13: Bildserie som visar hur vinkel till en identifierad krukväxt tas fram med hjälp av segmentering.

5. Resultat

I detta kapitel belyses resultat av tester och utveckling av prototypen samt redovisar projektets slutkoncept gällande identifiering och navigering.

5.1. Bildigenkänning av krukväxter

Med hjälp av egenkonstruerad kod som använder sig av Google CV samt webbkameran genomförs identifiering av krukväxter. Vid varje bildtagning delas bilden upp i ett rutnät bestående av 24 olika segment, tre vertikala och åtta horisontella uppdelningar, som visas i figur 14 nedan. De gröna rutorna i bilden representerar de segment där Google CV returnerade att det fanns en krukväxt.



Figur 14: En exempelbild över hur ett resultat från Google CV kan se ut. De gröna rutorna visar att i dessa segment finns det en krukväxt. [17]

Med hjälp av denna uppdelning kan roboten räkna ut dess vinkel gentemot krukväxten samt ökar sannolikheten att hitta krukväxter på längre avstånd vilket belyses i underkapitlet nedan. Kameran har som tidigare nämnts en horisontalvinkel på 64° och med antagandet att den horisontella vinkelförändringen är linjär är det möjligt att räkna ut vinkeln till varje segment vilket beskrevs i slutet av kap 4.3.6.

Tack vare bildens låga upplösning går det snabbt att lagra bilderna på den virtuella hårdisken samt behandla bilderna i Google CV. En uppladdning av en bild, vilket motsvarar tjugofyra enskilda segment, testades och tar ungefär 2 sekunder från bildtagning till resultat.

5.1.1. Längsta avstånd för att identifiera krukväxt

Tabell 4 nedan visar huruvida en krukväxt påträffades eller inte beroende på avståndet mellan krukväxten och kameran samt hur många segment som bilden delades upp i (rader x kolumner). De röda respektive gröna cellerna visar huruvida en krukväxt identifierades eller inte. För samtliga positiva svar kontrollerades att det var just krukväxten eller delar av denna som identifierades.

Tabell 4: Resultat från tester för att undersöka antalet segment bilden ska delas upp i. Röd cell med ”-” betyder att ingen krukväxt identifierats och grön cell med ”X” betyder att en krukväxt identifierats.

Segment	2.7 m	2.1 m	1.7 m	1.45 m	1.1 m	0.7 m	0.4 m
2x1	-	-	-	-	X	-	X
2x2	-	-	-	-	-	X	X
2x3	-	-	X	X	X	X	X
2x4	-	-	X	X	X	X	X
2x5	-	-	X	X	X	X	X
2x6	-	-	X	X	X	X	X
2x7	-	-	X	X	X	X	X
2x8	X	-	X	X	X	X	X
2x9	-	-	X	X	X	X	X
3x1	-	-	-	-	X	-	X
3x2	-	-	-	-	-	X	X
3x3	-	-	X	-	X	X	X
3x4	-	X	X	X	X	X	X
3x5	-	-	-	X	X	X	X
3x6	-	X	X	-	X	X	X
3x7	-	-	X	X	X	X	X
3x8	-	X	X	X	X	X	X
3x9	-	X	X	X	X	X	X

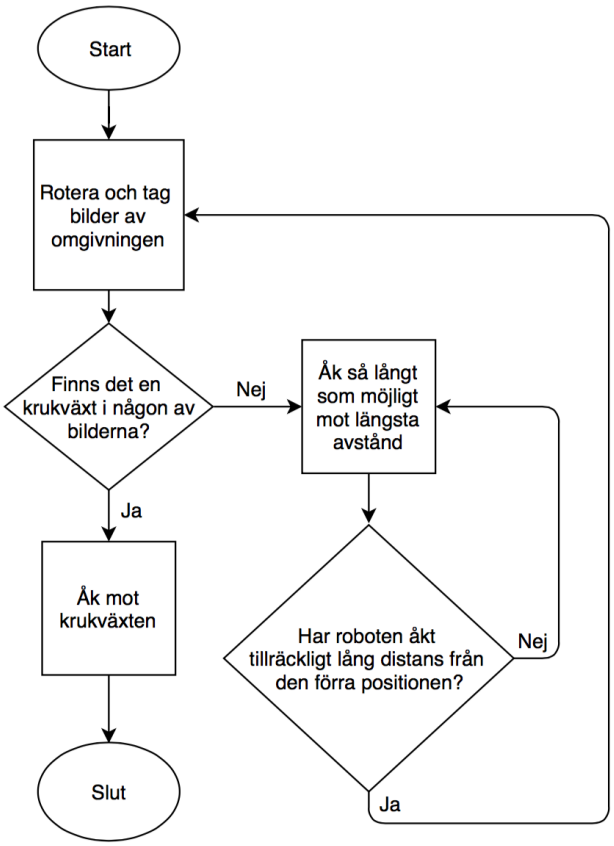
När avståndet ligger i intervallet 0.4 m till 2.1 m går det alltså att identifiera krukväxter med flera olika kombinationer av rader och kolumner.

5.1.2. Optimal rutnätstorlek

Tabell 4 ovan påvisar att rutnätstorlekarna 2x8, 3x4, 3x8 samt 3x9 ger identifiering av krukväxter på flest olika avstånd. För 2x8 förekommer dock ett glapp på avståndet 2.1 meter och valdes således bort till förmån för ett kontinuerligt intervall som till högre grad uppfyller målet 2 m. Av de tre återstående valdes de nät som hade stort antal kolumner, alltså 3x8 eller 3x9, för att dessa ger möjligheten att mer precist ange vinkel till roboten.

5.2. Styrning

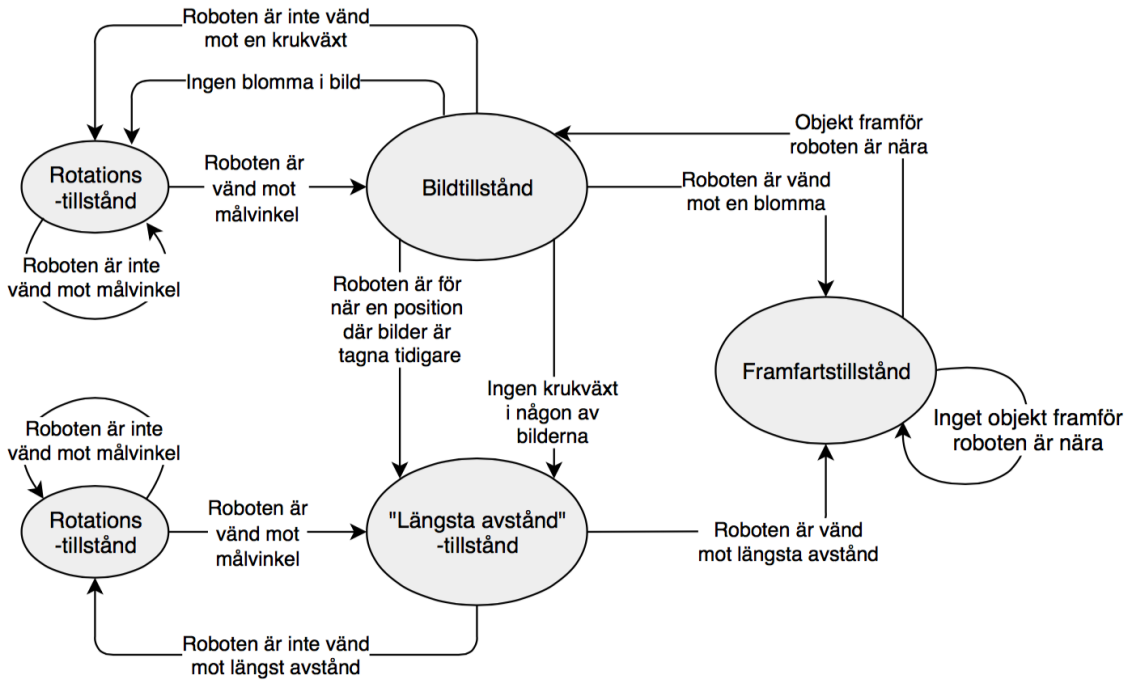
Robotens initialtillstånd är att fotografera omgivningen och söka efter krukväxten. Detta gör den genom att rotera på plats och ta bilder varje 60° eftersom bildens horisontalvinkel är ungefär detsamma. Roboten fortsätter ta bilder tills den hittar en krukväxt eller roterat sex gånger, alltså ett helt varv. När roboten hittar en krukväxt så avbryts rotationen för att navigera mot den. Om roboten inte hittar någon krukväxt åker den istället i riktningen med längst distans som den mäter med LiDAR:n. Ett krav i detta tillstånd är att roboten måste köra två meter från ursprungspositionen innan den tar kort igen. Detta är nödvändigt för att säkerställa att roboten kommer undersöka områden som inte setts innan. När roboten till slut hittar en krukväxt i bilden åker den framåt tills den är tillräckligt nära, då avslutas programmet. Ett övergripande flödesschema för hela processen ses nedan i figur 15.



Figur 15: Övergripande flödesschema för robotens styralgoritmer.

För att åstadkomma detta resultat kör roboten två processer parallellt. Den första processen uppdaterar positions- och vinkelvariabler med hjälp av SLAM-biblioteket och den andra använder dessa variabler för att bestämma vad roboten ska göra. Den sistnämna kommer fortsättningsvis kallas för styrtråden.

Styrtråden är i grund och botten en tillståndsmaskin som utför olika beräkningar och tillståndsbyten i varje steg baserat på sitt nuvarande tillstånd. Tillståndsmaskinen har bildtillståndet som ursprungstillstånd och följer en pil i figur 16 för varje steg. Stegen sker regelbundet, men kan ta olika lång tid då processorn är upptagen med annat. Alla tillstånden förklaras utförligt nedan.



Figur 16: Graf över tillståndsmaskinen. Varje nod motsvarar ett tillstånd och varje pil motsvarar ett steg. Bildtillståndet är ursprungsläget vid start.

5.2.1. Rotationstillstånd

Rotationstillståndet är ett hjälptillstånd för andra tillstånd och används för att rotera roboten mot en målvinkel. Vid varje steg roterar roboten ungefär fem grader i en given riktning. Om roboten roterat till den givna målvinkeln med en felmarginal på max fem grader återvänder programmet till det föregående tillståndet. För att veta vilket det föregående tillståndet är har varje tillstånd som behöver hjälp ett eget rotationstillstånd.

5.2.2. Framfartstillstånd

Så länge roboten befinner sig i framfartstillståndet åker den framåt. För varje steg avgör roboten hur långt det är till objekt och framförvarande hinder. Om föremålet framför roboten är inom 80 cm börjar roboten bromsa in och går in i bildtillståndet.

5.2.3. ”Längsta avstånd”-tillstånd

Det längsta avståndet baserat på den förbehandlade LiDAR-datan väljs och avståndet lagras. Roboten roterar mot det valda avståndet med rotationstillståndet. Skulle det finnas ett avstånd i den nya transformerade avståndsvektorn som är minst 80 cm längre än det sparade avståndet sparas det nya värdet som längsta avstånd, varpå roboten roterar mot denna riktning. Detta repeteras tills roboten står vänd mot vad den bedömt är det absolut längsta avståndet. När den roterat klart byter roboten till framfartstillståndet.

Skulle det längsta avståndet vara mindre än 80 cm backar istället roboten i 1,5 sekunder för att ta sig ur situationen.

5.2.4. Bildtillstånd

Vid uppstart av roboten så startas bildtillståndet. Detta har som första uppgift att avgöra om roboten befinner sig för nära en punkt där den tidigare tagit bilder. Om den gör det går roboten direkt till ”längsta avstånd”-tillståndet. Annars lägger roboten nuvarande position på minnet och börjar ta bilder. Roboten avgör detta endast vid den första av de bilder den tar i samma position.

Varje gång roboten ska ta bilder tar den sex bilder med 60° rotation mellan varje bild för att täcka ett helt varv. Finner roboten en krukväxt tidigare än den sista bilden avbryts resten av processen till förmån för att åka direkt mot krukväxten. Roboten roteras mot krukväxten tills denna bedöms vara rakt fram. En felmarginal på fem grader används för att undvika alltför stor rotation. Om roboten avgör att krukväxten är för långt bort byter den till framfartstillståndet. Är krukväxten tillräckligt nära har roboten hittat krukväxten och därmed är roboten färdig.

6. Diskussion

Efter slutfört projekt har vissa mål uppnåtts och andra prioriterats bort. Med hjälp av segmentering för bildigenkänningen, vilket presenterades i avsnitt 5.1.1, är det möjligt att konsekvent identifiera krukväxter inom 2.1 m avstånd. Vidare tester har visat att roboten även kan hitta krukväxter på ännu längre avstånd. I samma avsnitt visas att roboten kan hitta krukväxter med ett avstånd på 0.4 m. Målet som sattes, att hitta krukväxter inom en halv meter till två meter, anses därmed ha uppnåtts.

Med hjälp av de algoritmer, som tidigare presenterats, som syftar till att styra och undvika hinder är det nu möjligt att manövrera fram till identifierade krukväxter utan kollision. Styralgoritmerna tillåter roboten att köra i okända områden och förändringar i rummet mellan körningar ska inte påverka resultatet.

Roboten har i nuläget inställningen att hållas 80 cm från hinder men kan enkelt ändras till ett kortare eller längre avstånd. Eftersom datan från LiDAR:n uppdaterar långsamt behövs en felmarginal på $\sim$30cm för att reglera att roboten stannar på önskat avstånd. Detta resulterar alltså i att roboten stannar mellan 40-50cm från hinder eller krukväxten.

Det mål som inte uppnåtts är att roboten ska känna av och sluta köra när den lyfts upp. Eftersom roboten inte har någon gyro- eller accelerometer inkopplad är det inte möjligt att registrera att roboten lyfts och får därmed inget kommando att stanna. På grund av prioriteringsändringar har detta avgränsats i ett sent skede.

Tidtagningen för att hitta och navigera till krukväxter har tyvärr inte utförts med kvantitativa tester. På mässområdet där roboten ska demonstreras har vissa tester utförts där resultatet har varierat stort beroende på vart krukväxten placeras. I de flesta fall har roboten dock klarat målet, att hitta och navigera till krukväxt, men utan tester går det inte bekräfta att det alltid stämmer.

6.1. Återstående problem

Alla problem som uppdagats under framtagningen av prototypen beskriven i denna rapport har inte kunnat lösas. Ett omfattande problem som varit med från början är de oregelbundna servomotorerna. I startskedet fanns åtta motorer att tillgå men endast tre av dessa fungerade korrekt. Resterande fungerade inte alls eller presterade under förväntan. Den lämpligaste lösningen på detta problem är att implementera ett reglersystem för att motverka dessa oregelbundenheter. Ett sådant system skulle då regleras baserat på hastigheten hos de olika hjulen. Eftersom det inte varit möjligt att mäta hastigheten på varje hjul separat har det heller inte gått att skapa ett reglersystem för motorerna.

Vidare har det noterats att prototypens högsta punkt sträcker sig ovanför den nivå som LiDAR:n ger mätvärden på. Det här innebär att det finns en risk att prototypen kör in i saker eftersom de inte syns i LiDAR:ns mätdata. Samma problem uppstår om det är något föremål som ligger under LiDAR:ns laserstrålar. Dessa problem uppstår eftersom det tidigt i projektet gjordes en avgränsning att inte ta hänsyn till höjdskillnader.

På vissa underlag blev det tydligt att motorerna inte var kraftfulla nog att driva roboten. Detta för att hänsyn inte tagits till vikten hos datorn och det nya batteriet. Avsaknaden av kraft behöver dock inte bero på att motorerna i sig är för svaga utan det är mer sannolikt att utväxlingen är för hög. Detta har resulterat i att roboten är väldigt känslig mot skarvar och ojämnheter.

6.2. Ej genomförda moment

På grund av prioriteringsändringar har vissa moment och idéer under projektets gång valts bort men nämns ändå till förmån för eventuella arbeten i framtiden.

6.2.1. Lagring av SLAM-karta

En av dessa idéer är lagringen av SLAM-kartan som skulle syftat till att bekräfta att hela rummet har registrerats. När roboten har registrerat hela rummet samt vattnat alla krukväxter skulle den kunna gå in i viloläge. Till en slutprodukt hade detta varit önskvärt för att roboten bara ska behöva lära sig rummet en gång. I kombination med detta skulle även krukväxter kunna lagras för att underlätta och effektivisera körningen. Skulle rummet möbleras om eller roboten flyttas till annat rum återställs lagringen. Samma gäller även för krukorna om de skulle flyttas eller tas bort.

6.2.2. Säkerhet

I dagsläget finns det inga säkerhetsspärrar implementerade på roboten. Eftersom det är en prototyp ansågs inte dessa tillräckligt viktiga och prioriterades bort med fördel för andra, mer vitala, moment.

Diskussioner har dock förts vilket resulterat i diverse idéer kring potentiella säkerhetsförbättringar. Exempelvis skulle ett system för att stänga av roboten då den lyfts upp behövas. Tanken var att lösa det här problemet med hjälp av ett gyroskop för att registrera rörelse i höjdled.

6.2.3. Dynamisk segmentering

Det diskuterades även tidigt att göra uppdelningen av segment dynamisk beroende på vilket avstånd roboten har till väggen. Uppdelningen av bilden spelar stor roll för att kunna hitta krukväxter, detta presenteras i avsnitt 4.4, tabell 4. Fler uppdelningar innebär att objekt på längre avstånd kan urskiljas men också att processen tar längre tid. Vid kortare avstånd är det lika bra, om inte bättre, att ha färre uppdelningar för att hitta en krukväxt. Med hjälp av att göra systemet dynamiskt effektiviseras processen men det kan inte sägas ifall det blir dyrare eller billigare än nuvarande lösning utan vidare tester. Ifall det ofta är långa avstånd kommer det skapas fler segment vilket resulterar i att fler bilder skickas till Google CV.

6.3. Projektets bidrag

Projektet kan anses kunna bidra en liten del till teknikutvecklingen av nya produkter. En viktig del är att prototypen väldigt enkelt kan konfigureras till att identifiera i princip vad som helst. Av att endast ändra nyckelorden som programmet för bildigenkänning ska söka efter kan roboten på ett flexibelt sätt hitta alla möjliga typer av objekt. Det vill säga de objekt som Google CV:s neurala nätverk har lärt sig att hitta.

På detta sätt kan nya produkter utvecklas med hjälp av samma typ av teknik för att kunna identifiera ett önskat objekt. Exempelvis kan nyckelord som *”laundry”* och *”washing machine”* användas till en framtida tvättrobot. Dessa funktioner skulle också kunna kombineras på grund av enkelheten att ändra nyckelord, och därmed användas för att skapa en fullständig hemmarobot som utför samtliga sysslor som kan behövas.

Något annat som projektet bidragit med är grunden till styrningen av roboten i okända rum. I nuläget kan inte roboten känna av när den sett hela rummet, men med samma teknik och komponenter finns potentialen att uppnå detta. Denna styrningsalgoritm skulle då kunna appliceras till liknande projekt men också helt andra projekt som har behov av att utforska hela områden.

Ett stort fokus ligger på att utveckla autonoma robotar i nuläget [18] och utvecklingen av olika aspekter till detta är vad som driver tekniken framåt. Detta projekt kommer troligen inte bidra enormt mycket till teknikens framfart, men genom att ta fram en grund för nya områden inom den autonoma tekniken skapas en potential för framtida större verk.

6.4. Rekommendationer till vidare utveckling

Under projektet spekulerades det mycket kring potentiella vidareutvecklingar med framtida kandidatarbeten i fokus men även för den kommersiella slutprodukten.

6.4.1. Utveckling - Identifiering

Storleken på de segment bilderna delas in i är i nuläget till ett fixt värde. Det här är inte alltid optimalt eftersom avstånden i bilden kan variera relativt mycket från ena sidan till den andra. En vidareutveckling av det här området skulle kunna vara att anpassa storleken av segmenten till avståndet som mäts upp i den delen av bilden där segmentet är. Ett kortare avstånd skulle resultera i större segment och vice versa. Det här skulle ge en algoritm som är mer optimerad gällande hur stora, och hur många, segment bilden ska delas upp i. Optimeringen skulle i sin tur kunna resultera i att färre bilder behöver bearbetas av det neurala nätverket vilket reducerar bearbetningstiden.

I dagsläget kan roboten anses begränsad av Google CV eftersom det ibland har problem med att identifiera krukväxter i en miljö med mycket information såsom ett ostädat hem eller ett med mycket dekoration. I framtiden hade ett eget neuralt nätverk behövts för att göra robotens sökning efter krukväxter mer tillförlitlig. Det egna nätverket hade endast sökt efter krukväxter och behöver alltså inte störas av att andra objekt är tydligare i bilden. Detta kommer i sin tur troligtvis öka sannolikheten att roboten hittar en krukväxt i en stökigare miljö.

Även det faktum att bildigenkänning är från en tredje part kan anses begränsande. Detta är på grund av att det kostar $1.50 för 1000 bilder då färre än fem miljoner bilder skickas på en månad [19]. Det skapar också ett beroende av något utom vår kontroll vilket är oönskat.

En kamera med större bildvinkel för att täcka större område med varje bild skulle vara värt att undersöka samt, om detta innebär förvrängda bilder, hur detta påverkar resultatet av identifieringen.

Enligt avsnitt 1.2.1 har endast en krukväxt behövt identifieras under en körning. I framtiden kommer flera objekt att behöva identifieras under en körning samtidigt som deras respektive position i rymden behöver lagras. Eftersom det endast varit en krukväxt som identifierats har hänsyn inte behövt tas till det faktum att flera objekt behöver identifieras i samma bild. Problemen med att identifiera flera objekt i samma bild och placera dem i rymden uppstår främst vid felaktiga identifieringar. Det här beror på att de felaktigt identifierade objekten då sparas och placeras på kartan. Det är ett problem även nu men speciellt stort eftersom objekten inte sparas någonstans. Antalet felaktiga identifieringar är dock redan nu väldigt få och kan antas bli ännu färre i takt med att de neurala nätverken blir alltmer exakta.

Just nu resulterar uppdelningen av bilden i avlånga segment med kortare bas än höjd. Detta skulle kunna tänkas lämpa sig väl för just krukväxter vars silhuett är långsmal, men är ännu inte bekräftat att göra någon skillnad. Empiriskt testande för att testa olika geometriska figurer och vilken påverkan dessa har på identifieringen vore intressant. Exempelvis att anpassa segmentens form till att vara kvadratiska eller kanske rentav hexagonala segment för att identifiera objekt som fotbollar eller breda, låga segment för bilar.

6.4.2. Utveckling - Navigering

Vidare framåt rekommenderas en mer sofistikerad LiDAR att användas. Eftertraktade egenskaper hos denna vore exempelvis att inte se genom glas, eftersom det kan skapa stora problem med glasdörrar och liknande, eller att kunna mäta i höjdled för att uppfatta huruvida roboten får plats under vissa föremål eller inte. Ökad ljusstyrka på LiDAR:n skulle också kunna göra roboten mer anpassningsbar för ljusstarka miljöer.

För att på ett bättre sätt kunna ta fram positionen hos de identifierade objekten skulle en kamera med djupseende kunna vara användbar. En sådan kamera skulle kunna placera objekten i rummet på ett bättre sätt jämfört med kombinationen av en vanlig kamera och en LiDAR. Det här beror på att avstånden till objekten skulle räknas ut samtidigt som de identifieras, vilket i kombination med en LiDAR som inte mäter genom glas skulle kunna eliminera risken att roboten identifierar objekt som är utanför dess räckvidd. Ett enkelt sätt att göra detta är att LiDAR:n mäter avståndet i horisontell riktning mot objektet. Påträffas ett hinder närmare än objektet sätts detta avstånd som begränsning och objektet ignoreras. Höjdskillnader mellan roboten och objektet kommer även att behöva tas i beaktning men det kräver endast simpla trigonometriska beräkningar.

SLAM-implementationen kommer att behöva fördjupas sådan att roboten på ett bättre och mer effektivt sätt kan utforska okända miljöer. I dagsläget finns inget välutvecklat mönster i hur miljön undersöks och detta kan resultera i att roboten kör tillbaka till områden den redan besökt. Detta är oönskat eftersom det kan resultera i att områden inte undersöks samt att undersökningen av rummen blir synnerligen ineffektivt.

6.4.3. Utveckling - generellt

I nuläget finns inte någon automatisk uppstart eller laddningsstation vilket vore en förbättring för en slutprodukt. En idé som diskuterades tidigt i projektet var att starta roboten med hjälp av fuktsensorer i krukväxterna som signalerar när bevattning fordras. Det här skulle dock vara kontraproduktivt gällande att inte behöva ha något i krukan. Därför behöver villkor för robotens uppstart analyseras djupare i framtiden.

När roboten nu ska undersöka den närmaste omgivningen roterar hela roboten för att kunna ta de kort som behövs. En bättre lösning på detta skulle vara att enbart kameran roterar. Det här skulle resultera i att rotationen med största sannolikhet kommer gå betydligt fortare utan att sänka prestanda. Rotation av enbart kameran skulle även underlätta tack vare att roboten då inte behöver lika mycket plats för att kunna rotera samt förbrukar mindre energi.

Eftersom roboten, när den introduceras till en ny miljö, inte vet hur många krukväxter det finns eller var de finns måste det gå att avsluta körningen och återvända till laddningsstationen. Det mest lämpliga sättet vore att få roboten att utforska hela den miljö som den blivit placerad i. Detta skulle innebära en djupare implementation av SLAM där roboten kör mot outforskade områden. När den varit överallt och identifierat och vattnat de krukväxter som finns skulle detta naturligtvis innebära att den är klar.

Detta är dock inte optimalt alla gånger. Ponera att robotens placeras i ett stort rum men att krukväxter enbart finns i ett hörn. Det skulle vara väldigt onödigt att flera gånger undersöka hela rummet för att om och om igen notera att det inte finns några nya krukväxter i de andra hörnen. Det skulle därför vara lämpligt att implementera olika typer av körning för roboten. I den ena utforskar den hela sin omgivning och söker efter krukväxter och i den andra åker den till redan identifierade krukväxter för att spara tid. Användaren skulle i så fall kunna välja att informera roboten om att nya växter har placerats i miljön för att då låta den utforska hela omgivningen igen.

7. Slutsatser

Efter slutfört arbete anses de övergripande målen och syftet uppnåtts med goda resultat. Att använda en LiDAR samt SLAM-algoritm som lösning för att navigera i rummet har varit väl fungerande. Däremot har bildigenkänningen med hjälp av Google CV varit omständligt då externa lösningar har krävts för att hitta krukväxtens position. En kamera med djupseende och ett neuralt nätverk som returnerar koordinater för objekt hade varit mer optimalt. Likväl har våra lösningar för att hitta krukväxtens position fungerat över förväntan.

Projektet har också bidragit till en bra grund för framtida kandidatarbeten, där många idéer för vidareutveckling har lyfts i rapporten. Ursprungligen var arbetet syftat till att bygga en blomvattnarrobot, men har resulterat i en bra grund för olika typer av objektidentifiering samt navigation.

Som avslutande ord hoppas vi i projektgruppen på att framtida arbeten kommer att kunna fortsätta med vad vi har startat och dra nytta av vårt arbete.

Referenser

[1] L-Å. Josefsson. Svenskar reser mer än någonsin: Men det är inte längre thailand som lockar mest. Internet: https://www.aftonbladet.se/resa/article18348761.ab, 2014 [2018-04-17].

[2] S. Universitetssjukhus. Hälsa och inomhusmiljö: Faktablad från arbets- och miljömedicin, göteborg. Internet: https://www.gu.se/digitalAssets/1333/1333932_faktablad_im3.pdf, 2009-08 [2018-04-17].

[3] Edward G. King, Hilmer H. Shackelord Jr, Leo M. Kahl, "Stair climbing robot," U.S. Patent US4 993 912, 1991.

[4] Google. Cloud vision api. Internet: https://cloud.google.com/vision/, [2018-03-10].

[5] N. Robot. Nexus robot, looking to the future. Internet: https://www.robotshop.com/media/files/pdf2/user_manual.pdf.

[6] A. Yao. Teaching robots presence: What you need to know about slam. Internet: https://blog.cometlabs.io/teaching-robots-presence-what-you-need-to-know-about-slam-9bf0ca037553, [2018-04-18].

[7] Nexus Robots. 4wd mecanum wheel mobile arduino robotics car 10011. Internet: http://www.nexusrobot.com/product/4wd-mecanum-wheel-mobile-arduino-robotics-car-10011.html, [2018-03-24] [Image].

[8] Driving mecanum wheels omnidirectional robots. Internet: https://www.roboteq.com/index.php/component/easyblog/entry/driving-mecanum-wheels-omnidirectional-robots?Itemid=1208, [2018-04-19] [Image].

[9] Arduino. Arduino - introductions. Internet: https://www.arduino.cc/en/Guide/Introduction, Italy.

[10] Farnell. Np-series - valve regulated lead acid battery. Internet: http://www.farnell.com/datasheets/1916376.pdf.

[11] Roscomponents. Hokuyo urg-04lx-ug01. Internet: https://www.roscomponents.com/en/lidar-laser-scanner/83-urg-04lx-ug01.html, [2018-04-02].

[12] Hokuyo. Distance data output for urg-04lx-ug01. Internet: https://www.hokuyo-aut.jp/search/single.php?serial=166, [2018-04-28].

[13] Understanding and interpreting lux values. Internet: https://msdn.microsoft.com/en-us/library/windows/desktop/dd319008(v=vs.85).aspx, [2018-04-17].

[14] Hokuyo urg-04lx-ug01 scanning laser rangefinder. Internet: https:

//www.robotshop.com/en/hokuyo-urg-04lx-ug01-scanning-laser-rangefinder.html, [2018-04-17] [Image].

[15] Playstation. Playstation eye. Internet: https://sv.wikipedia.org/wiki/Playstation_Eye, [2018-03-24] [Image].

[16] Notebook. A notebook review. Internet: https://www.notebookcheck.net/Asus-Zenbook-UX330UA-Notebook-Review.182710.0.html, [2018-04-17], [Image].

[17] Originalbild - artificial plants for home decor. Internet: http://identitycampaigning.org/artificial-plants-for-home-decor.html, [2018][Image].

[18] C. Dahlberg. Autonoma system – en framväxande ny industriell revolution. Internet: https://kaw.wallenberg.org/forskning/autonoma-system-en-framvaxande-ny-industriell-revolution [2018-05-03].

[19] Google Cloud. Pricing. Internet: https://cloud.google.com/vision/pricing, [2018-04-06].

Bilagor

A. Framtagna koncept till elimineringsmatris

Koncept 1. A3 - B2 - C5 - D2
Rum-scanning, hittar till krukan med hjälp av dess utsatta position. Konceptet undviker hinder med hjälp av känselspröt.

Koncept 2. A4 - B5 - C3 - D2
Odefinierad rutt som undviker hinder med hjälp av ultraljud. Identifierar att det är en kruka med hjälp av kamera som ser färger (krukan har alltså en specifik färg).

Koncept 3. A2 - B5 - C7 - D3
Konceptet följer väggen och undviker hinder med hjälp av ultraljud. Identifierar vilken kruka det är med hjälp av en semiaktiv RFID. Det som befinner sig i krukan får ström genom solceller.

Koncept 4. A1 - B2 - C5 - D2
Fördefinierad rutt där konceptet åker runt genom att beräkna varven på hjulen. Undviker hinder med hjälp av känselspröt. Den hittar krukan genom att veta krukans position sedan innan.

Koncept 5. A4 - B5 - C6 - D4
Slumpvis körning och undviker hinder med hjälp av ultraljud. Konceptet identifierar vilken kruka det är med hjälp av en aktiv RFID och får energi från solceller.

Koncept 6. A4 - B4 - C4 - D2
Konceptet använder slumpvis körning med hjälp av en radar som den undviker hinder med. Den identifierar vilken kruka det är med hjälp av att krukan sänder ut en viss frekvens som drivs av solceller.

Koncept 7. A4 - B5 - C4 - D2
Slumpvis körning, positionsmätare som undviker hinder med hjälp av ultraljud. Identifierar krukor med hjälp av image recognition.

Koncept 8. A3 - B4 - C1 - D4
Scanna rummet eller ha det för-scannat och låt en unik ljud-frekvens som inte uppfattas av mänsklig hörsel sändas ut från varje kruka med hjälp av solceller. Undvik hinder med radar.

Koncept 9. A5 - B5 - C7 - D2
Rum-scanning i realtid, navigera och undvik hinder med hjälp av ultraljud. Varje kruka är utrustad med semiaktiva RFID-etiketter som är förprogrammerade utifrån växt att ge information om att de behöver vatten enligt varsin unika timer. Ingen energi i krukan behövs då.

Koncept 10. A5 - B3 - C4 - D2
Rum-scanning i realtid, navigera och undvik hinder med hjälp av LiDAR. Krukor identifieras med hjälp av image recognition.

Koncept 11. A5 - B3 - C3 - D2
Rum-scanning realtid, navigera och undvik hinder med hjälp av LiDAR. Krukor identifieras med hjälp av kamera och färgade krukor.

B. Elimineringsmatris

Chalmers	**Elimineringsmatris för:**						
Utfärdad av:			Skapad: 040510				
			Modifierad: 040830				
	Elimineringskriterier				Beslut		
	(+) Ja(tillräckligt)				(+) Fullfölj lösning		
	(-) Nej				(-) Eliminera lösning		
	(?) Mer information krävs				(?) Sök mera information		
	(!) Kontrollera kravspec				(!) Kontrollera kravspec		
	A: Löser huvudproblemet (hitta kruka)						
		B: Uppfyller alla krav					
			C: Realiserbar				
				D: Inom kostnadsramen			
					E: Tillräcklig info finns		
Lösning	**A**	**B**	**C**	**D**	**E**	**Kommentar**	**Beslut**
Koncept 1	+	-				Kunskap innan	-
Koncept 2	+	+	+	+	+		+
Koncept 3	+	-				Påverkar krukan	-
Koncept 4	+	-				Kunskap innan	-
Koncept 5	+	-				Påverkar krukan	-
Koncept 6	+	-				Påverkar krukan	-
Koncept 7	+	+	+	+	?		+
Koncept 8	+	-				Påverkar krukan	-
Koncept 9	+	-				Påverkar krukan	-
Koncept 10	+	+	+	+	+		+
Koncept 11	+	+	+	+	+		+