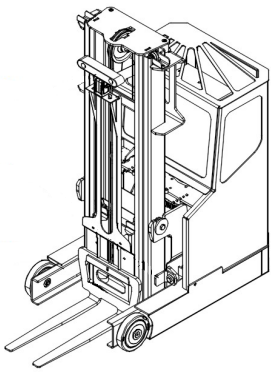
Systemskiss
Autonom styrning av gaffeltruck

Version 1.0

L.A.M.A.
12 oktober 2016



Status

Granskad	Samtliga projektmedlemmar	2016-09-21
Godkänd	Andreas Bergström	2016-09-23

Projektidentitet

Gruppmail:	jenst280@student.liu.se
Hemsida:	http://www.isy.liu.se/edu/projekt/reglerteknik/2016/forklift/
Beställare:	Andreas Bergström, ISY, Linköpings universitet **Telefon:** +46 (0)10-711 54 54, **Mail:** andreas.bergstrom@liu.se
Kund:	Emil Selse, Toyota Material Handling **Telefon:** +46 (0)702032254 , **Mail:** emil.selse@toyota-industries.eu
Kursansvarig:	Daniel Axehill, ISY, Linköpings universitet **Telefon:** +46 (0)13 284042, **Mail:** daniel@isy.liu.se
Projektledare:	Jenny Stenström
Handledare:	Erik Hedberg, ISY, Linköpings universitet **Telefon:** +46 (0)13 281338 , **Mail:** erik.hedberg@liu.se Samuel Lindgren, Toyota Material Handling **Telefon:** +46 (0)767614024 , **Mail:** samuel.lindgren@toyota-industries.eu

Gruppmedlemmar

Namn	Ansvarsområde	Telefon	Mail (@student.liu.se)
Johan Almgren	Testansvarig	070-206 72 45	johal611
Henrik Andersson	Dokumentansvarig	073-854 77 79	henan562
Gustav Elingsbo	Informationsansvarig	073-685 19 75	gusel411
Mikael Hartman	Integrationsansvarig	076-771 13 38	mikha130
Petter Landerhed	Designansvarig	070-627 82 52	petla189
Andreas Norén	Mjukvaruansvarig	072-394 89 95	andno111
Jenny Stenström	Projektledare	070-329 92 21	jenst280

Dokumenthistorik

Version	Datum	Ändringar	Utförd av	Granskare
0.1	2016-09-16	Första utkast.	L.A.M.A.	Samtliga projektmedlemmar
0.2	2016-09-19	Andra utkast.	L.A.M.A.	Samtliga projektmedlemmar
0.3	2016-09-21	Tredje utkast.	L.A.M.A.	Samtliga projektmedlemmar
1.0	2016-09-23	Första version.	L.A.M.A.	Samtliga projektmedlemmar

Innehåll

1 Inledning

Detta dokument innehåller en skiss över det system som ska levereras och uppfylla de krav som definieras i kravspecifikationen, se projektets kravspecifikation. Observera att detta enbart är en skiss vilket medför att samtliga designförslag är helt öppna för diskussion och kan komma att ändras i ett senare skede i projektet.

1.1 Syfte och mål

Syftet med detta projekt är att vidareutveckla en mindre truckplattform, känd som MiniReach, framtagen av Toyota. Projektet avser ge en ökad och förbättrad nivå vad gäller lokalisering, ruttplanering och autonomi. Med detta följer nödvändiga uppdateringar av mjukvarumodeller och simuleringsmiljö.

1.2 Användning

Det konstruerade systemet är tänkt att användas och vidareutvecklas ytterligare av kunden i utvecklingssyfte samt som demonstrationsobjekt för dess kunder och på mässor som t.ex. LARM 2017.

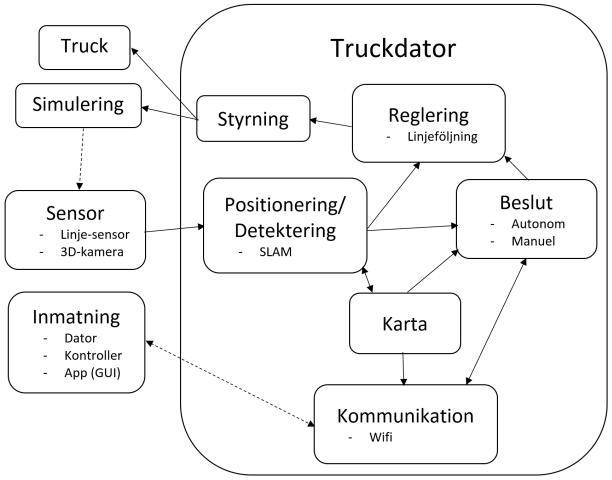
1.3 Definitioner

Nedan presenteras hur vi har definierat koncept för att använda i fortsatt beskrivning av systemet och de krav som ställs på detta.

Hinder	Ett objekt minst av dimension 0.5m x 0.5m x 0.5m som är placerat på lagergolvet.
Fast hinder	Ett hinder som är permanent, stationärt lokaliserat i lagret.
Karta	Information om omgivningen innehållande hinder samt pallar och pallplatser.
Lager	Den miljö där trucken ska arbeta.
MiniTruckApp	Den applikation som är körbar på Android-plattformar som används för att styra och ge kommandon till trucken.
Pall	En lastpall omskalad för att kunna lyftas av trucken till en höjd av cirka 100cm, där pallen med last inkluderad ej får överstiga en totalvikt på 10kg. Pallen har en unik AR-kod för identifiering placerad på de fyra vertikala sidorna.
Pallplats	En plats i markhöjd (om inte annat anges) där pallar kan placeras, platsens area är minst storleken av en pall med marginaler på 1dm åt varje håll. Platsen är försedd med en unik AR-kod placerad mitt i platsen och markerad med gul tejp runt platsens yttre kanter.
System	Syftar på den mjukvara som av projektgruppen utvecklas och integreras under projektets gång.
ROS	Robot Operating System, en Open Source programvara som möjliggör och underlättar kommunikation mellan komponenter och moduler vid robotutveckling.
Truck	En 1:3 nedskalad version av Toyotas ordinära truckar med liknande funktionalitet framtagen av Toyota och går under benämningen MiniReach.

2 Systemöversikt

Produkten är en mjukvara för autonom körning av MiniReach, samt motsvarande simuleringsmiljö, båda med tillhörande dokumentation. Systemet ska utifrån extern inmatning autonomt kunna utföra viss enklare lagerhantering. Viss funktionalitet finns redan implementerat som kommer att användas som en bas, även om denna i vissa fall kan behöva omarbetas. En översikt av det planerade systemet kan ses i figur 1.



Figur 1: Översikt av systemet.

2.1 Grov beskrivning av systemet

Hela systemet med undantag för MiniTruckAppen är placerad i truckens dator som sitter i trucken. Användaren kan använda sig av MiniTruckAppen för att se truckens status samt skicka instruktioner så som ”flytta pall A till pallplats B”. Mjukvaran i truckdatorn är uppdelad i sju olika moduler efter funktionalitet. De moduler som ska finnas är:

Beslutsmodul	Huvudprogram som tar in data från de andra modulerna och fattar beslut utifrån detta.
Regleringsmodul	Har i syfte att reglera rörelser enligt beslutsmodulens rutt-/rörelseplanering.
Styrningsmodul	Styr produktens aktuatorer, innehåller regulatorer som reglerar produktens rörelser.
Positioneringsmodul	Modul som håller koll på produktens positionering i rummet samt position relativt andra objekt. Behandlar signaler från linjesensorer samt eventuellt andra sensorer som kan komma att testas under projektets gång. Exempelvis 3D-kameran och en IMU, Inertial Measurement Unit.
Detektionsmodul	Används för att detektera och identifiera laster. Används även för att detektera eventuella hinder.
Kartläggningsmodul	Kartlägger utifrån sensordata området som produkten befinner sig i. Ger användare möjlighet att kartlägga ett nytt, okänt område.
Sensormodul	Tar in och behandlar sensordata.
Kommunikationsmodul	Agerar som interface mellan övriga moduler och MiniTruckAppen.

2.2 Designfilosofi

Produkten är en utvecklings- och demonstrationsplattform för kundens autonoma lagertruckar. På grund av detta kommer produkten vara modulärt uppbyggd för att ändringar lätt ska kunna göras på endast en del av produkten utan att funktionalitet i andra oberoende moduler påverkas.

2.3 Hårdvara

Miljön som systemet ska användas på är truckplattformen "MiniReach". Den är till utseende och funktion en 1:3 nedskalad truck baserad på de truckar som kunden tillverkar. Den funktionalitet som MiniReach saknar som dess fullskaliga motsvarighet har är möjlighet att finjustera lyftgafflarna i sidled. Omvänt så har MiniReach vid projektstart två sensorer som ordinarie truckar ej har, en laserskanner och en 3D-kamera. Ytterligare sensorer kan läggas till.

Mjukvaran för MiniReach körs på en Jetson TX1 (developer kit - LinuxPC) som är monterad i trucken. För att externt kontrollera och ge kommandon till MiniReach används en LinuxPC med Wi-Fi uppkoppling eller en Android-plattform.

2.4 Mjukvara

Truckens mjukvara kommer att utvecklas på utvecklingsplattformen ROS, där det existerar viss färdig funktionalitet som ofta används vid robotutveckling. All kod på trucken kommer att skrivas i C++ och Python. Koden ska följa Googles kodstandard.

Mjukvaran för MiniTruckAppen kommer att skrivas i Java med Android Software Development Kit (SDK) och även den koden kommer att följa Googles kodstandard.

2.5 Existerande funktionalitet vid projektstart

Trucken som detta projekt ska arbeta vidare med har viss ingående funktionalitet. De mest betydande (ej självklara) listas nedan.

- **Lokalisering:** Med paket i ROS görs lokalisering och kartläggning (SLAM) med ett partikelfilter, denna byggs upp med data från en laserskanner samt given områdeskarta.
- **Simulering:** Istället för att utföra tester med riktig hårdvara kan trucken simuleras. Den simulerade trucken har samma funktionalitet som den verkliga, dock med något annorlunda modellering vilket ger ändrade köregenskaper.
- **Styrning:** Med paket i ROS kan trucken utifrån kartan från lokaliseringen identifiera en väg att ta sig från sin position till en annan given punkt. Den kan sedan följa den linje som bestämts för att ta sig till den önskade destinationen samt rikta upp sig i önskat läge när målet nåtts. Den linjeföljande funktionen är, åtminstone i simuleringsmiljön, inte väl fungerande då den överstyr vid regleringen vilket leder till ett oscillativt beteende.
- **Lagerhantering:** För att identifiera olika laster i miljön har trucken funktionalitet att identifiera och lokalisera AR-koder. I viss mån kan den sedan lyfta upp och flytta runt lasten när den har hittats.
- **Kommunikation:** Trucken kan kommunicera med och styras av externa enheter via Wi-Fi.

3 Systemets komponenter

Under denna rubrik finns en mer utförlig beskrivning av de olika modulerna samt en del implementeringsförslag.

3.1 Sensor

För att trucken ska kunna agera autonomt krävs flera sensorer för att detektera och identifiera omgivande miljö. Trucken behöver kunna detektera hinder, väggar och andra fasta objekt för att undvika kollisioner. Den behöver även kunna detektera och identifiera pallar och pallplatser. Dessa markeras med unika AR-koder som behöver avläsas för identifiering, men även för att användas som referenssystem för truckens position i förhållande till pallen eller pallplatsen. Detta för att hög precision på positioneringen krävs för hantering av pall.

De sensorer som är installerade vid projektstart är:

- Laserskanner för avståndsmätning av modell Hokuyo URG-04LX-UG01 med räckvidd på 5.6m, 240^o synfält, uppdateringshastighet 10Hz och vinkelupplösning på 0.36^o.
- 3D-kamera av modell ZED stereo camera med en operativ räckvidd på 0.7-20m.

För att uppnå högre precision på lokalisering i lagret ska även nyttan av att lägga till en s.k. inertial measurement unit (IMU) undersökas.

Sensorernas position på trucken är även något som kommer undersökas men primärt är det laserskannerns uppgift att detektera hinder och den kommer därmed att vara placerad så den har god detektion i körriktningen. 3D-kameran är främst avsedd för detektering, identifiering och lokalisering av AR-koder för att möjliggöra hantering av pallar och pallplatser. Den kommer därför vara riktad i samma riktning som gafflarna.

Sensormodulen på trucken kommer att agera som ett gränssnitt mellan truck och sensorer vilket innefattar konvertering av data från sensorer till användbart format för övriga moduler. Modulen hanterar ingen tolkning av data utöver eventuell konvertering.

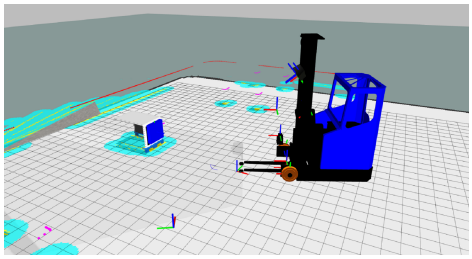
3.1.1 Sensor - potentiella noder

ar_track_alvar	Denna nod hanterar identifiering och spårning av AR-taggar.

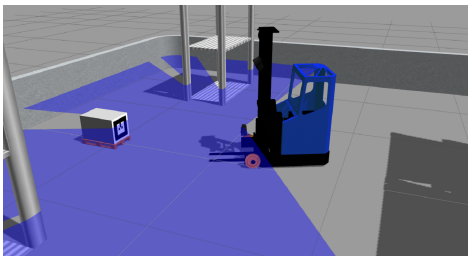
3.2 Simulering

Simuleringen tar emot data från styrmodulen och uppdaterar simuleringsvärlden därefter. Den förmedlar också data till sensormodulen vilket möjliggör navigering i den simulerade världen. Simuleringen sker i noden Gazebo vilket tillsammans med visualisering av data genom verktyget RViz ska efterlikna den verkliga trucken och dess arbetsmiljö i så stor utsträckning som möjligt. Det vill säga hårdvarans egenskaper ska återskapas i simuleringen. Till exempel ska sensorernas upplösning och prestandaegenskaper vara likvärdiga i båda miljöerna. Gazebo visar den simulerade miljön och RViz visualiserar sensordata.

Figur 2 och Figur 3 visar hur de olika programmen representerar och uppfattar samma simuleringsmiljö.



Figur 2: Figur som visar hur sensordata visualiseras i RViz. Blåa markeringar indikerar hinder. 3D kameran ser pallen framför trucken.



Figur 3: Figur som visar miljön i Gazebo. Det blåa området är det som uppfattas av laserscannern.

3.3 Reglering

Regleringsmodulen ser till att styrmodulen får lämpliga körinstruktioner givet ett färdmål eller uppdrag från beslutsmodulen. Denna modul ska inbegripa en del noder från paketet ros_controllers, där bland annat joint_position_controller finns. Denna nod jämför den data som är inmatad från exempelvis ett interface med den data som den hämtar ut från hårdvaran, t.ex. genom JointStateInterface. Detta används för att skapa en styrsignal som i sin tur kan skickas vidare för exekvering via EffortJointInterface. E-stop handling är aktuellt att införa för att uppfylla krav gällande beteende när plötsliga hinder dyker upp i potentiell bana.

3.3.1 Reglering - potentiella noder

joint_position_controller	Denna nod tar in önskad position som referenssignal samt en mätsignal och returnerar en styrsignal enligt angivna parametrar och reglermetod.
joint_effort_controller	Denna nod tar in önskad effekt som referenssignal samt en mätsignal och returnerar en styrsignal enligt angivna parametrar och reglermetod.
joint_velocity_controller	Denna nod tar in önskad hastighet som referenssignal samt en mätsignal och returnerar en styrsignal enligt angivna parametrar och reglermetod.
diff_drive_controller	Denna nod hjälper vid reglering då trucken ska färdas längs en linje.
Actuator State Interfaces	Denna nod ger information om tillstånden på truckens aktuatorer, vilka kan användas till reglering.
Position Actuator Interface	Denna nod används för att reglera positionen på en aktuator givet en styrsignal från aktuell regulator.

3.4 Styrning

Styrmodulen omvandlar den data som mottagits från regleringssmodulen till något som kan aktualiseras av Minireach respektive simuleringsmodulen. Exempel på aktuella noder i ROS är bland annat Base velocity controller och Base odometry controller, vilka tillsammans med beslutsmodulen uppfyller samtliga krav för att möjliggöra verkställande av en specifik hastighet för trucken.

I paketet move_base finns noden move_base_action, som försöker flytta roboten till en given position med en given orientering.

3.4.1 Styrning - potentiella noder

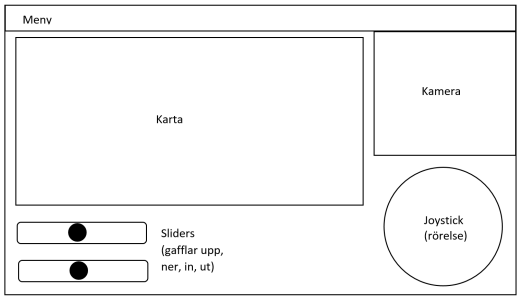
base_velocity_controller	Denna nod gör att truckens hjul uppnår en given hastighet alternativt vinkelhastighet. Den startas i standardläge samtidigt som aktuell robot.
base_odometry_controller	Denna nod startas samtidigt som aktuell robot och publicerar tillståndsinformation på tf.
move_base_action	Denna nod försöker flytta roboten till en given position och med en given orientering.

3.5 Beslut

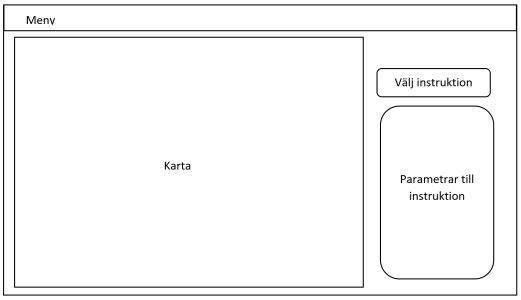
Beslutsmodulen fattar ett beslut baserat på mottagen data från övriga moduler, bland annat kommunikationsmodulen, kartmodulen samt positioneringsmodulen. Exempel på data kan vara avstånd till närmaste vägg eller huruvida trucken körs autonomt eller manuellt. Beslutet som fattats i beslutsmodulen skickas vidare för framtagning av aktuell aktuation i regleringsmodulen. Det är även i beslutsmodulen som ruttplaneringsalgoritmen och avsökningsalgoritmen kommer att finnas.

3.6 MiniTruckApp

Trucken ska kunna kontrolleras av en applikation, vilket är mest avsett för att användas vid demonstrationer. Det finns vid projektstart en existerande Android applikation som kan styra trucken i manuellt läge. Denna applikation ska vidareutvecklas för att även inkludera möjlighet till övervakning och ge kommandon vid autonom körning av truck. Skisser på MiniTruckAppens GUI kan ses i Figur 4 och Figur 5. Applikationen kan endast köras på Android plattformar med operativversion 4.0.3 (API 15) eller senare, den körs med fördel på en plattform med stor skärm.



Figur 4: Skiss av GUI vid manuell körning.



Figur 5: Skiss av GUI vid autonom körning.

Val av instruktioner ska göras med en dropdown-meny med instruktionerna ”*Söka av lager*” och ”*Flytta pall*”. Vid flyttning av pall behöver ytterligare instruktionsmöjligheter finnas där man kan välja pall och pallplats utifrån dem som identifierats i lagret.

Kommunikation med trucken ska ske via Wi-Fi till truckens lokala IP-adress och det är kommunikationsmodulen i trucken som kommer agera gränssnitt däremellan.

3.7 Karta

Kartläggningsmodulen har två huvudsakliga uppgifter, att läsa in och att skriva information till en karta. Den ska få in information från positioneringsmodulen samt sensormodulen för att kunna skapa sig en bild av var i rummet trucken befinner sig samt hur dess omgivning ser ut. Information om truckens plats och hastighet i rummet kommer vara viktig för att kunna placera ut objekt på kartan med god noggrannhet.

3.7.1 Karta - potentiella noder

slam_gmapping	Denna nod är anpassad för att få indata från en laserscanner och omvandlar datan till en två dimensionell karta som visar sannolikhet för hinder, så kallad OccupancyGrid.
map_server	Tar in OccupancyGrid och värderar pixlarna inom intervallet [0, 100] som motsvarar sannolikheten för att det finns ett hinder på den platsen. Okända pixlar ges värdet -1. Kartan kan sedan sparas i formatet YAML med hjälp av map_saver.

3.8 Kommunikation

Kommunikationen mellan truck och extern utrustning sker via Wi-Fi. Då den främsta kommunikationskanalen kommer att utgöras mellan truck och MiniTruckAppen ska denna modul utgöra ett gränssnitt för denna kommunikation. Trucken är även utrustad med en Bluetooth-mottagare för att kunna koppla upp sig mot en extern joystick som kan användas vid manuell styrning.

Referenser