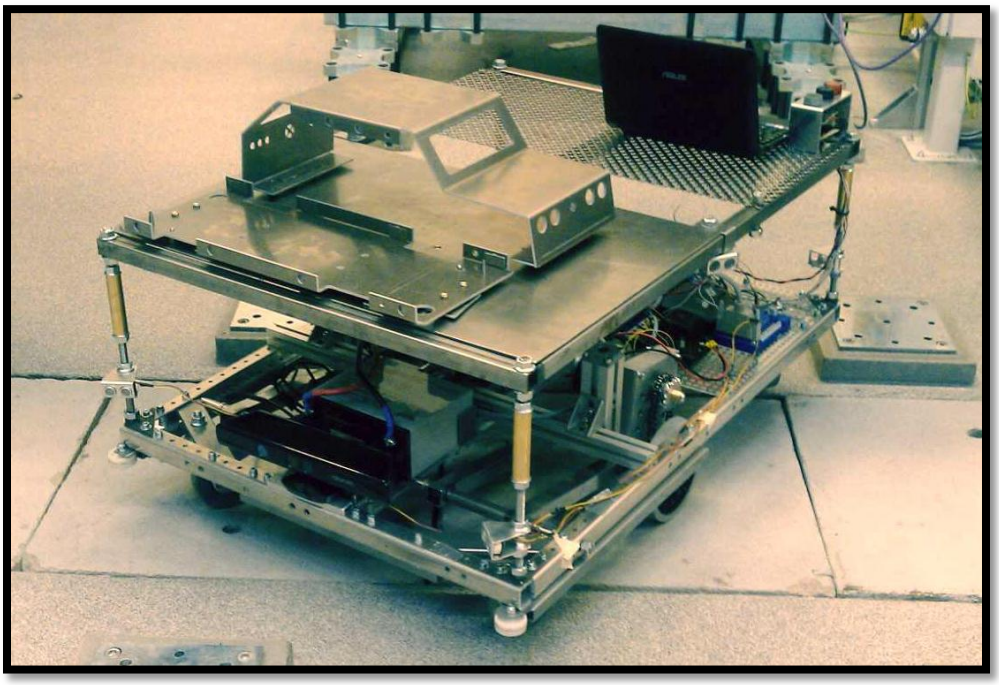
CHALMERS



Utveckling av AGV för materialtransport till produktionscell

med detektering av burkar

Kandidatarbete inom civilingenjörsprogrammet Automation och Mekatronik

Fredrick Bergström
Viktor Elisson
Jakob Fjellström
Erik Hultgren
Fredrik Jonazon
Patrik Werner

Forskargrupp
Institutionen för signaler och system
Chalmers tekniska högskola
Göteborg, Sverige 2012
Kandidatarbete SSYX02-12-10

Abstract

The increased use of automatic production in modern industries sets higher demands on a production system's security and flexibility. The use of Automated Guided Vehicles (AGV), which is a vehicle that can navigate automatically, enables a production system to operate continuously and minimizes the need for human interference. During the spring of 2012 an automated production system that produces a miniature car model in a variety of setups was built at Chalmers University of Technology. This thesis is covering the development of an AGV that supplies that production system with materials on request. The AGV also have a secondary mission when delivery is not needed, to find and pick up soda cans that are spread out in the vicinity of the production system. During the development new and old components from last year's project have been tested and evaluated. Programs were developed according to flowcharts made in an early stage of the project. The program for material transport was the priority since this was the main mission of the AGV. Due to an overall shortage of time the secondary mission was not fully developed. No pickup device for picking up soda cans was developed, however the AGV can find and drive to the soda cans. The end result of this project was an AGV able to accomplish its main mission and detect soda cans in the vicinity of the production system.

Sammanfattning

I takt med att fler och fler produktionslinor inför automatisk tillverkning ställs nya krav på säkerhet och flexibilitet. Med en Automated Guided Vehicle (AGV) som är en typ av självstyrande robot kan arbetet i en produktionscell påverkas utan stopp i produktionen och minska människors närvaro vid maskiner. Under våren 2012 byggs en produktionscell vid Chalmers tekniska högskola som slutligen skall producera en bil. Detta projekt syftar till att utveckla en AGV för transport av material till produktionscellen. Vid sidan om detta skall AGV:n detektera och samla upp burkar när inget material finns att leverera till cellen. Gruppen har konstruerat en AGV med framdrivning i mitten och swivelhjul fram och bak. Både nya som gamla komponenter har testats och utvärderats. Gruppen har skapat flödesscheman för de olika delarna av programmeringen och utifrån dessa skrivit de program som krävs för leverans av material. Under projektets gång prioriterades uppsamlingen av burkar bort då detta var den sekundera uppgiften. Detta resulterade i att AGV:n inte kan plocka upp och frakta bort de burkar som hittas. Slutresultatet blev en AGV som uppfyller huvudsyftet att leverera material till produktionscellen och detektera burkar.

Innehåll

1. Inledning

1.1 Bakgrund

I och med att fler och fler produktionslinor idag använder sig av automatiserad tillverkning ställs nya krav på säkerhet, flexibilitet samt effektivitet. Även om industrirobotar utvecklas och blir säkrare kan en miljö med interaktion mellan människor och robotar vara olämplig för människor att arbeta i. Med hjälp av en så kallad Automated Guided Vehicle (AGV) som är en typ av självstyrande robot kan arbetsplatsen bli både säkrare och mer effektiv. Genom att använda sig av en AGV istället för människor, i till exempel en produktionscell, kan man påverka arbetet i cellen utan att behöva stoppa produktionen, vilket man oftast måste då människor arbetar vid robotar. Detta medför ökad effektivitet. Eftersom människans närvaro vid roboten under körning minimeras, bidrar detta till en säkrare arbetsmiljö (HK systems, 2006).

AGV:erna blir alltmer sofistikerade med bättre sensorer och större beräkningskapacitet. De kan utföra mer komplicerade uppgifter vilket gör att användningsområdet kan utökas och eftersom AGV:n kan programmeras med kända rutter kan risken för kollision minimeras. De kan även ta över de uppgifter som är enformiga och slitsamma samt de uppgifter som kan vara skadliga eller ohälsosamma för en människa (HK systems, 2006).

Användningen av AGV:er började på 50-talet men det är först på senare år som mer avancerade modeller har utvecklats. Idag används AGV:er främst inom industri- och lagerverksamhet men troligtvis kommer det bli allt vanligare att se AGV:er inom andra områden då man på ett säkrare sätt kan hantera kontakten mellan människa och maskin (Elettric 80, 2008).

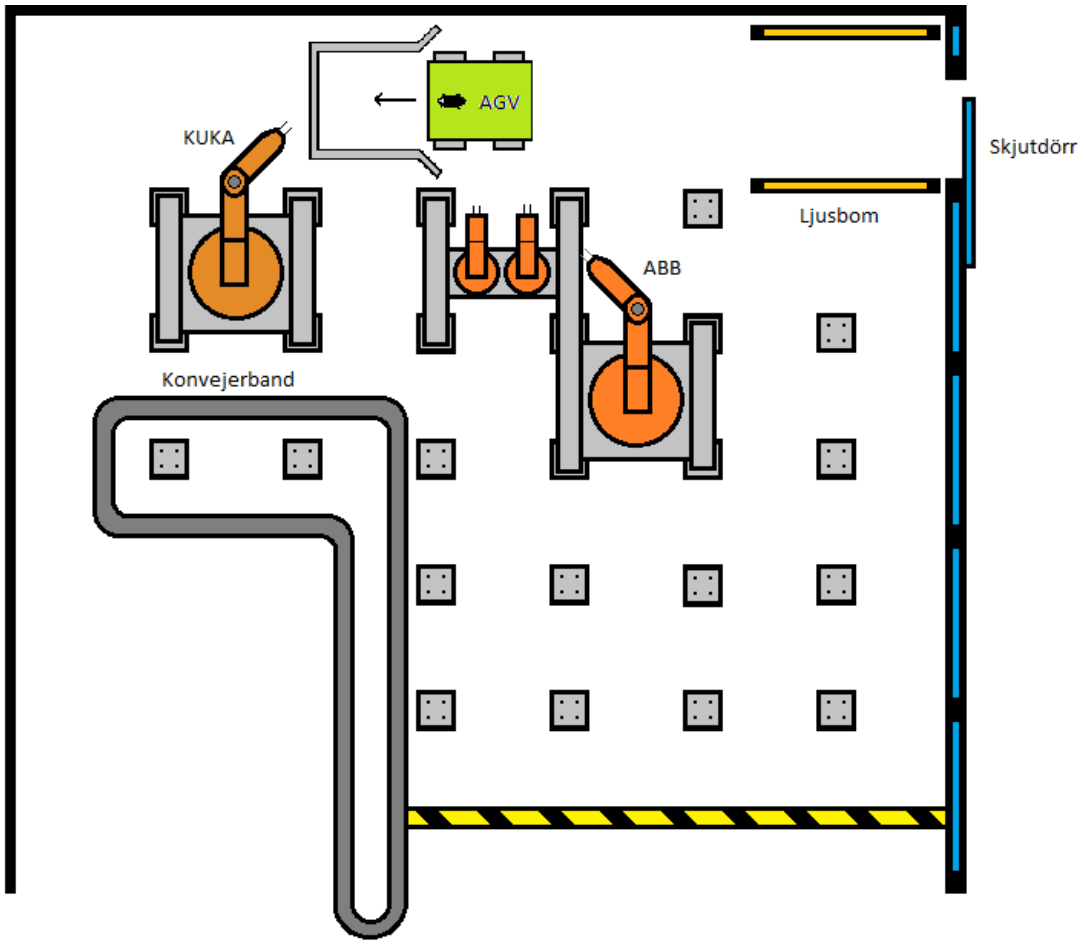
Under våren 2012 skall en ny produktionscell designas och byggas i Produkt- och Produktionsutvecklings laboratoriet (PPU) på Chalmers. I cellen skall ett antal bildelar sättas ihop, placeras i en fixtur och monteras med industrirobotar. En Programmable Logic Controller (PLC) kommer att styra det övergripande systemet och koordinera produktionen. Ett antal kandidatarbeten kommer att arbeta med utveckling av olika delar av cellen och det slutgiltiga resultatet skall demonstreras för representanter från lokala industriaktörer. Det här kandidatarbetet behandlar utvecklingen av AGV:n som ska användas till att leverera material till cellen och det bygger på ett tidigare kandidatarbete. (Andersson, 2011)

1.2 Syfte

Projektet syftar till att, genom sammansättning av mekanik, elektronik och datorsystem, utveckla en AGV som kan förse en produktionscell med nödvändigt material för tillverkning. AGV:n tar emot beställningar från robotcellen genom en centraldator och ska därefter leverera materialet till en förutbestämd plats. När AGV:n inte har någon beställning att bearbeta ska den hitta burkar och samla upp dessa. AGV:n ska under hela tiden undvika att kollidera med sin omgivning, vilket inkluderar en annan AGV och människor som rör sig i lokalen. AGV:n ska även utvecklas på ett sådant sätt att man under kommande år på ett enkelt sätt kan använda enheten som grund för ytterligare kandidatarbeten.

1.3 Problembeskrivning

Primäruppgiften för AGV:n är att leverera material till produktionscellen på beställning av huvuddatorn. När ingen beställning finns att ta om hand skall AGV:n som sekundäruppgift detektera och frakta bort burkar. För att lösa detta delas problemen in i de fyra underkategorierna: transport, kommunikation, konstruktion och detektering. För överblick av produktionscell se Figur 1: Produktionscell.



Figur 1: Produktionscell

I transport ingår ett flertal olika uppgifter. Det första och viktigaste är att AGV:n ska kunna transportera material mellan kända punkter i produktionscellen. Under denna transport måste roboten köra så pass lugnt att inte godset faller av, samtidigt som tiden för transporten från startpunkt till slutpunkt ska ta så kort tid som möjligt för att undvika stillestånd i produktionen. Då AGV:n är framme vid slutpunkten måste precisionen på denna punkt vara så pass bra att robotarna kan plocka upp materialet från AGV:n lika säkert varje gång. I övrigt måste AGV:n även kunna transportera upplockade burkar till en förvald plats.

Med kommunikation menas två saker: kommunikationen mellan AGV:ns olika komponenter samt kommunikationen mellan AGV:n och produktionscellens huvuddator. I kommunikationen mellan AGV:ns olika komponenter är uppgiften att testa och förstå hur de olika komponenterna fungerar samt vad de vill ha för in- och utsignaler. Utifrån detta ska gruppen sedan kunna programmera den AGV:ns styrdator CompactRIO från National Instruments. För kommunikationen mellan AGV:n och

huvuddatorn är uppgiften att lösa hur och när AGV:n ska prata med huvuddatorn. AGV:n måste kunna kommunicera med huvuddatorn för att veta när material ska levereras till robotcellen och för att huvuddatorn ska veta var AGV:n befinner sig.

Till projektet behövs fyra olika konstruktionsprojekt. AGV:ns chassi måste omkonstrueras för att den ska kunna ta sig fram i PPU-laboratoriets miljö. En anordning ska konstrueras på AGV:n för att hålla fast det material som ska transporteras i en fix punkt. Någon typ av positioneringssystem måste byggas för att AGV:n ska få tillräckligt hög precision i robotcellen. Till sist måste AGV:n på något sätt kunna plocka upp de burkar som hittas i lokalen.

Under kategorin detektering menas att AGV:n ska kunna undvika att kollidera med olika föremål, vilket gör att den på något sätt måste upptäcka människor och annan utrustning. AGV:n ska även kunna detektera och bestämma avstånd till de burkar som ska plockas upp.

1.4 Avgränsningar

I robotens anti-kollisions funktion kommer alla föremål som befinner sig inom dess detekterings-zon att antas vara stillastående och roboten behöver därför inte kunna avgöra om ett föremål rör på sig. Burkarna som roboten ska samla upp är avgränsat till röda burkar för att detekteringen inte ska bli för avancerad och tidskrävande. På grund av prioritering av huvuduppgiften konstruerades aldrig någon upplockningsanordning till AGV:n.

2. Hårdvara

För att AGV:n ska klara av sina uppgifter krävs diverse olika komponenter. Nedan kommer en beskrivning av de viktigaste komponenterna. Vissa av dessa har använts på AGV:n under tidigare år medan vissa är nya för detta år.

2.1 CompactRIO (NI cRIO-9074)

CompactRio är en realtidskontroller med en processor på 400 MHz som används för att kontrollera, logga och analysera system. CompactRio har två stycken ethernetportar, en RS232 serieport samt åtta modulplatser som kan användas för att utöka kompabiliteten. CompactRio kan programmeras i LabVIEW, C, C++ eller java (Wikipedia, 2012). Från tillverkaren National Instruments är LabVIEW det som rekommenderas då det är de som har gjort detta program (National Instruments, 2011).

Figur 2: CompactRIO

2.2 Digital I/O modul (NI9403)

Digital I/O modul är till för att skapa eller lägga till fler digitala ingångar till NI CompactDAQ eller CompactRIO (National instruments, 2012 (2)).

Figur 3: Digital I/O modul

2.3 Seriemodul (NI9870)

NI 9870 är en I/O modul som innehåller 4 stycken RS232 portar, den är kompatibel med CompactRIO (National instruments, 2012 (1)).

Figur 4: Seriemodul

2.4 Bildbehandlingsdator Asus

Datorn som används för bildbehandlingen på AGV:n är en ASUS EeePc 1011 px. Den är utrustad med 1 GB RAM minne, tre USB portar, en Atom N570 processor på 1,66 GHz samt en Ethernet port (Asus, 2012). Datorn används till att styra Kinectsensorn, vilken ansluts genom en av USB portarna.

Figur 5: Bildbehandlingsdator

2.5 Motorkontroller

MDL-BDC24 är en motorkontroller utvecklad av Texas Instruments. Den kan driva 12V och 24V DC-motorer med upp till 40A kontinuerlig ström. Motorkontrollen är utrustad med en RS232- till CAN-brygga vilket gör att två kontroller kan kopplas ihop och styras från samma RS232-port. Den har även en encoder-ingång som kan kopplas från en motors växellåda för att erhålla reglering av hastighet och få möjlighet att styra hur många varv hjulen ska snurra (Texas Instruments, 2011). Kontrollerna styrs via en seriell RS232-ingång där en sekvens av bytes säger vilken motor meddelandet avser och vad den ska göra.

Figur 6: Motorkontroller

2.6 Kinect

Figur 7: Kinect

Kinect är en kamera som i första hand är utvecklad till Xbox 360 av Microsoft för att kunna styra spel med hjälp av kroppen istället för en traditionell hankontroll. På Kinectsensorn sitter det tre kameror, en RGB-kamera för färgigenkänning och två kameror som används för djupseende. Enheten ansluts till dator via standard USB-kabel. Microsoft har nyligen släppt ett eget utvecklingskit, kallat Kinect SDK, för programmering av Kinectsensorn i Windowsmiljö (Microsoft, 2012(2)). Vid användning av detta bibliotek kan man utveckla program i C++, C# och Visual Basic.

2.7 Linksys WGA600N

Figur 8: Linksys

Linksys WGA600N är en trådlös nätverksadapter som kan användas för att koppla upp en dator eller spelkonsol etc. mot ett trådlöst nätverk i de fall dessa endast har ett trådbundet nätverkskort. Den stödjer de vanliga typerna av Institute of Electrical and Electronics Engineers-standarder (IEEE) för trådlöst nätverk. Den är bra att använda till produkter utan grafisk operativsystem då man kan ställa in vilket nätverk man vill ansluta till via en dator och sedan kommer den försöka ansluta till detta nätverk oavsett vilken enhet den är kopplad till så fort den blir påslagen (Cisco Systems, 2008).

2.8 Lasernavigering SICK NAV200

Figur 9: SICK NAV200

SICK NAV200 är en laserskanner som används för positionering i industrimiljöer. Lasern roterar 360 grader och detekterar fördefinierade reflektorer. Reflektorerna är placerade i rummet och med dem skapas ett koordinatsystem. Det krävs minst tre reflektorer för att räkna ut position och vinkel i koordinatsystemet. Noggrannheten för position är specificerad till 4 mm och 0.1 grader för vinkel (SICK, 2006).

2.9 Ultraljuds sensor SRF 05

Figur 10: Ultraljudssensor

SRF 05 är en sensor som med hjälp av ultraljud mäter avståndet till det närmste objektet i dess synfält. Den har en maxlängd på 4 meter och kan göra en ny mätning var 50 ms (Robotstorehk, 2012).

2.10 Power Distribution Board (PDB)

Figur 11: PDB

PDB ingår i de robotkit som säljs av andymark.com. På denna finns 8 säkrade utgångar upp till 40A respektive 12 utgångar upp till 30 A . Den agerar också som dc-dc omvandlare från 6-15V batterispänning till 5, 12 och 24V (USFIRST, 2011)

2.11 Digital sidecar AM-0266

Digital sidecar AM-0266 är ett kort som ansluts till Digital I/O modul (NI9403). Den är en äldre version av AM-0866(Andymark, 2012(1)). Kortet fungerar som en brygga mellan CompactRIO:n och diverse I/O-enheter t.ex. ultraljud och tryckknappar.

Figur 12: Digital sidecar AM-0266

3. Metod

Projektet har delats upp i ett flertal deluppgifter för att underlätta arbetet. För varje del har diverse information sökts fram och studier av föregående års kandidatarbeten gjorts. Den mesta informationen har hämtats från tillhandahållna datablad och manualer till de komponenter som finns samt från studier av andra liknande projekt.

För att lösa deluppgifter har kunskap från tidigare kurser använts. Främst kunskap från kurserna maskinorienterad och objektorienterad programmering har använts för att lösa de komplexa programmeringsuppgifter som projektet innefattar. Även kunskaper från systemkonstruktion och industriautomation har varit viktiga för att få en övergripande förståelse för hela produktionscellen och kunna använda ingenjörsmetodik i framtagningen av AGV:n.

Möten med övriga projektgrupper, relaterade till produktionscellen, har hållits på regelbunden basis där de aktuella gruppledarna representerar respektive projektgrupp. Vidare har också en kontinuerlig kontakt med övriga projektgrupper behövts för att kommunicera kring de gemensamma arbetsmomenten, där de berörda gruppernas samtliga medlemmar medverkar. Detta har krävts då de olika projektgrupperna ställer olika krav på varandras delar, som alla måste ta hänsyn till.

Gruppen har i första hand fokuserat på att lösa de problem som tillhör primäruppgiften, vilket var nödvändigt för att hela robotcellen skulle fungera.

3.1 Samarbeten

För att kandidatarbetet skulle vara möjligt att genomföra krävdes det att gruppen i olika grad samarbetade med alla grupper som tillhörde projektet med produktionscellen i PPU-laboratoriet.

Gruppen hade mest samarbete med den andra AGV-gruppen. Framgångar och tankar delades regelbundet mellan grupperna då båda gruppernas AGV:er har samma huvuduppgift. Denna uppgift kräver en hög precision inne i robotcellen, vilket gör att AGV:erna har samma yttre mått och en gemensam dockningsstation.

Det krävdes också en del samarbete med gruppen som programmerar industrirobotarna. Detta eftersom robotarna direkt interagerar med AGV:erna vid hämtning och avlämning av material. För detta bestämdes en fix punkt i robotcellen som fungerade för båda grupperna. I detta ingick även en placering av bildelarna på AGV:n som gör det möjligt för robotarna att plocka dem med sina verktyg.

PLC gruppen, som bestod av en gruppmedlem från samtliga projektgrupper, bestämde när de olika processerna i cellen kommer att utföras, till exempel inkörning av AGV i cell. Den utvalde gruppmedlemmen förmedlade denna information till resten av gruppen så att kommunikationen mellan AGV:n och PLC:n uppfylldes enligt specifikation från PLC-gruppen.

3.2 Transport

Metoden för att lösa de flesta av problemen som tillhör kategorin transport var programmering av CompactRIO:n och en Androidapplikation. Problem som precision och navigeringsförmåga byggde även på att vissa konstruktionsproblem löstes. Dessa behandlas senare i avsnittet konstruktion.

3.2.1 CompactRIO

Då stor del av AGV:n styrs från CompactRIO:n lades mycket tid till att inspektera och förstå kod som fanns tillgänglig från föregående år samt, genom olika instruktionsvideor och dokument, lära sig programmeringsspråket. Utifrån detta bestämdes vilka nya program som var tvungna att skrivas samt hur dessa skulle fungera. För att få detta överskådligt gjordes ett par flödescheman (se Bilaga 1: CompactRIO - Huvudrutin och Bilaga 2: CompactRIO – Burksamlingsrutin). Under själva programmeringstiden användes metoden ”Trial and Error”.

3.2.2 Androidapplikation

En applikation för Android som är ett operativsystem för mobiler och surfplattor utvecklades i projektets senare skede. Detta programmerades i programmeringsspråket Java med tillägget Android SDK (Google, 2012). Denna applikation utvecklades för att dels kunna styra AGV:n manuellt med en mobiltelefon men också för att kunna simulera en PLC-server för testkörning i cellen innan PLC-koden var färdig. Utvecklingsmiljön som användes var Eclipse (The Eclipse Foundation, 2012).

3.3 Kommunikation

För att komma underfund med hur kommunikationen internt och externt på AGV:n skulle fungera var alla komponenter tvungna att undersökas. Alla komponenter har undersökts med hjälp av tillhörande datablad. Till vissa komponenter som motorkontrollerna och SICK NAV200 fanns det även medföljande programvaror som underlättade arbetet. Med hjälp av dessa kunde en del initiala inställningar göras. Då komponenten sedan kopplats till CompactRIO:n användes enkla testprogram, skrivna endast för detta syfte, till att få en djupare förståelse i hur komponenten i fråga kommunicerar med CompactRIO:n.

3.4 Konstruktion

Konstruktionsmässiga beslut har tagits med hjälp av konstruktionsmetodik där de olika förslagen och dellösningarna har viktats mot varandra utifrån uppfyllande av den initialt skapade kravspecifikationen. Även andra tekniska beslut har tagits med ingenjörsmässiga metoder.
Under hela projektet har testning gjorts för att verifiera att rätt resultat har erhållits.

3.5 Detektering

Nedan följer en beskrivning av hur beslut har fattats samt vilka metoder som använts för att göra framsteg i projektet, mer specifikt inom området detektering.

3.5.1 Collision Avoidance

Detekteringen av hinder måste ske via någon typ av sensor. För att välja vilken typ som skulle användas jämfördes olika alternativ. Alternativen utvärderades och egenskaper som flexibilitet, synvinkel och möjlighet att se objekt av olika storlek och typ, värderades högt. Efter valet av sensor, utfördes även en del tester för att få en ännu klarare bild utav funktionalitet, styrkor samt brister hos sensorn. Då all information lades samman kunde beslut om sensorplaceringen på själva AGV:n tas.

3.5.2 Burkdetektering

Vid detektering av burkar används en Kinectsensor. Till denna gjordes först ett val av programmeringsspråk. För att få en överblick för programmets funktioner och när kommunikation med CompactRIO:n behövs gjordes ett flödesschema. Under programmeringsprocessen genomgicks olika lösningar för att ta fram en bra algoritm för att detektera röda burkar.

4. Lösningsförslag

Till de fyra problemområden som identifierades i problembeskrivningen behandlas här de lösningsförslag som övervägdes under utvecklingen av AGV:n. Metoderna som användes beskrevs i föregående kapitel.

4.1 Transport

Under problemen för transport var det ett problem som var tvunget att lösas i projektets start. Detta problem var val av språk för programmering av CompactRIO:n. Som nämnt i avsnitt 2.1 kan CompactRIO:n programmeras med fyra olika programspråk. Fördelarna med språket LabVIEW är att det är National Instruments egenutvecklade mjukvara vilket medför att det finns mycket dokumentation för programmering av CompactRIO till detta språk. Tidigare kandidatarbetens kod är också skriven i LabVIEW vilket förenklar processen av kodskrivandet. Nackdelarna med LabVIEW var att ingen av gruppmedlemmarna hade några förkunskaper inom språket. Då det gäller de andra programspråken är detta något alla gruppens medlemmar har viss erfarenhet av. Dock är dessa språk inte lika visuella och intuitivta som LabVIEW. Eftersom gruppen undersökte mycket av den, från förra året, befintliga koden fick gruppen mer erfarenhet i ämnet. Detta tillsammans med de tydliga kopplingarna mellan CompactRIO:n och LabVIEW gjorde att LabVIEW blev ett mer fördelaktigt språk att använda för ändamålet.

Övriga problem inom transport löstes utifrån det initialt gjorda flödesschemat för huvudrutinen (se Bilaga 4: Flödesschema för AGV-huvuddator). Denna huvudrutin gjordes på en generell nivå för att innehålla så många av AGV:ns funktioner som möjligt. Vid programmeringsarbetets start skrevs program för att lösa problem på en låg nivå, så som protokoll för kommunikation mellan CompactRIO och andra komponenter. Programmen innehöll gradvis fler funktioner för att täcka upp funktionerna i det tidigare nämna flödesschemat.

4.2 Kommunikation

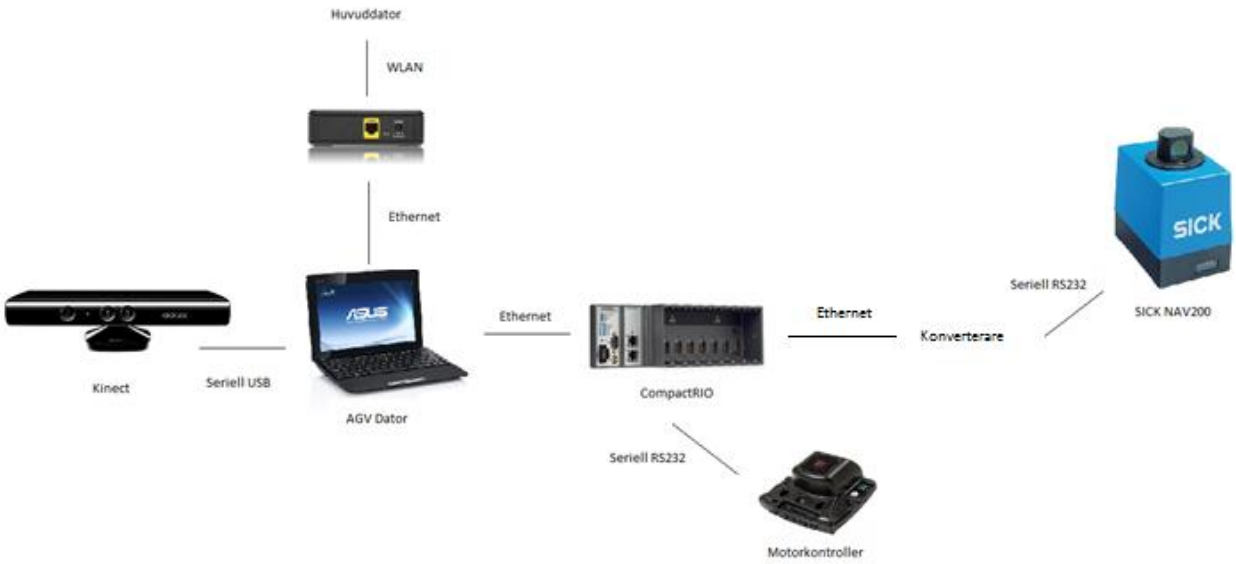
Kommunikationen på AGV:n delas upp i intern och extern kommunikation. Den interna kommunikationen syftar på kopplingar mellan komponenter på AGV:n och den externa syftar på sättet AGV:n kommunicerar med det övriga systemet i produktionscellen.

4.2.1 Intern kommunikation

Hur de olika komponenterna skulle vara kopplade till varandra utvärderades. Tillsammans med handledaren Alexey Voronov arbetades tre lösningsförslag fram som presenteras i Figur 13 - Figur 15. Lösningarna som arbetades fram tar hänsyn till att CompactRIO:n endast har en RS232 port, men det finns möjlighet att utöka antalet portar med hjälp av moduler. Både lasern och motorkontrollerna kommunicerar via RS232 vilket innebär att en lösning som möjliggör RS232-kommunikation för båda komponenter måste tas fram.

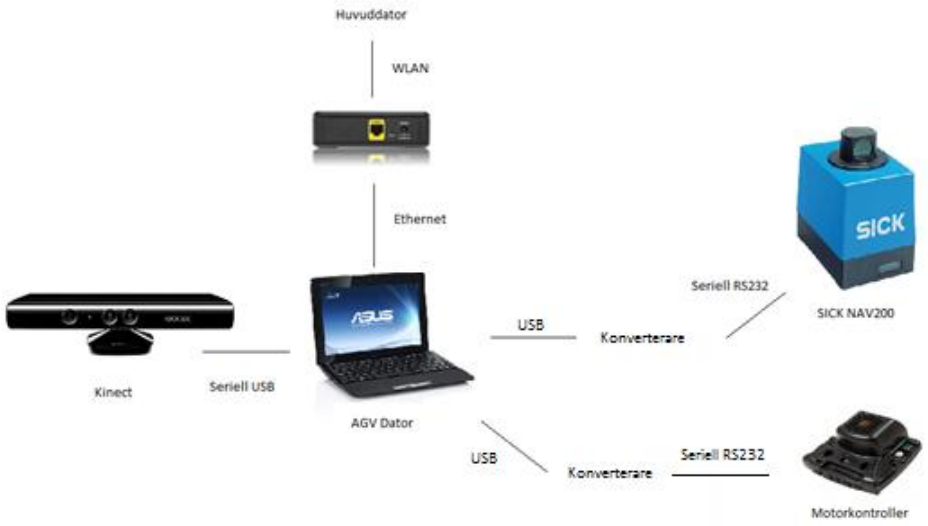
Förra årets kandidatarbete (Andersson, 2011) använde sig av en mikroprocessor, beagleboard, för bildbehandlingen. Handledarna har i år beslutat att den ska bytas ut mot en bärbar dator, ASUS EEEpc 1011px (avsnitt 2.4). Datorn ger nya möjligheter för att lösa problemet med för få portar. Exempelvis skulle USB eller konverterare kunna användas. Gruppen har därmed större möjlighet att ändra hur den interna kommunikationen ska gå till.

I lösningsförslag ett (Figur 13) används CompactRIO för att styra navigering med laser och motorkontroller. Motorkontrollerna använder då CompactRIOs lediga serieport och lasern är kopplad till ethernet-porten via en RS232-TCP/IP konverterare. Laptopen sköter bildbehandlingen som detekterar burkar och kommunicerar med CompactRIO via ethernet. Fördelar med denna lösning är att ingen extra modul behöver köpas in till CompactRIO och navigeringen kan göras i realtid med CompactRIO. Nackdelar blir att lösa interfacet mellan lasern och CompactRIO med en konveterare.



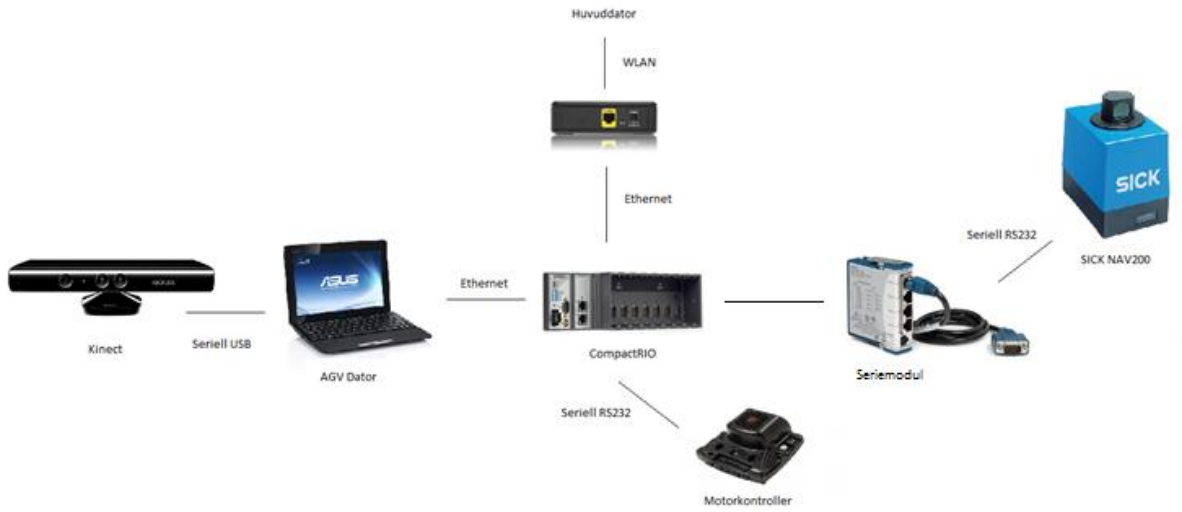
Figur 13: Intern kommunikation, lösningsförslag 1

Lösningsförslag två (Figur 14) är att använda laptopen som styrdator för hela systemet. Det skulle göra kommunikationen till lasern och motorkontrollerna enkel genom att använda datorns USB-portar. Den skulle också kunna programmeras i C vilket gruppen sen tidigare har erfarenhet av. Nackdelen blir att datorn inte hanterar processer i realtid på samma sätt som CompactRIO och att LabVIEW-program från tidigare år för styrning av motorkontroller inte kan användas eller kan behöva göras om.



Figur 14: Intern kommunikation, lösningsförslag 2

Lösningsförslag tre (Figur 15) använder CompactRIOs serieport för att styra motorkontrollerna. Kommunikationen med lasern sker via en RS232-hub alternativt inköp av modul för CompactRIO. Laptopen sköter bildbehandlingen som detekterar burkar och kommunicerar med CompactRIO via ethernet. Fördelen med detta förslag är att motorkontrollerna kan styras med tidigare års program och att navigeringen görs i realtid. Nackdelen är att lösa kommunikationen med lasern. Förra årets kandidatarbete (Andersson, 2011) använde de sig av en egentillverkad seriehubb och efter samtal med dem visades det sig att de hade haft problem med hubben och att koden måste skrivas specifikt för den hubben, vilket ger en låg kompabilitet och robusthet om den skulle gå sönder. Alternativet är att köpa in en seriemodul till CompactRIO:n som är en mer robust lösning eftersom det är en från samma tillverkare men den har samtidigt ett högt pris.



Figur 15: Intern kommunikation, lösningsförslag 3

Med detta som utgångspunkt valdes lösningsförslag 3 (Figur 15). I detta lösningsförslag styrs bildbehandlingen, vilken är icke tidskritisk men processkrävande, med hjälp av ASUS EEEpc. Den tidskritiska styringen, så som NAV200, ultraljud samt motorkontroller, styrs av CompactRIO. Bildbehandlingsdatorn och CompactRIO:n kommer att kommunicera via en ethernetkabel eftersom det finns en ledig sådan och ingen konvertering mellan gränssnitt kommer att behövas. Då styrningen bör göras i realtid förkastades alternativet att använda ASUS EEEpc för styrning av hela systemet.

Önskemål från examinator och handledare har varit att projektet under kommande år ska fokusera mer på utveckling än konstruktion samt att det ska byggas ett mer robust system. Seriemodulen ger då flest fördelar eftersom man enkelt kan ansluta fyra olika komponenter via RS232 utan extra modifiering. Gruppen beslutade med detta underlag att köpa in seriemodulen NI9870.

4.2.2 Extern kommunikation

Vid start av detta arbete fanns endast ett förslag för hur AGV:n ska kommunicera med resten av produktionscellen, vilket var via en trådlös nätverksadapter som kopplas direkt in i CompactRIO:n med en ethernetkabel (se avsnitt 2.7). I och med att en ny bildbehandlingsdator inhandlades uppkom möjligheten att kommunikationen skulle gå genom denna. Fördelen med detta är att det skulle bli en komponent mindre på AGV:n samt att det är väldigt enkelt att koppla upp en laptop mot ett trådlöst nätverk. Dock blir fördröjningarna längre om signalerna ska gå genom en laptop och nya program

måste tas fram för hur laptopen ska ta emot meddelanden från CompactRIO:n och skicka dessa vidare till det övergripande systemet i cellen. Då gruppen senare lärde sig att det var enkelt att installera nätverksadaptern till CompactRIO blev detta val mer fördelaktigt på alla sätt.

För programmeringen av PLC:n skapades en ny grupp bestående av fem medlemmar från olika kandidatgrupper ifrån produktionscellen. Gruppen hade bland annat i uppgift att lösa kommunikationen mellan PLC och AGV. Inför programmeringen av PLC:n gjordes operationslistor som beskrev vad som behöver hända i och runt cellen, samt vilka signaler som behövs påverkas vid respektive operation för att bilen skulle kunna produceras. Operationerna för AGV:n prestenteras i Bilaga 9: PLC, operationer för AGV. Eftersom det har varit många oklarheter kring PLC:n och hur kommunikationen i detalj ska fungera så har en alternativ lösning av kommunikationen tagits fram. Istället för att PLC:n kommunicerar direkt med AGV:n så kan man låta detta ske med hjälp av en dator eller mobil. I Human-Machine Interface (HMI) i PLC:n programmerar man virtuella knappar som man sedan trycker på manuellt när AGV:n behöver kommunicera med PLC:n. Med detta lösningsförslag kan man få ett fungerande system utan någon trådlös kommunikation med PLC:n.

4.3 Konstruktion

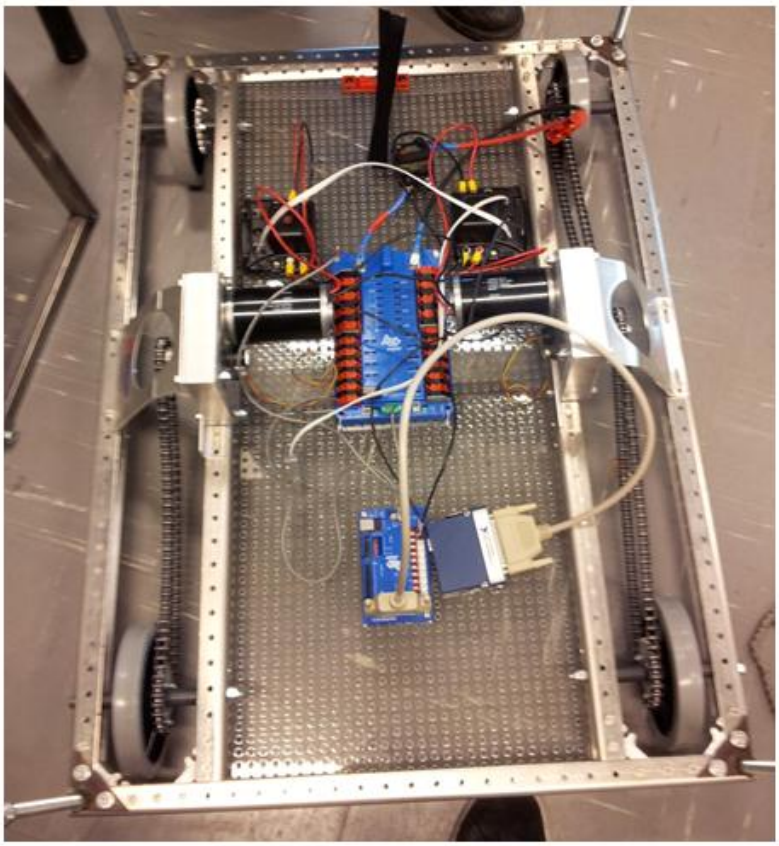
Efter kommunikation med deltagare i tidigare kandidatprojekt, elever och handledare, konstaterades det att AGV:ns mekaniska egenskaper i det skick som den köpt från leverantören inte var tillfredställande för dess uppgifter i robotcellen. Då golvet i lokalen som innehåller cellen är utrustat med kabelkanaler som är täckta med betongplattor uppstår ojämnheter och springor som AGV:n måste kunna hantera på ett bra sätt. Baserat på dessa aspekter togs en kravspecifikation (se Tabell 1) fram som sätter de krav och önskemål på AGV:n lämpligen bör klara.

Tabell 1: Kravspecifikation AGV

1.	**Prestanda**	
	1.1.	K. Ska kunna ta sig över springor i golvet
	1.2.	K. Ska kunna lämna delar på specifik position upprepade gånger
	1.3.	K. Ska kunna specifikt avgöra kring vilken punkt roboten svänger
	1.4.	K. Ska kunna köra i gånghastighet
	1.5.	K. Ska Kunna bära angiven last
	1.6.	Ö. Bör Kunna docka i flera stationer
	1.7.	Ö. Bör Liten svängradie
	1.8.	Ö. Bör ha en bra position på svängpunkt
	1.9.	Ö. Bör vara stabil
	1.10.	Ö. Bör ha god manövrerbarhet
2.	**Produktion**	
	2.1.	Ö. Bör återanvända så mycket som möjligt av befintligt material
	2.2.	Ö. Bör vara så billig som möjligt att tillverka
	2.3.	Ö. Bör ha en så låg mekanisk komplexitet som möjligt
3.	**Dimensioner**	
	3.1.	K. Skall kunna komma genom dörr och säkerhetsutrustning
	3.2.	K. Skall rymma lasten och nödvändig elektronik
	3.3.	Ö. SICK NAV 200 bör blockeras så lite som möjligt

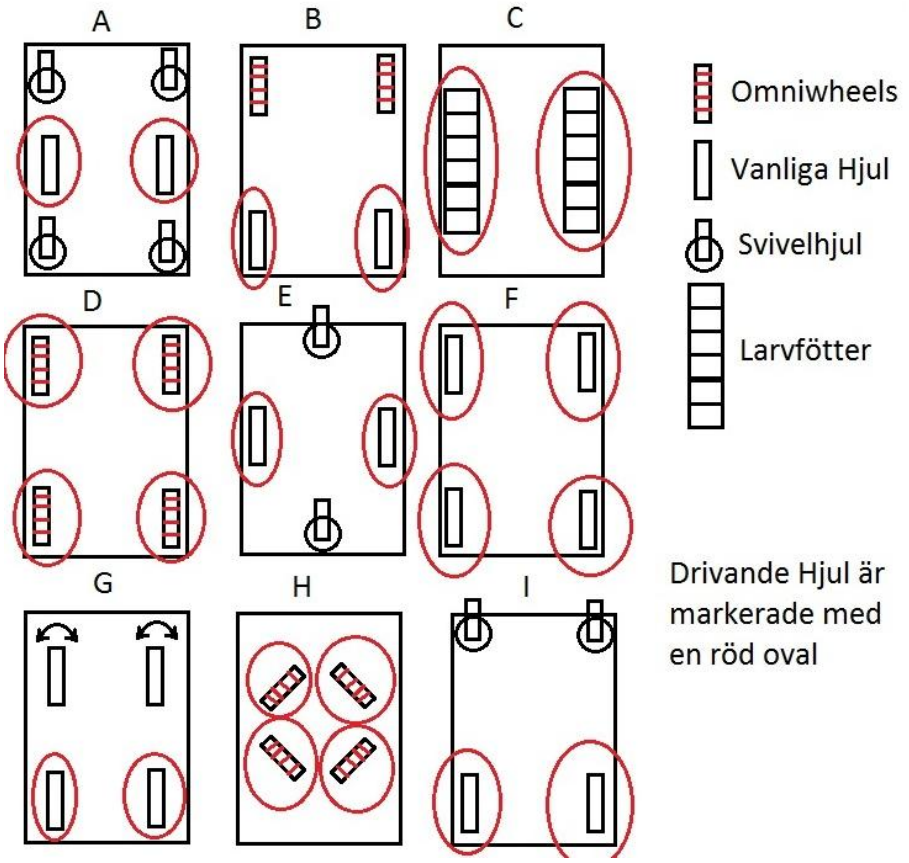
4.3.1 Hjulplacering och drivning

Den ursprungliga konfigurationen (se Figur 16) med fyra hjul drivna av två motorer via kedjor innebar ett antal problem. Det huvudsakliga var en dåligt definierad rotationspunkt vilket försvårade manövreringen och positioneringen med SICK NAV200. Dessutom innebar de fyra hjulen att ett relativt stort friktionsmotstånd var tvunget att övervinnas vid rotation kring en punkt. Detta i sin tur ledde till ett stort effektbehov från motorerna som resulterande i ett hackigt och ojämnt rörelsemönster. Med växellådorna monterade mitt mellan hjulen krävdes också långa kedjor vilka innebar ett visst slack och därmed en dödtid innan rotationen från motorn förmedlas till hjulen vid byte av rotationsriktning eller vid hastiga varvtalsförändringar.



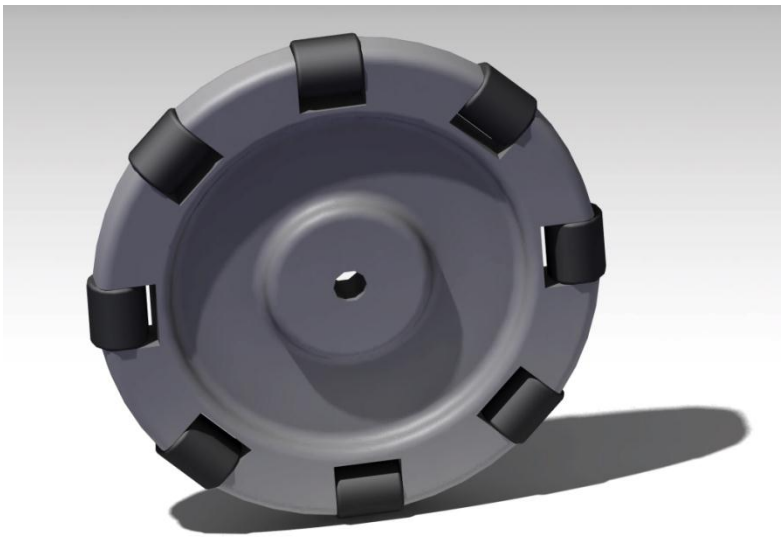
Figur 16: Ursprungligt Utseende

Utifrån kravspecifikationen användes enligt metodavsnittet konstruktionsmetodiska tillvägagångssätt för att ta fram förslag och välja ut det som är mest lämpligt. Ett antal delfunktioner observerades och till dessa togs dellösningsförslag fram. Dessa i sin tur kombinerades till totallösningsförslag enligt Figur 17. För en mer utförlig beskrivning av arbetsgången se Bilaga 7: Omkonstruktion av AVG.



Figur 17: Totallösningsförslag

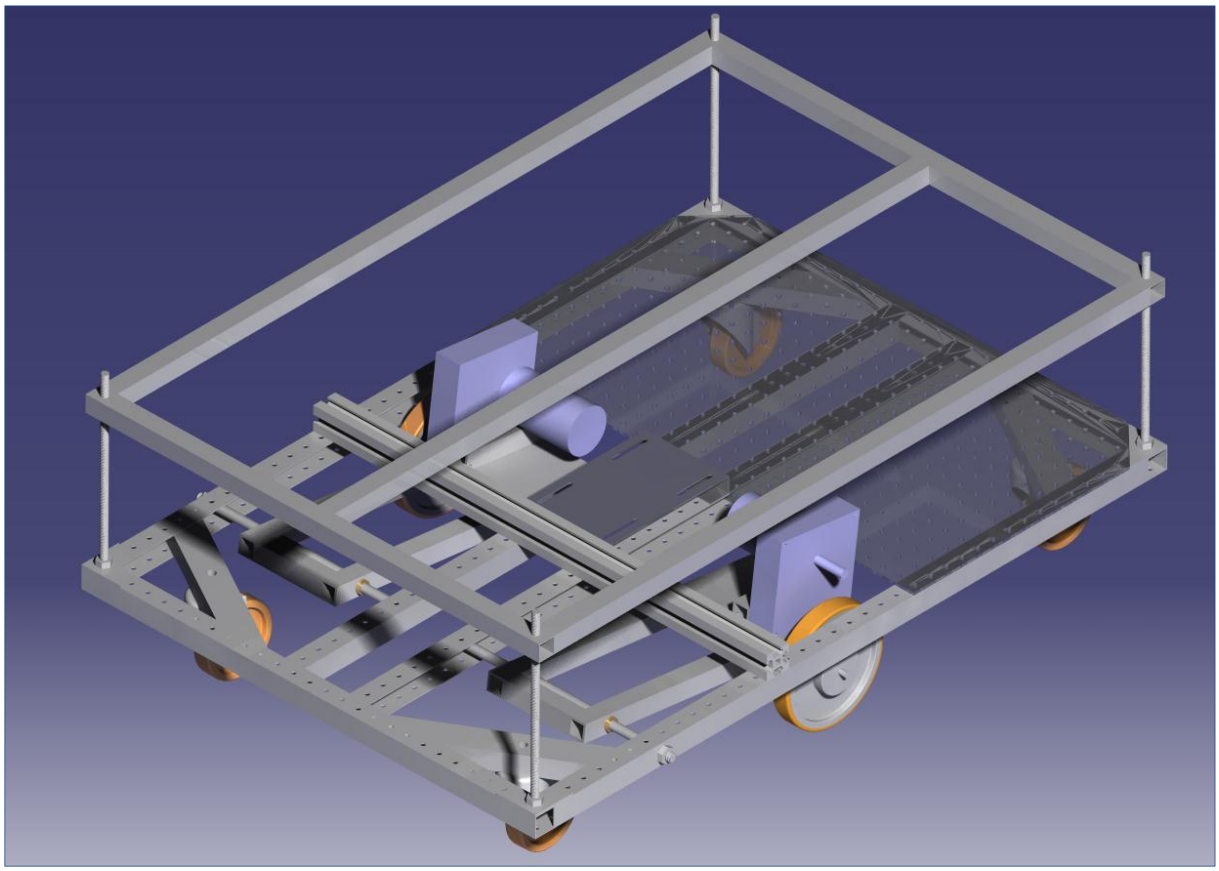
Två huvudsakliga spår kan observeras bland lösningsförslagen. Ett av de förslagen var byta ut de befintliga hjulen mot så kallade omnidirectional wheels (omniwheels) vilket i teorin skulle innebära en bättre definierad rotationspunkt och en minskad friktion vid rotation, lösningsförslag D ifrån Figur 17. Omniwheels (se Figur 18) är till utseende och storlek skiftande men gemensamt för denna typ av hjul är att de är utrustade med en mängd små rullar fördelade kring ett större hjuls ytterradie. Detta möjliggör förflyttning i flera riktningar. Dessa hjul skulle sedan kunna placeras enligt olika mönster med varierande antal motorer för att uppnå skilja egenskaper.



Figur 18: Omniwheel

Det andra förslaget var att byta fyrhjulsdriften mot tvåhjulsdrift med ett antal stödhjul, lösningsförslag A (Figur 17). Detta förslag skulle erbjuda en mycket väl definierad rotationspunkt om drivhjulen placerades i mitten och även lösa problemet med friktion i rotationsrörelsen. För att uppnå god framkomlighet över ojämnheter i golvet skulle detta dock innebära att en fjädrad upphängning av hjulen eller ramen skulle vara nödvändig för att inte drivhjulen skall bli hängande i luften vid körande över sådana. Vid placering av drivhjulen i någon ände av AGV:n skulle den mekaniska komplexitetten i ombyggnationen bli lägre men en icke önskvärd stor svängradie skulle uppstå även om rotationspunkten skulle bli väl definierad. Med tvåhjulsdriften möjliggörs montering av motorerna närmre drivhjulen och således kortas längden på kedjorna vilket innebär en minimering av slacket och dödtiden i drivningen.

Den lösning som fick mest poäng i utvärderingen visade sig vara lösning A som kom att bli den lösning som gruppen gick vidare med. En CAD modell ritades upp (se Figur 19) och ett antal beräkningar av motorerna och batteriets kapacitet gjordes för att försäkra att resultaten skulle bli de önskade. För en mer detaljerad beskrivning av beräkningarna se Bilaga 8: Beräkningar.



Figur 19: Lösningsförslag A

4.3.2 Dockningsstation

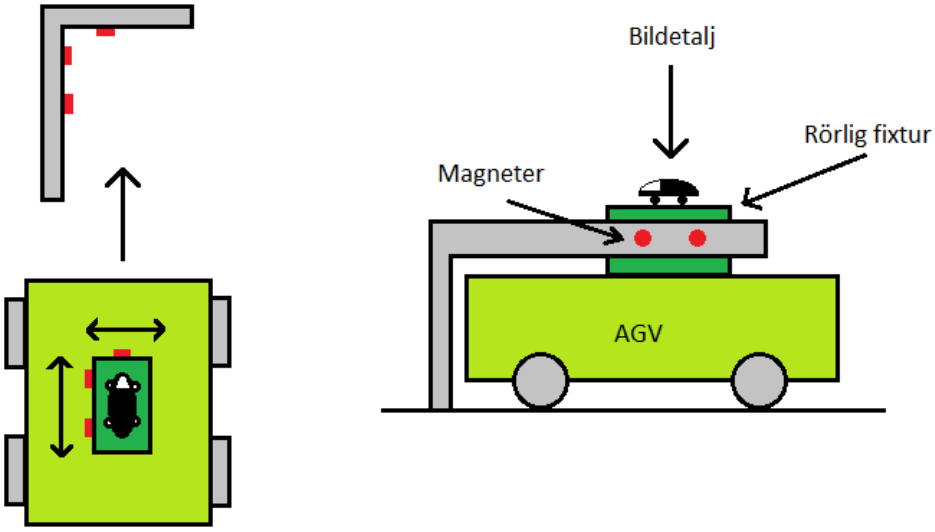
Vid avlämning av bildetaljer i produktionscellen ställs krav på AGV:ns positioneringsnoggrannhet. Då industrirobotarna programeras absolut efter fixa punkter krävs detta för att avlämning och upphämtning av material sker korrekt. Då SICK NAV200 inte erbjuder en tillräcklig noggrannhet måste någon form av hjälpmedel användas för att styra AGV:n till en väl definierad punkt. I detta fall genom att AGV:n låses fast i någon form av dockstation.

Som första steg togs en kravspecifikation fram för att väl definiera vilka krav och önskemål som ställdes på dockstationen (Tabell 2). Från denna kravspecifikation genererades ett antal lösningsförslag med hjälp av brainstorming och internetsökning.

Tabell 2: Kravspecifikation dockstation

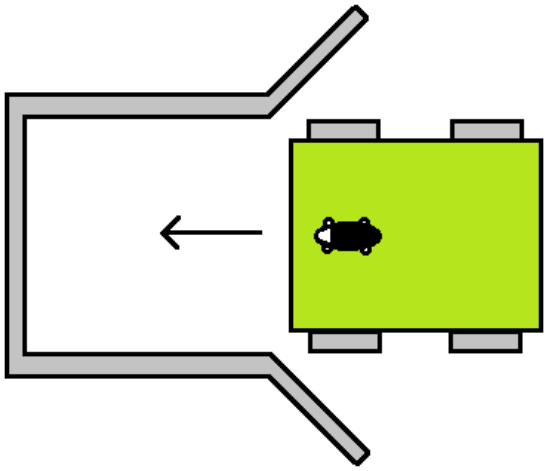
Kriterier	Målvärde	Krav/Önskemål
Konstruktionskostnad	<2000kr	Ö
Tillverkningsbarhet	Kunna tillverkas i prototyplaboratoriet	K
Tillverkningstid	Kunna färdigställas innan projektets slut	K
Tillverkningstid	Så kort tid som möjligt	Ö
Sammarbete med andra maskiner	Skall kunna kommunicera med övrig utrustning	K
Repeternoggrannhet	±0.5 mm & ±0.5 grader	K
Utrymmesbehov	Få plats i cellen och rymma AGV:n	K

Dockning med magnet:
En fixtur rörlig i ett plan parallellt med golvet placeras på AGV:n och utrustas med magneter. I höjd med fixturen placeras en trattformad dockningsstation vilken också utrustas med magneter. Genom att låta AGV:n navigera till en punkt under dockstationen styrs fixturen till rätt plats och låses till rätt position med magneterna. (Se Figur 20).



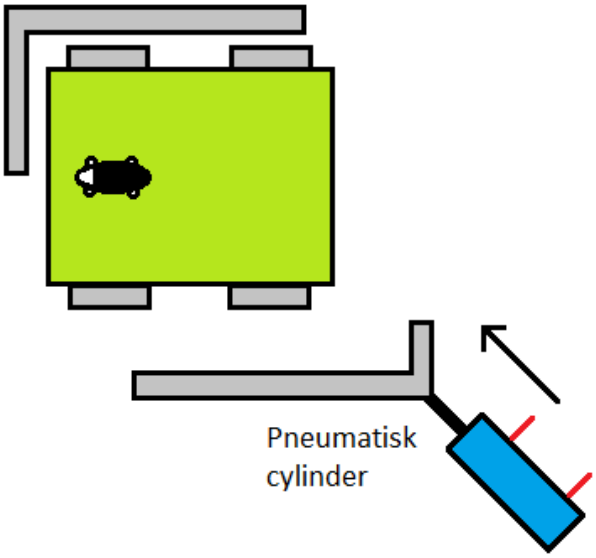
Figur 20: Dockning med magnet

Dockning genom pressning:
Genom att bygga en trattformad dockstation belägen på golvet i vilken AGV:n kan köra in med hjälp av egen motorkraft och justeras till en fix punkt kan en god noggrannhet uppnås. Antingen genom att AGV:n har glidskenor i sidorna för god passning eller med hjälp horisontellt monterade hjul. (Se Figur 21).



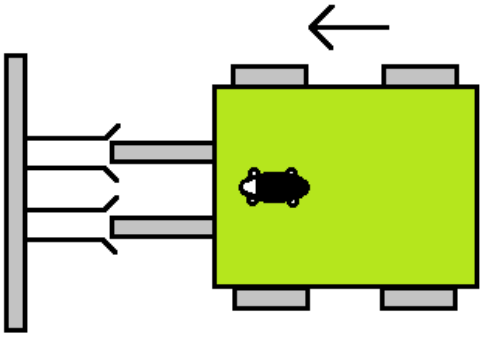
Figur 21: Dockning genom pressning

Dockning med pneumatiska cylindrar:
Om dockstationen förses med pneumatiska cylindrar som låser AGV:n i ett fix läge varje gång skulle en god repeterbarhet uppnås. En sådan lösning vore robust men arbetsinsatsen och kostnaden för konstruktion av en sådan får bedömmas som hög. (Se Figur 22).



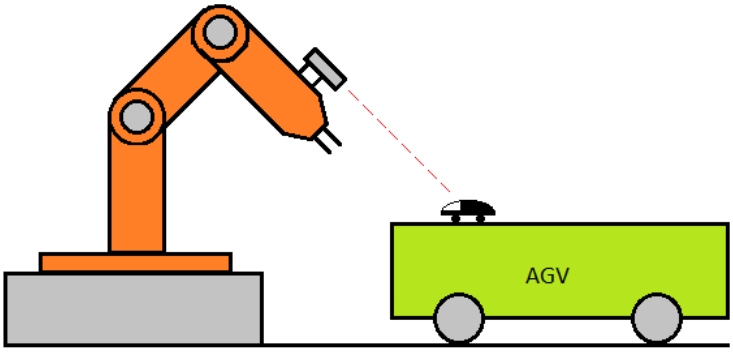
Figur 22: Dockning med pneumatiska cylindrar

Dockning genom styrstavar i AGV:ns front:
Om AGV:n förses med ett par stavar i fronten som passar mot ett par konor i en dockningsstation skulle AGV:n kunna styras till en specifik position med god noggrannhet. För att underlätta AGV:ns passning av stavarna i dockstationen kan sidskenor enligt förslag dockning genom pressning användas. (Se Figur 23).



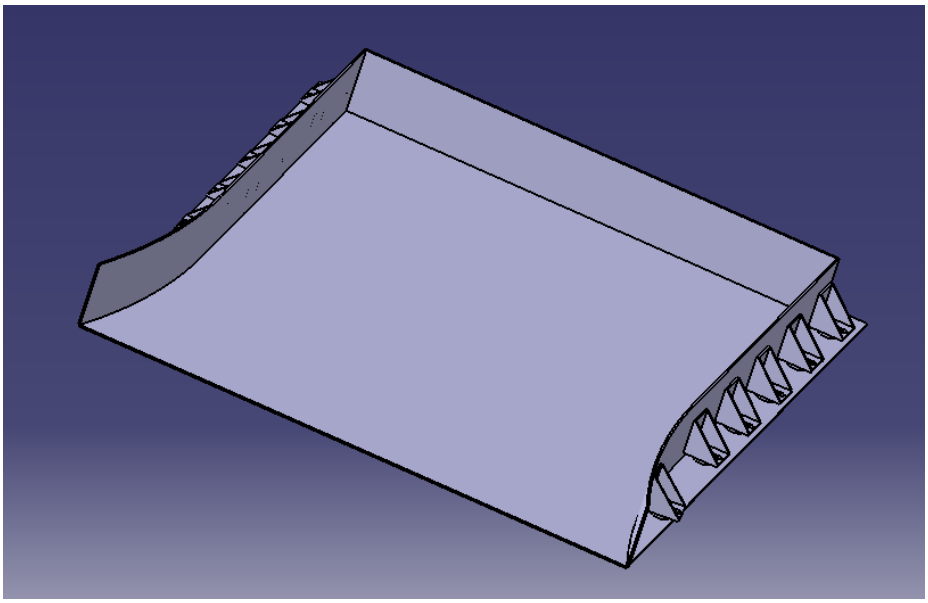
Figur 23: Dockning genom styrstavar

Robot mäter in AGV:ns position:
Om Industriroboten försågs med ett mätsystem för att kunna mäta in fixturens position skulle SICK NAV200s noggrannhet vara tillräckligt god. (Se Figur 24).



Figur 24: Robot mäter in AGV:ns position

Lösningseliminering:
Av lösningsförslagen uppfyller förslagen dockning med magnet, dockning genom pressning och dockning genom styrstavar i AGV:ns front kravspecifikationen. För att vidare avgöra vilket förslag som på bästa sätt uppfyller de ställda önskemålen vägdes förslagens för och nackdelar mot varandra. Samtliga kunde antas ha en produktionskostnad under 2000kr däremot vid avseende på tidsåtgång vid tillverkning fanns en större inbördes variation. Då i synnerlighet dockning via magnet som ansågs kräva en stor arbetsinsatts avseende den rörliga fixturen. De kvarvarande två alternativen måste anses ha en likvärdig tidsåtgång vid konstruktion. Slutligen valdes alternativet dockning genom pressning (se Figur 21) då det ansågs vara en robustare lösning som också var enkel att tillverka i prototyplaboratoriet. Denna lösning används i industrin för smalgångstruckar (Atlet, 2012).



Figur 25: Cad av dockstation

Enligt Figur 25 togs en cadmodell för dockstationen fram som detaljlösningsförslag. I denna är dockstationen i sin helhet beskriven gällande materialval och fastelement.

För att systemet skall veta när AGV:n är i rätt position i dockstationen har flera olika metoder diskuterats. De olika alternativen kan sammanfattas med att antingen förse AGV:n eller dockstationen med någon form av sensor som känner av när AGV:n åkt hela vägen in i dockan. Detta skulle kunna vara en mekanisk brytare eller en induktiv givare. Ett annat alternativ är att stanna när SICK NAV200 ger information om att en viss punkt är nådd.

Baserat på tillgängligheten på givare, material och tid för att bygga ett system som känner av då AGV:n nått hela vägen i dockan kommer i ett första skede enbart positionen från SICK NAV200 att användas. Det är dock ej uteslutet att det efter testning beslutas om ytterligare utrusning ska göras för att skapa ett robustare system.

För att se till att AGV:n glider lätt in i dockstationen kommer någon form av glidskena eller rullhjul monteras på dess sidor. För att säkerställa att AGV:n kommer att stå ordentligt i dockstationen hela vägen in mot bakstycket skall en skena eller plåt monteras på AGV:n mot vilken den stannar.

4.3.3 Fixtur för bildelar

För att kunna leverera material till robotcellen med god noggrannhet och utan risk för att tappa delar krävs någon form av fixtur som säkerställer detta. Diverse lösningsförslag diskuterades men i och med att den bil som togs fram av fixturgruppen ska sättas ihop med styrpinnar och magneter så var lösningsförslaget att utnyttja detta. Genom en plåt som placeras ovanpå AGV:n med uppborrade hål som passar med styrpinnarna på bilen uppnås en väldigt noggrann positionering.

Ett annat lösningförslag var att ha en separat vagn med fixturen på som AGV:n skulle lämna av och hämta inne i cellen. Detta hade gjort AGV:n mer flexibel genom att den inte blir låst att leverera just de delar som passar på fixturen på AGV:n utan man kan ha olika fixturer på olika vagnar.

4.3.4 Fästdetajer

För att skydda AGV:ns komponenter från att falla av eller utsättas för stötar under AGV:ns körning, tillverkades en rad olika fästanordningar. Många komponenter gick att skruva fast i basramens befintliga plastplatta, till exempel CompactRIO:n eller motorkontrollerna. Till de komponenter som antingen inte kunde fästas lika lätt, eller hade något speciellt krav på placering, tillverkades det speciella fästanordningar. Dessa komponenter var: batteriet, lamp- och knappanel, SICK NAV200 och samtliga ultraljudssensorer. Valet om hur varje fäste skulle skapas gjordes till skillnad ifrån de övriga delarna inom konstruktionskapitlet, inte genom konstruktionsmetodik utan genom diskussioner inom gruppen. Anledningen till beslutet var bedömningen att konstruktionsmetodiken i sig skulle ta så mycket tid i förhållande till tillverkningstiden och att fästets resultat inte skulle påverkas speciellt mycket.

Designen av samtliga fästen gjordes med tanken att de skulle vara så lätta som möjligt att tillverka, samtidigt som de skulle vara tillräckligt robusta och välgjorda att de inte skulle gå sönder och behöva göras om i framtiden.

4.4 Detektering

Under detektering har främst två olika problem hanterats. Dessa är val av Collision Avoidance-sensorer samt val av programmeringsspråk för Kinectsensorn. Detta avsnitt beskriver lösningsgången och genomförandet utav de två punkterna.

4.4.1 Collision Avoidence-sensorer

För att undvika kollisioner har tre olika alternativ utvärderas inom gruppen: Ultraljudssensorer, sensorer som använder sig av infrarött ljus samt den redan befintliga Kinectsensorn. Då förra årets projekt (Andersson, 2011) redan hade kollat på för och nackdelar med IR-sensorer gentemot ultraljudsensorer och kommit fram till att ultraljudsensorer var bättre till denna uppgift, valdes IR-sensorerna bort i ett tidigt skede.

Fördelarna med ultraljudsensorerna gentemot Kinecten är att de är relativt små och därför kan placeras på valfria platser på AGV:n, de är även enkla att koppla till CompactRIO:ns I/O-portar. Då det gäller Kinectsensorn finns det tillgång till kod som borde vara smidig att använda i AGV:n eftersom det finns ett annat kandidatarbete på Chalmers som specifikt är inriktat på Collision Aviodence med hjälp av Kinect. Eftersom bildbehandlingsdatorn inte används under utförande av primäruppgiften passar det bra att den istället används för att upptäcka hinder. En nackdel med Kinectsensorn är att den endast kan kolla åt ett håll i taget och utan något sorts vridtorn måste den vrida hela AGV:n för att kolla åt sidorna. Då man skulle behöva kolla sidorna innan man svänger runt, för att undvika kollision, kan detta bli ett stort problem. För och nackdelar vägdes samman och det beslutades att ultraljudssensorerna var det mest lämpliga alternativet.

4.4.2 Burkdetektering

Val av programspråk:

Programmering av Kinectsensorn gjordes i språket C# eftersom det finns bra och mycket dokumentation för det språket. Kinectsensorn, C# och även Kinect SDK är utvecklade av Microsoft, och från de efterforskningar gruppen gjort så har det framkommit att Microsoft rekommenderar att

man använder C#. Då båda grupperna valde att arbeta i C#, skapades större möjligheter till sammarbete då utbyte av kod var möjlig mellan grupperna. Utvecklingsmiljön som användes var Visual Studio 2010 (Microsoft, 2012 (1)).

Från olika sökningar på internet så verkade det inte finnas några färdiga funktioner i Kinect SDK för att sålla ut eller lokalisera den röda färgen på burken, och därmed hitta den, utan en egen funktion var tvungen att skrivas. En generell funktionsstruktur för hur programmet skulle fungera togs fram och ser ut enligt Figur 26.

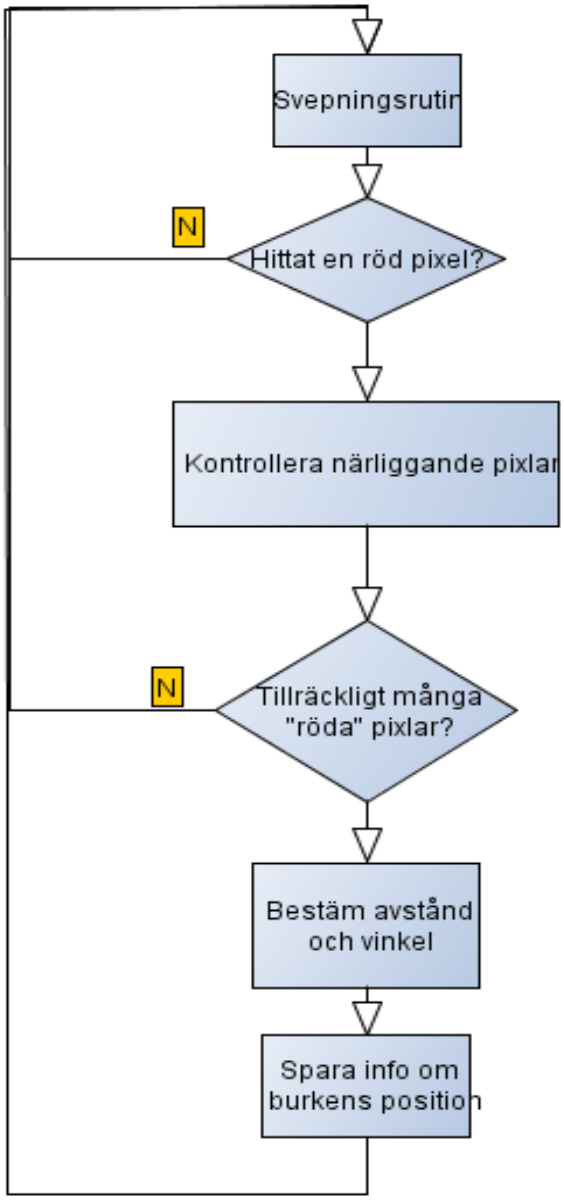
Svepningsrutin:

Här togs två förslag fram. Det ena var att slumpa en pixel och sedan kontrollera om denna är ”röd” och göra om detta ett antal gånger per bild (frame). Fördelen med detta är att det inte är tar särskilt mycket prestanda, dvs exekveringstiden blir kort dock kan det ta tid att hitta själva burken.

Det andra lösningsförslaget var helt enkelt att kontrollera alla pixlar i varje bild med möjlighet att ”hoppa över” pixlar om det kräver för mycket prestanda. Fördelen med detta är att man alltid hittar burken om det finns någon men nackdelen är att det kan gå långsamt att köra programmet. Hoppar man över pixlar i genomsökningen blir det svårare att hitta burken på längre avstånd.

Kontrollera närliggande pixlar:

Här togs endast ett lösningsförslag fram vilket var att utifrån den första påträffade röda pixeln ska ett område i form av en rektangel undersökas. Om det blev för beräkningsintensivt så skulle endast konturen av rektangeln analyseras.



Figur 26: Flödesschema för burkdetektering

Bestäm avstånd och vinkel:

Avståndet till varje pixel kan i princip fås direkt av rådatan från djupkameran. Djupdatan från varje pixel är två bytes stort, där de första 13 bitarna representerar avståndet i millimeter (James, 2012). För att få ett bra värde så ska medelvärdet av avståndet till de pixlar som kontrollerades i ”kontrollera närliggande pixlar” funktionen användas.

För att räkna ut eller uppskatta vinkeln till burken togs det fram två lösningsförslag. Det första är att endast ta reda på om burken ligger i mitten, till vänster eller till höger i bilden och ha ett intervall för dessa värden. Ett annat alternativ är att dela kinectens synfält, vilket är 57 grader (Sriharsh Biswal, 2011) med den totala pixelbredden och räkna ut vinkeln enligt:
$\varphi = [grad/pixel] \ * pixel\_xled - \ synfält/2.$

5. Resultat

Detta kapitel bygger på de metodval och lösningsgångar som föregående kapitel tagit upp och beskriver resultatet av projektets olika delar

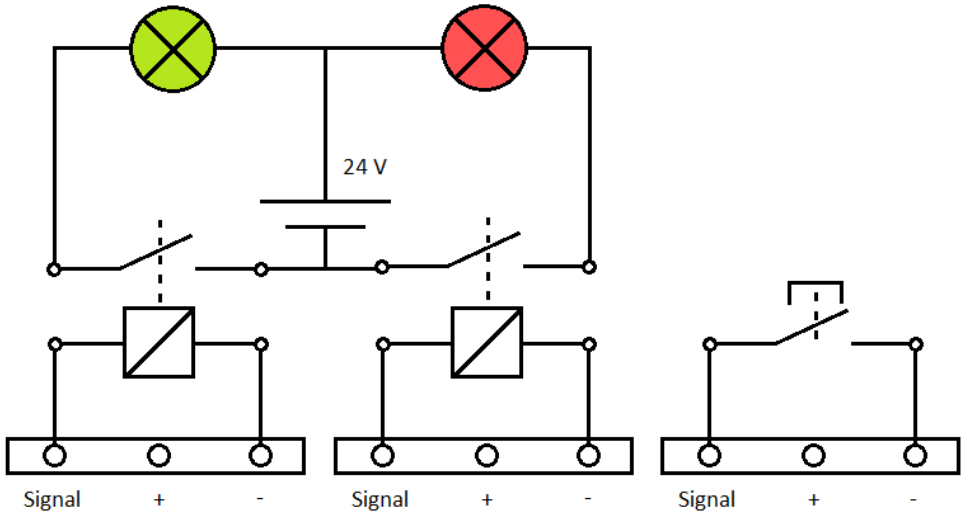
5.1 Transport

För att förklara programmen bakom transport har ämnet delats in i två delar: LabVIEW och Androidapplikation. Båda avsnitten är till för att visa på resultatet av programmeringen samt få en bild av hur programmen fungerar.

5.1.1 LabVIEW

Två olika flödesscheman togs fram för att beskriva informationsflödet i CompactRIO:n. Huvudrutinen (se Bilaga 1: CompactRIO - Huvudrutin) tar hand om leverans av bildelarna till produktionscellen samt startar burksamlingsrutinen (se bilaga 2) om material ej ska levereras. Om AGV:n håller på att leta burkar då nytt material ska levereras kommer AGV:ns koordinater och vinkel att sparas undan. Huvudrutinen kan när som helst avbryta burksamlingsrutinen om AGV huvuddatorn meddelar att delar ska levereras till produktionscellen. Då sedan AGV:n ska fortsätta att leta burkar, kommer dessa koordinater att anges som startkoordinater. Detta gör att AGV:n inte behöver börja om från början efter varje leverans av material.

Vid avhämtning av material samt kvittering av fel har en knappanel utrustad med två lampor och en strömbrytare tagits fram. Två reläer styrda av I/O modulen bryter eller sluter två kretsar anslutna till 24v dc-dc omvandlaren på AGV:ns power distribution board enligt kretsschemat i Figur 27.

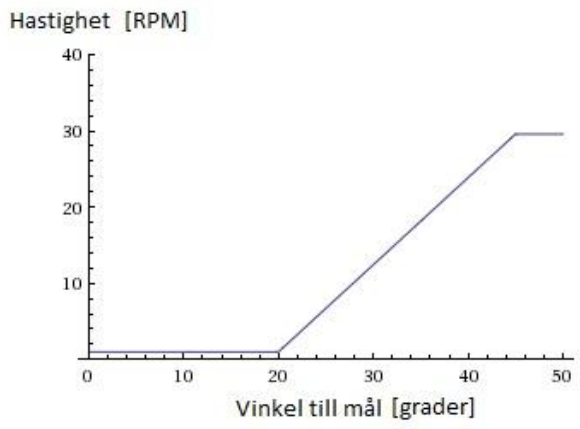


Figur 27: Kretsschema

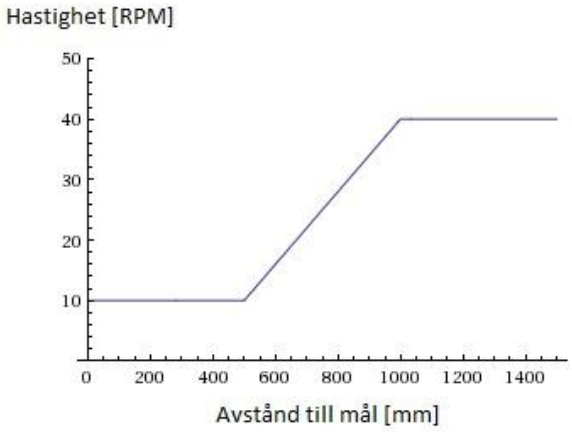
Från flödesschemat i Bilaga 1: CompactRIO - Huvudrutin utvecklades ett huvudprogram för AGV:n där det finns möjlighet att på kommando från PLC leverera material eller leta burkar. I programmet som sköter leveransen av material till produktionscellen används ett sub-program som navigerar till givna positioner och på så sätt kan en förutbestämd rutt följas in i produktionscellen. Navigeringsprogrammet läser via SICK NAV200 av AGV:ns position och beräknar, genom

trigonometriska samband, vinkeln på den linje som bildas mellan nuvarande position och målpositionen. Därefter sätts en hastighet på motorkontrollerna för att rotera AGV:n. Vinkeln uppdateras sedan kontinuerligt och när målet närmar sig roterar AGV:n långsammare för att sedan stanna när bestämd tolerans är uppnådd, vilket visas i Figur 28. AGV:n står då i rätt vinkel till målpositionen och börjar köra framåt. Position uppdateras kontinuerligt från SICK NAV200 och utifrån det regleras hastigheten på motorkontrollerna beroende på avståndet till målet, vilket visas i Figur 29. Om AGV:n under sin färd från start till mål avviker från kursen kommer motorerna beroende på vinkelavvikelsen rotera i olika hastigheter för att reglera kursen. Detta följer också kurvan i Figur 28. När AGV:n är inom en bestämd radie från målposition stannar AGV:n och går vidare till nästa steg i programmet.

Burksamlingsrutinen sköter styrning av AGV:n samt upplockningen av burkarna genom att kommunicera med bildbehandlingsdatorn. Efter att en burk har hittats kommer först AGV:ns vinkel att ställas in, sedan kommer AGV:n att köra fram mot burken och markera att en burk har hittats. En upplockningsanordning har inte tillverkats på grund av tidsbrist.



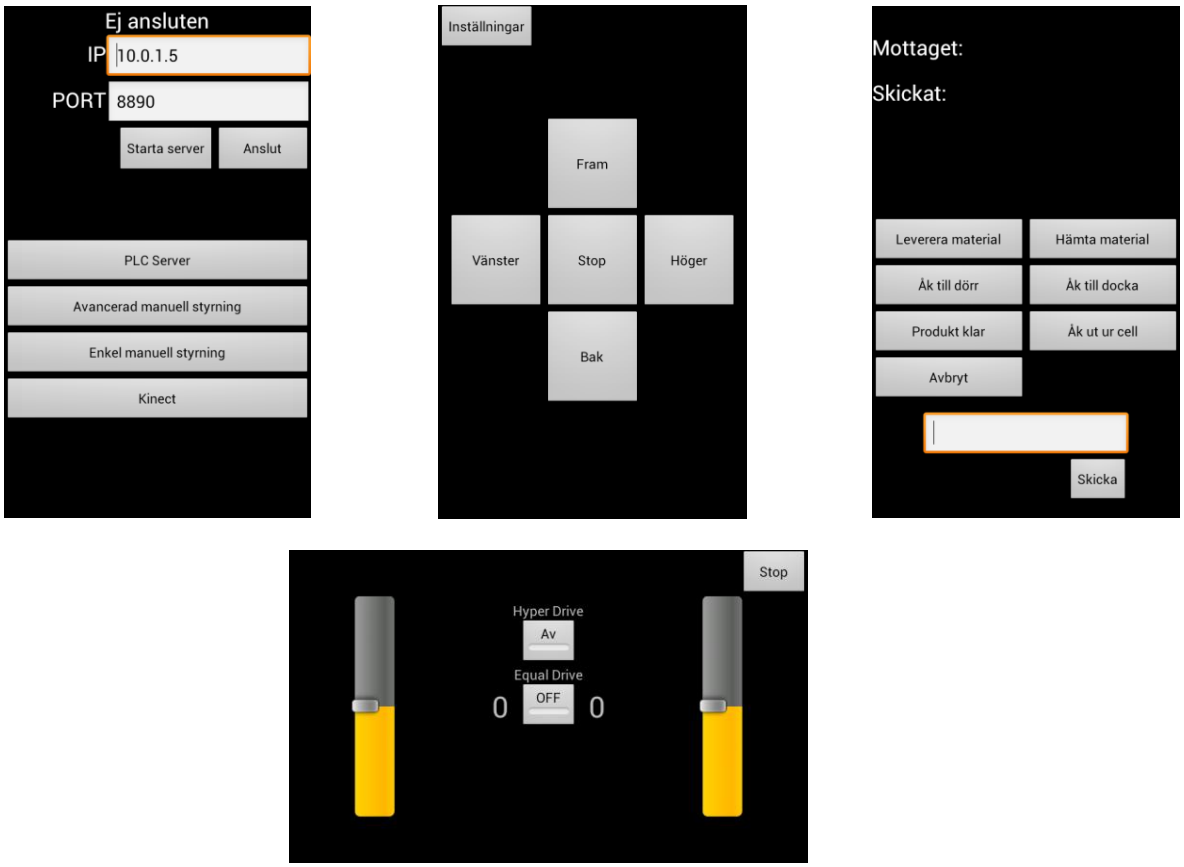
Figur 28: Reglering av vinkel



Figur 29: Reglering av avstånd

5.1.2 Androidapplikation

Androidapplikationen som utvecklades består av ett enkelt användargränssnitt där man kan antingen ansluta till en server genom att ange IP- och portnummer eller starta en server på en specifik port. Efter att en anslutning har etablerats kan man navigera sig fram till antingen ”Enkel styrning”, ”Avancerad styrning” eller ”PLC-server”. Även ”Kinect” finns att välja, där tanken var att man skulle kunna se bilden från Kinecten men tidsbrist gjorde att det inte blev klart. I nuläget måste olika program köras på CompactRIO:n för respektive alternativ av styrning. TCP/IP-kommunikationen fungerar genom att en textsträng följt av ett nyradstecken skickas och AGV:n som utför olika handlingar beroende på vilken sträng som skickas. Applikationen är i nuläget känsligt för tappade anslutningar, då den hänger sig och en omstart av applikationen krävs. Användargränssnittet för applikationen syns i Figur 30.



Figur 30: Androidapplikationens olika gränssnitt

5.2 Kommunikation

I detta avsnitt behandlas resultaten av metoderna för kommunikation samt de lösningsförslag som fanns inom problemområdet. I intern kommunikation beskrivs olika komponenter och hur dessa kommunicerar med CompactRIO:n. I extern kommunikationen beskrivs hur AGV:n kommunicerar med resten av produktionscellen

5.2.1 Intern kommunikation

Lösningsförslag tre, Figur 15 från kapitel 5, var det som valdes tillsammans med att köpa in seriemodul för RS232 kommunikation med SICK NAV200. I Figur 15 illusteras AGV:ns komponenter och vilken typ av kommunikation som finns mellan dem.

5.2.1.1 Motorkontrollerna

Motorkontrollerna testades med hjälp av ett program från tillverkaren, kallat bdc-comm-92.exe. Motorkontrollerna kopplas in via en seriellport på datorn som kör programmet. I detta program kan motorkontrollerna tilldelas ett ID-nummer för senare användning samt så kan det i programmet sättas en valfri hastighet för att se om motorkontrollerna reagerar som väntat. Motorkontrollerna kopplades in i CompactRIO:n där de testades mot olika program som fanns tillgängligt från föregående år. I dessa program fanns färdig kod för kommunikation till motorkontrollerna via deras seriellport.

Motorkontrollerna styrs från CompactRIO:n. CompactRIO:n beräknar, utifrån data från SICK NAV200, hur snabbt AGV:n ska köra med varje motor. Denna information skickas sedan till motorkontrollerna

genom en CAN-buss. I meddelandet står det vilken hastighet de olika motorerna ska köra i. Motorkontrollerna får sedan återkoppling från växellådan, vilket gör att den kan reglera varvtalet med den inbyggda PID-regleringen.

5.2.1.2 SICK NAV200

SICK NAV200 testades med programmet RealTerm vilket används för att lyssna och skriva på en dators serieport. Genom att läsa i tillverkarens datablad hittades inställningar för hur kommunikationen skulle fungera (SICK, 2006). I dessa datablad finns även en lista över de kommandon som används för att prata med SICK NAV200 (SICK, 2009). Från tillverkaren, SICK, finns även ett program kallat ”NAV200 Setup” som används för att läsa in positionen av de olika reflektorerna. Programmet kan då användas för att kolla var i koordinatsystemet SICK NAV200 befinner sig.

SICK NAV200 fungerar som AGV:ns radar. Då reflektorerna placerats ut på önskade platser skapas ett koordinatsystem som sparas i minnet på SICK NAV200 genom den ovan nämnda programvaran. Koordinatsystemet fungerar sedan som en karta för SICK NAV200. Ur denna karta beräknar SICK NAV200 på begäran fram sin position i form av x- och y-värden samt en vinkel. Dessa skickas sedan till CompactRIO:n.

5.2.2 Extern kommunikation

Kommunikationen mellan AGV:n och produktionscellen kommer förhoppningsvis att ske via en OPC-server som det var tänkt från början. Detta är i skrivande stund fortfarande oklart då det nödvändiga förarbetet i PPU-labbet fortfarande inte är färdigt på grund av förseningar från olika företag. Servern ska kontinuerligt styra vad AGV:erna gör utifrån vad produktionscellen behöver. När produktionscellen behöver leverans av material meddelas servern om detta som i sin tur skickar ett meddelande till den AGV som ska utföra leveransen. Vid in- och utpassering samt positionering i cellen sker alla klareringar via serven. Serven kontrollerar även att en AGV inte kör in i cellen förrän den andra är ute. När en AGV väntar på att få leverera material meddelar serven denna att utföra sin sekundäruppgift, att leta efter burkar.

I och med att OPC – servern inte har levererats så har ett Ladderprogram skrivits i PLC:n. Programmet använder sig av Siemens egna funktionsblock för att upprätta en TCP/IP kommunikation. Programmet skrevs för att endast kunna hantera en AGV för att inte onödiga fel skulle uppstå eftersom programmet inte kunde testas när det implementerades då PLC:n saknade trådlös kommunikation. Den trådlösa kommunikationen kunde först testas då PLC:n utrustades med en Apple Network extreme router och resultatet blev att programmet inte fungerade.

För att kunna testa kommunikationen mellan AGV och servern skapades en androidapplikation som simulerar en server. Utefter detta kunde huvudprogram för CompactRIO:n fortsätta att utvecklas även då den slutgiltiga lösningen för hur serven kommer fungera inte är färdig. Vid skapande av program för servern följdes det initialt skapade flödesschemat för AGV-huvuddator (se Bilaga 4: Flödesschema för AGV-huvuddator). Flödesschemat bygger på att AGV:n och PLC-servern har ett handskakningssystem för att båda parter ska ha koll på vilken del i rutinen man är.

5.3 Konstruktion

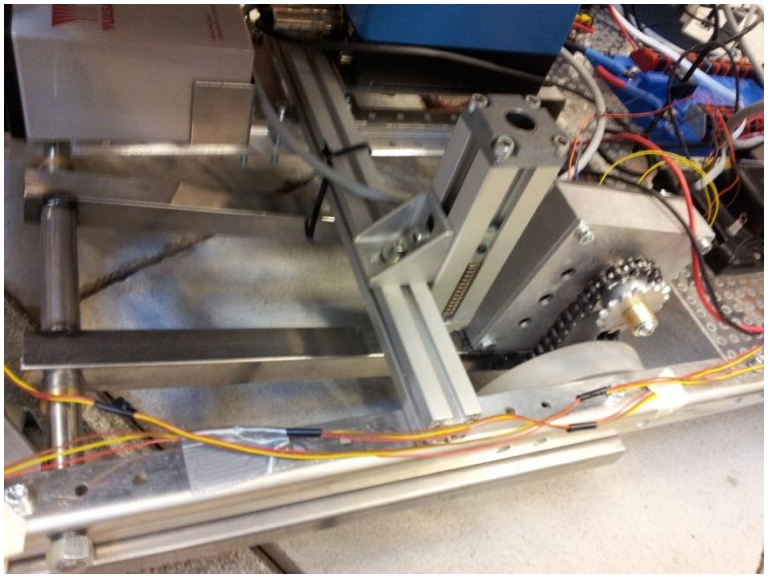
Enligt de tidigare skapade lösningsförslagen arbetades inom respektive område en slutgiltig lösning fram som tillverkades eller köptes in. I kapitlet som följer beskrives dessa resultat på en översiktlig nivå gällande alla intressanta delar.

5.3.1 Hjulplacering och drivning

Lösningsförslag A, vilket var det som valdes, har sex stödpunkter, i form av hjul, med de drivande hjulen i mitten. De ojämnheter som finns i golvet leder till en risk för att drivhjulen tappar kontakt med underlaget med detta antal stödpunkter. För att undvika detta konstruerades en fjädrad hjulupphängning som låter motor och hjul röra sig nära lodrätt i en vid cirkulär bana i förhållande till de övriga delarna på AGV:n. Upphängningen består utav en rektangulär ram med motorn monterad på den ena sidan (se Figur 31). På samma sida under motorn är hjulet monterat på en svarvad axel som låses med en låsmutter och en kedja ansluter de båda.

Ett lager till hjulupphängningen sitter på den motsatta sidan till motorerna på motorramen, detta låter ramen röra sig runt en axel. Det är denna rörelse som bidrar till kompensationen för ojämnheter. Lagret består utav ett stålrör med en bussning i vardera ända. Bussningarna är svarvade i mässing och har bearbetats invändigt genom brotschning för att få en fin ytjämnhet och en god tolerans mot stålaxeln som löper genom bussningen. Axeln är i sin tur bearbetat till en ytterdiameter nära bussningens innerdiameter och dess ändar är försedda med gängor för att kunna låsas fast i basramen. För att hålla motorramarna på plats mellan de två längsgående aluminiumprofilerna som utgör AGV:ns basram har fyra distanser i rostfritt stål svarvats i sådana längder att hjulen tangerar insidan av aluminiumprofilerna längs AGV:ns ytterkant.

Själva fjädringen kommer ifrån spiralfjädrar monterade i spåren på två Bosch Rexroth aluminiumbalkar. Dessa har i sin tur skruvats fast på en tvärgående Rexrothbalk som hjälper till att hålla ramen styv. Detta ser till att kompensera för de eventuella ojämnheterna och hjälper samtidigt, utöver motorramens egentyngd, att trycka drivhjulen nedåt för att uppnå erforderligt grepp mot underlaget.



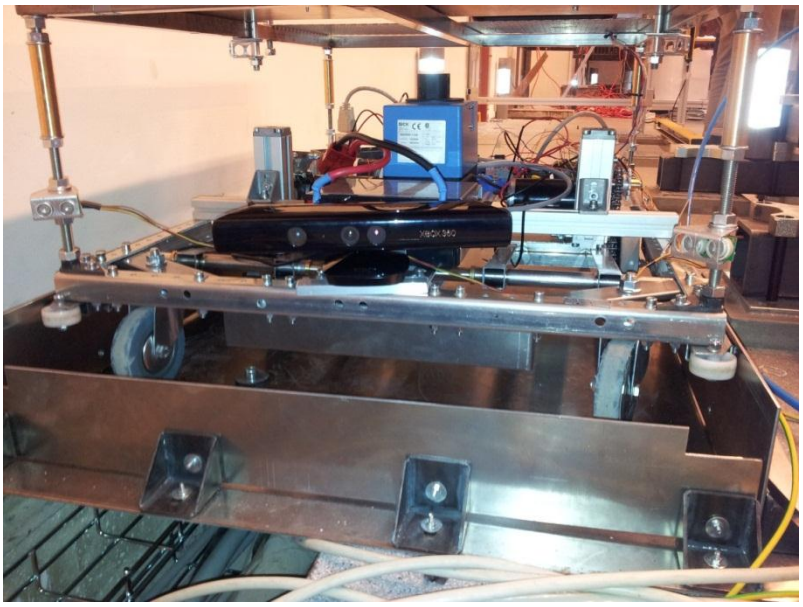
Figur 31: Hjulupphängning

5.3.2 Dockningsstation

Utifrån ritningen till dockstationen från lösningsförslaget byggdes en dockningsstation bestående av två välvda sitstycken, ett bakstycke med urtag för glidskenor och en bottenplåt. Dessa fogades samman med ett antal vinkelprofiler som skruvades med försänkta M4 skruvar. Dockstationen i sig skruvades i golvet med hjälp av två skruvar i befintliga hål i betongplattorna över kabelkanalerna (se Figur 32)

Figur 32: Dockstation

För att AGV:n skall passa väl mot dockstationens sidor tillverkades två glidskenor som monterats under den främre halvan på de yttre längsgående aluminiumprofilerna i basramen (se Figur 33). I framkant av dessa monterades två plasthjul för att minska friktionen mot dockstationen och att hjälpa till och styra in AGV:n i dockstationen. I fronten inskjutet ca 5 cm är en plåtprofil monterad för att göra ett tydligt stopp då AGV:n nått tillräckligt långt in i dockstationen.

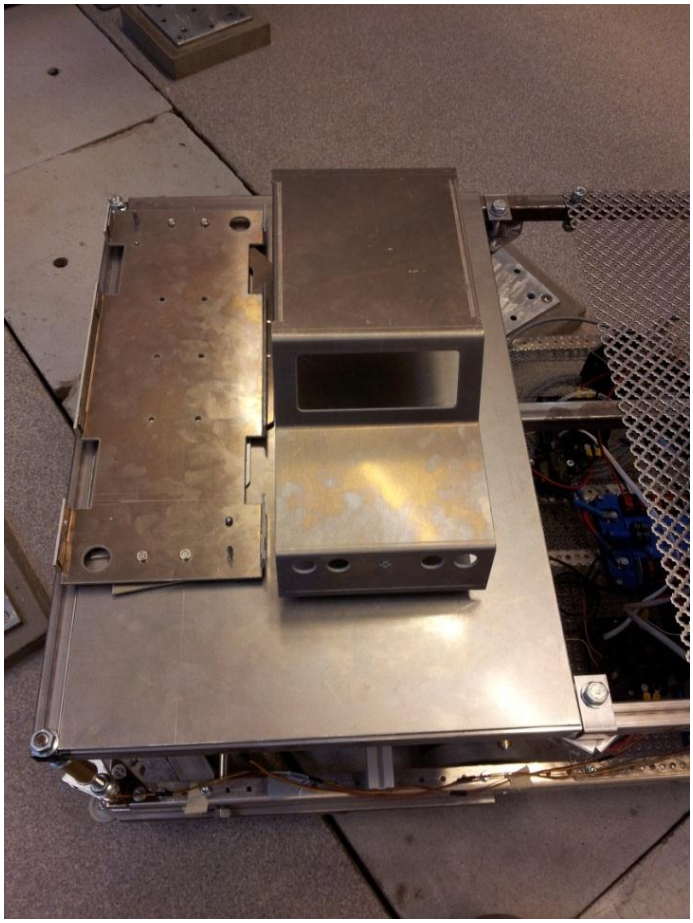


Figur 33: AGV i dockstation

5.3.3 Fixtur för bildelar

Fixturen för bildelar är konstruerad av en 3 mm tjock stålplåt i vilken ett antal hål är borrade. Vissa för att rymma styrstavar från komponenterna som skall transporteras och vissa för att fästa styrpinnar i. Genom denna lösning kan bilens sidor och bottenplatta staplas på varandra (se Figur 34). Vid sidan av dessa placeras bilens kaross.

Fixturplattan är placerad ovanpå AGV:ns överram och täcker cirka 50 % av dess längd. För att uppnå att SICK NAV200 fortfarande placeras kring AGV:ns rotationscentrum har denna flyttats ned från överramen till basramen. Detta leder även till att den är skyddad vid plockning av detaljer från fixturen.



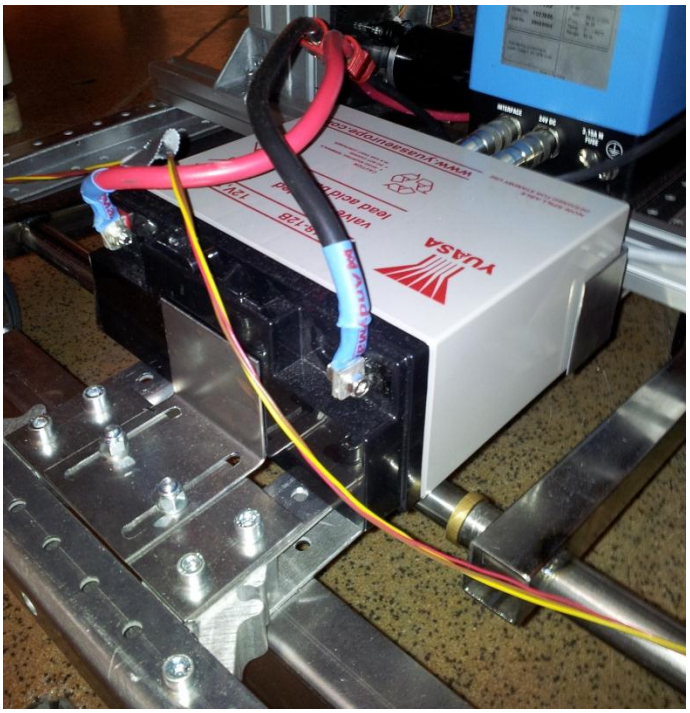
Figur 34: Fixtur

5.3.4 Diverse fästanordningar

För att hålla SICK NAV200 på plats har ett fäste av stål bockats i vilken fyra spår är frästa för att kunna justera SICK NAV200 till AGV:ns rotationspunkt. Fästet i sin tur skruvas med fyra M6 skruvar i AGV:ns basram.

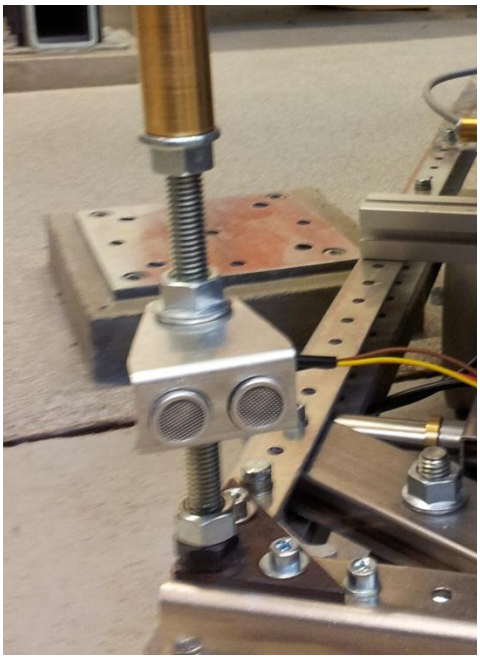
Figur 35: Fäste för SICK NAV200

Ytterligare ett fäste är gjort för att hålla batteriet på plats. Denna är placerad i mitten av AGV:n på motsatt sida om den perforerade transparenta plastskiva på vilken huvuddelen av all elektronik är fastsatt. Även denna är konstruerad av bockad stålplåt.



Figur 36: Fäste för batteri

Fästen gjordes för att kunna sätta fast ultraljudssensorerna på AGV:n. De tillverkades så att de kunde monteras på de befintliga gängstängerna som håller överramen, detta innebär att de både kan höjas och sänkas längs hela hängstången, men också vridas runt gängstången. Detta ger i sin tur att sensorernas placering kan justeras efter behov. Plastdistanser gjordes även så att sensorerna skulle ligga jämt på fästena, minimera risken för kortslutning samt så att de inte skulle sticka ut för mycket då de kan gå sönder vid en eventuell kollison.



Figur 37: Fäste för ultraljudssensor

Knapp- och lampsatsen är en bockad aluminiumplåt i vilken tre hål är borrade för att passa en grön lampa, en röd lampa, och en knapp. Den är i sin tur fäst i AGV:ns överram där den blir mest tillgänglig att nå samt lätt att se.

Figur 38: Knappsatts

5.4 Detektering

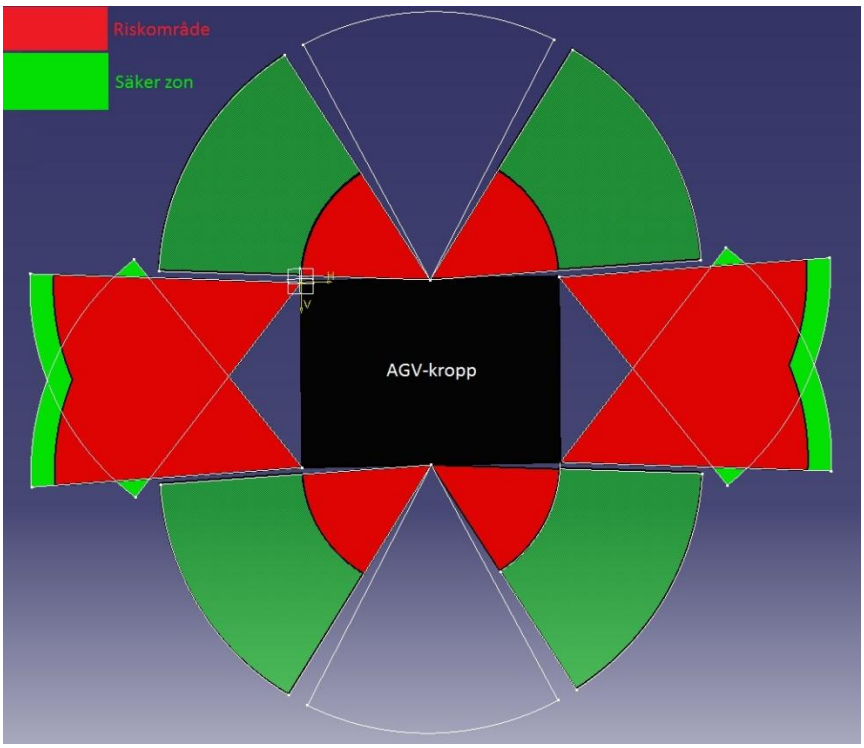
Här redogörs för de olika resultat som gjorts inom de två områdena som behandlades i kapitel 4.4, sensorval för Collision Avoidence samt programmering av Kinectsensorn.

5.4.1 Collision Avoidance

Ultraljudssensorerna ser på mindre än en meters avstånd med en vinkel större än 55 grader (Robotstorehk, 2012). Detta är en stor fördel då man kan täcka av stora områden med få sensorer. Då ultraljudssensorerna bara returnerar avståndet till det närmaste objektet och inte var i området objektet befinner sig (Robotstorehk, 2012), kan det stora synfältet också verka negativt. Detta innebär i sin tur att placeringen av ultraljudssensorerna är av största vikt, då en dålig placering skulle innebära att man inte får reda på något.

I den lösningen som utvecklades, se Figur 39, fokuserades det mycket på att få information som verkligen kan användas. De främre och bakre sensorerna är vinklade lite inåt, både för att utöka synområdet närmare AGV:n men också för att undvika att sensorerna ser något på sidan om AGV:n. Detta innebär att om de sensorerna returnerar att ett objekt är nära, då vet man att det objektet är rakt framför/bakom AGV:n.

Sensorerna vid sidan om AGV:n är till för att se om AGV:n kan svänga utan att krocka, det röda området visar var det inte får finnas något objekt för att svängning skall vara möjlig, medan i det gröna området skall svängning vara tillåten. Hur utbrett det röda området skall vara är något som kan ställas in i programkoden, detta gäller både för sido-, fram- och baksensorerna.
De sista sensorerna som syns i figuren, de ofärgade, har ett annat användningsområde än de tidigare nämnda. De är till för att mäta avståndet till ett föremål i sidled, detta för att i framtiden kunna runda statiska objekt som står i AGV:ns väg.



Figur 39: Ultraljudsplacering

5.4.2 Burkdetektering (Kinect)

Programmet för bildbehandling har två trådar. En tråd har hand om analyseringen av bilderna från Kinecten och den andra tråden agerar som server för att kunna skicka och ta emot data till och från CompactRIO via TCP/IP. Detta gör att serverfunktionen och bildanalysen exekveras parallellt, vilket är nödvändigt då programmet hela tiden måste lyssna om CompactRIO vill ha information.

Analys av bilder:
Kinecten genererar två olika typer av data; avstånd och färg. Rådata från de två kamerorna hämtas varje gång det finns en bild från dessa som är synkroniserade. I färgbilden analyseras varje enskild pixel som sparas i ett endimensionellt fält där varje pixel är fyra element; blå-, grön-, rödkomponent samt transparenskomponenten, även kallad alpha. Färgblandningen är additiv det vill säga maximalt värde av RGB (255,255,255) ger vit färg.

Programmet börjar med att söka igenom varje pixel i fältet (bilden) därefter kontrolleras om den aktuella pixeln innehåller rätt färgkombination. Den röda komponenten måste vara minst 100 och den blåa- och gröna komponenten måste vara max en tredjedel så stort som den röda. För att få dessa värden gjordes ett testprogram som sökte av ett område på colaburken och sedan togs medelvärdet av de olika färgkomponenterna. Därefter gjordes mycket testning och prövning.

Om pixeln uppfyller villkoret analyseras färgen hos intilliggande pixlar i ett område i form av en kvadrat. Storleken av kvadraten avgörs av hur många bilder som analyserats utan att programmet har hittat någon colaburk. Detta har implementerats så att programmet kan hitta en burk på längre avstånd. Om mer än hälften av pixlarna i den undersökta kvadraten uppfyller kraven för rött antas det vara en colaburk. När burken hittats ska vinkel och avstånd bestämmas men på grund av tidsbrist och prioriteringar av primärfunktionen hos AGV:n så har inte någon fungerande funktion för detta implementerats.

Figur 40: Burk hittad (*Den svarta kvadraten indikerar var programmet har hittat colaburken)*

Servertråd:
I servertråden upprättas en uppkoppling till en redan förbestämd port och IP nummer vid programmets start. Tråden ligger sedan och lyssnar om en efterfrågning av information har skett. Om klienten (CompactRIO) har gjort en förfrågan skickas ett meddelande om en burk har hittats och i så fall avståndet och vinkeln till den. Dock har inte detta testats med CompactRIO:n utan ett klientprogram har skrivits i C# till ASUS EeePC, vilket har fungerat tillfredsställande.

6. Diskussion och slutsats

I detta kapitel diskuteras de olika problemkategorierna för att se om metoder och lösningar gav det resultat gruppen önskade. Till sist diskuteras arbetet som helhet för att se om syftet med projektet uppfylldes.

För kategorin transport var det huvudsakliga problemet att kunna transportera material mellan olika punkter i produktionscellen. Detta löstes med hjälp av att programmera realtidskontrollern CompactRIO. Det första val gruppen tog inom detta var att bestämma programmeringsspråk. Språket som valdes var LabVIEW eftersom detta språk har använts under tidigare år och färdig kod fanns tillgänglig. Eftersom gruppen inte hade någon tidigare erfarenhet i detta språk lades det mycket mer tid än väntat på att lära sig språket, förstå tidigare års program och programmera de nya program som behövdes. På grund av att det inte fanns någon tydlig beskrivning på hur de befintliga programmen fungerade gjordes det kanske fler nya program än vad som behövts. Då gruppen väl förstått hur programmeringsspråket fungerar bestod problemen snarare av en rad mindre fel i programmen. Dessa fel löstes genom att kontinuerligt testa programmen mot AGV:n. Denna metod var nödvändig eftersom det bland annat är svårt att veta exakt hur olika delar av AGV:n reagerar på olika kommandon utan att ständigt testa och uppdatera programmen.

Den slutliga versionen av programmen löser det huvudsakliga problemet för transporten. Det som inte har uppfyllts är att AGV:n ska kunna transportera bort upplockade burkar. Detta tillsammans med konstruktionen av robotarmen prioriterades bort på grund av tidsbrist då detta tillhörde sekundäruppgiften. Något som skulle kunna förbättras är tiden det tar för AGV:n att utföra transportenav material. Om programmeringsprocessen hade gått snabbare så hade AGV:ns hastighet kunnat optimeras för att både vara pricksäker och snabb.

Att lösa den interna kommunikationen var något som initialt lades stor vikt vid på grund av problemen med tidigare års seriehubb. Den valda lösningen med inköp av en seriemodul till CompactRIO för kommunikation med lasern blev en bra lösning på grund av dess robusthet. Det negativa var att det tog längre tid än planerat att implementera ny kod på grund av svårigheter med programvaran. Gruppen lärde sig dock mycket under detta skede vilket var till stor nytta med programmeringen av materialtransportprogrammet.

Om OPC-serven hade levererats i tid hade AGV:n med största sannolikhet kunnat kommunicera med PLC:n. Mycket av tiden som lades till PLC:n gick åt att få det trådlösa nätverket att fungera. Det var även svårt att veta hur man skulle programmera PLC:n då gruppen hade få kunskaper inom ladder-programmering och inte möjlighet att testa koden mot AGV:n.

Snabbt efter att gruppen fått tillgång till fjolårets AGV: er konstaterades det att en ombyggnation skulle krävas för att uppnå den prestandan och robustheten i systemet som efterfrågats av samordnarna av kandidatarbetena i PPU. Särskilt stort fokus sattes då ursprungligen på att klara att åka över de springor och ojämnheter som finns i golvet i labbet. Ganska snabbt konstaterades det dock att det inte var ett problem utan att det var de glappet i drivningen som ledde till stora dödtider samtidigt som fyrhjulsdriftsystemet krävde att hjulen slirade då man vill vrida sig kring sin egen axel. Detta ledde till att hög prioritet lades på punkten i kravspecifikationen om just en väldefinierad rotationspunkt. Detta har också visat sig vid provkörning var mycket fördelaktigt och underlättar mycket vid utformning av program och rutiner.

En viss oro fanns för att den nya designen med bara tvåhjulsdrift skulle resultera i en försämrad förmåga att ta sig över ojämnheterna. Det har dock visat sig att detta inte var ett problem då den nya konstruktionen på ett bra sätt tar sig över betydligt högre hinder än vad den kommer att utsättas för.

Dockningsstationen som konstruerades hade ett noggrannhetskrav på 0.5 mm samt 0.5 grader. Hur vida detta är uppnått är svårt att säga, eftersom det är svårt att mäta utan någon speciell utrustning. Något som däremot är helt klart är att noggrannheten har ökat mot om dockningsstationen inte hade skapats. Som dockningsstationen är konstruerad finns det stora möjligheter att modifiera den i fall den inte skulle nå upp till kraven. Alternativ kan vara exempelvis att fästa en givare för att försäkra sig om att AGV:n är helt i position, byta ut sidorna till en tjockare och mindre flexande plåt alternativt göra högre stag till plåten, bygga någon form av arm som griper tag i AGV och positionerar den rätt, samt många andra saker, beroende på vad som kan behöva ändras.

Som tidigare nämnt har mycket fokus under projektet lagts på att skapa ett robust system, därför lades mycket tid på konstruktionen. En stor del gick till konstruktionen av drivningen men mycket tid gick också till de övriga mindre komponenter som tillverkades. Då alla komponenter är gjorda med robusthet i tankarna, är förhoppningen att ingen skall behöva konstruera om något som skapats till AGV:n så länge den inte får ett annat användningsområde.

Vid programmeringen av Kinecten gick det åt mycket tid till att förstå hur data från kamerorna skulle användas. Dokumentationen från tidigare kandidatarbete var svår att använda då denna ej var skriven för bildbehandlingsdatorn som används detta år. Om gruppen i ett tidigare skede hade gått över till att skriva programmet i Windows Presentation Foundation (WPF) hade mycket tid kunnat sparas. Den senaste versionen av bildbehandlingsprogrammet klarar av att hitta burkar inom ett visst interval, dock tros ibland andra röda föremål vara en burk. Det finns i skrivande stund ej heller någon tillhörande kod i LabVIEW för att styra AGV:n till den hittade burken. Då detta tillhörde sekunderuppgiften lades inte lika stor vikt vid att denna funktion skulle fungera.

Det slutliga allmänna resultatet av projektet uppfyller huvudsyftet att en AGV ska utvecklas för att leverera material till en produktionscell. Sekundäruppgiften uppfylldes delvis med att burkar kan detekteras men en upplockningsanordning gjordes aldrig. Den utifrån syftet skapade problembeskrivning var till stor hjälp för att se vad som skulle göras under respektive område. Vissa delar fick omprioriteras i och med tidsbristen då andra delar tog längre tid än beräknat. Detta berodde på att stor vikt lades vid att skapa en robust slutprodukt vilket krävde en större omkonstruktion och inköp av ny hårdvara som behövde programmeras. Detta har skapat goda möjligheter till ytterligare utveckling under kommande års kandidatarbeten.

7. Avslutning

Till kommande år är det möjligt att rikta ett större fokus på utveckling av mjukvaran till AGV:n på grund av bättre förutsättningar konstruktionsmässigt och hårdvarumässigt. Då gruppen detta år har utvecklat en Quickstart Guide för projektet (se bilaga 6) bör det även gå lättare att börja med programeringen på en gång. Rekommendationen till nästa år skulle till exempel kunna vara att utveckla Collision Avoidance med algoritmer för att ta sig runt hinder och skapa metoder för vilken rutt som AGV:n ska köra. Då detta år inte lyckats göra alla delar av sekundäruppgiften finns det vissa saker att utveckla även i detta område.

Källförteckning

Andersson, V. et. al. (2011), Programmering av AGV för skräpuppsamling och transport av däck, Göteborg: Chalmers Tekniska Högskola (Kandidatarbete inom institutionen för signaler och system)

AndyMark (2012(1)), Digital sidecar AM-0866 http://www.andymark.com/product-p/am-0866.htm (2012-05-15)

AndyMark (2012(2)), 2.5" CIM Motor, http://www.andymark.com/product-p/am-0255.htm (2012-02-29)

Asus (2012), Asus Eee PC 1011PX, http://www.asus.se/Eee/Eee_PC/Eee_PC_1011PX/#specifications (2012-05-08)

Atlet (2012), Forte URF, http://www.atlet.com/se/trucks/forte-urf-smalgangstruck-med-svaenggafflar-gaffeltruck#node-2909 (2012-05-11)

Cisco Systems (2008), USER GUIDE Dual-Band Wireless-N Gaming Adapter http://team358.org/files/programming/ControlSystem2009-/setup/Linksys-WGA600N-manual.pdf (2012-05-11)

Elettric 80 (2008), A Guide to Robot Logistics,http://www.elettric80.com/products-applications/AGuidetoRoboticLogistics.aspx (2012-02-02)

Google (2012), Android Developers, http://developer.android.com/index.html (2012-05-11)

HK systems (2006), Designing Material Flow For Maximum Agility, http://www.hksystems.com/resources/educational/ES_AGV_2006.pdf (2012-02-02)

James, M. (2012), Getting started with Microsoft Kinect SDK – Depth.I-programmer.info, http://www.i-programmer.info/programming/hardware/2714-getting-started-with-microsoft-kinect-sdk-depth.html?start=2 (2012-05-14).

Microsoft (2012(1)), Visual Studio, http://msdn.microsoft.com/en-us/vstudio/ (2012-05-11)

Microsoft(2012(2)), Kinect for Windows, http://www.microsoft.com/en-us/kinectforwindows/ (2012-05-11)

National Instruments (2011), NI cRIO-9074, http://sine.ni.com/ds/app/doc/p/id/ds-204/lang/en (2012-03-01)

National instruments (2012 (1)), 4-Port, RS232 Serial Interface Module for CompactRIO http://sine.ni.com/nips/cds/view/p/lang/en/nid/204259 (2012-05-08)

National instruments (2012 (2)), C Series 32-Ch, 5 V/TTL Bidirectional Digital I/O Module http://sine.ni.com/nips/cds/view/p/lang/en/nid/208811 (2012-05-08)

Nice, K (2011), How Tires Work, http://auto.howstuffworks.com/tire4.htm (2012-02-29)

Robotstorehk (2012), SRF05 - Ultra-Sonic Ranger Technical Specification, http://www.robotstorehk.com/sensors/doc/srf05tech.pdf (2012-05-08)

SICK (2006), Laser Positioning for Navigational Support http://www.sick-automation.ru/images/File/pdf/DIV05/manual%20NAV200.pdf (2012-03-01)

SICK (2009), Telegrams for Configuring and Operating the NAV200 Laser Positioning, System http://www.robotsinsearch.com/file/SICK/NAV200-1132/NAV200-1132-TelegramListing.pdf (2012-05-08)

Sriharsh Biswal (2011), Xbox 360 Kinect review, specification & price http://www.techibuzz.com/xbox-360-kinect-review/ (2012-05-14)

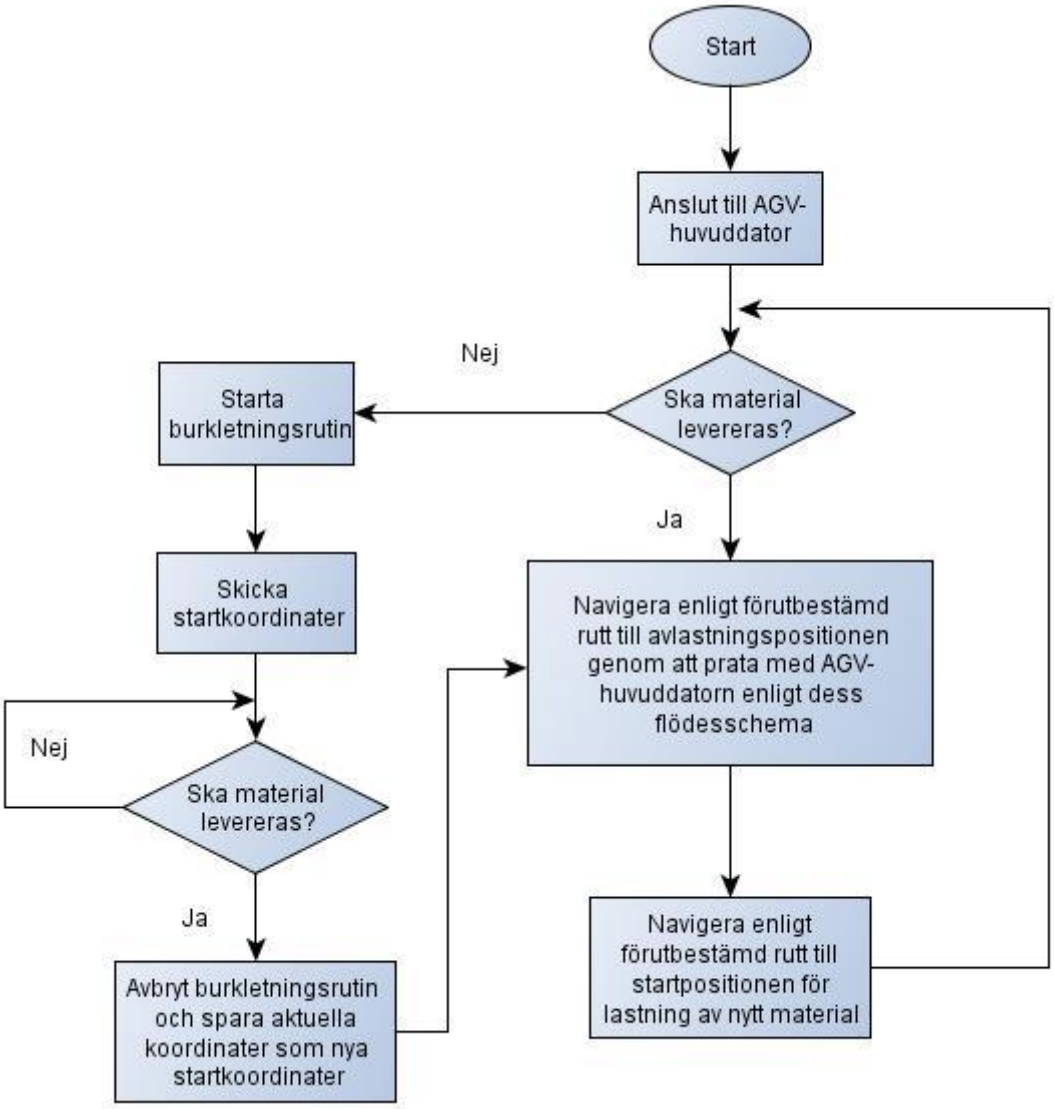
Texas Instruments (2011), Stellaris® Brushed DC Motor Control Module with CAN (MDL-BDC24), http://www.ti.com/lit/ml/spmt197a/spmt197a.pdf (2012-05-08)

The Eclipse Foundation (2012), Eclipse, http://www.eclipse.org/ (2012-05-11)

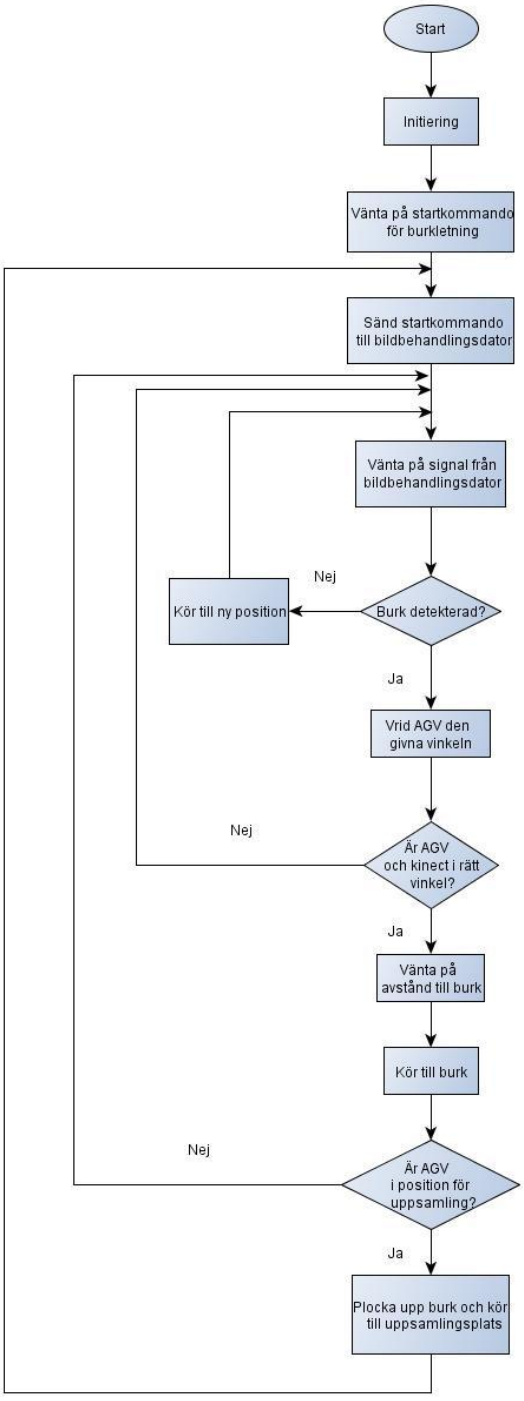
USFIRST (2011), Power Distribution Board Data Sheet 2011 FRC – Logo Motion, http://www.usfirst.org/uploadedFiles/Robotics_Programs/FRC/Game_and_Season__Info/2011_Assets/Kit_of_Parts/Power_Distribution_Board.pdf (2012-05-15)

Wikipedia (2012), CompactRIO, http://en.wikipedia.org/wiki/CompactRIO#Software (2012-05-11)

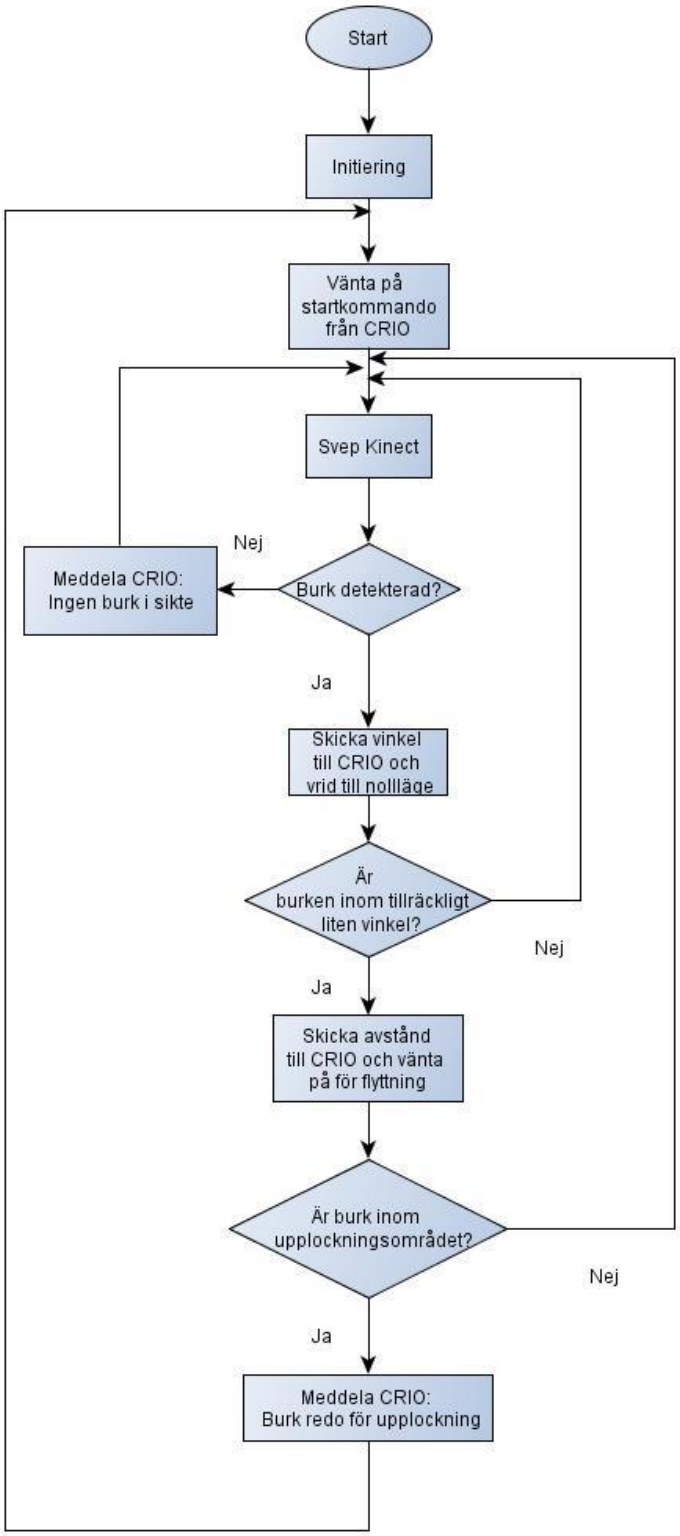
Bilaga 1: CompactRIO - Huvudrutin



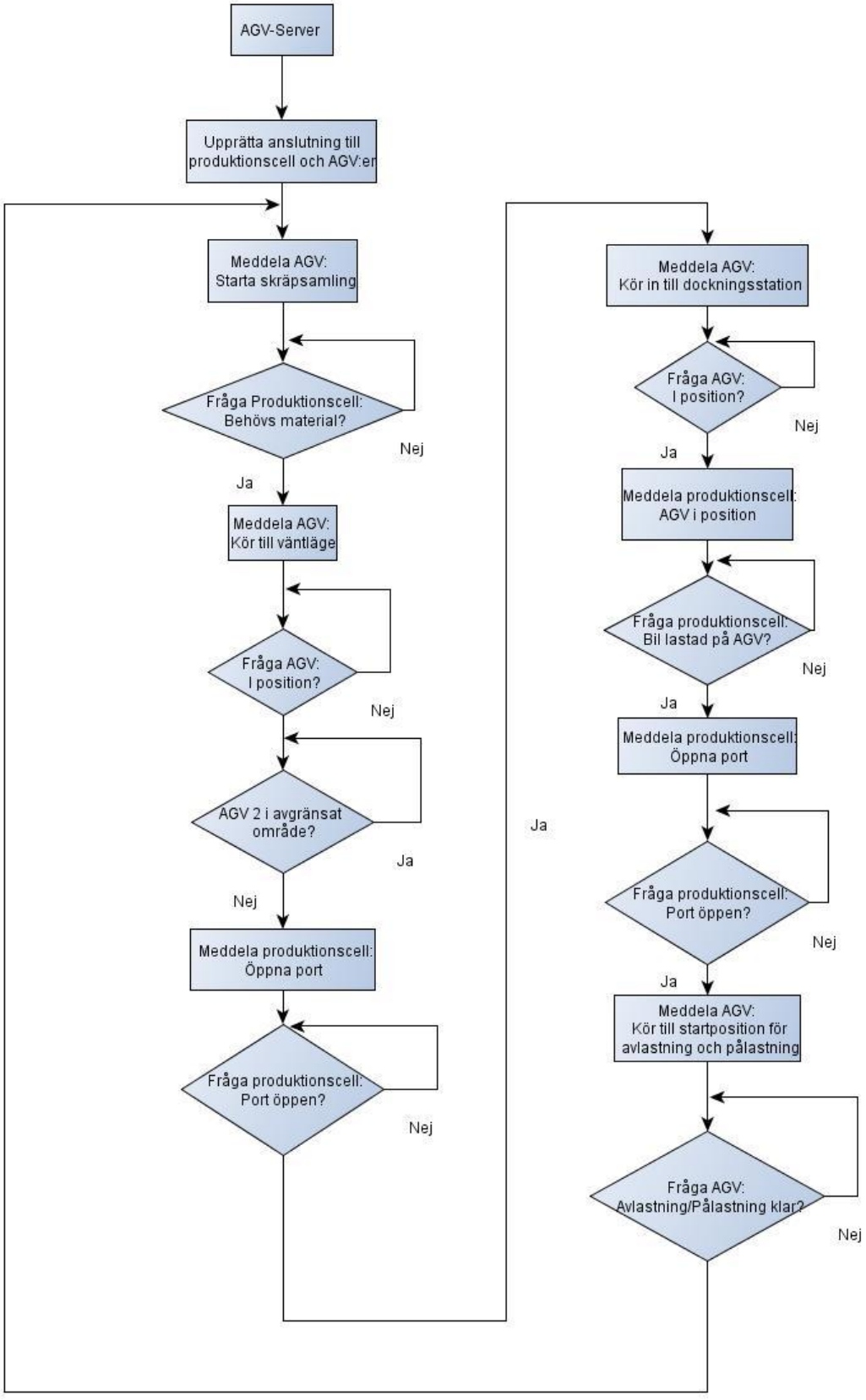
Bilaga 2: CompactRIO – Burksamlingsrutin



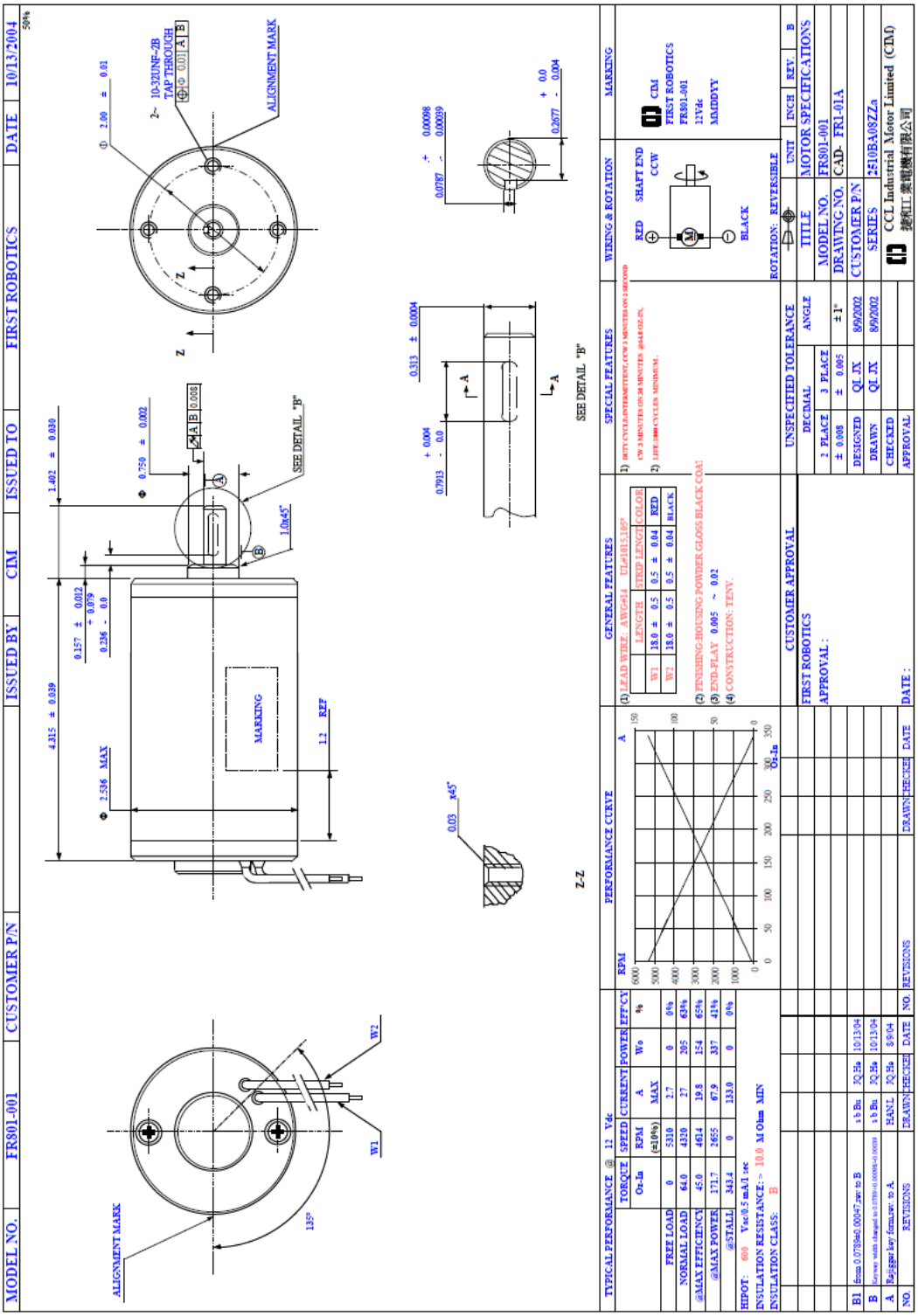
Bilaga 3: Bildbehandlingsrutin



Bilaga 4: Flödesschema för AGV-huvuddator



Bilaga 5: Motorspecifikationer



Bilaga 6: Quickstart guide

På grund av att vi själva hade en lång uppstartstid av vårt arbete beslutade vi att göra en snabbguide för att förenkla för nästkommande kandidatarbete som ska vidareutveckla AGV:n. Detta så att arbetet snabbt kan komma igång utan krångel med vilka programvaror och komponenter som använts etc. Denna bilaga avsedd specifikt till nästa års kandidatarbete som ska använda AGV:n med swivelhjul.

Programvaror som använts, vad dem används till och vart man hittar dem:

LabView 2009, används till programmering av CompactRIO:n. Anledningen till att vi valde LabView 2009 och inte 2010 var för att tilläggen **RT** och **FPGA** krävs vid programmering av RIO:n och dessa fanns inte tillgängliga för LabView 2010. LabView 2009 och tillägg hämtas på http://studentfile.portal.chalmers.se/ för chalmersstudenter. Utöver dessa tillägg behövs också NI-VISA (http://joule.ni.com/nidu/cds/view/p/id/2914/lang/en) och NI-RIO (http://joule.ni.com/nidu/cds/view/p/id/2815/lang/en).

Visual Studio 2010, används till programmering av bildbehandlingen(Kinecten). För att göra program för Kinecten måste **Kinect SDK** också laddas ner. Visual Studio 2010 hämtas på https://student.portal.chalmers.se/sv/kontaktochservice/it-passerkort-kopiering/egenadministrerad/programvaror/Sidor/default.aspx , klicka er vidare till **MSDNAA**. På http://www.microsoft.com/en-us/kinectforwindows/ hittar ni **Kinect SDK**, här finns också nyttiga videos och exempel som är nyttiga att kolla på för att komma igång med programmeringen.

Eclipse, användes till programmering av Android-applikationen. Detta var inget som egentligen ingick i projektet men är en bra grej att använda för att testa kommunikationen med och det är även bra att kunna köra AGV:n manuellt. För att programmera till Android krävs även **Android SDK** och en plugin till Eclipse. Eclipse hämtas på http://www.eclipse.org/downloads/ och det är **Eclipse IDE for Java Developers** som vi har använt. Android SDK hämtas på http://developer.android.com/index.html här finns också en guide för hur du installerar Android-plugin i Eclipse.

dc-comm-92.exe kan använda för att styra motorkontrollerna från en PC via serieport. Mycket användbart för att se att motorkontrollerna fungerar som de ska och ge ID till motorer.

NAV200Setup kan användas för att styra lasern ifrån en PC via serieport. Det är användbart för att ”mappa” upp reflektorer som bildar ett koordinatsystem och se att position och vinkel visar rätt värde.

En del mindre program har använts men dessa finns tillgängliga i dropboxen som ni kommer ha tillgång till. Där kommer även finnas readme filer med kort info om vad respektive program används till.

Hårdvara som använts och vart man hittar information om dem:

CompactRIO (NI cRIO-9074): http://sine.ni.com/psp/app/doc/p/id/psp-421/lang/en

Seriemodul (NI9870): http://sine.ni.com/nips/cds/view/p/lang/en/nid/204259

Digital I/O modul (NI9403): http://sine.ni.com/nips/cds/view/p/lang/en/nid/208811

Motorkontroller MDL-BDC24: http://www.ti.com/tool/mdl-bdc24

SICK NAV200:
https://www.mysick.com/eCat.aspx?go=FinderSearch&Cat=Row&At=Fa&Cult=English&FamilyID=344&Category=Produktfinder&Selections=34430,34243

Ultraljuds sensor SRF 05: http://www.robotstorehk.com/sensors/doc/srf05tech.pdf

Kinect: http://www.microsoft.com/en-us/kinectforwindows/

Linksys WGA600N: http://homesupport.cisco.com/en-us/wireless/lbc/WGA600N

Första testerna

Till att börja med bör ni kolla så att alla komponenter är på plats. När ni har gjort detta så kan ni börja med att testa komponenterna för sig för att få bättre koll på hur de funkar. Både motorkontrollerna och lasern kan kopplas in direkt till datorn via en serieport och testas med program från tillverkaren. Här gör man också alla inställningar så som att ställa in ID på motorkontrollerna och hitta nya reflektorer till SICK-naven etc. Dessa program finns tillgängliga på dropboxen.

Efter ni har testat komponenterna för sig kan ni gå vidare och testa lite mindre program från LabVIEW-koden. Dessa kan kompileras på CompactRIO:n trådlöst via ett wlan, något som vi insåg sent i projektet. För att göra detta måste man ställa in den trådlösa nätverksadaptern Liksys Wireless Gaming Adapter genom att koppla in den till en dator och ställa in vilket nätverk den ska ansluta till. När detta är inställt ska, om allt stämmer, CompactRIO:n dyka upp i ”Automation and Measurement” i LabView där man kan ansluta till den. Efter att den är ansluten i Automation and Measurement Explorer kan en omstart genom ”Automation and Measurement” behövas för att kunna connecta i LabVIEW. När man sedan kör ett program i LabVIEW kommer detta att kompileras på CompactRIO:n automatiskt.

Använda FPGA

För att använda seriemodulen NI9870 och digitala I/O modulen NI9403 behöver man i LabVIEW 2009 använda tillägget FPGA. I LabVIEWprojektet skapas FPGA-programmen under ”FPGA-target” där vi skapat olika program för att kommunicera med laser, ultraljud och tryckknapp. FPGA programmen behöver kompileras en gång innan de körs till skillnad från de vanliga ”Real-Time” programmen. Kompileringen kan ta upp till en timme för större program. När de är kompilerade kan de köras ifrån ”Real-Time”-programen för att läsa och skriva till hårdvaran.

Köra manuellt med Android applikationen

När man kör AGV:n manuellt krävs att ni kopplar upp AGV:n och mobilen till samma nätverk. Beroende på om ni ska köra Avancerad eller Enkel styrning finns två olika program i LabView. Båda LabView-programmen lyssnar på en specifik port, alltså AGV:n är server. Genom att skriva in samma port och AGV:ns IP nummer i applikationen så kopplar mobilen upp mot AGV:n. Sen är det bara att trycka på den styrning man skall använda och tuta och köra. Vid simulering av PLC-server är det tvärtom, alltså att AGV:n kopplar upp mot mobilen och då behöver endast port anges i applikationen.

Vid frågor kontakta oss gärna via mail:

Fredrik Jonazon, Kinect och Android	frejona@student.chalmers.se
Patrik Werner, Kinect och PLC	patrikw@student.chalmers.se
Jakob Fjellström, LabView	jakobfj@student.chalmers.se
Viktor Elisson, LabView	eviktor@student.chalmers.se
Erik Hultgren, Konstruktion	erikhu@student.chalmers.se
Fredrick Bergström, Konstruktion	freberg@student.chalmers.se

Bilaga 7: Omkonstruktion av AVG

Ett antal krav och önskemål som AGV:n skall respektive bör uppfylla så väl som möjligt listades.

1. **Prestanda**
   - 1.1. K. Ska kunna ta sig över springor i golvet
   - 1.2. K. Ska kunna lämna delar på specifik position upprepade gånger
   - 1.3. K. Ska kunna specifikt avgöra kring vilken punkt roboten svänger
   - 1.4. K. Ska kunna köra i gånghastighet
   - 1.5. K. Ska Kunna bära angiven last
   - 1.6. Ö. Bör Kunna docka i flera stationer
   - 1.7. Ö. Bör Liten svängradie
   - 1.8. Ö. Bör ha en bra position på svängpunkt
   - 1.9. Ö. Bör vara stabil
   - 1.10. Ö. Bör ha god manövrerbarhet
2. **Produktion**
   - 2.1. Ö. Bör återanvända så mycket som möjligt av befintligt material
   - 2.2. Ö. Bör vara så billig som möjligt att tillverka
   - 2.3. Ö. Bör ha en så låg mekanisk komplexitet som möjligt
3. **Dimensioner**
   - 3.1. K. Skall kunna komma genom dörr och säkerhetsutrustning
   - 3.2. K. Skall rymma lasten och nödvändig elektronik
   - 3.3. Ö. SICK nav200 bör blockeras så lite som möjligt

Önskemålen viktades mot varandra för att kunna placeras i ordning efter hur viktiga de kan anses vara. Som tabell 3 visar har detta gjorts genom att alla lösningsförslagen vägdes mot varandra och tilldelades poängen 0 eller 1 beroende på omönskemålet kunde ses som viktigare eller mindre viktigt än det som det jämfördes med. Därefter summerades resultaten i spalten längst till höger.

Tabell 3: Jämförande av önskemål

Önskemål	1.7	1.8	1.9	1.10	1.11	2.1	2.2	2.3	3.3	4.1	Resultat
1.7	-	0	0	0	0	0	0,5	0	0	0	0,5
1.8	1	-	0	0	0	1	1	1	0	1	5
1.9	1	1	-	1	1	1	1	1	0	1	8
1.10	1	1	0	-	0,5	1	1	1	0	1	6,5
1.11	1	1	0	0,5	-	1	1	1	0	1	6,5
2.1	1	0	0	0	0	-	1	1	0	0	3
2.2	0,5	0	0	0	0	0	-	0	0	0	0,5
2.3	1	0	0	0	0	0	1	-	0	0	2
3.3	1	1	1	1	1	1	1	1	-	1	9
4.1	1	0	0	0	0	1	1	1	0	-	4

Önskemålen placeras baserat på viktningen i en tjugo gradig skala. Därefter delas skalsiffran med summan av alla skalsiffror för att erhålla en normerad viktfaktor.

Skala	20	19	16	15	12	10	8	7	3	2
Önskemål	3.3	1.9	1.11	1.10	1.8	4.1	2.1	2.3	2.2	1.7
Viktfaktor	0,179	0,17	0,143	0,134	0,107	0,089	0,071	0,063	0,027	0,018

Problemet delas upp i delproblem och med brainstorming tas olika lösningar fram. De olika lösningarna sammanställs. Sedan används systematisk konstruktion med olika matriser för att ta fram den mest lämpliga lösningen.

Tabell 4: Sammanställning av dellösningsförslagen

Delfunktion					
Hjultyp (stödjande)	Swivelhjul	Omniwheels	Vanliga hjul	Larvfötter	
Hjultyp (drivande)	Omniwheels	Vanliga hjul	Larvfötter		
Hjulkonfiguration	Holonomic	Rektangulär (4 st)	Diamant	Rektangulär (6 st)	
Framdrivning	framhjulsdrift	Bakhjulsdrift	mitthjulsdrift	fyrhjulsdrift	
Styrning	Vridbara hjul	Bandvagnsprincip			
Transport av delar	På tak i fixtur	Rack på vagn	Separat flak med fixtur		
Bra precision vid materialavlämning	Dockstation	Magneter	SICK nav 200	Räls	Tejp

Dellösningsförslagen kombinerades till ett antal totallösningar. Beskrivna i tabell 5.

Tabell 5: Totallösningsförslag

A Hjultyp (stödjande): Swivelhjul Hjultyp (drivande): Vanliga hjul Hjulkonfiguration: Rektangulär (6) Framdrivning: Tvåhjulsdrift Styrning: Bandvagnsprincip	B Hjultyp (stödjande): Omniwheels Hjultyp (drivande): Vanliga hjul Hjulkonfiguration: Rektangulär (4) Framdrivning: Tvåhjulsdrift Styrning: Bandvagnsprincip
C Hjultyp (stödjande): Larvfötter Hjultyp (drivande): Larvfötter Hjulkonfiguration: Rektangulär (4) Framdrivning: Tvåhjulsdrift Styrning: Bandvagnsprincip	D Hjultyp (stödjande): Omniwheels Hjultyp (drivande): Omniwheels Hjulkonfiguration: Rektangulär (4) Framdrivning: Fyrhjulsdrift Styrning: Bandvagnsprincip

E		F	
Hjultyp (stödjande):	Swivelhjul	Hjultyp (stödjande):	Vanliga hjul
Hjultyp (drivande):	Vanliga hjul	Hjultyp (drivande):	Vanliga hjul
Hjulkonfiguration:	Rektangulär (4)	Hjulkonfiguration:	Rektangulär (4)
Framdrivning:	Tvåhjulsdrift	Framdrivning:	Fyrhjulsdrift
Styrning:	Bandvagnsprincip	Styrning:	Bandvagnsprincip
G		H	
Hjultyp (stödjande):	Vanliga hjul	Hjultyp (stödjande):	Omniwheels
Hjultyp (drivande):	Vanliga hjul	Hjultyp (drivande):	Omniwheels
Hjulkonfiguration:	Rektangulär (4)	Hjulkonfiguration:	Diamant
Framdrivning:	Tvåhjulsdrift	Framdrivning:	Tvåhjulsdrift
Styrning:	Vridbara Hjul	Styrning:	Bandvagnsprincip

För att sålla bort de förslag som inte uppfyllde kraven ställdes tabell 6 upp. Genom att de lösningsförslagen som inte uppfyllde något av önskemålen, A-H, ströks ur den vidare utvärderingen.

Tabell 6: Förslag som inte uppfyller kraven sållas bort

	1.1	1.2	1.3	1.4	1.5	1.6	3.1	3.2
A	x	x	x	x	x	x	x	x
B	x	x	x	x	x	x	x	x
C	x	x	-					
D	x	x	x	x	x	x	x	x
E	x	x	x	x	x	x	x	x
F	x	x	-					
G	x	x	-	x	x	x	x	x
H	x	x	x	x	x	x	x	x

Det konstaterades att alla lösningsförslagen kunde uppfylla kraven och önskemålen för att navigera och bära last likvärdigt oberoende de andra valen så de har frånsetts under den fortsatta utvärderingen för att återkomma till efter färdigställande av basramen.

För att få fram den bästa slutgiltiga lösningen poängsätts lösningarna A, B, D,E, G och H. Genom detta system kan den lösning som uppfyller flest önskemål på bästa sätt väljas ut

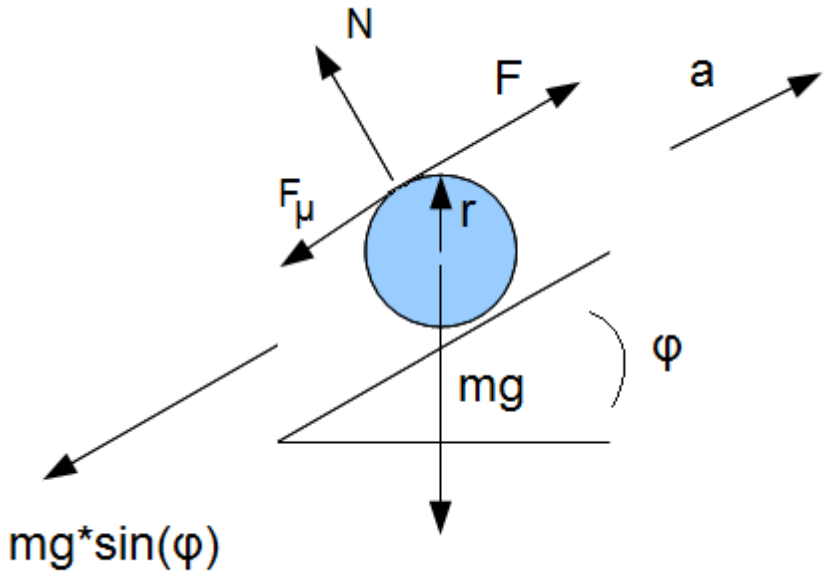
Genom att poängsätta på en skala mellan ett och fem hur väl de olika totallöningsförslagen uppfyllde önskemålen, multiplicera med viktfaktorn och summera för respektive totallösning nås en slutpoäng för varje förslag.

Tabell 7: Lösningarnas uppfyllande av önskemål

		A		B		D		E		H		I	
Önskemål:	Vikt	poäng	värde	poäng	värde	poäng	värde	poäng	värde	poäng	värde	poäng	värde
Kunna docka i flera stationer	0,018	5	0,09	5	0,09	4	0,072	5	0,09	4	0,072	5	0,09
Svängradie	0,107	5	0,535	5	0,535	5	0,535	5	0,535	5	0,535	5	0,535
Position på svängpunkt	0,117	5	0,585	1	0,117	5	0,585	5	0,585	5	0,585	1	0,117
Återanvända material	0,071	3	0,213	5	0,355	3	0,213	3	0,213	1	0,071	5	0,355
Tillverkningskostnad	0,027	2	0,054	4	0,108	1	0,027	2	0,054	1	0,027	4	0,108
SICK nav ska blockeras så lite som möjligt	0,179	3	0,537	3	0,537	3	0,537	3	0,537	3	0,537	3	0,537
(stabilitet)	0,134	5	0,67	5	0,67	4	0,536	3	0,402	4	0,536	5	0,67
(mekaniskkomplexitet)	0,063	2	0,126	4	0,252	2	0,126	2	0,126	1	0,063	5	0,315
(kodningskomplexitet)	0,089	5	0,445	3	0,267	4	0,356	5	0,445	1	0,089	3	0,267
Manövrerbarhet	0,143	4	0,572	2	0,286	5	0,715	4	0,572	5	0,715	2	0,286
Summa			3,827		3,217		3,702		3,559		3,23		3,28

Lösning A får mest poäng i uteslutningsmatrisen. Detta innebär att lösning A uppfyller kraven och flest önskemål på bästa sätt.

Bilaga 8: Beräkningar



Figur 41: Friläggning av AGV i lutning

I **Error! Reference source not found.** 41 visas vilka krafter som verkar på AGV:ns hjul då den rullar i en uppförsbacke. F_μ är det bromsade rullmotståndet och den framdrivande kraften F genereras av motorns moment som verkar på hjulet.

Det bromsande rullmotståndet approximeras med antagande att den förhåller sig till normalkraften på plant underlag enligt följande formel. Här uppskattas vikten för AGV:n till 25 kilo.

$$F_\mu = C_\mu * N = 0{,}01 * 9{,}82 * 25 \approx 2{,}45\ N \qquad (1)$$

Där Cμ antas vara 0,01 vilket skulle motsvara ett bildäck som färdas på ett underlag av betong.

Ojämnheter i labbets golv leder till ytterligare motstånd vilket approximeras med konstant körning/start i en uppförsbacke med vinkel φ.

För att robotens accelerationsfas inte skall bli för lång sätts ett önskemål om att max hastighet skall nås efter 0,5 meters körning. Dessutom önskas en maxhastighet motsvarande gångfart vilket skulle motsvara en människas prestanda. Med gånghastighet avses $1 m/s$

$$a = \frac{v^2 - v_0^2}{2 * d} = \frac{1 - 0}{2 * 0{,}5} = 1 m/s^2 \qquad (2)$$

Krav på att AGV:n skall klara att hålla en viss toppfart ger oss således en gräns för hur fort hjulet måste rotera.

$$\frac{n * 2\pi r}{60} = v \rightarrow n = \frac{v * 60}{2\pi r} = 128\, rpm \qquad (3)$$

Friläggningen ger följande ekvationer

$$F_{acc} - mg * \sin(\varphi) - F_{\mu} = ma \rightarrow F_{acc} = ma + mg * \sin(\varphi) \qquad (4)$$

$$F_{konst} - mg * \sin(\varphi) - F_{\mu} = 0 \rightarrow F_{konst} = mg * \sin(\varphi) + F_{\mu} \qquad (5)$$

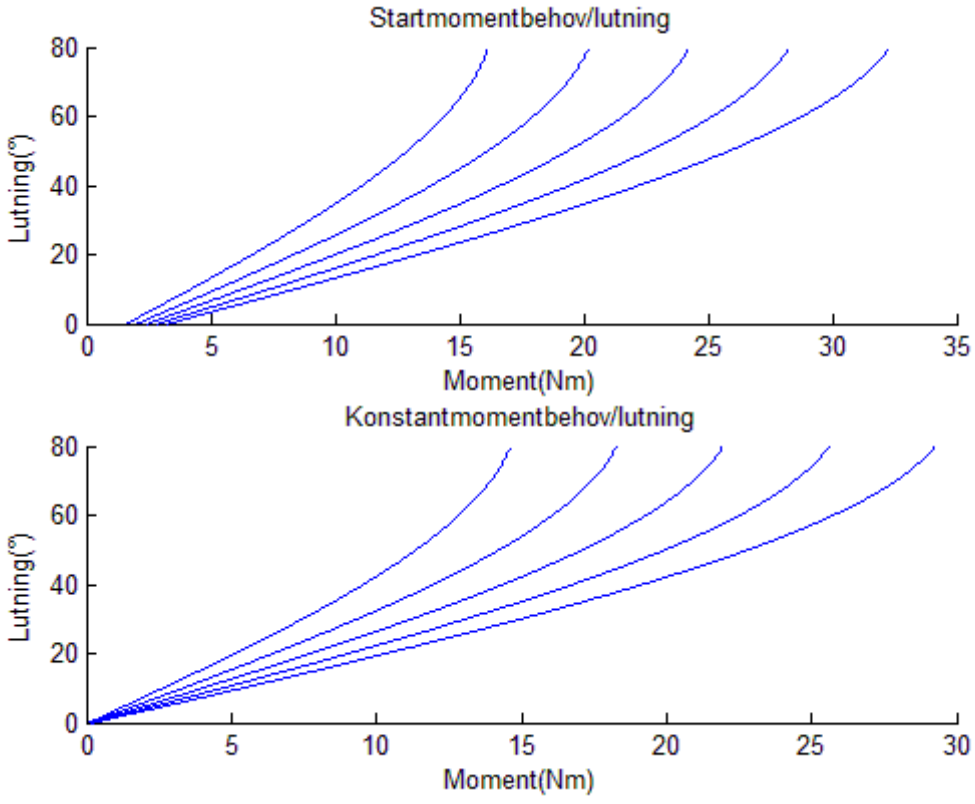
Vilket ger moment behoven på hjulen med massträghetsmomentet för hjulen inräknat.

$$\dot{\omega} = \frac{a}{r} \qquad (6)$$

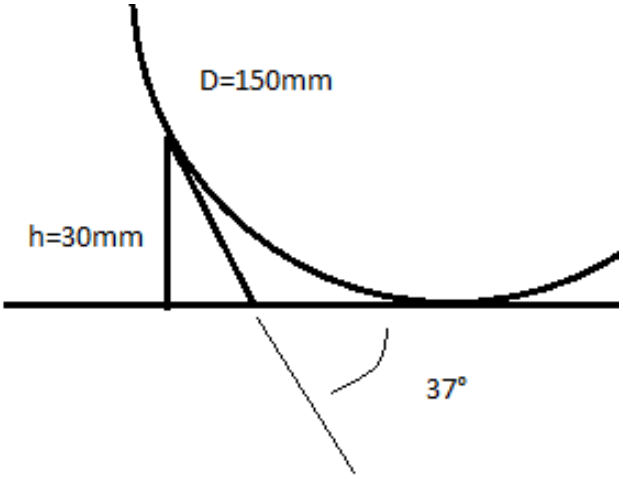
$$M_{acc} = F_{acc} * r + J\dot{\omega} = M + \frac{m}{2} * r^2 * \frac{a}{r} \qquad (7)$$

$$M_{konst} = F_{konst} * r \qquad (8)$$

Enligt ovanstående ekvationer räknas momentbehovet i förhållande till lutningsvinkeln ut. Detta för vikterna 15kg-35kg.



Figur 42 - Momentbehov 15 - 35 kg



Figur 43 - Hjul mot kant

Enligt figur 43 kan lutningen 37 grader beräknas som den ekvivalenta lutningen mot kanten på 30mm som AGV:n bör ta sig över.

Därmed kan accelerationsmomentet för AGV:n bestämmas om vikten antas vara 25 kilo. Detta enligt ekvation 1 samt 4-8.

$$M_{acc} = 13{,}15\ Nm$$
$$M_{konst} = 11{,}27\ Nm$$

Således måste det nominella momentet överstiga M_{konst} och startmomentet överstiga M_{acc} på hjulaxeln.

Då två motorer skall användas är det halva totala framräknade momentet som blir dimensionerade av kapaciteten för respektive motor.

Växellådan har utväxling i=14.81:1 och utväxlingen från kedjeöverföringen är 22/15=1,47 => i=1.47:1 vilket medför en total utväxling $i_{tot} = 21{,}77{:}\ 1$. Verkningsgraden antas till 85 %.

$$M_{motor.acc} = \frac{M_{tot}}{2 * i_{tot}} * \frac{1}{\eta} \qquad (9)$$
$$M_{motor.konst} = \frac{M_{tot}}{2 * i_{tot}} * \frac{1}{\eta} \qquad (10)$$

Vilket ger behoven $M_{motor.acc} = 0{,}36\ Nm$ och $M_{motor.konst} = 0{,}31\ Nm$.

Enligt tillverkaren har motorerna följande specifikationer (AndyMark, 2012(2)):

Physical Specs:

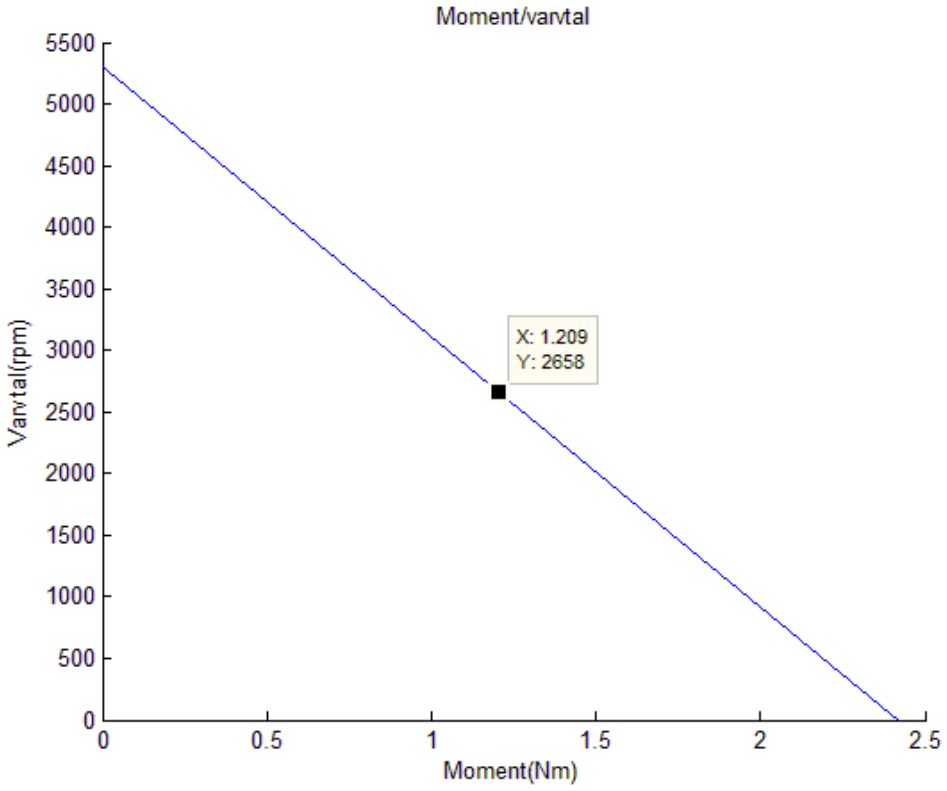
- Size: 2.5 inch diameter, 4.34 inch long body
- Output Shaft size: 0.313 +/- 0.0004, with 2mm keyway
- Weight: 2.82 pounds
- Mounting Holes: #10-32 tapped holes (2), on a 2" bolt circle

Performance Specs:

- Voltage: 12 volt DC
- No load RPM: 5,310 (+/- 10 %)
- Free Current: 2.7 amps
- Maximum Power: 337 Watts (at 2655 rpm, 172 oz-in, and 68 amps)
- Stall Torque: 2.42 N-m, or 343.4 oz-in
- Stall Current: 133 amps

För mer information, se datablad i bilaga 8.

Motorerna kan ge 2,42 Nm vid start och 1,253 Nm vid det efterfrågade varvtalet på hjulen enligt ekvation 2 vilket ger $n_{max} = 128 * 20{,}77 = 2658{,}6\, rpm$ då hänsyn tages till utväxlingen.



Figur 44 - Moment/varvtal

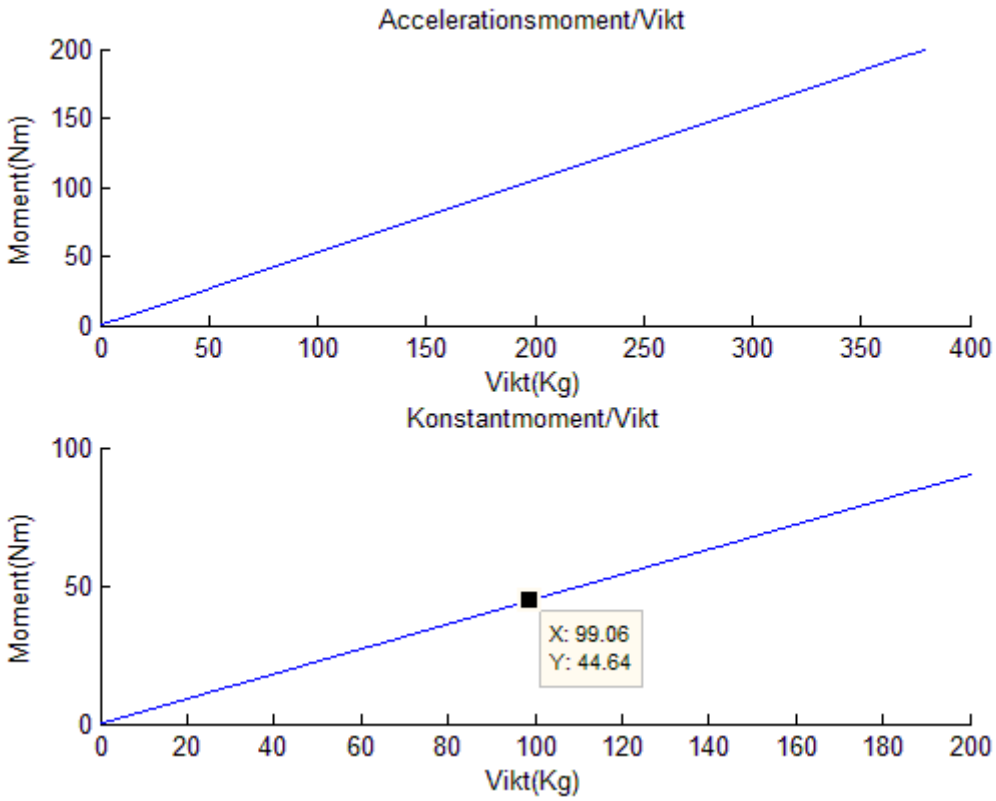
Detta innebär att en god kapacitet finns för att lasta roboten då motorn kan leverera ett moment som är ca fyra gånger så stort som behovet under konstantfart och nästan sju gånger mer under accelerationen.

Om momenten sätts enligt maxvärdena 2,42 Nm vid acceleration och 1,21 Nm och dessa räknas om till moment på motoraxeln enligt utväxling, verkningsgrad och antal motorer fås de maximala gränserna för vilka vikten kan läsas av ur nedanstående grafer.

$$M_{max.acc} = 178{,}3\ Nm$$

$$M_{max.konst} = 44{,}6\ Nm$$

Här är det lägre momentet för konstanthastigheten dimensionerande



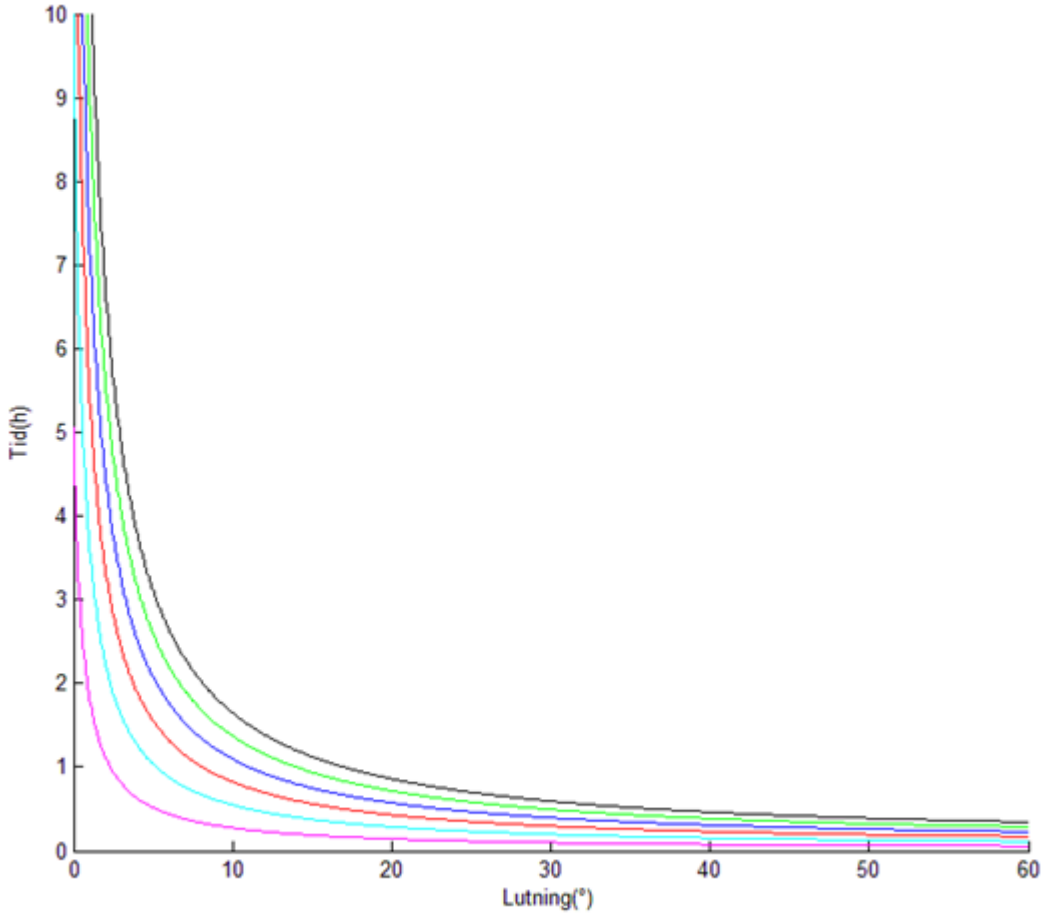
Figur 45 - Maximal vikt

Batteritidsberäkning

Momentkonstanten ges av lutningen på ström/momentkurvan

$$\frac{\Delta M}{\Delta A} = \frac{24{,}25 - 0}{133 - 2{,}7} = 0{,}186\ Nm/A$$

Om vikten för roboten sätts till 25 kilo kan batteritiden plottas som en funktion av lutningen där en ekvivalent lutning får bestämmas för att motsvara en genomsnittlig körning. Då underlaget på vilken roboten kommer att färdas är relativt platt är det inte orimligt att anta att denna lutning är liten. Därmed kan man konstatera att batteritiden för 20 Ah batteriet som valdes och köptes är tillräcklig för det demonstrativa syfte som cellen skall uppfylla.



Figur 46 - Batteritid/lutning. 10-35 Ah

Bilaga 9: PLC, operationer för AGV

Värd:	Operationer:		Triggande insignaler:	Blockerande insignaler:	Utsignaler	Övriga signaler	Operationsbeskrivning:
AGV	AGV vid dörr	70	delarAGV,agvDorr		öppnaDörr		Vad som krävs för att dörren till cellen ska få öppnas
	Åka in i cell	71	SH323, delarAGV, cellTom	SH324			Vad som krävs för att kunna åka in i cell
	Åka ur cell	72	SH323, bilLamnad	SH324		not bilLamnad	Vad som krävs för att kunna lämna cell
	I position i cell	73	agvPlats		stängaDörr	not delarAGV	Vad som krävs för att stänga dörr vid ingång
	Lasta AGV	74	bilBestalld, laststalleTom,	bilLamnad		delarAGV	Vad som krävs för att lasta AGVn med karossdelar
	Lämna färdig bil	75	avlaststalleTom		bilKlar		Vad som krävs för att AGVn ska kunna lämna den färdiga bilen
	AGV tom	76	bilLamnad, bilBortplockad		not bilLamnad		AGV har lämnat bil och är redo för ny runda
	Stänga dörr vid utgång	77	agvUtanfor		stängaDörr		Vad som krävs för att stänga dörren vid utgång