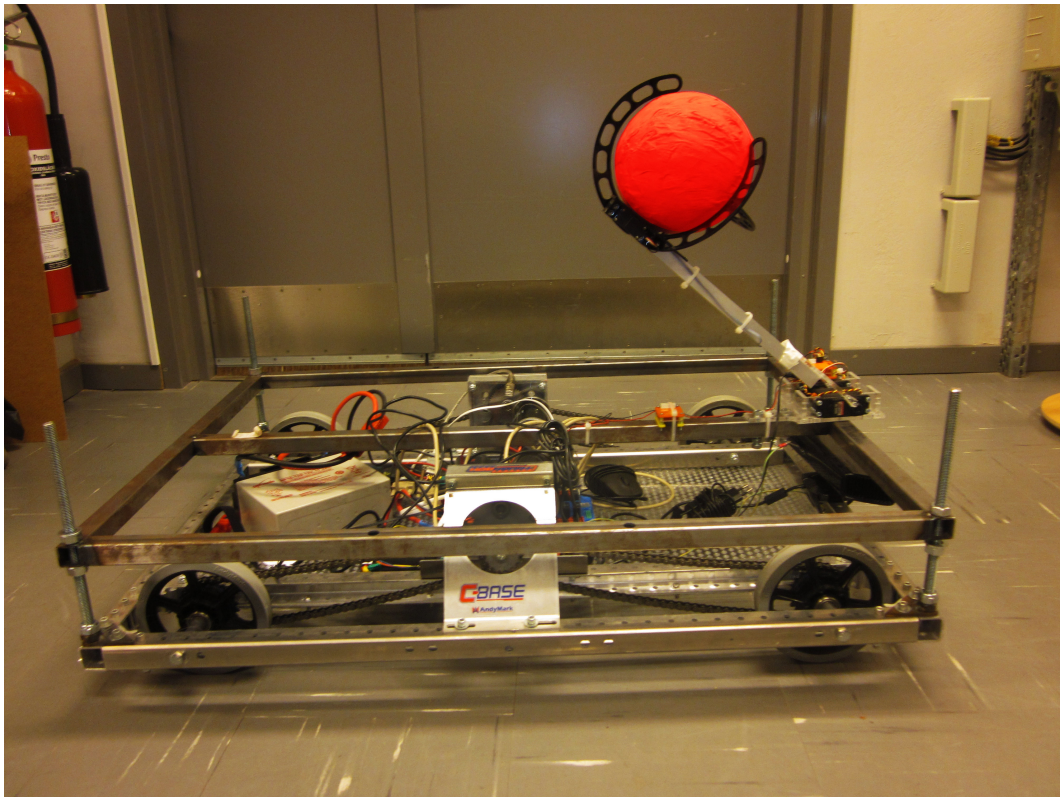
CHALMERS



Styrning av autonom robot

- utveckling av search and rescue-prototyp

MADELEINE CZARNECKI
UNNI ENGEDAHL
JOHAN LJUNGGREN
VICTOR PÅSSE

Institutionen för signaler och system
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2012
Kandidatarbetesrapport SSYX02-12-29

Sammanfattning

Projektet riktar sig till att bygga en grund för framtida utveckling av en *search and rescue*-robot som kan vara användbara i situationer där det inte lämpar sig att människor löser problemet, till exempel vid minröjning. En prototyprobot utvecklas med syfte att kunna söka efter, identifiera och plocka upp utvalt objekt på en begränsad yta. Den autonoma roboten slumpvandrar genom rummet och identifierar objekt samt slutposition genom färgidentifiering som är förfinat med brus- och kompensationfilter samt storleksbedömning. Själva upplockningen utförs av en elektromekanisk arm med tillhörande gripklo. Samtliga använda tekniker fungerar samtidigt som det finns utrymme för utveckling av både hård- och mjukvara.

Abstract

The project is aimed at providing a base for development of a search and rescue-robot which could be useful in situations where it is too risky for humans to interact, for example in mine clearance. A prototype robot is developed in order to search for, identify and retrieve a selected object in a limited area. The robot randomly walks through the room, identifying the objects and the end position by color detection, refined with noise and compensating filters in addition to sizing. The pick-up is done by an electromechanical arm with an associated pincer. All used techniques fill their purpose, while still leaving room for further development of both hardware and software.

Innehåll

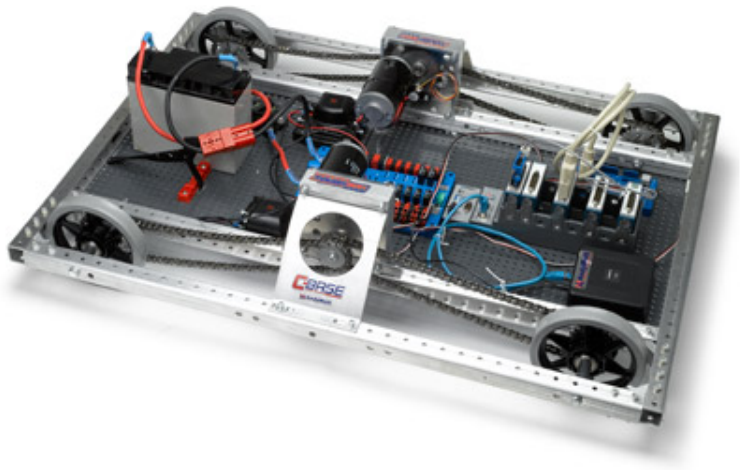
Kapitel 1

Inledning

Det finns idag många robotar vars huvudsyfte är att identifiera och interagera med föremål. Dessa robotar är vanliga i tillämpningar så som vid minröjning (Hirose m.fl., 2005) och i räddningssammanhang (Kitano m.fl., 1999) då man av säkerhetsskäl inte vill skicka in människor. En plattform utvecklad av National Instrument står som grund för utveckling av en förenklad version av en sådan så kallad *search and rescue*-robot. Problemet går ut på att finna ett objekt, med känd färg, form och storlek, och okänd placering i rummet för att föra det i säkerhet. Eftersom detta är en reducering av de tidigare nämnda tillämpningarna kan det resultat som uppnås i detta projekt ligga till grund för vidareutveckling, antingen i form av en utvecklingsplattform för att testa mer avancerade system och algoritmer eller i tillämpningar som att hitta personer i riskfyllda miljöer eller identifiering av gömda bomber och minor. Samtliga tillämpningar är av uppenbart intresse för till exempel räddningstjänst, polis och militär. Även i den privata sektorn finns det tillämpningar som till exempel lagerarbetare i lokaler där farliga ämnen förvaras.

1.1 Befintlig plattform

Den plattform som ligger till grund för projektet är en *NI ROCK* (*NI Robotics CompactRIO Kit*) som kan ses i figur 1.1. Den tillverkas av National Instruments och styrs av en NI CompactRIO realtidsprocessor. Varje system innehåller en omkonfigurerbar *FPGA* (*field-programmable gate array*) vilket möjliggör egen anpassning av timing, triggning och bearbetning av robotens styrbara AndyMarkställdon, så som DC-motorer, servon, reläer, solenoider och kompressorer.



Figur 1.1: *Den befintliga plattformen är ett* NI Robotics CompactRIO Kit *vilken är utvecklad av National Instruments i syfte att skapa en enkel grund för studenter att programmera och vidareutveckla.*

1.2 Kinect kamera

Kinect kameran är utvecklad för och delvis av Microsoft. De egenskaper som kommer till nytta i detta projekt är dock utvecklade av företaget PrimeSense (MsPress, 2010). Det är främst den bildbaserade 3-D rekonstruktionen kallad *Light Coding* som kommer till nytta för kamerans djupseende. Detta möjliggör bland annat avståndsbedömning. Kamerans specifikationer återfinns i tabell 1.1.

Tabell 1.1: *Kinect kamerans specifikationer (Microsoft 2012a)*

Horisontellt synfält	57°
Vertikalt synfält	43°
Möjlig fysisk lutning	27°
Avståndssensorns intervall	$0,9 - 3$ m
Dataströmmar	320x240 16-bit djup @ 30 frames/s, 640x480 32-bit färg @ 30 frames/s

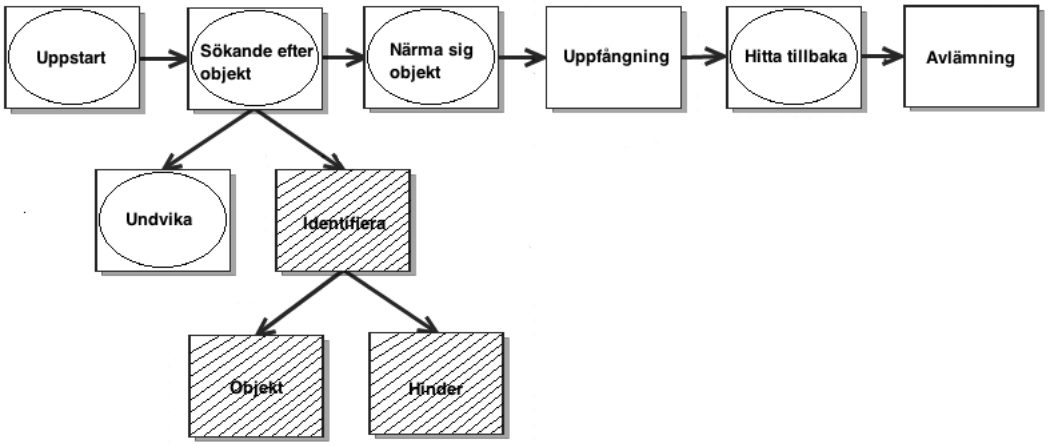
1.3 Syfte

Projektet syftar till att få en robot att autonomt identifiera och inhämta ett objekt på en begränsad yta. Hur olika former av bildbehandling, reglerteknik, program-

mering och elektromekanisk konstruktion kan implementeras på befintliga komponenter för att lösa uppgiften och utvecklas till en prototyprobot ska undersökas.

1.4 Problemformulering

För att underlätta sökandet av lösningar har problemet identifierats med en generell lösning som delar in det i tre huvuddelar – navigeringssystemet, identifieringsprocessen och uppfångningsmekanismen – som visas grafiskt i figur 1.2. Projektets



Figur 1.2: *Flödesdiagram över den generella lösningen. De randiga rutorna ingår i identifieringsprocessen, de blanka i den mekaniska uppfångningen och övriga i navigaringen.*

mål är att få en robot att autonomt hitta och hämta upp ett förbestämt objekt för att sedan transportera det till en slutposition utan att skada objektet. Processen ska utföras i ett fyrkantigt rum, med en maximal area på 100 kvadratmeter där objektet ligger stilla på marken, under en tidsram som begränsats utifrån uppskattningar och mindre tester på hastigheten, ytans storlek och körningssättet till maximalt tio minuter och en önskad tid på fem minuter. För att minska skaderisken på sig själv och omgivningen ska roboten kunna undvika kollision med väggar och andra hinder. Bortsätt från väggarna ska roboten kunna undvika tio eller färre hinder som är utplacerade på den angivna ytan. Placeringen av dessa kommer vara slumpmässig med begränsningen att de inte blockerar robotens åtkomst av outforskade ytor.

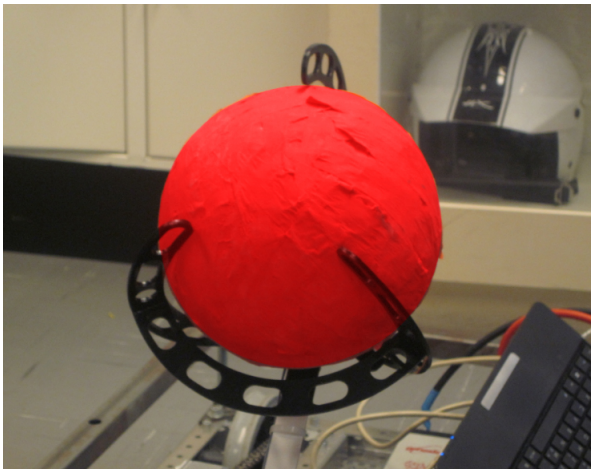
Ur miljösynpunkt ska lösningen till problemet vara sådan att den befintliga roboten kan återställas och återanvändas till 100 procent. Lösningen ska inte heller

skräpa ner eller ge några direkta utsläpp i omgivningen. Det ska även vara möjligt att vidareutveckla mjukvaran oberoende av hårdvaran och tvärtom.

Samtliga avgränsningar och krav på lösningen finns sammanställda i kravspecifikationen i Bilaga A.

1.5 Objekt

Gripklons utformning påverkar vilka objekt som kan plockas upp. Det objekt som valdes för uppgiften, se figur 1.3, är en helröd skumboll med en diameter på 180 millimeter och en vikt på cirka 200 gram.



Figur 1.3: *Det valda objektet som roboten ska finna och sedan plocka upp för att föra tillbaka det till utgångspositionen.*

1.6 Läsanvisningar

Rapporten behandlar i huvudsak lösningarna på tre olika delproblem. I avsnitt 2 redovisas de metoder som använts för att framställa lösningarna. De tekniska specifikationerna för de valda lösningarna presenteras i avsnitt 3 och den resulterande helhetslösningen redogörs i avsnitt 4. Det praktiska resultatet av de teoretiska lösningarna, det vill säga resultatet av den implementerade koden, samt av den mekaniska lösningen redovisas i avsnitt 5. Alternativa lösningar, förbättringsmöjligheter och tänkbar vidareutveckling för hårdvara samt mjukvara diskuteras i avsnitt 6. I avsnitt 7 redogörs för till vilken grad projektets mål och syfte uppfyllts.

Kapitel 2

Metod

Projektet genomförs med befintlig utrustning bestående av den robot som beskrivs i avsnitt 1.1 samt en Kinect kamera, avsnitt 1.2. En mindre litteraturstudie genomförs i syfte att ta reda på hur de problem som är förknippade med kandidatarbetet kan lösas. Litteraturen inkluderar tidigare års kandidatarbetsrapporter vilka behandlar samma plattform, artiklar som rör styrning av autonoma robotar samt fakta kring de befintliga komponenterna. Utifrån dessa studier utvecklas och genomförs tester av bildbehandling och motorstyrning samt optimering av dessa på roboten.

2.1 Utveckling av helhetslösning

Utgående från tillgänglig utrustning genomförs en *brainstorming* för att komma fram till tänkbara fristående lösningar på samtliga delproblem beskrivna i avsnitt 1.4, med hänsyn enbart tagen till problemen och inte till de specifika kraven lösningen måste uppfylla. De lösningsförslag som inte är genomförbara på grund av budgetbegränsningen, realiserbarhet och tidsramen för projektet sållas bort innan de kvarstående ställs mot kravspecifikationen i Bilaga A. I den morfologiska matrisen (Lindstedt och Burenius, 2003), se Bilaga B.1, kombineras de olika dellösningarna till realiserbara helhetslösningar vilka sedan ställs mot varandra i Pughs matris (Lindstedt och Burenius, 2003), se Bilaga B.2, där krav och önskemål från kravspecifikationen viktas beroende på hur pass viktiga de är för projektet. Den helhetslösning med högst poäng är den lösning till problemet som lämpas bäst utifrån givna förutsättningar, krav och önskemål.

2.1.1 Tekniska lösningar

För att hitta de, för identifieringsdelen av projektet (se avsnitt 1.4), mest lämpade filtren samt för att se hur pass funktionella och tillförlitliga dessa är för att nå det eftersträvade resultatet, simuleras olika typer av färgfilter, brusreduceringar och masscenterberäkningar i MATLAB. Även simuleringar av förmågan att söka upp objekt i rum av olika storlekar och form görs i MATLAB. Olika typer av navigeringsalgoritmer undersöks och simuleras för att finna en passande sökningsalgoritm för att få roboten att undvika väggar och hindren på lämpligaste sätt.

2.1.2 Datorkommunikation

Dokumentation på kommunikationsprotokollet som vanligen används för att styra den befintliga robotens motordrivare har inte kunnat finnas och därför avlyssnas kommunikationen mellan motordrivarna och testprogrammet BDC-COMM (TexasInstruments, 2010) med hjälp av en *serial port monitor* (HHDSoftwear, 2007). Signalerna analyseras och som resultat kan kod som återskapar kommunikationen utvecklas och därigenom kan motordrivarna styras.

2.1.3 Elektromekanisk lösning

Lösningen till problemet, uppfångning av objekt, väljs utefter den beskrivna lösningsgången i avsnitt 2.1. Ritningar på den valda lösningen på uppfångningsmekanismen görs i CAD som sedan används för att utveckla och konstruera en verklig prototyp. Därefter konstrueras ett kretskort som tillåter styrning av uppfångningsmekanismen. Koden struktureras i flera trådar för att få ut maximal prestanda ur datorn. Att skriva ett program multitrådat ger dock vissa problem med minnesåtkomsten då flera trådar samtidigt kan försöka komma åt samma minnesutrymmen. För att undvika detta problem används speciella system (Microsoft, 2012b) som finns inbyggda i C#. Utvecklingsmiljön som används på datorn är *Visual Studio Express* (Microsoft, 2011). Även den kod som körs på processorn på kretskortet mellan datorn och uppfångningsmekanismen utvecklas i C#.

Kapitel 3

Tekniska specifikationer

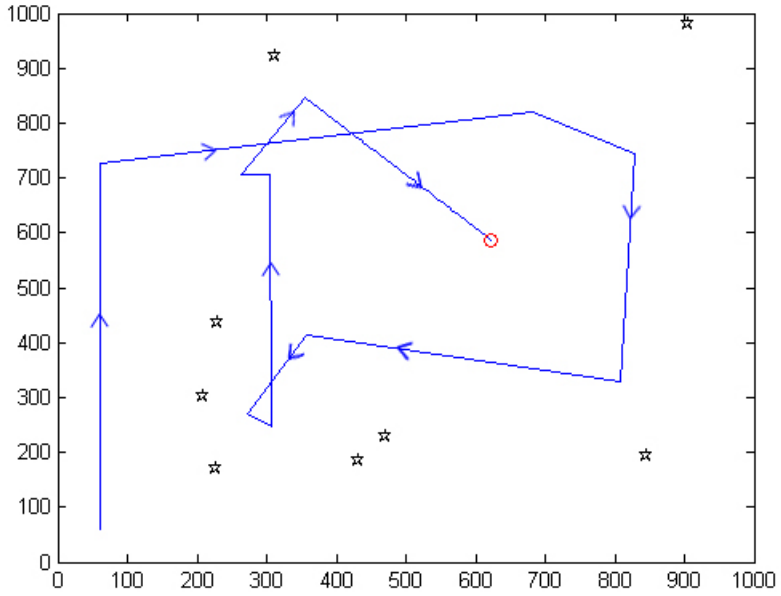
För att få roboten att uppfylla sitt syfte används en rad av olika tekniker. Dessa tekniker varierar från mjukvara till hårdvara och specificeras nedan.

3.1 Navigering

I sökandet efter objektet scannar Kinect kameran av det område som roboten har framför sig. För att kunna göra detta över hela den yta objektet kan befinna sig på behöver roboten förflytta sig i rummet. Detta gör den med hjälp av ett navigeringssystem som simulerats i MATLAB för att finna det lämpligaste rörelsemönstret. Simuleringarna resulterade i ett rörelsemönster där roboten vid initierad rörelse rör sig rakt fram tills hinder eller vägg uppkommer inom tre meter från kamerans synfält. Roboten stannar och analyserar vilken sida hindret till största del ligger på och svänger sedan åt motsatt håll med en vinkel som baseras på en slumpmässig svängningstid i intervallet $1 - 6$ sekunder med en hastighet på cirka $0,33$ m/s. Roboten kommer därför att slumpvandra genom rummet tills den funnit objektet. Samma sak gäller för sökandet av hempositionen.

3.1.1 Navigationssimuleringar

För att finna ett bra och rimligt sätt att få roboten att navigera i rummet simuleras ett rörelsemönster i MATLAB, se figur 3.1. Eftersom roboten inte kan positionera sig används ett slumpmässigt rörelsemönster som kan begränsas i utgångsposition, startriktning, längd på raksträcka roboten åker, rummets storlek, antal hinder, position på hinder och svängningsvinklar. Eftersom roboten använder sig av samma navigeringssystem för att finna objektet som för att komma tillbaka till utgångspositionen simuleras endast objektsökningen.



Figur 3.1: *Bild på hur simuleringens grafiska del ser ut. Stjärnorna symboliserar hinder, den röda ringen är objektet och den blå linjen är sträckan roboten åkt för att finna objektet.*

Simuleringen beräknar sträckan roboten kört för att finna objektet och sedan används den verkliga hastigheten på cirka 0,33 m/s för att bestämma tiden det tog för roboten att hitta objeket. Eftersom det även tar tid att svänga roboten beräknas antalet svängar som den utfört och sedan läggs det på tre sekunder[1] i tidsberäkningen för varje svängning. Simuleringen tar inte hänsyn till om roboten ser objektet samtidigt som den svänger vilket gör att tiden i vissa fall blir kortare i praktiken. Varje simulering görs 50 gånger för att få ett tillräckligt bra resultat och sedan beräknas medelvärdet på tidsåtgången. Eftersom navigeringen beror på slumpen kan varje försök att finna objektet inte alltid hamna inom tidsramen och därför sätts en felmarginal på simuleringarna. Av de 50 simuleringarna som görs är det acceptabelt att under 10 procent av dessa överstiger en tid på sex minuter.

Resultaten för varje simulering sammanställs i stapeldiagram som visar antalet sökningar som sker inom tidsintervallen $0-3$ minuter, $3-6$ minuter och över 6 minuter där tider under tre minuter är bra, under sex minuter är respektabla tider samt över sex minuter kan överskrida tidskravet i kravspecifikationen. Samtliga simuleringsresultat hittas i Bilaga D.

[1] Snittiden det tar för roboten att svänga mellan 0 – 180° baserat på verkliga tester på roboten.

3.2 Identifiering

Identifieringen av objektet går i flera steg vilka finns illustrerade i figur 3.2. Det första av dessa steg är färgidentifiering.

Färgfilter

Ett färgfilter läggs på den obearbetade bilden från kinect kameran, se figur 3.2(a). Filtret kan bara identifiera grundfärgerna röd, blå och grön. För att finna objektet aktiveras det röda identifieringsfiltret och utdatan blir en bild där allt som inte identifierats som rött elimineras, se figur 3.2(b). Samma sak sker vid identifieringen av hempositionen men med blå som identifieringsfärg. Två olika filter har utvecklats och utvärderas där det ena filtret (figur 3.3(c)) är mer avancerat och tar hänsyn till vitbalans som därmed hanterar reflektioner från lampor på bollen bättre. Detta filtret använder dock flyttal vilket gör att algoritmen går långsammare att beräkna. Det andra filtret (figur 3.3(b)) tar inte hänsyn till vitbalans men är i gengäld väldigt enkel och använder endast heltal vilket reducerar beräkningstiden.

Brusreducering

För att minska brus i den bearbetade bilden genomförs en brusreducering. Detta filter kallas ofta för erroderingsfilter och tar bort alla de pixlar som har identifierats som målfärgen men som har mindre än fyra omkringliggandes pixlar i samma färg. Detta leder till resultatet i figur 3.2(c) där det även syns att bollen blivit mindre.

Kompensation

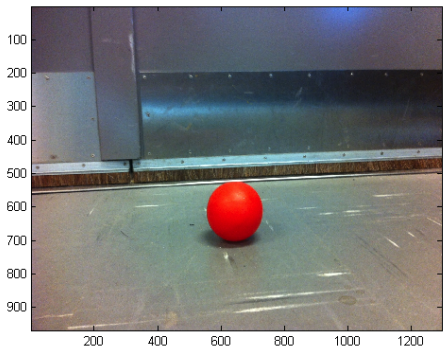
När bruset i bilden elimineras reduceras antalet pixlar hos bilden på objektet. För att inte kompromissa med tillförlitligheten i identifieringen kompenseras detta genom att återinföra de borttagna pixlarna hos den massa som identifierats som objektet.

Identifiering av masscentrum

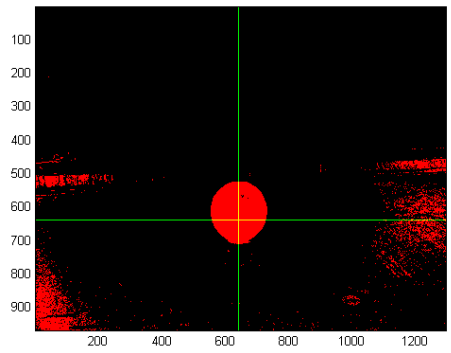
För att räkna ut objektets masscentrum integreras det över antalet pixlar, först över hela bilden och sedan i X- respektive Y-led. Resultatet av denna uträkning blir skärningspunkten för de två gröna linjerna som går över bilden i figur 3.2(b) och 3.2(c).

Storleksbedömning

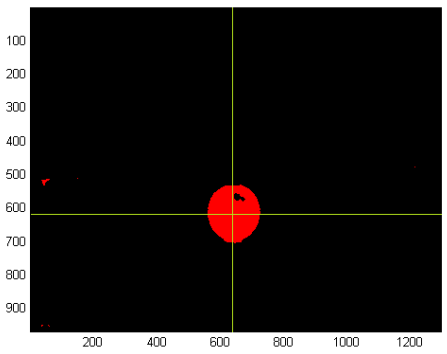
Om objektet ligger inom $0,9-3$ meters räckhåll från roboten känner den till avståndet till den röda massan samt hur stor den är och kan med denna data approximera massans verkliga storlek. Denna approximation jämförs sedan med storleken på det verkliga objektet, och om denna jämförelse stämmer överens antas det korrekta objektet vara i bild.



(a) Steg 1: Obearbetad bild från kameran.

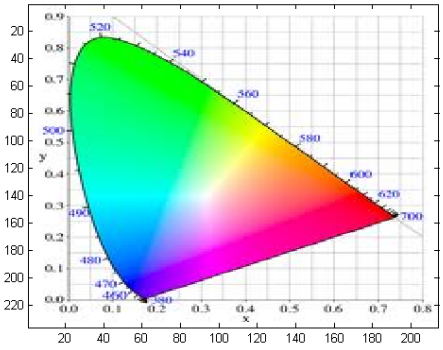


(b) Steg 2: Identifiering av färgen röd samt identifiering av masscentrum.

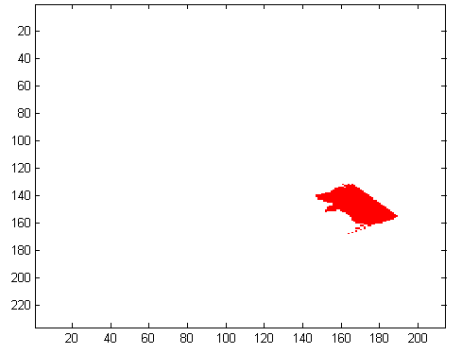


(c) Steg 3: Brusreducering samt identifiering av masscentrum.

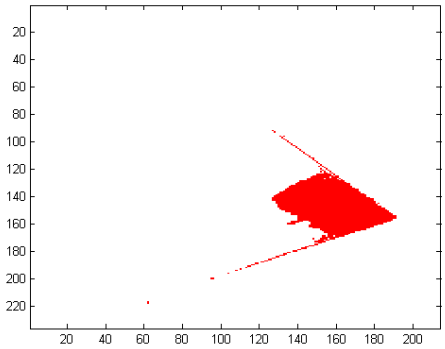
Figur 3.2: *Föremålsidentifiering utgående från objektets färg. (Obs, ej valt objekt!)*



(a) Färgrymden CIE 1931, används för filter-test.



(b) Filtertyp 1, snabbt men med liten precision.

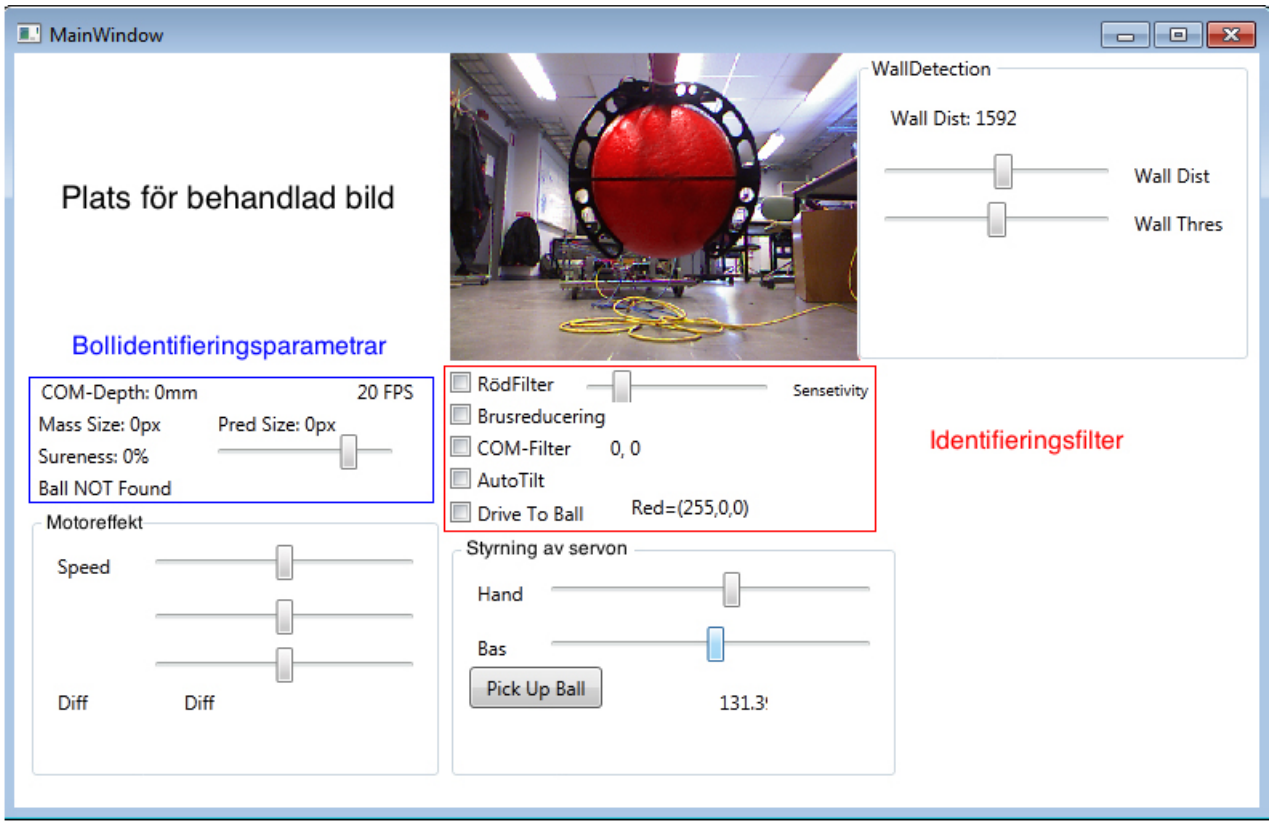


(c) Filtertyp 2, långsamt men med bra precision.

Figur 3.3: *Olika färgfilter*

3.2.1 Användargränssnitt

Det grafiska användargränssnittet, som kan ses i figur 3.4, har utvecklats för att en operatör ska ha möjlighet att observera utdata och ställa in parametrar efter behov. Dessa parametrar definierar hur olika filter och reglersystem ska fungera vilket påverkar robotens styrning och reglering vid utförandet av sitt uppdrag. Programmet används även för testning av de olika subsystemen som utgör den färdiga kontrolloopen. Koden är utvecklad i C# och är multitrådad för att möjliggöra exekvering på den prestandamässigt undermånliga dator som projektgruppen har erhålligt. Data från Kinect sensorn är den enda indatan som programmet kan läsa och utdata består dels av effekt till respektive motor samt position på den arm som ska plocka upp bollen. På den grafiska panelen kan operatören bestämma vilka av de ovan beskrivna filtren som ska användas samt hur känslig färgidentifieringen ska vara. Operatören kan även styra effekten hos robotens motorer och på vilket avstånd den ska svänga bort från hinder och väggar. Till vänster om den obearbetade bilden Kinect kameran levererar syns den bearbetade bilden, detta ger operatören möjlighet att se effekten av de olika filtren.



Figur 3.4: *Användargränssnitt med reglage för färgidentifiering.*

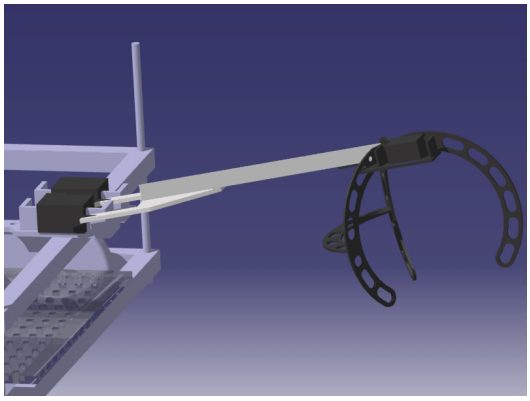
3.3 Uppfångningsmekanism

En kloförsedd arm tillverkad i plexiglas används för att utföra uppfångningen. Denna elektromekaniska konstruktion är utförd enligt figur 3.5(a) och 3.5(b) med två rörliga leder och en gripklo ytterst på armen bestående av två fasta och ett rörligt finger. De två lederna styrs av servomotorer dimensionerade för respektive led, se Bilaga C. Det servo som sitter närmast roboten vrider armen relativt roboten i vertikalplanet och det servo som sitter på gripklon styr det ledade fingret, se figur 3.5. Ett kretskort designas och tillverkas för att möjliggöra styrning av servomotorerna via USB.

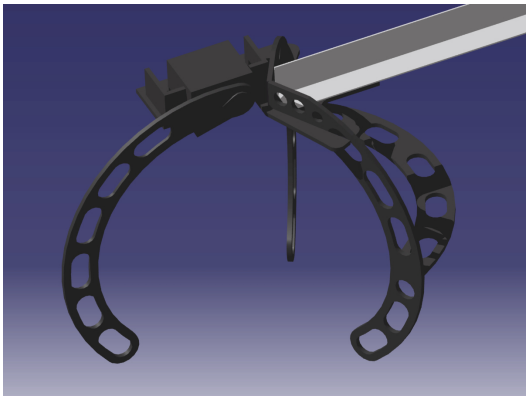
Griparmen är placerad mitt på robotens ena kortsida, se figur 3.5(d). För att inte kompromissa med bärkraften hos armen skruvas den fast i robotens egna konstruktion. Rakt under armen är Kinect kameran placerad för att inpassningen mot objektet inte ska behöva ta hänsyn till en skillnad mellan robotens ”öga” och arm i horisontalled samt för att gardera sig mot att klon eventuellt misslyckas med uppfångningen av objektet, till exempel att den tappar det. Efter att ett objekt framgångsrikt greppats av gripklons fingrar kommer armen att vridas tills den står i vertikalläge och på detta sätt transporteras bollen till avlämningsplatsen.

3.3.1 Begränsningar på objektet till följd av gripklons utformning

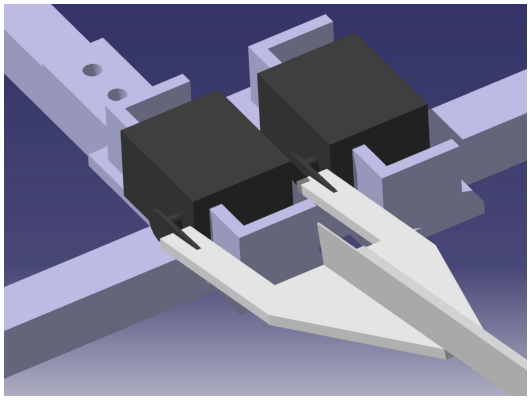
Den maximala vikten som det yttersta servot klara i en vinkelrät applicering är 230 gram. Konstruktionen klarar dock 550 gram, se Bilaga C, då tyngden från objektet fördelas relativt lika mellan de tre fingrarna. Avståndet mellan de tre fingrarna och staget mellan de två fast fingrarna tillåter att objektets volym är mellan 0,113 och 0,314 dm^3. Vad gäller objektets form måste klon kunna greppa runt hela objektet alternativt runt en kant för att det inte ska glider ur klons grepp. En stående pyramid kan till exempel ha större volym än den som är angiven ovan förutsatt att den är uppochner-vänd samtidigt som en rättvänd pyramid inte kan plockas upp alls.



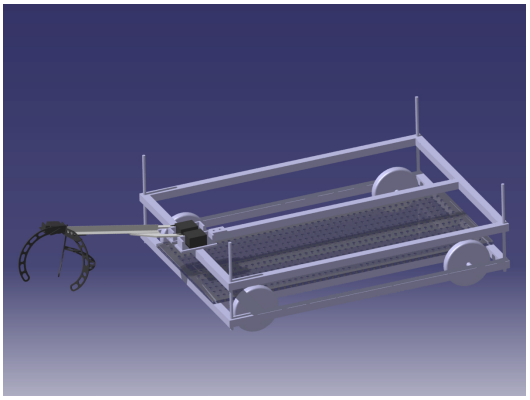
(a) Uppfångningsmekanismens arm.



(b) Uppfångningsmekanismens klo bestående av tre fingrar, varav ett är rörligt i vertikalled.



(c) Armens fastsättning på roboten.



(d) Hela roboten med uppfångningsmekanismen påmonterad.

Figur 3.5: *Bilder på uppfångningsmekanismen konstruerade i CAD.*

Kapitel 4

Helhetslösning

Det uppdrag som roboten ska utföra kan delas upp i ett antal mindre delar där varje del löser ett specifikt delproblem som roboten måste utföra för att slutföra uppdraget. Alla dessa delproblem måste lösas på ett effektivt sätt och i korrekt ordning. Nedan beskrivs delproblemen och hur roboten kommer att lösa dessa.

Uppstart
: Vid start kommer roboten att initiera all hårdvara, det vill säga Kinectsensorn, uppfångningsmekanismen och motordrivarna. När dessa är igång börjar roboten söka igenom rummet.

Söka efter objektet
: Efter uppstart ska roboten söka efter objektet i den omgivande miljön. Detta moment delas in i tre delmoment.

  Köra runt
  : Under sökningsprocessen kör roboten raka sträckor i miljön den befinner sig i fram tills hinder eller vägg uppkommer. Detta sker integrerat med identifieringen av objektet som beskrivs nedan under rubriken ***Identifiera objekt***.

  Identifiera och undvika hinder/väggar
  : Under pågående sökning ska roboten kunna identifiera väggar och andra hinder samt undvika dessa för att minska skaderisken för sig själv och omgivningen. I en miljö där det finns fler hinder än enbart rummets begränsningsytor färdas roboten framåt tills en vägg eller ett hinder befinner sig mellan $0 - 1,2$ meter framför den. Roboten kommer då att identifiera på vilken sida hindret främst befinner sig på och sedan svänga åt motsatt håll med en slumpmässigt vald tid mellan 1 - 6 sekunder för

att sedan fortsätta köra en rak sträcka. Detta leder till att roboten slumpvandrar genom rummet.

Identifiera objekt
För att finna objektet under sökandet ska roboten kunna identifiera och urskilja det ur miljön det befinner sig i. Detta sker genom färg- och storleksidentifiering enligt avsnitt 3.2.

Närma sig objektet
En PI-regulator implementeras för att sköta insvängningen mot objektet. Kinect sensorn vinklas upp och ner beroende på bollens placering i förhållande till robotens för att hålla dess masscentrum mitt i bilden. Detta minskar risken att förlora objektet ur sikte samt underlättar uppfångningen. När avståndet till objektet är mindre än ca 800 millimeter kan Kinect sensorn inte mäta distansen till bollen. Istället kommer avståndet approximeras baserat på hur stor del av bilden som upptas av objektet och dess fysiska storlek. Inpassningen är färdig när objektet ligger på ett avstånd av 290 millimeter från Kinect kameran.

Uppfångning av objektet
När bollens masscentrum ligger inom en felmarginal på 100 millimeter i sidled och 290 millimeter i avstånd framåt fälls armen ner och roboten försöker greppa bollen. Gripklon höjs upp från marken och med hjälp av kameran avgörs huruvida bollen ligger korrekt i gripklon. Om upplockningen lyckats håller roboten fast bollen och börjar leta tillbaka till utgångspositionen. Om upplockningen misslyckats återupptar roboten letandet igen, då den troligtvis har puttat iväg objektet från sin ursprungliga position.

Hitta tillbaka
Under transport av objektet ska roboten söka efter sin hemposition vilket kommer vara en liknande sökningsprocess som den beskriven ovan under rubrik **Söka efter objektet**, fast med en blå målfärg. Sökprocessen är avslutad när roboten nått målpositionen.

Avlämning av objekt
När roboten nått sin slutposition fäller den ner armen, öppnar gripklon och bollen rullar mot positionen.

4.1 Återkoppling

För att veta vad roboten tänker göra under processen behövs en återkoppling mellan robot och operatör. Återkopplingen som valts till detta projekt är att roboten

ska kunna tala för att berätta för operatören vad den tänker göra i olika situationer. Tillfällen då den ska ge återkoppling är:

- Vid möte av vägg eller hinder och berätta på vilken sida hindret uppkommer.
- När objektet uppkommer i robotens synfält.
- När roboten tänker köra mot objektet.
- När roboten hamnat i en tillfredsställande position mot objektet och tänker plocka upp det.
- Då målet där objektet ska lämnas av uppkommer i robotens synfält.
- När roboten tänker lägga ner objektet.

Återkoppling är nödvändig för att på ett enkelt och direkt sätt få ut resultat på robotens hantering av olika situationer för att senare kunna utvärdera det.

Kapitel 5

Resultat

Här presenteras resultatet av hur slutprodukten löser problemet. Slutprodukten består av den robot som är beskriven i avsnitt 1.1 med en påmonterad Kinect kamera, en elektromekanisk arm med tillhörande gripklo och med implementerad kod som styr de tidigare nämnda delarna, se avsnitt 3 och 4 för mer information om hur konstruktionen. För att få resultat på hur roboten löser problemet har tester utförts i ett rum på 30 kvadratmeter med två stycken hinder placerade på ett sådant sätt att roboten kan nå outforskade ytor. De frågor som beaktats under tester av hur roboten utför hela processen[1] var:

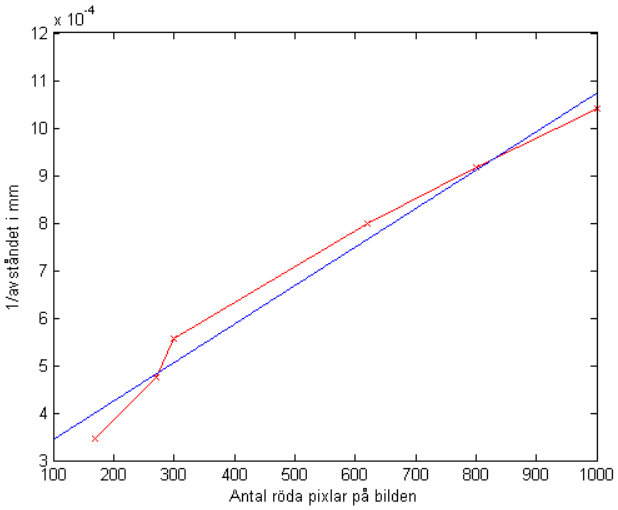
- Hur ofta undviker roboten hinder eller väggar utan att krocka?
- Hur ofta identifierar roboten rätt sak som objektet och hur ofta misslyckas den med att identifiera objektet trots att det är i synfältet?
- Hur ofta fångar roboten upp objektet på första försöket?
- Hur ofta identifierar roboten rätt mål som slutposition? Hur ofta misslyckas den med att identifiera slutpositionen trots att den är i synfältet?
- Hur ofta lämnar roboten av objektet vid slutpositionen?
- Hur ofta utför roboten hela processen inom den maximala tidsramen på tio minuter?
- Hur ofta utför roboten hela processen utan misslyckanden?

Resultatet på testerna finns att se i tabell 5.1 där antalet lyckade försök för varje beaktad fråga visas i förhållande till antalet tester. Tiden det tog för roboten att

[1]Med hela processen menas det från att roboten börjar söka efter objektet till att den har lämnat av det vid slutpositionen.

Tabell 5.1: *Resultatet på testerna då roboten utför hela processen, från att börja söka efter objektet till att lämna det vid utgångspositionen.*

	Antal försök	Antalet lyckade försök	Andel lyckade försök [%]
Undvika hinder och väggar:	35	28	80
Identifiera objektet:	15	12	80
Fånga upp objektet:	12	11	92
Identifiera hempositionen:	15	11	73
Lämna av objektet:	11	11	100
Utförandets tidsram	11	10	91
Totala utföranden	13	9	69



Figur 5.1: *Graf över förhållandet mellan den ytan som bollen tar upp på bilden från Kinect sensorn och avståndet till bollen. Den blå linjen representerar det ideala förhållandet mellan bollyta och avstånd medan den röda representerar det verkliga förhållandet.*

genomföra hela processen sammanställdes för varje test och medelvärdet blev ungefär fyra minuter.

För att uppskatta avståndet från roboten till bollen då den inte befinner sig i det avståndsintervall som Kinect sensorn fortfarande kan avståndsbedömma i används en funktion som är en linjarisering av de mätpunkter i figur 5.1. Grafen visar förhållandet mellan bollens upptagna yta på bilden från Kinect sensorn och avståndet till bollen.

Kapitel 6

Diskussion

De tester som utförts på roboten, vars resultat presenteras i avsnitt 5, visar att roboten i 69 procent av fallen lyckas utföra processen utan något misslyckande i de olika delprocesserna. Anledningen till att vissa delprocesser inte utförs korrekt av roboten diskuteras nedan.

Uppfångningsmekanismens förändringar

Den första prototypen av uppfångningsmekanismen var designad för mindre objekt samt med tre frihetsgrader där den extra frihetsgraden var placerad mellan armen och gripklon, vilket tillät gripklon att röra sig i vertikalled. Detta gjorde att flera objekt kunde plockas upp under samma sökprocess och förvaras i en behållare ovan på roboten. Det visade sig dock att Kinect sensorn, på grund av objektens begränsade storlek, inte kunde få ut djupdata relaterad till bollarna. Detta då kameran kräver en minimistorlek på 0,0250 kvadratmeter för avståndsbedömning. Uppfångningsmekanismen behövde således designas om för att passa ett större objekt med 180 millimeter i diameter och en vikt på 200 gram. För att kompensera viktökningen hos objektet utrustades den nya uppfångningsmekanismen med enbart två frihetsgrader, i syfte att hålla den totala vikten nere.

Bristen på en tredje frihetsgrad medför att roboten inte kan lämna av objekten i behållaren uppe på roboten. Det är dock fullt möjligt att lägga till ett servo och utöka frihetsgraderna men det skulle resultera i att armen blir cirka 100 gram tyngre vilket får till följd att servorna i basen på armen behöver jobba hårdare. Det beslutades att armen inte skulle modifieras med ett till servo, dels då det ökar risken att de befintliga servorna går sönder och dels för att det inte anses som en kritisk funktion i projektet att roboten ska kunna transportera fler objekt

Brister i identifieringsprocessen

Objekt som ligger inom ett avstånd på 0,9 meter framför kameran kan inte avståndsbedömas och kommer då inte att upptäckas av kameran. Detta kan inträffa bland annat när roboten svänger vilket skapar en del problem vid identifiering av boll, hemposition och hinder. Om bollen hamnar för nära eller för långt bort ifrån kameran kan den inte bekräfta att den röda massan i bilden verkligen är bollen. Det skulle vara möjligt att få roboten att backa eller köra framåt tills den får djupdata men det skulle medföra att den behöver göra samma sak för alla röda objekt den ser, till exempel brandsläckare eller röda dörrar. Vi anser det mer effektivt för den att leta i rummet tills dess att bollen förekommer inom avståndsintervallet $0,9 - 3$ meter där djupseendet fungerar och roboten lättare kan fastställas att det röda objektet som syns är bollen. Samma problem återkommer vid sökande av hempositionen. Om det är ett föremål annat än objektet som befinner sig för nära kameran initierar roboten inte en sväng bort från hindret eftersom den inte uppfattar att hindret befinner sig i vägen. Detta leder till att roboten kan köra på hinder och skada sig själv eller miljön och det är därför rimligt att ha ytterligare en sensor som kan bedöma avståndet till objekt som ligger närmare än 0,9 meter.

Kamera och andra sensorer

Det finns flera begränsningar i den Kinect sensor som har använts i projektet. Dessa problem inkluderar till exempel att kameran har dålig färgåtergivelse och det mesta kameran ser har en tendens att bli väldigt mörkt eller väldigt ljust. Kameran kan inte heller ge djupdata på små eller reflekterande föremål. Vissa av dessa problem är dock enkela att kringgå. Avståndsbedömningen kan underlättas genom att som komplement använda en eller flera ultraljudssensorer för att ge djupdata. Sensorerna ger endast avståndet till det närmaste objektet vilket medför att en vanlig kamera måste användas för att kunna bedöma om det är en boll eller ett hinder som har identifierats. Om denna lösning impementeras istället för Kinect är valet av utvecklingsmiljö och programmeringsspråk mer öppet, detta begränsas annars av Kinect kamerans stöd utanför Microsofts produkter.

Geometrifilter

Roboten gör i dagsläget ingen geometrianalys på objekten kameran ser. Detta leder till att roboten inte kan särskilja objekt annat är på deras färg. Ett geometrifilter hade applicerat en tänkt rektangel på alla föremål med samma eller liknande färg som objektet och sedan anpassat rektangelns sidor efter föremålet. Ju mer lik en kvadrat denna rektangel är efter anpassningen desto större chans är det att föremålet är det sökta objektet. De föremål med avvikande geometri kan med denna

process filtreras bort.

Ett annat tänkbart filter är ett som beräknar förhållandet mellan röda och icke röda pixlar i den rektangel det tidigare potentiella filtret har beräknat. Om antalet röda pixlar utgör lika stor yta i rektangeln som en cirkel, cirka 78 procent, så är det troligtvis en boll. Om båda dessa filter implementerades skulle roboten ha lättare att urskilja bollen från andra röda föremål som ofta finns i vanliga utrymmen, till exempel brandsläckare och tavlor. Anledningarna till att dessa typer av filter inte implementerats i den nuvarande lösningen är dels att de inte ansetts vara nödvändiga då de objekt som är röda kunnat avlägsnas innan testar av roboten påbörjats samt att de är processorkrävande. Den dator som finns tillgänglig för projektet inte skulle klara av alla de nödvändiga beräkningarna med en tillfredsställande uppdateringshastighet.

Val av identifieringsfärg

Det färgfiltret som är implementerat på roboten fungerar bäst för färger nära grundfärgerna rött, blått och grönt. Att det valda objektets färg är röd gör att grön eller blå bör användas för att identifiera hempositionen. Grön fungerar inte som identifikationsfärg då kameran är väldigt känslig för nyansskillnader av grönt. Kameran tolkar dock solljus som blått vilket gör att det inte heller lämpar sig som identifikationsfärg i rum med fönster som ej går att täcka. Eftersom kamera inte är tänkt för de användningsområden den ges i detta projekt är dess färgåtergivning inte tillräckligt bra. En lösning som har testats är att även låta hempositionens färg vara röd. Detta har även implementerats med lyckade resultat, med undantag för när roboten plockar upp objektet som markerar hempositionen och avlämnar det på bollen.

PC istället för LabVIEW

Robotplattformen baserades ursprungligen på compactRIO och LabVIEW. När alla beräkningar togs över av en dator blev dock de enda arbetsuppgifterna för compactRIO och LabVIEW att ta emot motorinformation från PCn för att sedan skicka det till motordrivarna. Med cirka fyra rader kod i C# gavs även denna uppgift till datorn och behovet av compactRIO och LabVIEW eliminerades.

De olika navigeringsalternativen

Den metod som valdes för implementering på roboten är slumpmässig vandring vilket innebär att roboten svänger under ett slumpmässigt tidsintervall när den möter en vägg eller hinder. Den lösningen är dock inte optimal då roboten kan

komma att besöka samma position flera gånger.

Det finns många olika metoder för att söka av ett rum. De flesta sofistikerade sökalgoritmerna kräver dock någon form av positionering, någonting som roboten helt saknar. Återkoppling blir då omöjligt och hela navigeringen tvingas vara en öppen loop. En möjlighet skulle vara att implementera död räkning för att hålla koll på robotens aktuella position.Hastighetsvektorn integreras då över tiden och den momentana positionen beräknar relativt startpositionen. Tyvärr sladdar samtliga fyra hjul på roboten då den svänger vilket omöjliggör denna metod. Hade roboten istället haft två fast hjul och ett, till exempel, kundvagns hjul som kan rotera i två dimensioner skulle det möjligtvis gå att implementera dödräkning och därmed få en stängd loop över positionen i rummet.

En annan metod som är möjlig är SLAM (*Simultaneous localization and mapping*)(Thrun, 2008), som bygger på att roboten hittar landmärken och konstruerar en karta baserat på dessa samtidigt som den åker runt i rummet. Även denna metod skulle ge möjlighet till reglering med en stängd loop. Både de ovan nämnda metoderna ökar dock komplexiteten på projektet signifikant vilket skulle öka risken att projektets syfte inte uppfyllts med avseende på dess tidsram och det valdes därför att inte implementera detta på roboten.

Fasskift/utveckling av reglering

På grund av fasskift[1] mellan kameran och motorerna är det svårt att få reglerloopen snabb men fortfarande stabil. Detta kan observeras då roboten ibland har en tendens att oscillera när den närmar sig bollen. En lösning på detta är att minska P- eller I-faktorn i reglerloopen men det medför en långsammare reglering. En annan lösning är att använda en bättre dator som kan behandla bilderna från kameran snabbare och på så sätt minska fasskiftet.

Svårigheter med tester av roboten

En del svårigheter uppkom i samband med att få fram resultat på hur rummets storlek och antal hinder påverkar roboten då tillgången till större, tomma rum inte fanns. Det finns därför inga resultat på hur pass bra roboten fungerar i miljöer större än 30 kvadratmeter samt om den uppfyller ytstorleks- och hinderkravet i kravspecifikationen. Om roboten däremot varit mindre i storlek och kunnat ta sig runt på mindre ytor skulle testning av den underlättats. Storleken som roboten

[1]Med fasskift menas att alla delar av regler-systemet, så som kamera, dator och motorer, har en förskjutning mellan in och ut signaler.

har i dagsläget fyller ingen funktion och skulle utan problem kunna minskas rejält. Att roboten är stor begränsar även batteritiden då motorerna måste jobba hårdare för att snurra hjulen som kommer längre ifrån rotationscentrum. Robotens storlek och utformning ökar även risken att skada personer och egendom vid sökprocessen.

6.1 Utvecklingsmöjligheter

Om roboten hade impementerat någon form av positionsåterkoppling, till exempel SLAM, skulle en stängd loop reglering kunna implementeras. Det skulle medföra att roboten kan navigera efter någon form av mönster för att söka i rummet efter målet. Vilket skulle kunna medföra att objektet identifieras snabbare.

Det finns en påtaglig risk att föremål av liknande storlek och i samma färgskala som bollen identifieras som bollen. Valet av identifikationsfärgen röd innebär att även färger som tillhör brunskalan, till exempel vissa träslag, identifieras som röda. Detta skulle kunna lösas genom att använda en kamera som har bättre vitbalans och dynamik än den kameran som är integrerad i Kinect sensorn.

Den dator som används för alla beräkningar är kraftigt underdimentionerad och uppfyller inte de krav som krävs för att kunna användas ihop med Kinect sensorn. Om datorn skulle vara kraftfullare skulle flera, mer avancerade algoritmer kunna implementeras. Även upplösningen från Kinect sensorn skulle kunna höjas.

Ytterligare en frihetsgrad kan läggas till på den befintiliga upplockningsmekanismen, Detta skulle möjliggöra avlämning av bollen uppe på roboten i en behållare vilket gör att roboten kan plocka upp flera bollar. Dock bör en sådan behållare vara konstruerad så att roboten även kan plocka ur bollarna från den.

Kapitel 7

Slutsats

Följande huvudslutsatser kan dras från projektet:

- Informationen som fås av Kinect sensorn har en del brister som begränsar projektets tillförlitlighet och försvårar lösandet av problemet.
- Färgfiltret behöver förbättras för att kunna använda fler och bättre identifiaktionsfärger.
- Att använda en Kinect kamera som enda sensor är inte ett bra val för realtidsapplikationer.
- Robotens mekaniska utformning försvårar positionsreglering.
- Förbättring av mjukvara kräver en bättre dator.
- Programmering direkt i PC utan att använda LabVIEW fungerade bra för projektet.
- Roboten kan undvika hinder och väggar på ett tillfredsställande sätt.
- Både hårdvaran och mjukvaran kan utvecklas separat.
- Slumpmässig vandring med begränsingar är inte ett optimalt navigeringssystem men löser ändå problemet.

Roboten uppfyller således sitt syfte trots vissa brister i detaljutförandet.

Kapitel 8

Litteraturförteckning

HHDSoftwear. Monitor and Analyze Your Serial Ports and Hardware Devices, 2007. `http://www.serial-port-monitor.com/`, 5 mars 2012.

Hirose, A., Toh, J.A., och Hara, T. Plastic landmine identification by multistage association. *IEIC Technical Report (Institute of Electronics, Information and Communication Engineers)*, 104(758):113–118, 2005.

Jonsson, B. *Formel- och tabellsamling*. Värmdö Gymnasium, 2012. `http://www.bjornjonsson.se/pdf/fyabformelsamling.pdf`, 5 mars 2012.

Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., och Shimada, S. Robocup rescue: Search and rescue in large-scale disasters as a domain for autonomous agents research. I: *Systems, Man, and Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on*, band 6, ss 739–743. IEEE, 1999.

Lindstedt, P. och Burenius, J. *The value model: how to master product development and create unrivalled customer value*. Nimba, 2003. ISBN 9789163063497.

Microsoft. Microsoft Visual Studio Express, 2011. `http://www.microsoft.com/express`, 5 mars 2012.

Microsoft. *Kinect for Windows*. Microsoft, 2012a. `http://www.microsoft.com/en-us/kinectforwindows/`, 1 februari 2012.

Microsoft. Microsoft Mutex, 2012b. `http://msdn.microsoft.com/en-us/library\slashsystem.threading.mutex.aspx`, 5 mars 2012.

MsPress. PrimeSense Supplies 3-D-Sensing Technology to "Project Natal" for Xbox 360, 2010. 31 mars 2010.

TexasInstruments. *Stellaris® Brushed DC Motor Control Module with CAN (MDL-BDC24)*. Texas Instruments, 2010.
`http://www.ti.com/tool/mdl-bdc24`, 5 mars 2012.

Thrun, S. Simultaneous localization and mapping. *Robotics and cognitive approaches to spatial mapping*, ss 13–41, 2008.

Bilagor

Bilaga A

Kravspecifikation

Kraven som roboten ska uppfylla är sammanställda tillsammans med önskemålen i en kravspecifikation. Önskemålen är viktade mellan 1-5 där 5 räknas som viktigaste önskemålet att uppfylla.

Specifikation		**K/Ö**	**Viktning**
1. Objekt	1.1 Röda objekt	K	
	1.2 Objektets maxvikt: 550 g	K	
	1.3 Objekt med varierande form	Ö	2
2. Hitta hem	2.1 Identifiera blå skylt	K	
	2.2 Utan identifiering	Ö	3
3. Identifiera och undvika hinder	3.1 Väggar	K	
	3.2 Alla statiska hinder	K	
4. Omgivningen	4.1 Fyrkantigt rum	K	
	4.2 Tomt rum	K	
	4.3 Ett rum med 10 okända föremål	Ö	2
	4.4 Rummets maxstorlek: 100 m^2	K	
5. Närma sig objekt	5.1 Utan att köra över objektet	K	
	5.2 Finna objekts exakta position	Ö	4
6. Fånga upp objektet	6.1 Ett känt objekt	K	
	6.2 Olika objekt	Ö	4
	6.3 Flera objekt samtidigt	Ö	3
7. Transportera objekt	7.1 Identifierat objekt	K	
	7.2 Till utgångspositionen	K	
	7.3 Utan att tappa objektet	K	
	7.4 Ta snabbaste vägen	Ö	3
	7.5 Flera objekt samtidigt	Ö	3
8. Avlämning av objekt	8.1 Så att en människa kan ta den från roboten	K	
	8.2 Läggas bredvid roboten	Ö	4
	8.3 Läggas bredvid roboten på förbestämd plats	Ö	3
9. Tidsaspekt	Ovanstående ska ske inom en förbestämd tidsram:		
	9.1 10 min	K	
	9.2 5 min	Ö	2
10. Miljöaspekter	10.1 Återanvändbar till 100%	K	
	10.2 Ingen nedskräpning	K	
	10.3 Inga direkta utsläpp	K	
11. Potential	11.1 Mjukvara och hårdvara ska kunna vidareutvecklas separat	K	

Bilaga B

Urvalsmatriser

För att den lösningen som väljs till de olika delproblemen, och även lösningen som helhet, ska vara den mest lämpade under de givna förutsättningarna, avsnitt 1.4, och efter kraven specificerade i kravspecifikationerna, bilaga A, sätts samtliga lösningar in i en morfologisk- och sedan Pughs matris.

En morfologisk matris gör det möjligt att skapa många helhetslösningar till projektet genom olika kombinationer av lösningar till de olika delproblemen. För att från dessa helhetslösningar få ut den mest lämpade sätts förslagen in i Pughs matris. Detta är en elimineringsmatris i vilken lösningsförslagen jämförs med varandra beroende på hur väl de uppfyller kraven och de mest eftertraktade önskemålen (bilaga A) på lösningen. Viktningen bestämmer vilket krav eller önskemål som anses vara viktigast för projektet.

B.1 Morfologisk matris

B.2 Pughs matris

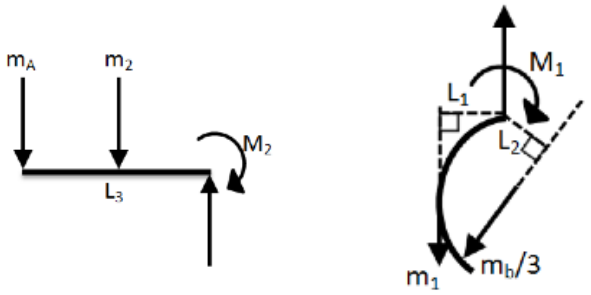
Krav/Önskemål	Viktning	Lösning 1(ref)	Lösning 2	Lösning 3	Lösning 4	Lösning 5	Lösning 6
Identifiera röda objekt	2		0	–	0	–	0
Hitta hem	4		–	+	+	0	0
Undvika väggar	5		–	0	0	0	0
Undvika hinder	3		–	–	0	0	0
Köra i ett fyrkantigt rum	5		0	0	+	0	0
Utan att köra över objektet	5		–	0	–	0	0
Finna objektets exakta position	2		–	0	–	0	0
Fånga upp ett känt objekt	5		0	+	–	0	+
Fånga upp olika objekt	2		+	+	0	0	+
Transportera identifierat objekt	5		0	0	0	0	0
Transportera utan att tappa	4		0	+	0	0	+
Transportera flera objekt samtidigt	4		0	–	0	0	+
Lämna objektet så att en människa kan ta det	4		+	+	–	0	+
Läggas bredvid roboten	1		+	+	0	0	0
Tidsram på 8 min	4		0	0	0	0	0
Återanvändbar 100 %	5		0	0	0	0	0
Ingen nedskräpning	5		0	0	0	0	0
Inga utsläpp	5		0	0	0	0	0
Mjukvara och hårdvara ska kunna vidareutvecklas separat	4		+	+	–	0	+
Antal +		0	4	7	2	0	6
Antal –		0	5	3	5	1	0
Summa		0	-1	4	-3	-1	6
Viktad summa		0	-8	15	-11	-2	23

Bilaga C

Dimensionering av armens elmotorer

De två servomotorerna som styr armens rörelser sitter på roboten vid armens bas samt mellan armen och gripklon. Det första och största servot styr armens rörelser i vertikalled. Friläggning av de krafter som utger påfrestningar på servot visas i figur C.1(a).

Längderna L_1, L_2 och L_3 är 60, 80 respektive 393 (arm plus axelled) millimeter



(a) Krafterna det kraftigaste servot utsätts för.

(b) Servomotorn som styr det rörliga fingret dimensioneras efter krafterna på bilden.

Figur C.1: *Friläggningar för de två servomotorerna.*

och vikterna står för det rörliga fingrets massa ($m_1 = 13.216$ gram)[1], massan över L_3 ($m_2 = 78.309$ gram)[2], summan av objektets, klons och det lilla servots massa ($m_A = 69.76$ gram + objektets massa) samt massan på objektet, m.

[1] $1.12 \cdot 10^{-5}$ m^3 plexiglas med densitet 1.18 g/cm^3 ger massa på $1.12 \cdot 10^{-5} \cdot 10^6 \cdot 1.18 = 13.216$

[2] Armens massa, $1.209 \cdot 10^{-5}$ m^3 aluminium med densitet 2.7 g/cm^3 ger massa på $1.209 * 10^{-5} \cdot 10^6 \cdot 2.7 = 32.643$ g. Axelledens massa, $3.87 \cdot 10^{-5}$ plexiglas med densitet 1.18 g/cm3 ger massa på $\cdot 10^{-5} \cdot 10^6 \cdot 1.18 = 45.666$ g (Jonsson 2012).

Vinkeln på armen har betydelse för vilket moment den uträttar på motorn. Eftersom momentet ges av den applicerade kraften multiplicerat med det vinkelräta avståndet mellan angreppspunkten och momentpunkten fås störst moment då armen befinner sig horisontellt läge. Dimensionerna för armen kan direkt användas för att beräkna det maximala momentet respektive motor utsätts för. Momentjämvikter enligt friläggningen i figur C.1 ger

$$M_1 = \frac{m_b}{3} L_1 - m_1 L_1 \tag{C.1}$$

$$M_2 = m_A L_3 + m_2 \frac{L_3}{2} \tag{C.2}$$

För de aktuella motorerna gäller de värden som återfinns i tabell C.1. Insättning

Tabell C.1: *Specifikationer för de två servomotorerna som styr uppfångningsmekanismen.*

Fingrets motor	**Armens motor**
Vikt: 8.8 g	Vikt: 110 g
Max vridmoment 1.4 kgcm	Max vridmoment 13.2 kgcm

i ekvationen (C.1) ger maximal tillåten objektvikt för servot som styr det rörliga fingret, 550 gram. Insättning i ekvationen (C.2) ger den maximala totalvikten för arm inklusive objekt, 9070 gram.

Bilaga D

Simulering

D.1 Navigering

Resultaten för de olika simuleringarna som beskrivs nedan har satts samman för att hitta det mest lämpade navigeringssystemet till roboten. Om inget annat nämns visas resultaten för varje simulering i ett stapeldiagram där resultaten, det vill säga tidsåtgången för varje simulering, sorteras i tidsintervallen $0-3$ minuter, $3-6$ minuter och över 6 minuter. Stolparna i diagrammet visar antalet simuleringsresultat som hamnat inom varje intervall.

D.1.1 Antal hinder och ytstorlekens betydelse

En simulering på hur roboten tar sig runt i olika stora rum och med varierande antal hinder görs för att se hur det påverkar roboten när den navigerar. Enligt kravspecifikationen ska roboten kunna hantera maximalt tio stycken hinder i ett rum som placerats ut slumpmässigt på ett sådant sätt att roboten kan undersöka alla ytor och samtidigt hitta objektet. Specifikationer för simuleringarna finns i tabell D.1.

Tabell D.1: *Tabell över specifikationer vid simulering av rumsstorleken och hinderantalet.*

Hinder:	10 stycken slumpmässigt utplacerade.
Storlek på rum:	10×10 m^2 8×8m^2 5×5m^2
Svängningsvinklar:	Slumpmässigt mellan $20° - 135°$ åt höger eller vänster.
Startriktning:	Längs en vägg.

Figur D.1: *Stapeldiagram som visar simuleringsresultatet för rumsstorlekens och hinderantalets betydelse.*

Simuleringsresultat

Enligt simuleringsresultaten i figur D.1 är det möjligt att ha tio hinder i ett rum med den maximala arean på 100 m2 för att roboten fortfarande ska kunna hitta objektet och föra det till utgångspositionen inom tidsramen på tio minuter. Resultatet visar dock att fyra procent av simuleringarna inte är under sex minuter i detta fall vilket ligger under felmarginalerna som sökningen har. Om arean däremot minskas men antal hinder består blir tidsåtgången för att finna objektet större eftersom roboten lättare fastnar mellan hindren och inte når den yta som objektet befinner sig på. Detta sker då avståndet mellan hindren minskar vilket gör att det blir svårare för roboten att ta sig förbi dem på grund av dess storlek och når därför inte alla outforskade ytor. Det krävs därför att antalet hinder minskas då ytan blir mindre så att hindren inte står på ett sådant sätt att de blockerar ytor för roboten. I tabell D.2 kan medelvärdena för simuleringarna jämföras.

Tabell D.2: *Tabell över medelvärdena för simuleringsresultaten.*

	10 hinder, 100 m^2	10 hinder, 64 m^2	10 hinder, 25 m^2	5 hinder, 25 m^2
Medelvärde:	2.35 min	3.45 min	6.32 min	1.41 min

D.1.2 Svängningsvinkel vid undvikandet av hinder

Svängningsvinkeln är den vinkel som roboten svänger med för att undvika en vägg eller ett hinder. Olika sätt för hur roboten kan svänga vid möte av ett hinder eller en vägg simuleras för att hitta det bästa svängningssättet. De svängningsvinklar och svängningssätt som är lämpliga för projekter och som således har simulerats är:

1. Slumpmässig svängning inom ett vinkelintervall.
    - Den maximala svängningsvinkeln.
    - Den minimala svängningsvinkeln.
2. Svängningsriktningar
    - Slumpmässigt höger eller vänster.
    - Endast åt höger
3. Fasta svängningsvinklar.
    - Jämfört med svängningsvinklar som slumpmässigt väljs ur ett vinkelintervall.

Simuleringsresultat

Fasta svängningsvinklar
Fasta svängningsvinklar simuleras för att se hur resultatet skiljer sig från det med slumpmässiga svängningsvinklar. De vinklar som testats är 50°, 80° och 100°. Anledningen till varför räta vinklar inte har simulerats är för att de i fyrkantiga rum gör att roboten fastnar i ett rörelsemönster som inte genererar att den finner det sökta objektet. Specifikationerna som används i simuleringen visas i tabell D.3.
Simuleringsresultaten ur figur D.2 visar att slumpmässig rundvandring är ett mer effektivt sätt att ta sig runt i ett rum för att hitta ett objekt. Det beror främst på att det är lättare att ta sig ur ytor där roboten annars kan fastna med ett rörelsemönster. I tabell D.4 finns medelvärdet för att tydligt visa vilken som hade kortast tidsåtgång.

Svängning åt endast höger
En simulering på hur svängning åt endast höger skulle påverka resultatet görs för att sedan jämföras med svängning åt slumpmässigt höger eller vänster. Specifikationerna som används i simuleringarna visas i tabell D.5.

Tabell D.3: *Tabell över specifikationer vid simulering av fasta svängningsvinklar.*

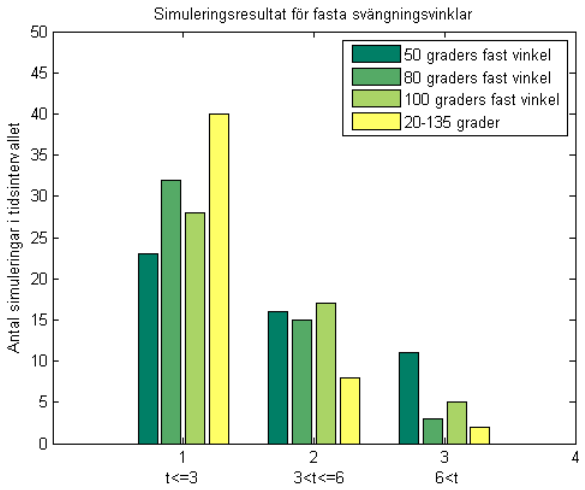
Hinder:	10 stycken fast utplacerade.
Storlek på rum:	10×10 m^2
Svängningsvinklar:	50° 80° 100° Slumpmässigt åt höger eller vänster.
Startriktning:	Slumpmässig

Tabell D.4: *Tabell över medelvärdena för simuleringsresultat vid fasta svängningsvinklar.*

	50°	80°	100°	Slumpmässigt
Medelvärde:	4.92 min	2.81 min	3.08 min	1.87 min

Tabell D.5: *Tabell över specifikationer vid simulering av svängning endast åt höger.*

Hinder:	10 stycken fast utplacerade.
Storlek på rum:	10×10 m^2
Svängningsvinklar:	Slumpmässigt mellan $20° - 135°$ åt endast höger. Samma vinklar men åt slumpmässigt höger eller vänster.
Startriktning:	Slumpmässig



Figur D.2: *Stapeldiagram som visar simuleringsresultatet för hur fasta svängningsvinklar påverkar navigeringsprocessen jämfört med slumpmässiga.*

Resultatet på simuleringarna ger att det i snitt tar längre tid att göra sökningen genom att endast svänga åt höger eftersom det då är svårare att nå mittre delen av rummet. Medelvärdet kan avläsas i tabell D.6 där det tydligt syns att det tar längre tid att endast svänga åt ett håll. I figur D.3 finns tidsfördelningen på simuleringarna tydligt.

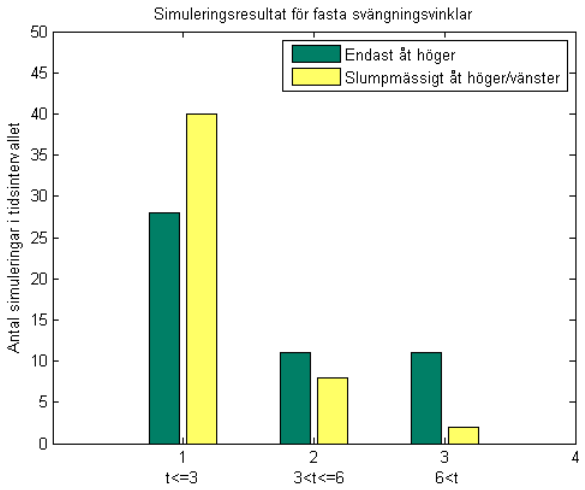
Tabell D.6: *Tabell över medelvärdena för simuleringsresultat vid svängning endast till höger.*

	Endast höger	Höger eller vänster
Medelvärde:	3.89 min	1.87 min

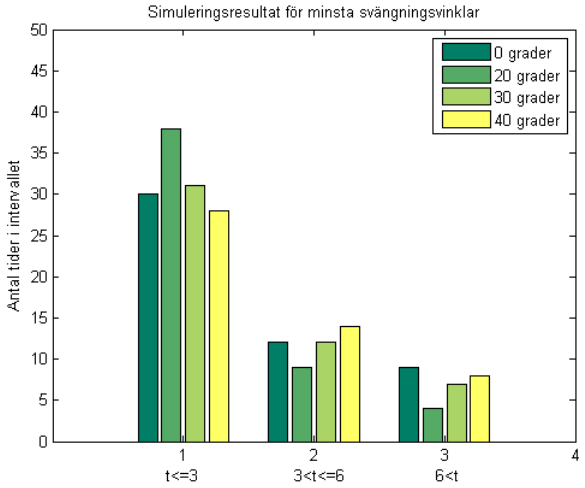
Minimala svängningsvinkeln

För att ta reda på vilken den minimala svängningsvinkeln är i vinkelintervallet vid slumpmässig svängning simuleras olika intervall med samma maximala svängningsvinkel men med varierande minimala svängningsvinklar. Specifikationerna som används vid dessa simuleringar finns i tabell D.7.

Resultatet på simuleringarna över den minsta svängningsvinkeln ett vinkelintervall bör ha för att få roboten att hitta objektet snabbast visar att svängningsvinklar under 20° kan vara onödiga eftersom roboten i de flesta fall då inte kommer loss från det hinder som den ska undvika. Det vinkelintervallet har fått minsta medelvärdet, se tabell D.8, och ur figur D.4 visas det tydligt att det intervallet fått flest bättre tider.



Figur D.3: *Stapeldiagram som visar simuleringsresultatet för hur svängning åt endast höger påverkar navigeringsprocessen jämfört med slumpmässigt åt höger eller vänster.*

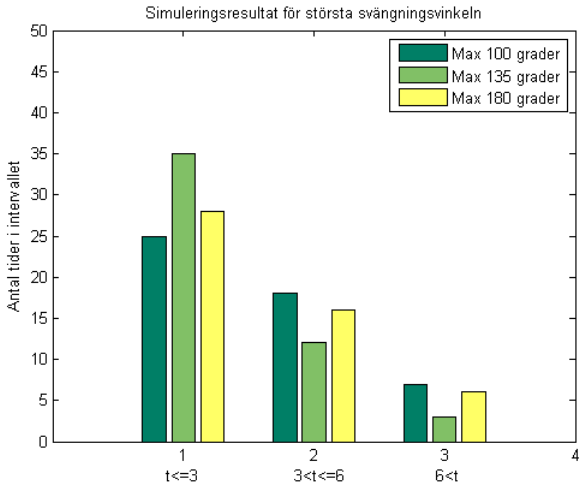


Figur D.4: *Stapeldiagram som visar simuleringsresultatet för minimala svängningsvinkeln i ett vinkelintervall.*

Maximala svängningsvinkeln

För att ta reda på vilken den maximala svängningsvinkeln är i vinkelintervallet vid slumpmässig svängning simuleras olika intervall med samma minimala svängningsvinkel men med varierande maximala svängningsvinklar. Specifikationerna som används vid dessa simuleringar finns i tabell D.9.

Resultatet på simuleringarna över den maximala svängningsvinkeln ett vinkelintervall bör ha för att få roboten att hitta objektet snabbast visar att bäst resultat nås med en maxvinkel på ungefär 135°. Det kan främst bero på att vinklar över det gör att roboten missar för stora ytor eftersom den då nästan helt vänder om till samma riktning som den precis körde. Det är det vinkelintervall som har fått minsta medelvärdet, se tabell D.10, och ur figur D.5 visas det tydligt att det fått flest bättre tider.



Figur D.5: *Stapeldiagram som visar simuleringsresultatet för maximala svängningsvinkeln i ett vinkelintervall.*

D.1.3 Utgångspositionen och startriktningen

Utgångspositionen som valts är placerad i ett av de fyra hörnen på rummet dels för att symbolisera att roboten kommer genom en dörr och dels för att det underlättar uppstartandet av roboten. Sträckorna roboten kör är raka för att det är lättast för roboten att hantera. Svängningen sker sedan stillastående runt dess egen mittpunkt.

Riktningen som roboten åker vid start kan regleras på tre sätt:

- Köra längs ena väggen
- Köra åt motsatt hörn
- Köra åt ett slumpmässigt håll

Tabell D.7: *Tabell över specifikationer vid simulering av minsta svängningsvinkeln.*

Hinder:	10 stycken fast utplacerade.
Storlek på rum:	10×10 m^2
Svängningsvinklar:	$0 - 135°$
	$20 - 135°$
	$30 - 135°$
	$40 - 135°$
	$50 - 135°$
Startriktning:	Slumpmässig

Tabell D.8: *Tabell över medelvärdena för simuleringsresultat vid minimal svängningsvinkel.*

	0°	20°	30°	40°
Medelvärde:	3.58 min	2.66 min	3.07 min	3.35 min

Tabell D.9: *Tabell över specifikationer vid simulering av störst svängningsvinkeln.*

Hinder:	10 stycken fast utplacerade.
Storlek på rum:	10×10 m^2
Svängningsvinklar:	$0 - 135°$
	$20 - 80°$
	$20 - 100°$
	$20 - 135°$
	$20 - 150°$
	$20 - 180°$
Startriktning:	Slumpmässig

Tabell D.10: *Tabell över medelvärdena för simuleringsresultat vid maximal svängningsvinkel.*

	100°	135°	180°
Medelvärde:	3.36 min	2.69 min	3.44 min

Simuleringsresultat

Simuleringen för vilken riktning roboten ska köra vid start simuleras med specifikationerna i tabell D.11.

Resultatet på simuleringarna angående vilken startriktning roboten ska ha

Tabell D.11: *Tabell över specifikationer vid simulering för startriktningen.*

Hinder:	10 stycken fast utplacerade.
Storlek på rum:	10×10 m^2
Svängningsvinklar:	$20-180°$ slumpmässigt åt höger eller vänster.
Startriktning:	Längs ena väggen. Mot motsatt hörn. Slumpmässigt.

vid sitt sökande gav inte så stor skillnad mellan de olika testade riktningarna. Minsta medelvärde, se tabell D.12, fick den riktning som går längs ena väggen på rummet. Tidsfördelningen kan ses i figur D.6 där det tydligt syns att skillnaden mellan de olika är markant men att den snabbaste riktningen hade mindre tider över sex minuter.

Tabell D.12: *Tabell över medelvärdena för simuleringsresultat av startriktningen.*

	Längs väggen	Mot motsatt hörn	Slumpmässigt
Medelvärde:	2.23 min	2.73 min	2.45 min

Figur D.6: *Stapeldiagram som visar simuleringsresultatet för startriktningen.*