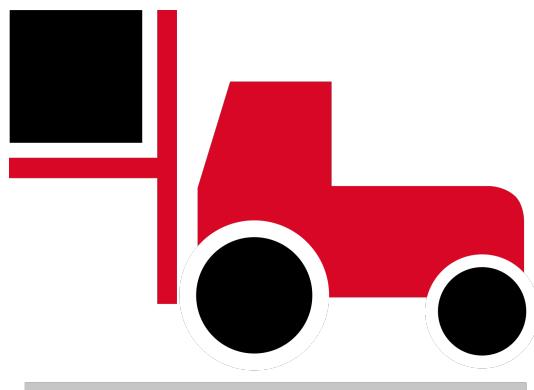


Teknisk rapport

Redaktör: Sofie Dam

Version 0.1



GRUPPTRUCK

Status

Granskad	Dokumentansvarig	2017-12-14
Godkänd		

PROJEKTIDENTITET

2017/HT, GruppTruck

Tekniska högskolan vid Linköpings universitet, ISY

Gruppdeltagare

Namn	Ansvar	Telefon	E-post
Gabriel Fredriksson	Projektledare	076-294 04 49	gabfr905@student.liu.se
Sofie Dam	Dokumentansvarig	070-422 32 57	sofda068@student.liu.se
Johannes Bodin	Designansvarig, Uppdrags- ansvarig Delområde 1 & 4	070-246 05 66	johbo346@student.liu.se
Daniel Nilsson	Mjukvaruansvarig	070-733 23 10	danni768@student.liu.se
Emil Relfsson	Testansvarig	070-635 08 37	emire260@student.liu.se
Max Antonsson	Uppdragsansvarig Delom- råde 2	070-781 77 75	maxan749@student.liu.se
Jasmina Hebib	Uppdragsansvarig Delom- råde 3	073-672 66 28	jashe481@student.liu.se

Kund: Toyota Material Handling Manufacturing Sweden AB, 595 81 Mjölby

Kursansvarig: Daniel Axehill, 013-28 40 42, daniel.axehill@liu.se

Handledare: Erik Hedberg, 013-28 13 38, erik.hedberg@liu.se

Beställare: Andreas Bergström, 010-711 54 54, andreas.bergstrom@liu.se

Innehåll

1 Inledning	6
1.1 Parter	6
1.2 Bakgrund	6
1.3 Syfte och mål	6
1.4 Definitioner	7
2 Översiktlig beskrivning av systemet	8
2.1 Hårdvara	8
2.2 Simuleringsmiljö	8
3 Delområde 1: Truckbeskrivning i simuleringsmiljö	10
3.1 Modelluppbyggnad	10
3.1.1 Modellbeskrivning	11
3.2 Uppmätning av drivhjulets förskjutning	11
3.2.1 Validering mot tester	13
3.3 Införande	15
3.3.1 Parameterinförande	15
3.4 Modellvalidering	15
3.4.1 Problem med hårdvarugränssnitt mot simuleringsmiljö	15
3.4.2 Inställning av regulatorparametrar	16
3.5 Resultat	18
3.6 Analys	20
3.6.1 Ändrad effektiv förskjutning vid förslitning av hjul	20
3.6.2 Fortsatt dålig modell trots införd hjulförskjutning	20
3.7 Vidareutveckling	21
4 Delområde 2: Tillståndsmodell	22
4.1 Ursprunglig Idé	22
4.2 Förenklad fysikalisk modell med PI-regulator	23
4.2.1 Problemuppställning	24
4.2.2 Fysikalisk ansats	24
4.2.3 PI-regulator	25
4.2.4 Vidareutveckling av tillvägagångssätt	26
4.3 Grey Box Model	26
4.3.1 Hastighet till hastighet	27
4.3.2 Rotationshastighet till Rotationshastighet	29

4.3.3	Rotationshastighetens påverkan på hastigheten	33
4.3.4	Från hastighet och rotationshastighet till position	37
4.3.5	Hela modellen	37
4.3.6	Finjustering av parametrar	38
4.4	Resultat	39
4.5	Analys	40
4.5.1	Jämförelse med modell av låg ordning	40
4.5.2	Använda modellen till simulerings i Gazebo	41
4.6	Slutsats	42
4.7	Vidareutveckling	42
5	Delområde 3: Precisionsinkörning	43
5.1	PD-regulator	44
5.2	Hastighetsreglering	45
5.3	PD-regulator med framkoppling	45
5.3.1	Framkoppla ett steg	45
5.3.2	Framkoppla med en första ordningens dynamisk modell	46
5.3.3	Framkoppla med en högre ordningens dynamisk modell	46
5.4	Dimensionsreducering	46
5.5	MPC-regulator	47
5.6	Linjär MPC-regulator	47
5.7	Olinjär MPC-regulator	48
5.8	Planeraren	49
5.9	Resultat	51
5.10	Slutsats	53
Referenser		54

Dokumenthistorik

Version	Datum	Utförda förändringar	Utförda av	Granskad
0.1	2017-12-14	Första utkast	Alla	Sofie Dam

1 Inledning

1.1 Parter

Andreas Bergström, doktorand på Institutionen för Systemteknik (ISY), avdelningen för reglerteknik vid Linköpings Universitet, är projektets beställare.

Kontaktuppgifter:

Kontor: 2A:501, B-huset, Campus Valla

Telefon: 010-711 54 54

E-post: andreas.bergström@liu.se

1.2 Bakgrund

Projektet har genomförts som en del av kursen TSRT10 Reglerteknisk projektkurs, CDIO vid Linköpings universitet i samarbete med Toyota Material Handling i Mjölby. Toyota har utvecklat ett system med nedskalade versioner av deras truckar som de tillhandahållit projektgruppen.

Systemet som användes i projektet har tidigare utvecklats av bland annat ett CDIO-projekt förra året samt en grupp på fyra studenter som genomfört ett sommarjobbsprojekt. Projektarbetet inleddes med ett besök på Toyota och systemet introducerades av sommarjobbsgruppen där de berättade vad de arbetat med och vad de tyckte vore en lämplig fortsättning.

1.3 Syfte och mål

Syftet med detta projekt var att vidareutveckla Toyota Material Handlings koncept med förarlösa truckar. Med bakgrund av att truckar i verkligheten är väldigt tunga och dess system är komplicerade, så används förminskade gaffeltruckar till utvecklingen. Den förminskade gaffeltrucken har samma grundfunktioner som den fullstora modellen så att utvecklingen kan ske på ett både smidigt och säkert sätt. I samband med vidareutvecklingen ska även uppdateringar av mjukvarumodeller och simuleringsmiljö implementeras. Arbetet delades upp i tre olika delområden, dessa introduceras nedan.

Delområde 1: Truckbeskrivning

Den fysiska trucken finns beskriven i en modell som används i simuleringsprogrammet Gazebo. Vid starten av projektet överensstämde modellen inte helt med det verkliga systemet eftersom ingen hänsyn togs till dess hjulkonfiguration. Under projektets gång har en förbättrad modell tagits fram där trucken tar hänsyn till denna hjulkonfiguration. Detta innebär att truckens nuvarande beteende i simuleringen efterliknar det verkliga beteendet bättre jämfört med vad den gjorde innan modifieringen gjordes.

Delområde 2: Tillståndsmodell

För att reglera precisionsinkörningen (se delområde 3) kommer en MPC-regulator utvecklas. Denna regulator behöver en modell av trucken för att fungera. Målet med delområde

2 är därför att utveckla en tillståndsmodell av trucken till denna regulator.

Delområde 3: Precisionsinkörning

Målet med delområde 3 är att utveckla en regulator som är så robust och exakt att man ska kunna placera sig framför pallar och därmed möjliggöra upphämtning. Målet är dessutom att göra regleringen så robust att även skarpa kurvor kan tas. För att åstadkomma detta har en hastighetsreglering implementerats i två delar, en återkopplande och en framkoppling. Detta möjliggjorde en aggressivare regulator för att klara skarpare svängar. Även framkoppling för vinkelhastigheten och MPC har undersökts men ej implementerats.

1.4 Definitioner

ROS Robot Operating System: ramverk för utveckling av mjukvara till robotar.

Nod Delprogram i ROS.

Topic Kanal där ROS-noder kan publicera data. Medium för kommunikation mellan ROS-noder

Publisher Nod som skriver data till en topic

Subscriber Nod som läser data från en topic

H2H Önskade linjära hastighetens påverka på den uppmätta linjära hastigheten

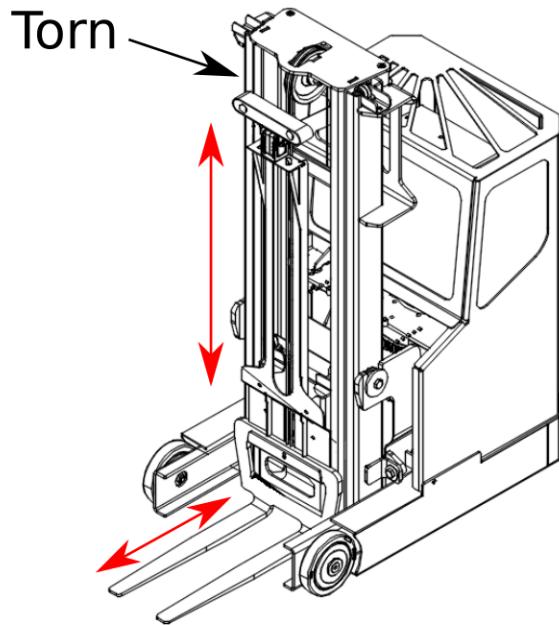
R2R Önskade rotationshastighetens påverka på den uppmätta rotationshastigheten

R2H Önskade rotationshastighetens påverkan på den uppmätta linjära hastigheten

2 Översiktlig beskrivning av systemet

2.1 Hårdvara

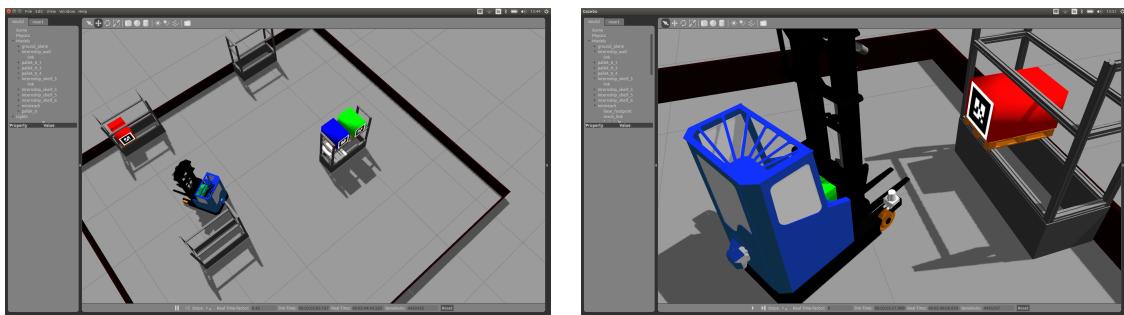
En Reach-truck är en typ av gaffeltruck som har möjligheten att köra ut sitt torn och på så sätt göra det möjligt att hantera pallar på ställen som man inte kan köra in under. En Reach-trucks övergripande konstruktion och möjliga gaffelrörelser visas i Figur 1.



Figur 1: Reach-truckens konstruktion och möjliga gaffelrörelser.

2.2 Simuleringsmiljö

Projektgruppen har endast haft två tillfällen att arbeta med den riktiga trucken. Därför har mycket av arbetet utförts i en simuleringsmiljö som funnits till systemet. Simuleringsmiljön är framtagen med applikationen Gazebo som är en fysikmotor som simulerar rörelser hos fysikaliska modeller med egenskaper som massa, hastighet och friktion [5]. Trucken i simuleringsmiljön i Gazebo kan ses i figur 2. De fysikaliska modellerna som beskriver trucken är uppbyggda med Robot Operating System, *ROS* som är en uppsättning programbibliotek och verktyg för att bygga robotapplikationer.

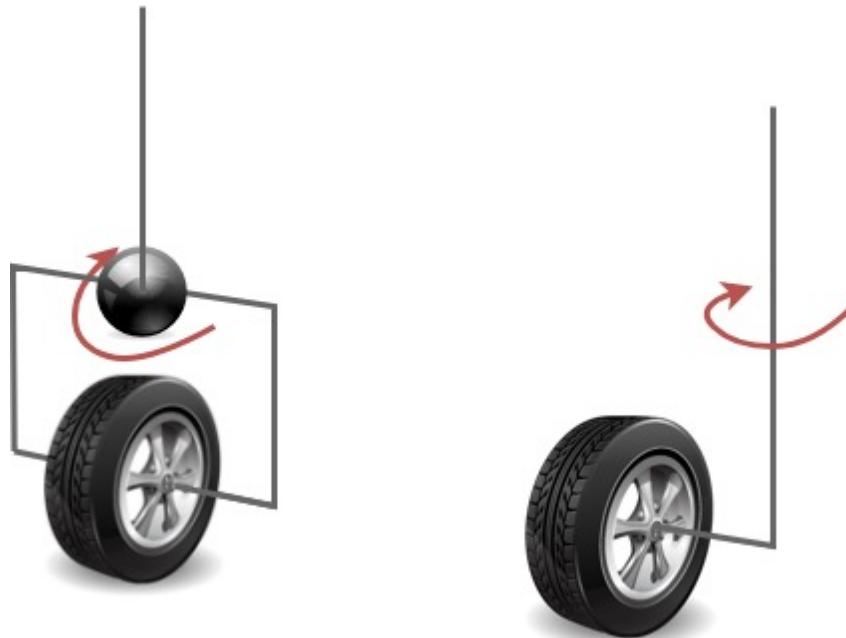


Figur 2: Trucken i simuleringsapplikationen Gazebo.

3 Delområde 1: Truckbeskrivning i simuleringsmiljö

Detta avsnitt behandlar projektets första delområde, nämligen utveckling av den modell som används i simuleringsmiljön Gazebo. Då större delar av projektet är baserad på att modellera och utveckla systemet i en simuleringsmiljö, är det ytterst viktigt att ha en bra simuleringsmodell som kan beskriva det verkliga systemet så exakt som möjligt. Ett typiskt scenario där små marginaler i truckrörelsen har en stor betydelse är när trucken står framför en pall och ska utföra en pallupphämtning. Pallens trånga gaffelgångar medför att trucken behöver positionera sig väldigt exakt framför pallen innan en upphämtning av pall kan utföras. Därför är truckens vinkelställning speciellt viktig, där endast små differenser i vinkel får förekomma.

Största fokuset med detta delområde lades på att ta fram en bättre simuleringsmodell som ska efterlikna det verkliga systemet mer. En tydlig skillnad som observerades i simuleringsmodellen jämfört med den verkliga trucken var att truckens bakre hjul inte var beläget på samma ställe. Hjulet som beskrivs i modellen sitter precis rakt under truckens rotationsaxel, vilket visas till vänster i Figur 3. I den verkliga trucken är däremot hjulet något förskjutet i axiell riktning, vilket visas till höger i Figur 3. En uppgift var därför att mäta upp denna hjulförskjutning (offset) samt implementera denna i modellen så att den tar hänsyn till förskjutningen.



Figur 3: Bilden till vänster visar hur den nuvarande modellen i Gazebo har det bakre hjulet konfigurerat medan den högra bilden visar hur hjulet är konfigurerat på den verkliga trucken.

3.1 Modelluppbryggnad

Den modell som används i simuleringsmiljön är definierad enligt s.k. URDF-format (*Unified Robot Description Format*). I detta format specificeras olika fysiska kroppar och hur

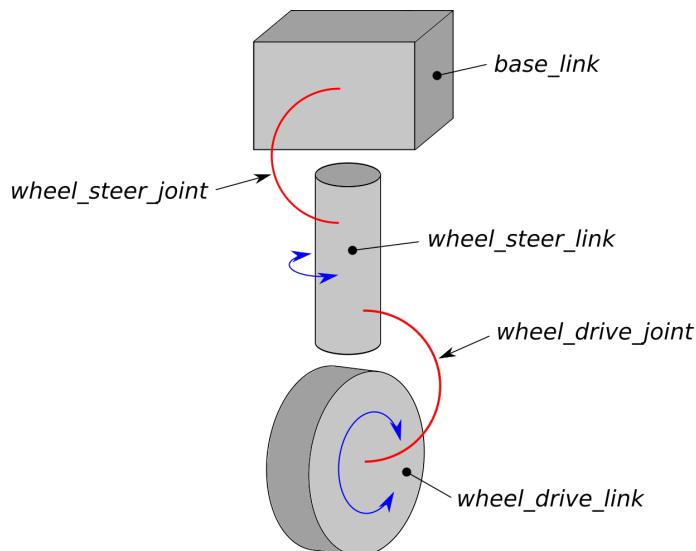
de är kopplade till varandra. Minireach-modellen är uppdelad i ett flertal olika filer för att logiskt separera de olika undermodellerna från varandra och på så sätt underlätta vidareutveckling och felsökning. Filerna finns i paketet *minireach_description* i huvudrepositoriet *minireach*. Nedan presenteras de viktigaste filerna för detta delområde av projektet.

- *minireach.xacro*: huvudfil som definierar olika parametrar och kallar på undermodellsfiler
- *assemblies/drive_wheel_assembly.xacro*: undermodell för drivhjulet och dess infästning

Den sistnämnda av dessa filer specificerar, foga förvånande, drivhjulet och hur detta är kopplat till resten av trucken.

3.1.1 Modellbeskrivning

Modellen i Gazebo är av stor betydelse vid införandet av parametern som modellerar hjulförskjutningen. Denna är uppbyggd av både länkar (engelska: links) och ledar (engelska: joints). Lederna förbindrar länkarna och sätter därmed ihop flera objekt till ett enhetligt objekt. Uppbyggnaden av undermodellen *drive_wheel_assembly.xacro* presenteras i Figur 4. I figuren kan man till exempel se att leden *wheel_drive_joint* förbindrar länkarna *wheel_steer_link* och *wheel_drive_link*, samtidigt som leden *wheel_steer_joint* förbindrar länkarna *wheel_steer_link* och *base_link*.



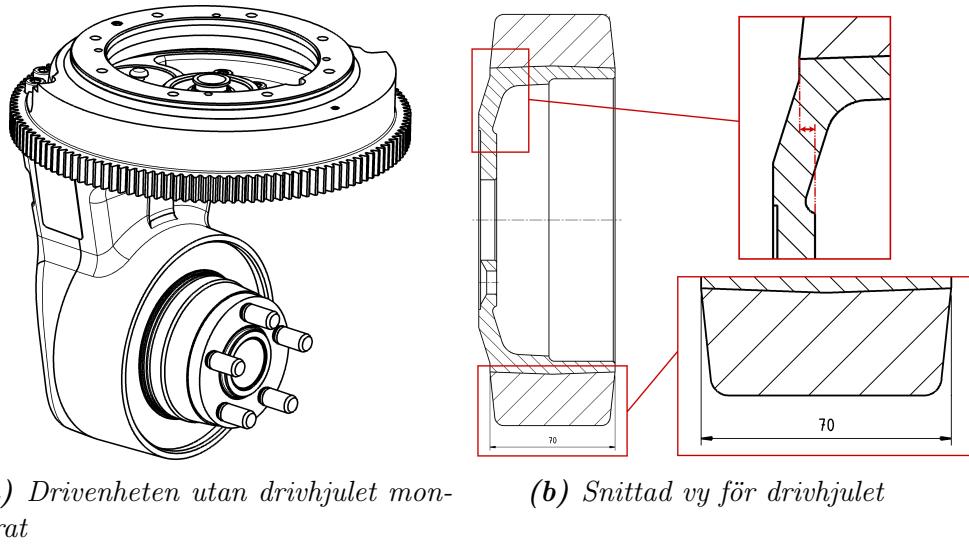
Figur 4: Modellen i Gazebo med länkar (links) och ledar (joints).

3.2 Uppmätning av drivhjulets förskjutning

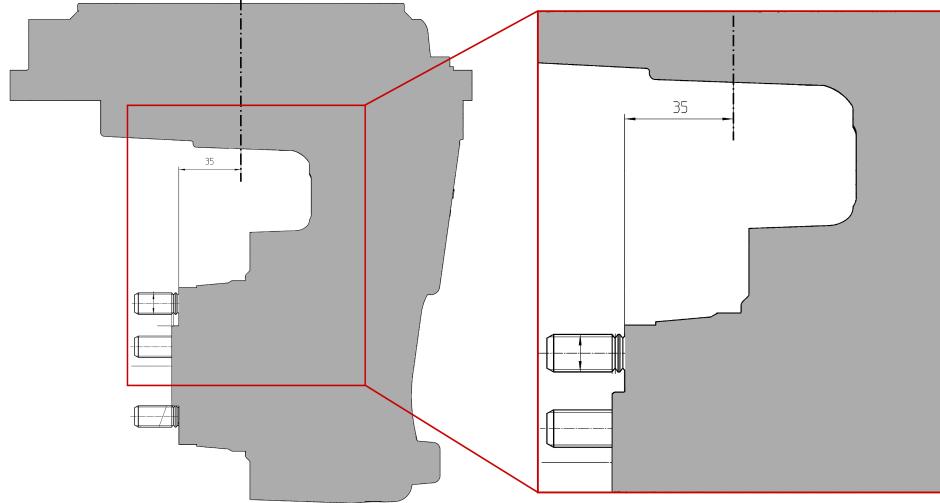
För att uppskatta storleken på hjulförskjutningen fanns både ritningar och CAD-filer på drivhjulet och dess montage att tillgå. I Figur 5a visas en översiktsvy av truckarnas drivenhet utan själva drivhjulet monterat. Denna enhet står både för framdrivning av trucken

samt styrning då den yttre kuggkransen som syns i figuren är kopplad till ett styr servo.

I Figur 5b ses att drivhjulets slitbana är 70 mm bred. Samtidigt ses i Figur 6 att avståndet från drivhjulets infästningsplan till centrum av enhetens rotationsaxel är 35 mm, dvs. hälften av drivhjulets bredd. Slutsatsen är alltså att drivhjulet hade varit centrerat under styrervots rotationsaxel om hjulets infästningsplan och kanten på hjulets slitbana hade varit i linje med varandra. Så är dock inte fallet, något som blir tydligt från den övre högra delen av Figur 5b, där man ser att dessa är förskjutna i förhållande till varandra.



Figur 5: Ritningsvyer för drivenheten samt drivhjulet



Figur 6: Snittvy av drivenheten utan drivhjulet monterat

Mått från ritningarna och mätningar i en CAD-fil av drivhjulet visade att denna förskjutning var 3,7 mm.

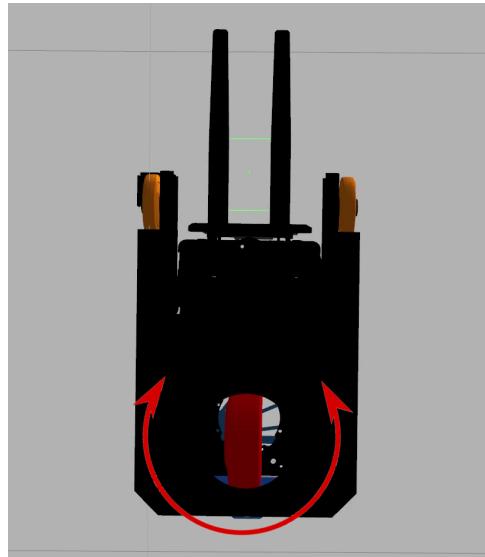
3.2.1 Validering mot tester

För att validera storleken på hjulförskjutningen samt för att studera dess påverkan på truckens beteende skrevs ett testprogram som kunde köras både i simuleringen och på den verkliga trucken. Detta program fungerade på följande sätt:

- Sätt maxhastigheten på drivhjulet till 0 rad/s för att förhindra att trucken körs framåt eller bakåt
- Rotera drivhjulets styr servo åt ena riktningen så att drivhjulet ställer sig tvärs körriktningen
- Rotera servot i motsatt riktning så att drivhjulet ställer sig tvärs körriktningen fast 180° vridet åt andra hållet
- Upprepa de föregående två punkterna flertalet gånger

Anledningen till att testprogrammet lades upp på detta sätt var för att det tidigare hade observerats att trucken tydligt rörde på sig när drivhjulets servo drevs fram och tillbaka (såsom beskrivet i testprogrammet), och att detta därför var ett lämpligt fall att studera.

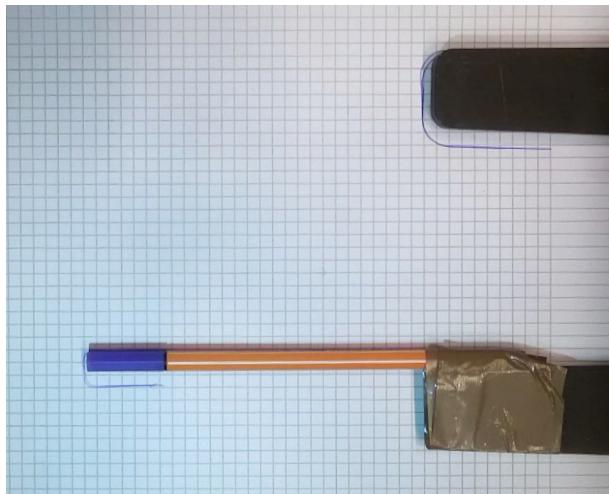
I Figur 7 visas en principiell skiss över hur testprogrammet fungerar.



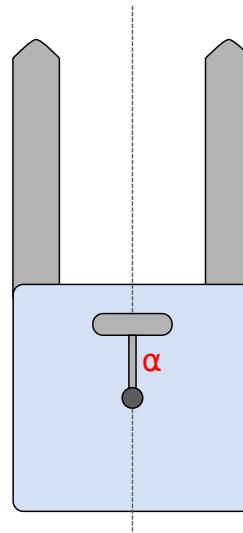
Figur 7: Principiell skiss av testprogrammets funktion. Drivhjulets servo drivas växelvis fram och tillbaka för att åstadkomma en vinkling av drivhjulet, medan resten av trucken står stilla.

När testprogrammet kördes studerades truckens beteende. Något som syntes tydligt var att trucken rörde sig fram och tillbaka när programmet kördes, samtidigt som den vinklades en aning. I Figur 8 och 9 visas positionen hos truckens gaffelspetsar efter dessa rörelser. Eftersom att hjulet vrider sig 180° under denna rörelse borde trucken förflytta sig dubbelt så långt som storleken på drivhjulets förskjutning (markerad med α i Figur 8b). Om man studerar t.ex. hur spetsen hos den fastmonterade pennan rör sig ser man att den förflyttar

sig omkring 7 mm i truckens gaffelriktning. Detta verifierar att det uppmätta värdet på hjulförskjutningen (3,7 mm) är i rätt storleksordning, eftersom att detta värde i teorin skulle ge en förflyttning i längsriktningen på 7,4 mm.

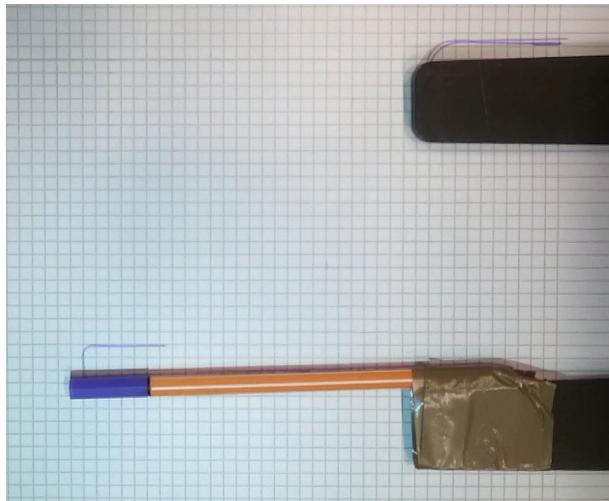


(a) Gaffelspetsarnas position

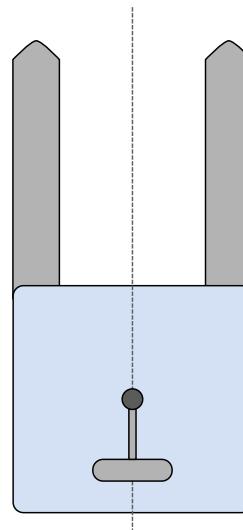


(b) Drivhjulets position

Figur 8: Gaffelspetsarnas position efter att drivhjulet vräddits upp till den övre positionen.



(a) Gaffelspetsarnas position



(b) Drivhjulets position

Figur 9: Gaffelspetsarnas position efter att drivhjulet vräddits ner till den undre positionen.

3.3 Införande

3.3.1 Parameterinförande

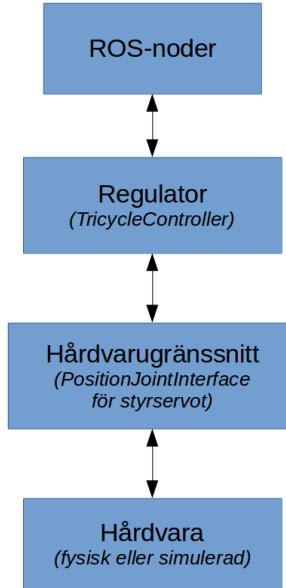
Det var av stort intresse att införa en parameter för hjulförskjutningen som står till förfogande då man vill hänvisa till denna i andra sammanhang, till exempel i andra ställen i koden eller andra filer. Parametern har en särskilt stor betydelse då man vill ändra värdet av denna, vid till exempel förslitningar på hjulet. Storleken på hjulförskjutningen kan då enkelt modifieras genom att ändra värdet på parametern. Införandet av parametern passar bäst i huvudfilen *minireach.xacro* och kan därmed användas av andra underfiler, som till exempel underfilen *drive_wheel_assembly.xacro*. Det är i denna underfil som den faktiska förskjutningen av leden *wheel_drive_joint* i y-riktningen sker.

3.4 Modellvalidering

När hjulförskjutningen var uppmätt, validerad och implementerad undersöktes modellens överensstämmelse med den verkliga trucken. Testprogrammet kördes i simuleringsmiljön och truckens rörelse studerades. Förväntningen var att trucken nu efter implementering av förskjutningen skulle röra sig likt den verkliga trucken. Detta blev initialt dock inte fallet. Trots utökad modell stod trucken helt stilla när drivhjulet vinklades fram och tillbaka, dvs. den simulerade trucken var helt opåverkad av att hjulförskjutningen introducerats.

3.4.1 Problem med hårdvarugränssnitt mot simuleringsmiljö

Regulatorstrukturen för truckens styrsystem består av flera lager. I Figur 10 visas ett blockschema för denna uppbyggnad. På högsta nivån ligger ROS-noderna (t.ex. *move_precise.py* i fallet för precisionsinkörningen) som beräknar och ställer ut referenssignaler för truckens linjärhastighet och rotationshastighet. Dessa signaler används sedan i huvudregulatorn *TricycleController* för att reglera drivhjulets hastighet och styrervots vinkel. Denna regulator ger i sin tur styrsignaler till s.k. hårdvarugränssnitt på underliggande nivå. Dessa gränssnitt kan vara av olika typ (PositionJointInterface, VelocityJointInterface, EffortJointInterface) och fungerar som ytterligare en regulatornivå där specificerad referensstorhet (position, hastighet, motorström, etc.) regleras ner mot hårdvarunivå.



Figur 10: Principiell skiss över hur systemet regleras från ROS-noder på högsta nivå till hårdvara på lägsta nivå

I fallet för styrservot var hårdvarugränssnittet av typ *PositionJointInterface*, dvs. *TricycleController*-regulatorn ställde ut önskad vinkel som hårdvarugränssnittet sedan reglerade styrservot efter (insignalen för servot på den fysiska trucken var just styrvinkel så denna regulatoruppgögnad var lämplig). Tester då detta gränssnitt ändrades till *VelocityJointInterface*-typ visade att den simulerade trucken då betedde sig mer likt den verkliga trucken, så slutsatsen att det var något problem med *PositionJointInterface*-implementeringen kunde dras. Orsaken till problemet visade sig till sist ligga i att vissa inställningar av PID-parametrar saknades för just denna styrvinkelsregulator för att simuleringen skulle fungera på önskat sätt. Om dessa inställningar saknades skippade simuleringssmiljön helt att simulerat fysiken (kontakt med underlaget, friktion, etc.) när hjulet vinklades och ställde hjulet helt förutsättningslöst i den önskade vinkel. När dessa PID-parametrar ställdes in började dock den simulerade truckens beteende genast likna det verkliga systemets.

Regulatorinställningarna skulle göras i filen *minireach_control.yaml* i *minireach_control*-paketet i *minireach*-repot. Nedan visas ett exempel på det kodstycke som infördes.

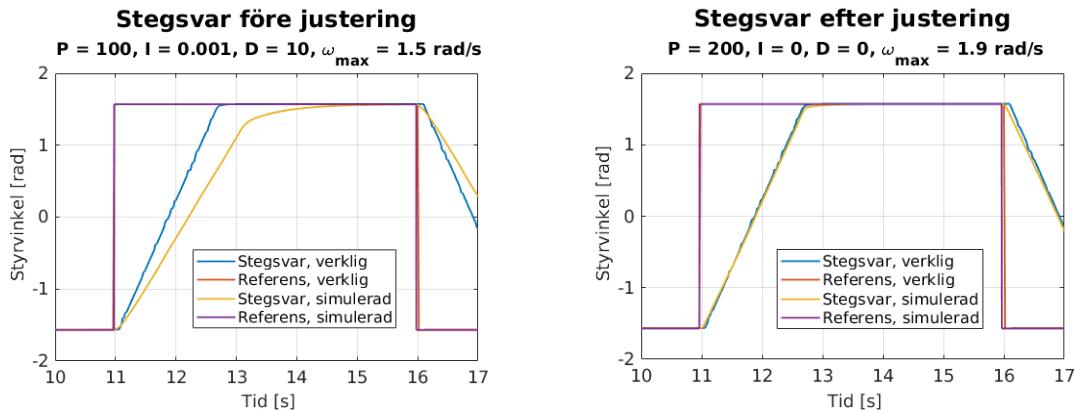
```

1 ...
2 # Needed to make PositionJointInterface work for wheel_steer_joint
3 gazebo_ros_control:
4     pid_gains:
5         wheel_steer_joint: {p: 100.0, i: 0.01, d: 10.0}
6 ...
    
```

3.4.2 Inställning av regulatorparametrar

De regulatorinställningar som behövde göras för styrservots *PostionJointInterface* måste naturligtvis matcha inställningarna som fanns på den fysiska trucken för att simuleringssmodellen skulle efterlikna verkligheten så bra som möjligt. Därför utfördes ett stegsvarsex-

periment på det verkliga systemet så att parametrarna kunde trimmas in. Testprogrammet som beskrivits under avsnitt 3.2.1 återanvändes men tiden mellan vridningarna ökades så att värdena fick tid att stabilisera sig. Sedan spelades både referensvinkel utställd av *Tri-cycleController* och faktisk vinkel på hjulet in. I Figur 11 presenteras resultatet av denna mätning (blå kurva i de båda figurerna).



(a) Stegsvar före justering av PID-parametrar och maximal vinkelhastighet. Det är tydligt att den simulerade rörelsen (gul linje) är mycket långsammare än det verkliga systemet (blå kurva).

(b) Stegsvar efter justering av PID-parametrar och maximal vinkelhastighet.

Figur 11: Tuning av regulatorparametrar samt maximal vinkelhastighet för styrservot

I Figur 11a ser man att stegsvaret i simuleringsmiljön var betydligt långsammare än hos den verkliga trucken innan justeringar utförts. Samtidigt verkade de initialt satta PID-inställningarna vara felaktiga, något som inses framför allt genom att studera krökningen hos den gula kurvan innan den når stationärt värde.

Först gjordes ett försök att snabba upp systemet genom att ändra PID-parametrarna. Resultatet av dessa justeringar var att systemet blev aning snabbare, men dock inte alls så snabbt som det verkliga. Slutsatsen var att någon begränsning på vinkelhastigheten hos styrservot måste varit specificerad någonstans i modellen och/eller regulatorn. Detta visade sig senare stämma. I modellen fanns en maximal hastighet hos servot (eller rättare sagt hos den modellerade leden *wheel_steer_joint*) satt till 1,5 rad/s, vilket också stämde bra överens med lutningen på kurvan vid stegsvarsexperimentet i simuleringen (gul kurva i figuren) som var ungefär 1,5 rad/s.

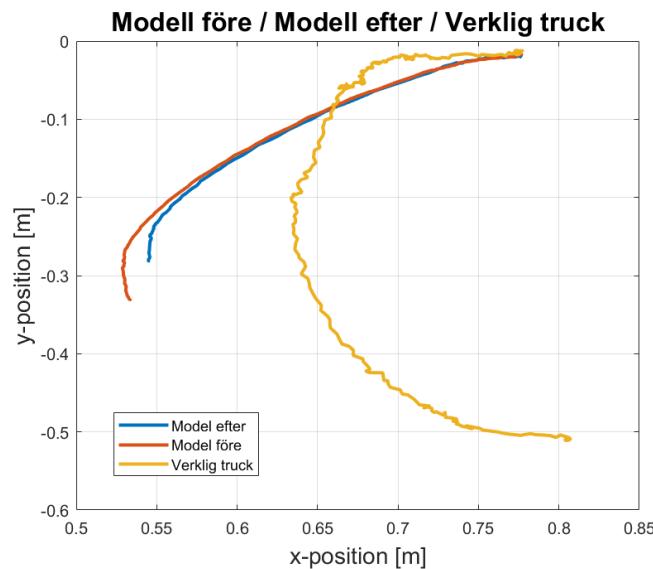
Lutningen på stegsvaret för den verkliga kurvan mättes upp till ca 1.9 rad/s och modellen uppdaterades sedan med denna maxhastighet. Efter att slutligen ha trimmat PID-inställningarna för regulatorn åstadkoms det stegsvar som presenteras i Figur 11b (gul kurva). Man kan där se att simulerat och verkligt stegsvar överlappar varandra till stor del och att inställningarna därför är tillfredsställande. Följande parametrar visade sig lämpliga:

- P-värde: 200
- I-värde: 0
- D-värde: 0
- Maximal vinkelhastighet för styrservot ω_{max} : 1,9 rad/s

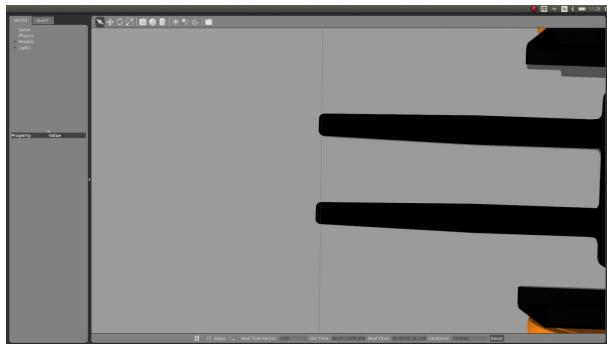
3.5 Resultat

Efter implementering av hjulförskjutningen, med erforderliga inställningar gjorda och validering av modellen genomförd har modellen utökats så att den simulerade trucken beter sig mer likt den verkliga. I Figur 13 och 14 visas hur gaffelpetsarna rör sig i simuleringen när testprogrammet (beskrivet i avsnitt 3.2.1) körs. Här ser man att truckens gafflar rör sig fram och tillbaka under rörelserna samtidigt som de vinkelställs en aning, likt det mönster som tidigare observerats hos det fysiska systemet (jämför med Figur 8 och 9).

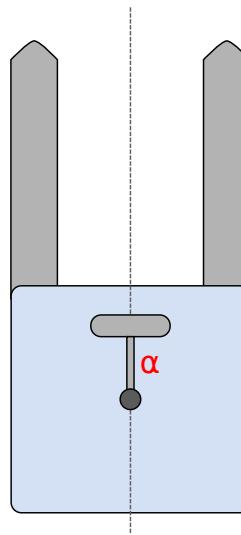
I Figur 12 visas en jämförelse mellan hur väl den simulerade modellens beteende överensstämde med den verkliga truckens, innan respektive efter implementering av hjulförskjutningen.



Figur 12: Graf som visar hur modellen före resp. efter implementeringen av hjulförskjutningen står sig i förhållande till den verkliga trucken.

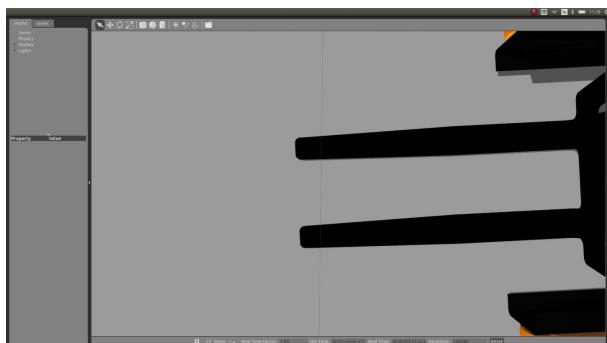


(a) Gaffelspetsarnas position

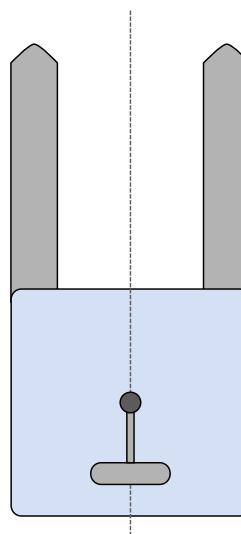


(b) Drivhjulets position

Figur 13: Gaffelspetsarnas position i simuleringen efter att drivhjulet vridits upp till den övre positionen.



(a) Gaffelspetsarnas position



(b) Drivhjulets position

Figur 14: Gaffelspetsarnas position i simuleringen efter att drivhjulet vridits ner till den undre positionen.

3.6 Analys

Resultatet ovan leder till ytterligare diskussioner vad gäller modellens riktighet och dess flexibilitet. Modellen uppvisar visserligen liknande beteendemönster som det fysiska systemet, men det innebär inte att den är fullständig, utan snarare att det fortfarande finns en hel del brister och utvecklingsområden. Samtidigt har den nuvarande modellen möjligheter till enkla modifieringar som tar hänsyn till framtida förekommande faktorer som kan äventyra modellens trovärdighet.

3.6.1 Ändrad effektiv förskjutning vid förslitning av hjul

Den införda parametern för hjulförskjutningen har för närvarande ett värde som utgår från mätningar i ritningar och observationer vid experiment i verkligheten, vilket genererar en bra överensstämmelse mellan de båda. Detta värde passar i dagens läge, men det är viktigt att betona att det troligen inte är fallet i framtiden. Det uppmätta värdet i ritningarna utgår från ett hjul i nyskick och ger därmed en idealiserad bild av ett hjul vid framtida bruk av trucken. Detta gäller eftersom att hjulet slits i takt med att det används, vilket i sin tur innebär att det effektiva värdet av hjulförskjutningen ändras. För att upprätthålla en verlig bild av trucken i simuleringen gäller det därför att man har uppsikt över hjulets förändringar. Därefter kan man göra nödvändiga modifieringar av den införda parametern och därmed ta hänsyn till förslitningar av hjulet. Dessa modifieringar är inte heller svåra att utföra på grund av parameterns lättillgänglighet. Vidare är det av intresse att hålla koll på möjliga förslitningar av hjulet, inte minst för att kunna skildra trucken på ett rättvist sätt i simuleringen, utan även för att hålla koll på truckens prestanda i övrigt.

3.6.2 Fortsatt dålig modell trots införd hjulförskjutning

Som kan ses i Figur 12 var modellens överensstämmelse med verkligheten riktigt dålig innan hjulförskjutningen blivit implementerad. Jämför man sedan modellöverensstämmelsen efter implementationen ser man att modellen har blivit bättre, men endast marginellt. Modellen är alltså fortsatt riktigt bristfällig. Innan projektet hade man uppfattningen att drivhjulets förskjutning var den viktigaste anledningen till att modellen inte uppförde sig som i verkligheten och visst, det har en påverkan men inte alls en särskilt stor sådan. Slutsatsen är att det finns andra brister i modellen som har betydligt större inverkan än just hjulförskjutningen.

Under projektets gång upptäcktes att den maximala vinkelhastigheten för styrservot var satt alldeles för lågt (1,5 rad/s istället för 1,9 rad/s) vilket visade sig vid stegsvarsexperimentet. Detta märktes även vid arbete inom andra delområden; gruppen som arbetade med delområde 3 (reglering av precisionsinkörning) hade trimmat in en regulator med hjälp av simuleringssmiljön och när den sedan testades på den fysiska trucken visade den sig knappt fungera alls därför att trucken rörde sig mycket snabbare i verkligheten än i simuleringen. Detta är ett tecken på att många fler av de parametrar som finns i modellen (maximala hastigheter, accelerationer, jerks...) är felaktigt inställda. En användbar vidareutveckling på detta delområde skulle därför vara att undersöka modellens överensstämmelse med verkligheten ytterligare och försöka åtgärda många av de modellfel som finns. T.ex. skulle stegsvarsexperiment och liknande tester behöva göras för varje aktuator på trucken.

3.7 Vidareutveckling

Det är viktigt att betona modellens möjligheter till utveckling med bakgrund av att det finns så många. Utgångspunkten för detta arbete ligger först och främst i att undersöka modellens brister och begränsningar och framför allt göra mer djupgående undersökning om hur bra modellen faktiskt representerar det fysiska systemet (se avsnitt 3.6.2 ovan). Utan förbättringar på modellen blir det svårt att t.ex. utveckla precisa regulatorer modellbaserat som sedan förväntas fungera lika bra på den verkliga trucken.

4 Delområde 2: Tillståndsmodell

I detta kapitel beskrivs arbetet och resultatet inom projektets andra delområde. Detta arbetsområde har utvecklat en tillståndsmodell som ska kunna användas i en regulator. Tillståndsmodellen hanterar önskad linjär hastighet och önskad rotationshastighet som insignal till systemet och position för trucken i x- och y-led samt truckens rotations som utsignal.

I designspecifikationen [1] beskrevs hur en modell skulle uttryckas med utgångspunkt från fysikaliska storheter hos trucken med olika vinklar som tillstånd hos trucken. Under arbetets gång visade sig detta för svårt att genomföra. Istället utvecklades en modell som beskriver förhållandet mellan de önskade hastigheterna och de uppmätta hastigheterna hos trucken. Denna modell benämns *black box model* eftersom den inte tar hänsyn till hur truckens utseende utan bara ser insignalerna till en ”svart låda” som ger utsignaler. Utsignalerna från den svarta lådan användes sedan för att beräkna positionsförändring mellan varje mätning och uppdatera truckens position.

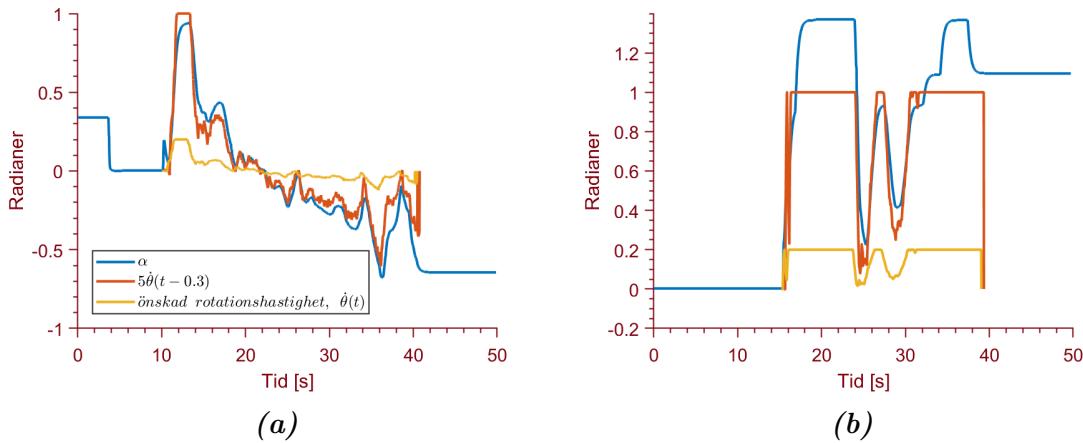
Den resulterande tillståndsmodellen brukar benämñas ”grey box model” eftersom den både består av delar som estimerats utifrån mätdata och delar som utgår från fysikaliska samband. Den framtagna tillståndsmodellen är olinjär och består av flera mindre tillståndsmodeller som sammanfogats. Två linjära modeller används för att beskriva överföringen från önskad linjär hastighet respektive rotationshastighet som insignal till den faktiska linjära hastighet och rotationshastighet som trucken får. Därutöver finns en olinjär delmodell som beskriver den önskade rotationshastighetens påverkan på den linjära hastigheten. Sedan används ett olinjärt samband för att uttrycka truckens position med hjälp av hastigheterna och rotationen.

Efter beskrivningen av hur tillståndsmodellen utvecklats presenteras ett resultat över hur väl tillståndsmodellen efterliknar verkligheten. Därefter följer en analys av resultatet och den framtagna tillståndsmodellen. Kapitlet fortsätter med en jämförelse av trucken i simuleringsprogrammet i Gazebo och trucken i verkligheten för att avgöra om modellen kan användas för reglering även i simuleringsprogrammet. Sist i kapitlet diskuteras möjliga vidareutvecklingar som kan göras på detta delområde.

4.1 Ursprunglig Idé

Den ursprungliga idén som diskuterades i designspecifikationen var att beskriva truckens dynamik utifrån hur de önskade insignalerna överfördes via hjulens läge och position till faktiska positionsändringar för trucken. I designspecifikationen beskrevs en tillståndsmodell med fyra tillstånd, position i x- och y-koordinat i förhållande till en nollpunkt i truckens omgivning, truckens rotation θ i förhållande till denna omgivning och vinkeln α på truckens styrhjul.

Tyvärr visade det sig tidigt i arbetet med att utveckla tillståndsmodellen att vinkeln α på styrhjulet var framräknad utifrån olika parametrar i flera steg. För att beskriva α som en



Figur 15: Vinkeln α mellan drivhjulet och trucken som funktion av rotationshastigheten i två olika fall.

funktion av insignalerna undersöktes några olika funktioner. Den som gav bäst resultat var

$$\alpha(t) = 5\dot{\theta}(t - 0.3)$$

där $\dot{\theta}$ är önskad rotationshastighet. I figur 15 visas funktionen och α för två olika fall. I fall (a) ger funktionen väldigt bra värden på α men i fall (b) ger inte funktionen korrekt värde för stora värden på vinkeln α . I detta fall har xbox-kontrollen använts för att generera styrsignaler. Det är tydligt att det finns en begränsning på hur hög rotationshastighet som xbox-kontrollen ger. I projektet har det inte undersökts hur väl vinkeln α kan beskrivas av den önskade rotationshastigheten då en användare manuellt förser systemet med styrsignaler. Det är möjligt att detta hade gett ett bättre resultat då man hade kunnat ge en maximal önskad rotationshastighet som anpassades efter den maximala vinkeln på styrhjulet.

Även om den resulterande vinkeln α i figur 15 inte upplevs komplex var koden som beräknade vinkeln α komplicerad och därför ansågs det svårt och tidskrävande att beskriva den i en tillståndsmodell. Istället undersöktes möjligheter med förenklade modeller som inte hanterade alla egenskaper hos trucken.

4.2 Förenklad fysikalisk modell med PI-regulator

I processen att konstruera en tillståndsmodell angreps problemet från ett annat håll och ett förslag till lösning utvecklades. Vidare arbete i det här avsnittet övergavs då användbart resultat framställdes snabbare med tillvägagångssättet benämnt ”grey box model” i nästa avsnitt. Fortsatt arbete med den här metoden kan potentiellt ge en bra resulterande tillståndsmodell inom önskade begränsningar, men det är inget gruppen har helt utforskat.

Idéen med den här metoden är att istället för att detaljerat försöka modellera enskilda hjul och enskilda delar så ansätts hela trucken som ett rörligt objekt med en viss linjär

hastighet och rotationshastighet mot en specificerad punkt. Vad som eftersöks i slutändan är en tillståndsmodell för truckens position med hastighet och rotationshastighet som styrsignaler. Med hjälp av välkända fysikaliska postulat för dessa storheter samt en PI-regulator som reglerar felet mellan önskad styrsignal och den aktuella storheten. Det viktiga att ta med ur det här är att regulatorn kan justeras enligt eftertraktat beteende hos trucken för att uppnå liknande stegsvar med avseende på hastighet och vinkelhastighet.

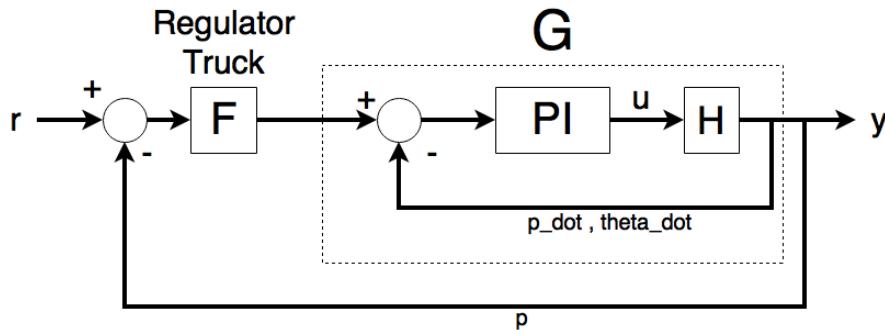
4.2.1 Problemuppställning

Eftersökta tillstånds ansätts i en tillståndsmatris x enligt:

$$x = \begin{bmatrix} x_1 = p \\ x_2 = \dot{p} \\ x_3 = \theta \\ x_4 = \dot{\theta} \end{bmatrix} \Rightarrow \dot{x} = \begin{bmatrix} \dot{x}_1 = \dot{p} = x_2 \\ \dot{x}_2 = \ddot{p} = g(\dots) \\ \dot{x}_3 = \dot{\theta} = x_4 \\ \dot{x}_4 = \ddot{\theta} = h(\dots) \end{bmatrix}$$

Här anger p position, \dot{p} hastighet samt θ och $\dot{\theta}$ vinkel och vinkelhastighet. Vad som där efter sökes är uttryck för acceleration (\dot{x}_2) och vinkelacceleration (\dot{x}_4). I figur 16 visas en överblick av hela systemet där G är huvudsakliga enheten vilket ger önskade tillstånd som sina utfall. Systemet G består av två delar, en PI-regulator och ett förenklat fysiskt system H vars uttryck beskrivs nedan. Det är även två tillstånd (\dot{x}_1 och \dot{x}_3) som återkopplas vilket regulatorn reglerar felet på, och det är de tillstånd vilket ges som önskad insignal till hela systemet G .

Det huvudsakliga målet med en tillståndsmodell och därmed tillståndsåterkoppling är att tillfoga information till truckens regulator och med hjälp av modellbaserad reglering förbättra precisionen och robustheten hos regulatorn. Figuren visar även en återkoppling av position p vilket vill användas av truckens regulator.

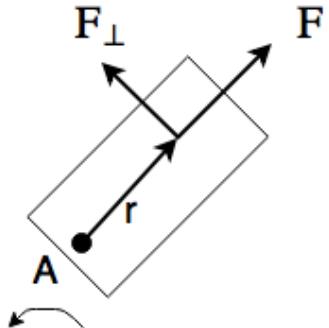


Figur 16: Överblick av systemet G , delsystem samt återkopplade tillstånd.

4.2.2 Fysikalisk ansats

För att hitta uttryck för truckens acceleration samt vinkelacceleration användes välkända fysikaliska postulat tillsammans med ett generellt och grovt uppskattat perspektiv av systemet. Figur 17 visar på en ansats för krafter som påverkar ett godtyckligt objekt i form av ett rätblock. En kraft F i det linjära ledet som påverkar linjära hastigheten

och accelerationen samt en vinkelrät kraft F_{\perp} som i sin tur påverkar vinkelhastighet och vinkelacceleration.



Figur 17: Överblick av krafter som påverkar truckens dynamik.

Här används nu Newtons andra lag samt Eluers ekvation för kraft- och tröghetsmoment.

$$\begin{aligned} \text{Newton II : } & F = ma \\ \Rightarrow \ddot{p} &= \frac{F}{m} - \frac{k}{m}\dot{p} \end{aligned}$$

$$\begin{aligned} \text{Euler II : } & M_A = I\dot{\omega} \\ I\dot{\omega} &= r \cdot F_{\perp} \Rightarrow \ddot{\theta} = \dot{\omega} = \frac{r}{I}F_{\perp} \end{aligned}$$

För att ge en så detaljerad beskrivning som möjligt ansätts en friktionskonstant k i ekvationen med linjär acceleration. I ekvationen för vinkelacceleration kan tröghetsmomentet I uppskattas grovt med formel för ett rätblock med dimensioner likt trucken samt momentarmen r vilket är avståndet som uppmäts från rotationspunkt till drivhjul. Här är truckens massa m .

Det finns fortfarande okända krafter i ekvationerna och det är felet i dessa krafter som PI-regulatorn har som jobb att justera. Det ligger i regulatorns natur att minimera felet i de tillstånd som används att styra systemet, därav behöver inte dimensionen på tillstånd och krafter tas hänsyn till.

4.2.3 PI-regulator

Med de fysikaliska postulaten ovan anges ekvationer för acceleration och vinkelacceleration enligt:

$$\begin{cases} \dot{x}_2 = \ddot{p} = \frac{1}{m}u_1 - \frac{k}{m}x_2 \\ \dot{x}_4 = \ddot{\theta} = \frac{r}{I}u_2 \end{cases}$$

Här utgör u_1 och u_2 styrsignalerna för systemet. Felet (e) i önskad och aktuell hastighet och rotationshastighet regleras av en PI-regulator enligt:

$$u = K_p \cdot e + K_i \int e$$

Därefter kan regulatorparametrarna för både u_1 och u_2 justeras individuellt för att efterlikna inspelade steg hos den riktiga trucken.

4.2.4 Vidareutveckling av tillvägagångssätt

Det nämntes i början av avsnittet att vidare utveckling av det här tillvägagångssättet övergavs när resultat från nästa metod gav bättre resultat snabbare. Regulatorparametrarna har inte trimmats utförligt för att ge önskvärd precision i simuleringar, även om detta är något som anses utförbart om vidare utveckling av metoden skulle önskas.

Vad som beskrivits här är en generell och grov metod för att beskriva de önskade tillstånden. En stor fördel med tillvägagångssättet är att ingen detaljerad information om dynamik på låg nivå, exempelvis vinkel på förskjutning av drivhjulet. Därmed är metoden applicerbar med enbart kördata (hastighet och rotationshastighet) för trucken.

En av de stora nackdelarna är att med krav på kördata blir modellen känslig mot utväntiga förändringar och slitage av trucken. Metoden kräver även arbete med given data för att trimma regulatorparametrar samt andra tillhörande modifieringar för att modellen ska bli tillförlitlig. Det här arbetet är svårt att automatisera och därmed svårt att eventuellt sköta online på trucken. Det är varför nästa tillvägagångssätt som beskrivs med ”Grey box” och MATLAB-applikationerna gav bättre resultat och anses mer användbart för det här projektet.

4.3 Grey Box Model

Modellen består av flera mindre delmodeller som tillsammans bygger upp hela tillståndsmodellen. Dessa mindre delmodeller och hur de tagits fram beskrivs i detta avsnitt.

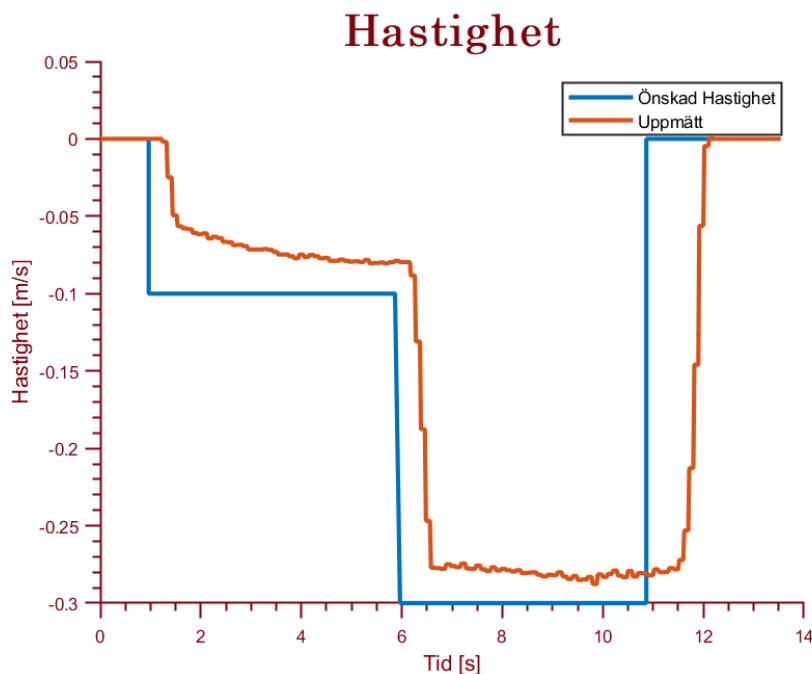
För att bestämma modellen har data från tester som utfördes på den verkliga trucken samlats in och använts. Testerna utfördes med två olika tillvägagångssätt. Den första metoden var att skicka in en sekvens av förutbestämda insignaler under en förutbestämd tid. Denna metod användes för att utföra stegsvarsexperiment och kunna få en uppfattning om truckens dynamik. I det andra tillvägagångssättet användes en xbox-kontroll för att styra trucken. Syftet var att efterlikna truckens verkliga rörelsemönster då den autonomt hämtar pallar i lagret. Data från dessa experiment användes för att verifiera olika modeller för att se hur väl de efterliknade verkligheten. Daten sparades i bag-filer. För att extrahera dessa i Matlab användes Robotics System Toolbox.

Två linjära modeller för att beskriva överföringen från önskade insignaler till faktiska värden på hastighet och rotationshastighet utvecklades. Dessa modeller har bestämts

med System Identification som är en toolbox till Matlab som används för att bygga matematiska modeller av dynamiska system [2]. System Identification användes också för att bestämma en linjär överföring från absolutbeloppet av derivatan av den önskade rotationshastigheten till linjär hastighet.

4.3.1 Hastighet till hastighet

Figur 18 visar resultatet från ett stegsvarsexperiment på den linjära hastigheten. Resultatet visar att det finns en fördräjning i systemet. Flera stegsvar har undersökts och värdet för fördräjningen bestämdes till 0.27 s. Förutom fördräjningen visar stegsvaret även på ett statiskt fel hos systemet. Det statiska felet tycks ligga kring 0.02 m/s oavsett vilken hastighet man önskar.

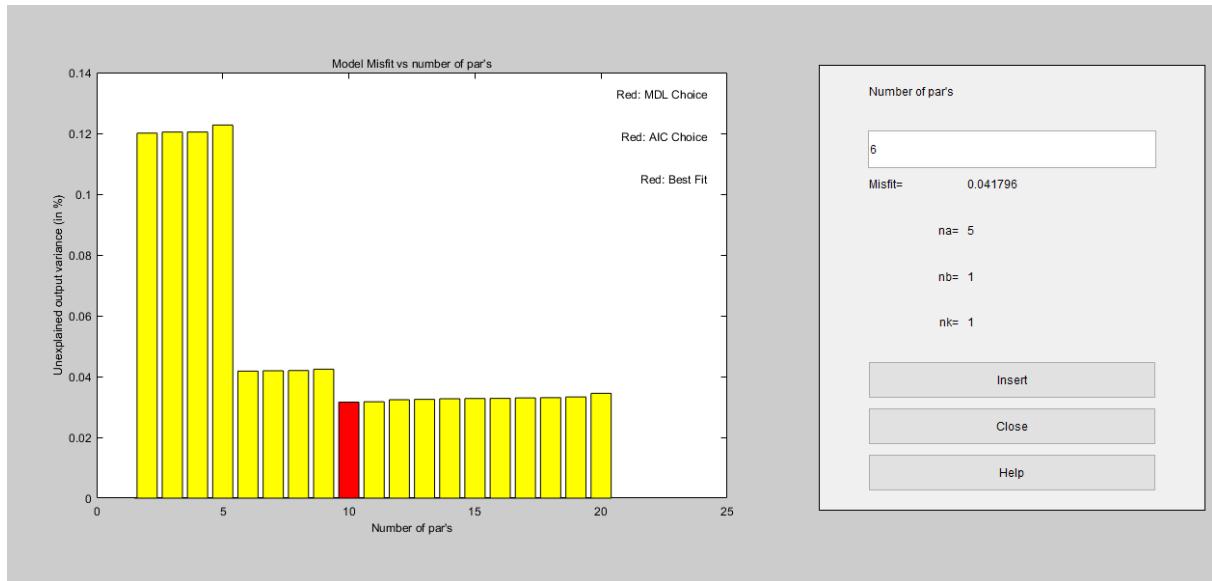


Figur 18: Stegsvar på truckens linjära hastighet.

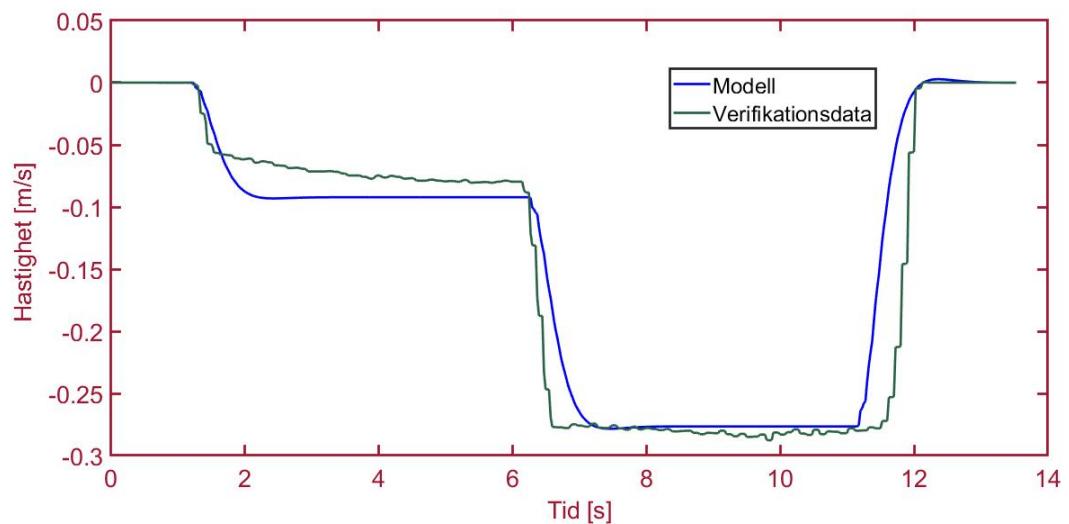
För att avgöra hur många tillstånd som ger en bra tillståndsmodell användes System Identifications metod *Order Estimation* för ARX-modeller. Denna funktion beräknar en lämplig modell av önskat gradtal. Varje modell jämförs genom att undersöka hur stor del av utsignalen som modellen inte beskriver. Resultatet kan ses i figur 19. En riktlinje för val av antal tillstånd är att välja det antal poler som ger ett betydligt mindre fel än modellen av en grad lägre. I vårt fall var det en ARX-modell av grad sex med fem poler och därför valdes antalet tillstånd till fem.

Därefter användes System Identification för att skatta en tillståndsmodell med fem tillstånd. För att validera att denna modell gav ett bra resultat användes data från ett annat stegsvar för att jämföra modellen med verkligheten. Resultatet kan ses i figur 20.

För att avgöra den maximala linjära hastigheten som trucken kan uppnå gjordes ett linjärt steg där den önskade linjära hastigheten var 1.0 m/s och ett där den önskade linjära

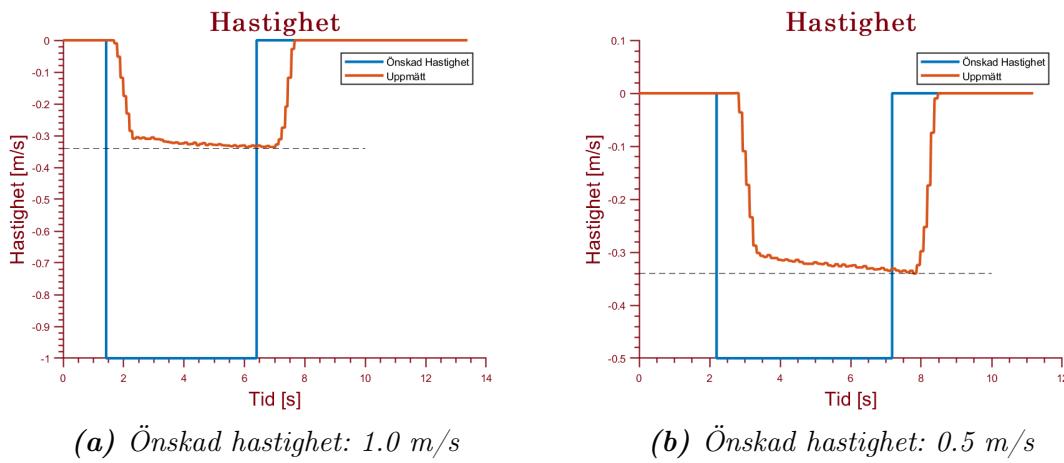


Figur 19: Resultat för uppskattning av grad för en ARX modell på ett stegsvar på den linjära hastigheten. X-axeln är antalet grader hos modellen och y-axeln beskriver hur stor del av utsignalen som inte beskrivs av modellen. Den grad som ger en stor nedgång i stapelhöjd anses lämplig vilket motsvarar en modell av grad sex med fem poler.



Figur 20: Modellerad hastighet jämfört med verifikationsdata för framtagen tillståndsmodell för överföringen från önskad hastighet till verlig hastighet.

hastigheten var 0.5 m/s. Den resulterande hastigheten visas i figur 21. Den maximala hastigheten som uppnåddes i båda fallen var 0.34 m/s.



Figur 21: Stegsvar för att undersöka den maximala linjära hastigheten trucken kan uppnå. Två olika önskade linjära hastigheter undersöktes för att säkerställa att de gav samma värde på den maximala hastigheten. Båda testerna gav att den maximala hastigheten var 0.34 m/s .

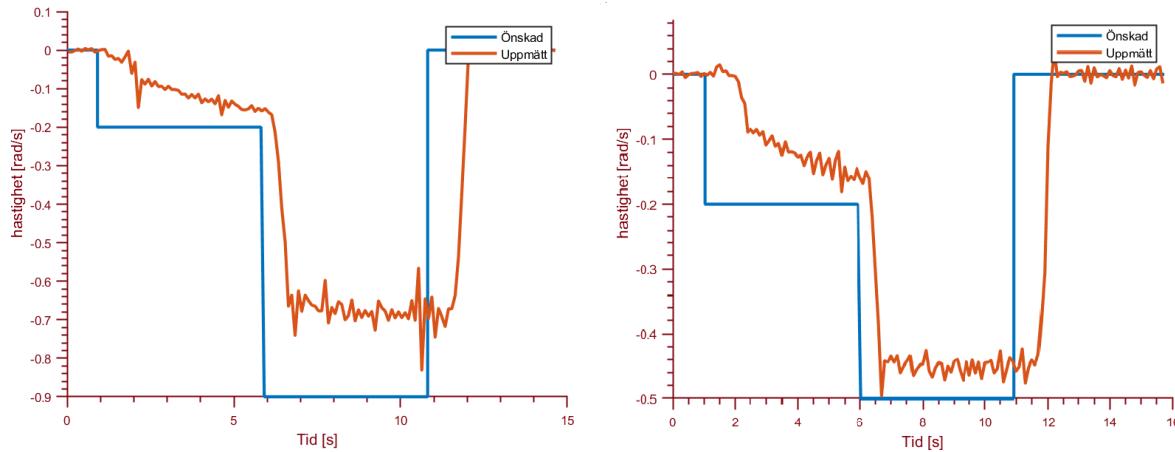
4.3.2 Rotationshastighet till Rotationshastighet

Denna sektion beskriver modellen för den önskade rotationshastighetens påverkan på den uppmätta rotationshastigheten. Precis som i det linjära fallet användes toolboxen System Identification i Matlab för att hitta en bra modell. Innan det kunde göras studerades stegsvarsexperimenten och några intressanta egenskaper hos systemet identifierades som påverkande vilken data som användes till modellidentifieringen.

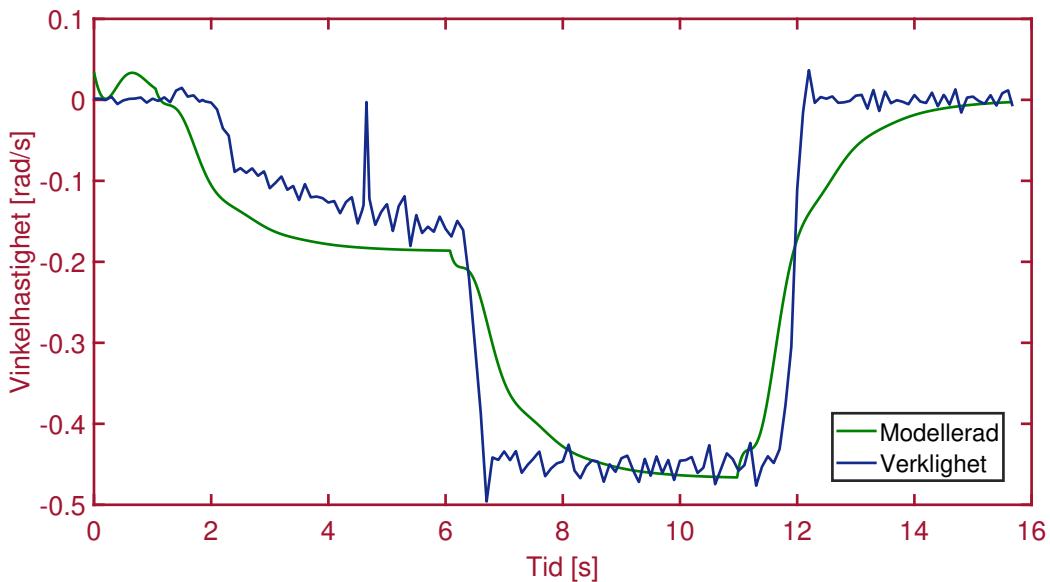
Rotationshastigheten under två steg visas i figur 22. I det vänstra görs ett steg till -0.2 rad/s och sedan till -0.9 rad/s. I det högra är nivåerna istället -0.2 rad/s och -0.5 rad/s. Precis som i fallet med den linjära hastigheten existerar en födröjning och ett statistiskt fel. Födröjningen är densamma som i det linjära fallet, $0,27$ s. Det statiska felet ligger kring $0,05$ rad/s oavsett vilken rotationshastighet som önskas. I det vänstra diagrammet ser man tydligt att detta inte är fallet för andra steget där den önskade rotationshastigheten är $-0,9$ rad/s men den uppmätta rotationshastigheten ligger kring $-0,68$ rad/s. Detta visar att den maximala rotationshastigheten för trucken ligger kring $0,68$ rad/s.

I båda stegsvaren är det tydligt att i det första steget när trucken går från stillastående till -0.2 rad/s är födröjningen betydligt större än i det andra steget. Dessutom är accelerationen upp till den önskade rotationshastigheten inte lika distinkt som i det andra steget. Det är tydligt att trucken beter sig olika beroende på om den tar ett steg från stillastående eller då den redan har en rotationshastighet. I figur 23 visas resultatet då en modell tas fram med båda stegen. Modellen följer inte verkligheten särskilt väl då den försöker anpassa sig till både det långsamma första steget och det snabbare andra steget.

I stegsvaren är det också tydligt att fördräjningen är större då den önskade rotationshastigheten går från nivån med hög rotationshastighet till stillastående. Vad detta beror på är inte säkert men en möjlig förklaring är att i experimenten skickades ingen ny önskad rotationshastighet som angav att den skulle vara noll. Därför gjordes ingen kraftig in-



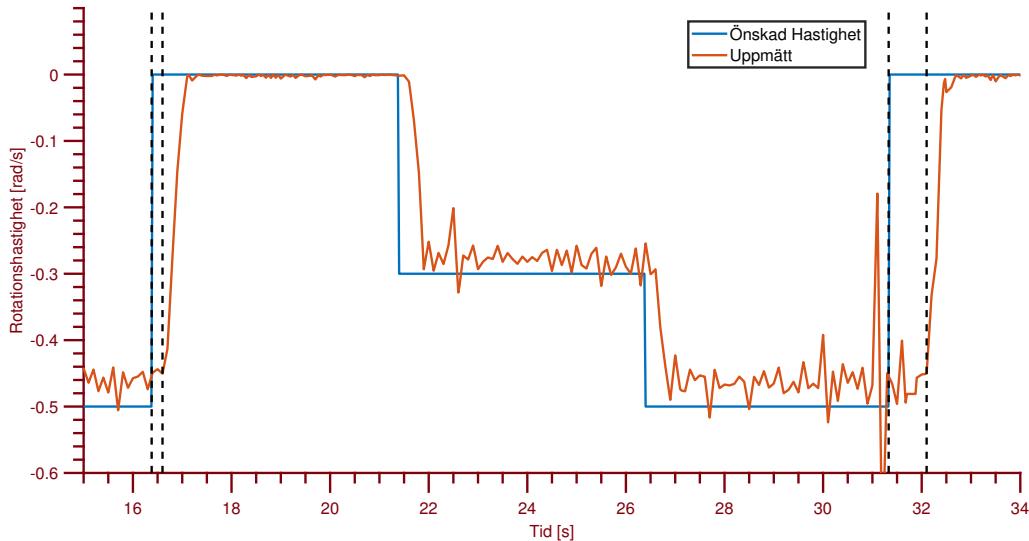
Figur 22: Rotationshastighet då den önskade linjära hastigheten är noll samtidigt som den önskade rotationshastigheten går från en nivå till en annan.



Figur 23: Utsignal från modell jämfört med verlig data då trucken startar i stillastående läge.

bromsning mot noll utan rotationshastigheten fick istället ”dö ut”. För att undersöka om så är fallet gjordes ett nytt stegsvar där en önskad rotationshastighet på 0 rad/s skickas till trucken direkt efter stegsvaret är färdigt för att undersöka. I figur 24 visas resultatet från experimentet. Efter det första steget skickas en önskad rotationshastighet på 0 rad/s men efter det andra steget skickas ingen ny insignal. Resultatet bekräftar att rotationshastigheten sjunker till stillastående snabbare om man skickar en önskad rotationshastighet på 0 rad/s jämfört med att inte skicka någon signal.

I figur 24 finns ytterligare ett intressant resultat. Efter det första steget har gjorts och den önskade rotationshastigheten till trucken är 0 rad/s och ett nytt steg görs så fås ingen fördöjning i stegsvar likt de i figur 22. Så länge trucken får en kontinuerligt flöde av önskade rotationshastigheter så beter den sig som förväntat även om dessa rotationshastigheter



Figur 24: Inbromsning av rotationshastigheten då en önskad rotationshastighet på 0 rad/s (till vänster) respektive ingen rotationshastighet (till höger) skickas till trucken.

är 0. Om man däremot slutar skicka kommandon till trucken får man en svårmodellerad fördräjning då man återigen börjar skicka kommandon.

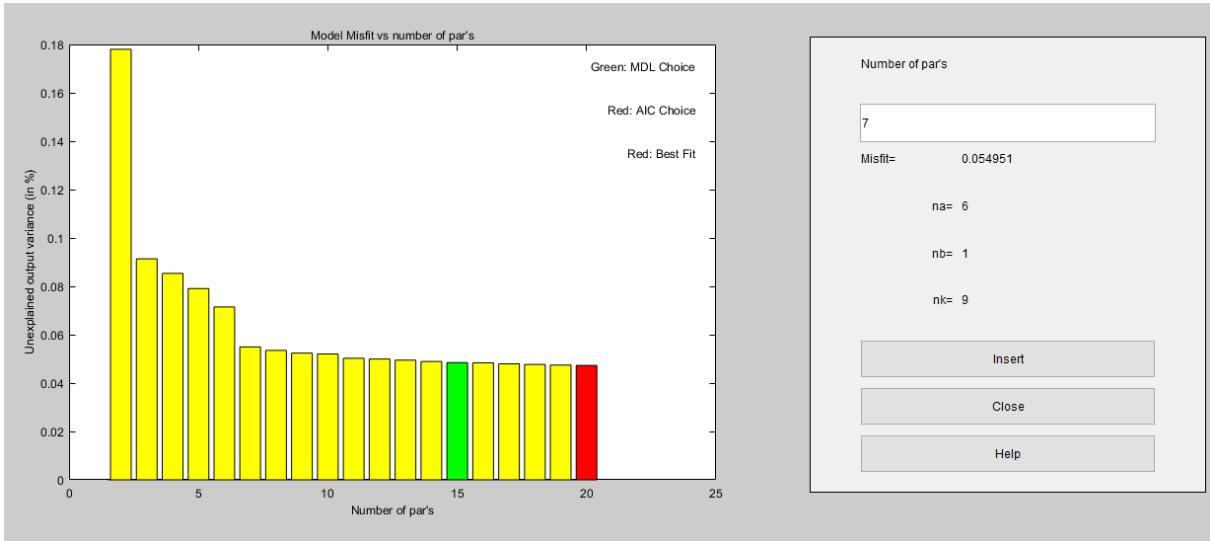
Orsaken till att trucken accelererar långsammare vid uppstart kan bero på att styrhjulet måste roteras för att hamna i rätt vinkel. Det kan också finnas fler komponenter i trucken som måste stå i rätt läge och orsakar en liten fördräjning under tiden de ställs in.

Eftersom trucken kommer röra på sig och få ett kontinuerligt flöde av styrkommandon då modellen används i regulatorn används data som uppfyller detta för att få en modell som är anpassad för reglering.

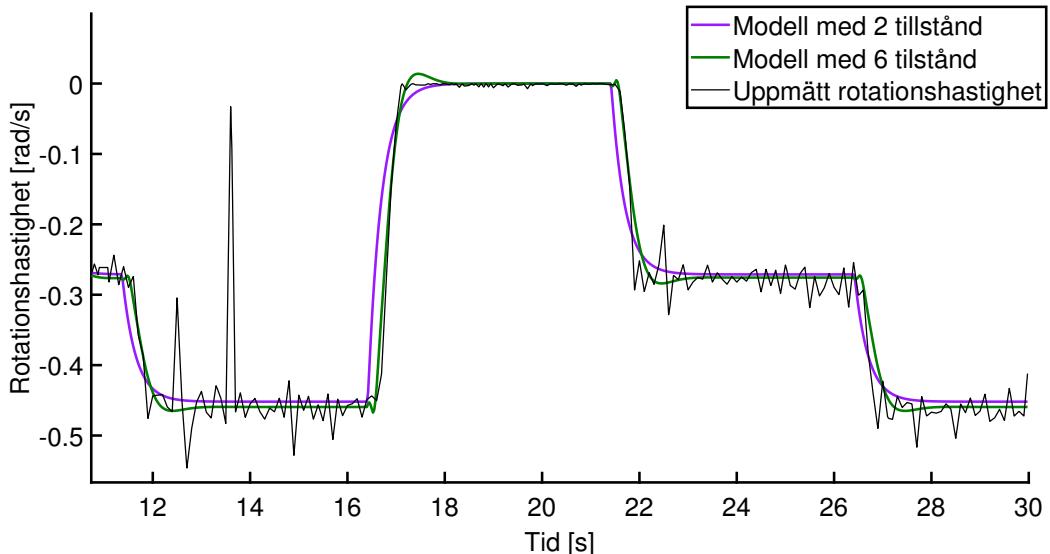
Resultatet från uppskattningen av antalet tillstånd från System Identification visas i figur 25. Denna uppskattning gjordes precis som i det linjära fallet genom att ta fram modeller av olika gradtal och undersöka vilka modeller som ger lägst avvikelse från utsignalen, se sektion 4.3.1 för en ordentlig beskrivning av tillvägagångssättet.

Två modeller ger en kraftig nedgång i stapelhöjd. Det är modellen av ordning tre med två poler och modellen av ordning sju med sex poler. För att avgöra vilken av modellerna som lämpade sig bäst att använda jämfördes resultatet från modellerna med verifikationsdata i figur 26.

I jämförelsen ger modellen med sex tillstånd ett något bättre resultat. Ytterligare en anledning till att välja modellen med sex tillstånd fås om man studerar funktionen *residual analysis* i System Identification. Residualen beräknas för varje tidpunkt som differensen på värdet som modellen förutspår att nästa sampel ska bli och det faktiskt uppmätta värdet. Den första delen av residualanalysen består av att undersöka autocorrelationen mellan residualerna. Denna analys visar om det finns något samband mellan residualerna vid olika tidpunkter. Den andra delen undersöker korskorrelationen mellan insignalen och



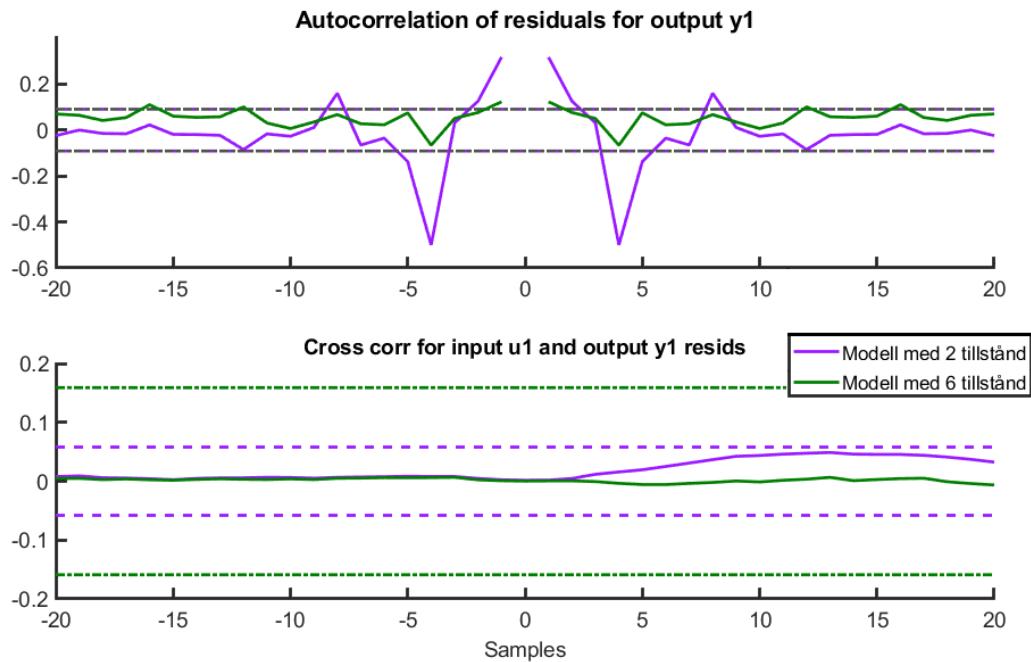
Figur 25: Resultat för uppskattning av grad för en ARX modell för stegsvar på rotationshastigheten.



Figur 26: Jämförelse av hur en modell med 2 respektive 6 tillstånd efterliknar utdata från riktig data.

residualen. Detta visar om det finns något samband mellan insignalen vid en tidpunkt och utsignalen vid en annan tidpunkt som modellen inte tar hänsyn till [3].

Resultatet från residualanalysen i figur 27 visar att det finns en korrelation mellan residualerna med 1 och 4 sampeldifferens för modellen med 2 tillstånd vilket betyder att det finns ett samband mellan felet vid tidpunkt k och felet vid tidpunkt $k - 1$ respektive $k - 4$. Det innebär att det finns ett samband mellan utsignalen från modellens nuvarande värde och tidigare värden som modellen inte utnyttjar. Detta talar emot att använda modellen med 2 tillstånd eftersom modellen ska använda så mycket av den information som finns tillgänglig som möjligt för att beskriva trucken.

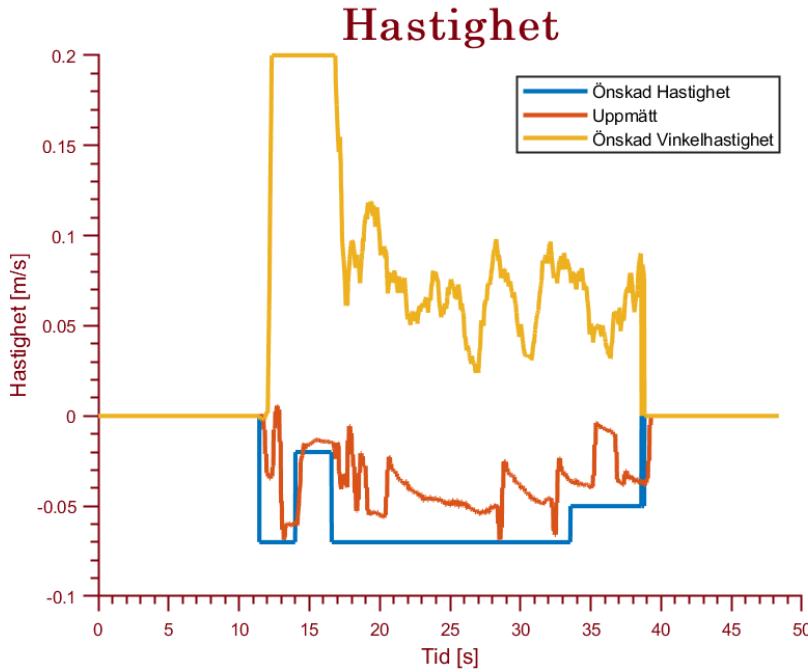


Figur 27: Residualanalys av modeller med 2 respektive 6 tillstånd.

Med denna analys som grund valdes att använda en tillståndsmodell med sex tillstånd eftersom den gav bättre resultat än modellen med två tillstånd och de extra tillstånden är motiverade.

4.3.3 Rotationshastighetens påverkan på hastigheten

I figur 28 visas den uppmätta linjära hastigheten då både den önskade linjära hastigheten och rotationshastigheten varierar. Figuren visar att den önskade rotationshastigheten påverkar vilken linjär hastighet trucken får. Det är rimligt att anta den linjära hastigheten sjunker då trucken svänger på grund av att styrhjulet ändrar vinkel vilket ger en ökad hastighet i rotationsriktningen men minskad hastighet framåt. Då styrhjulet vinklas kan också friktionen mot underlaget öka vilket minskar framdrivningskraften.



Figur 28: Uppmått hastighet då den önskade rotationshastigheten varierar.

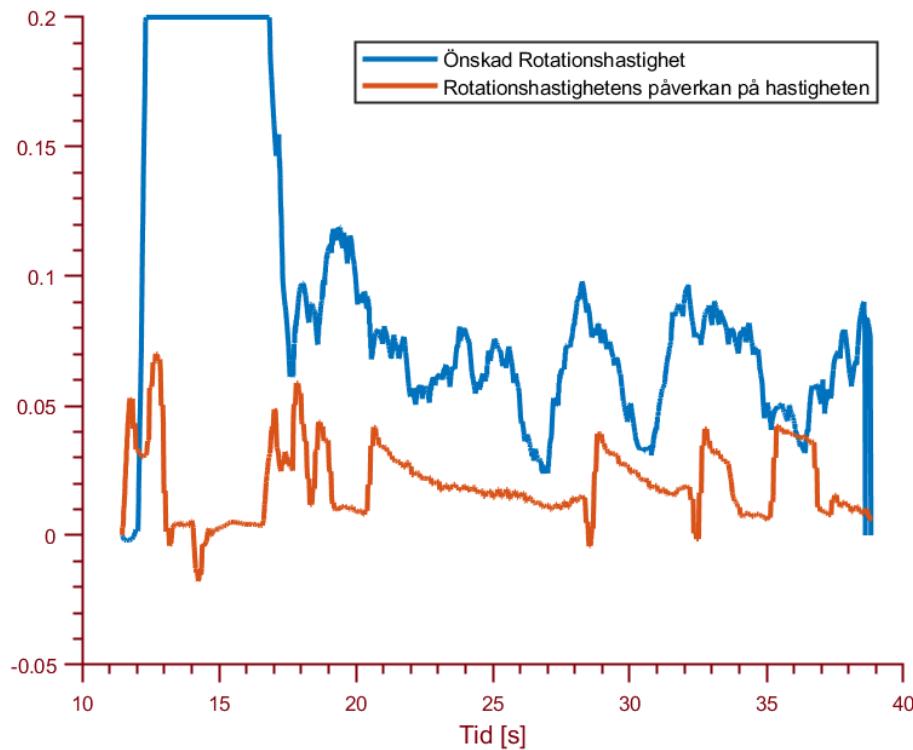
För att undersöka om det går att modellera den önskade rotationshastighetens påverkan på hastigheten användes även här System Identification. Då rotationen antas påverkar den linjära hastigheten likadant oavsett vilket håll rotationen sker har absolutbeloppet av den önskade rotationen använts som insignal.

För att endast undersöka rotationshastighetens påverkan på den linjära hastigheten subtraherades den önskade linjära hastighetens påverkan från den linjära hastigheten. Detta gjordes genom att använda modellen som utvecklats för överföringen mellan önskad linjär hastighet och uppmätt linjär hastighet.

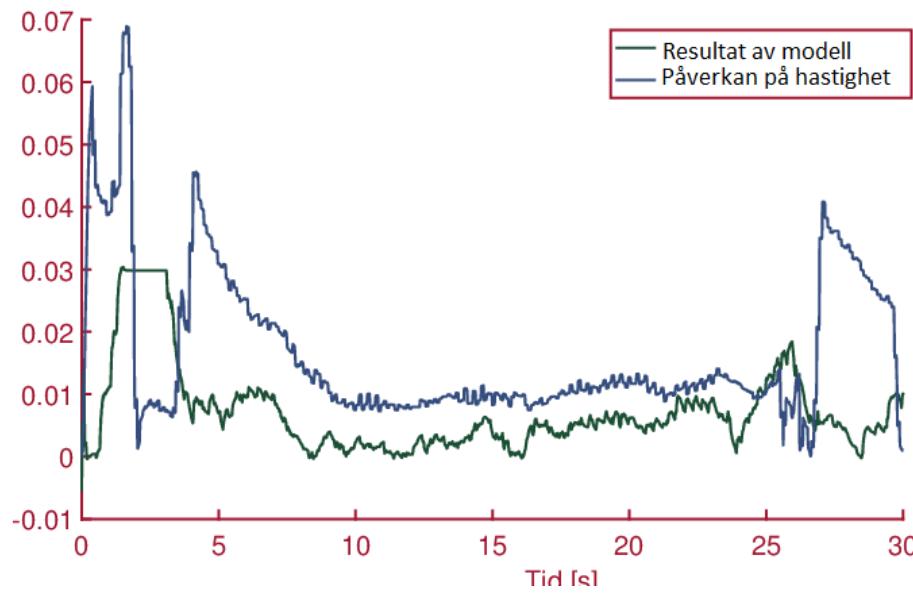
Figur 29 visar rotationshastigheten och dess påverkan på den linjära hastigheten. Det är svårt att se ett tydligt samband mellan linjerna. Då denna data användes i System Identification för att välja en modell och ett annat dataset användes som verifikation erhölls resultatet i figur 30 för en modell med 4 tillstånd. Modellen följer inte verkligheten särskilt väl.

Då figur 29 undersöktes noggrant upptäcktes att påverkan på den linjära hastigheten var som störst då den önskade rotationshastigheten ändrades. För att undersöka om detta kunde användas för att modellera rotationshastighetens påverkan användes absolutbeloppet av derivatan av rotationshastigheten som insignal till systemet. Resultaten för en modell med endast två tillstånd bestämd i System Identification finns i figur 31. Denna modell ger betydligt bättre resultat än då den önskade rotationshastigheten användes som insignal. Topparna i början representeras mycket bra. Tyvärr lyckas inte modellen hantera påverkan på den linjära hastigheten i slutet särskilt väl.

För att avgöra om modellen som utgår från derivatan för att hantera rotationshastighetens påverkan på hastigheten var tillräckligt bra för att användas i modellen över trucken

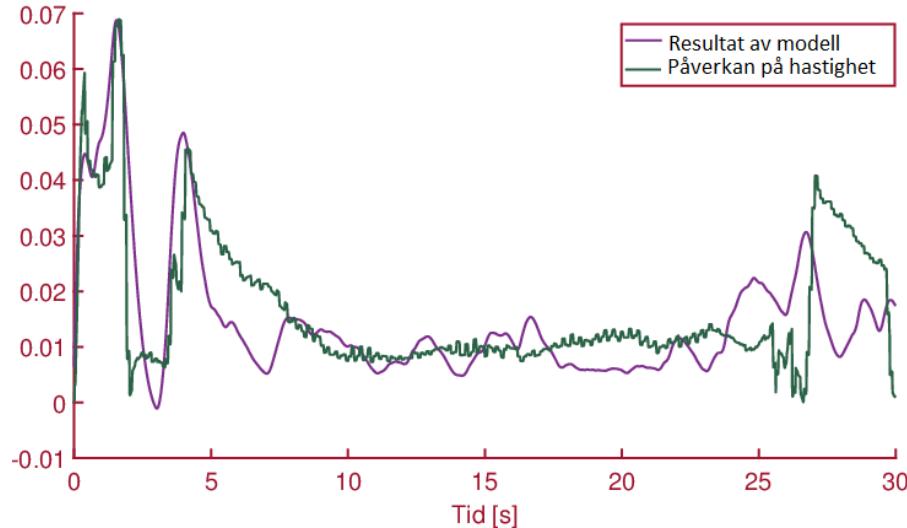


Figur 29: Önskad rotationshastighet och dess påverkan på den linjära hastigheten

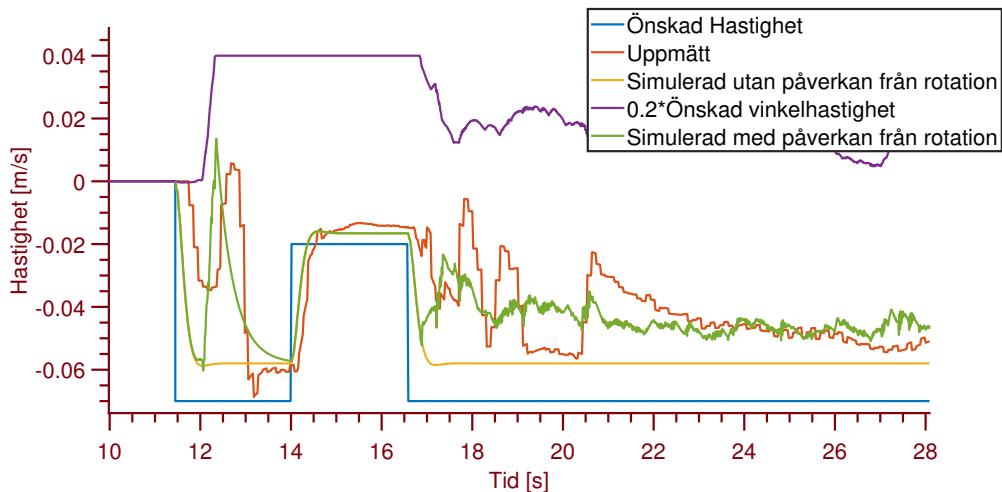


Figur 30: Modell med önskad rotationshastighet som insignal till påverkan på den linjära hastigheten.

gjordes en jämförelse av hastigheten i ett testfall med xbox-kontrollen. Resultatet i figur 32 visar att modellen inte ger ett perfekt resultat men det är ändå mycket bättre än i fallet då modellen inte används. Med detta som underlag beslutades det att inkludera denna modell i tillståndsmodellen.



Figur 31: Resultat då en modell med derivatan av önskad rotationshastighet används som insignal.



Figur 32: Jämförelse av den simulerade linjära hastigheten då modellen för att hantera rotationshastighetens påverkan används respektive inte används.

För att använda derivatan som insignal till den linjära modellen utökades den totala tillståndsmodellen med ytterligare ett tillstånd x_{17} som sparar föregående värde på den önskade rotationshastigheten. Derivatan av rotationshastigheten beräknades genom differensen av nuvarande och föregående värde dividerat med samplingstiden T_s .

$$\dot{\theta} = \frac{u_2 - x_{14}}{T_s}$$

Som insignal till modellen användes sedan absolutbeloppet av derivatan. Totalt resulterar detta i att modellen utökas med tre tillstånd för att hantera hur den önskade rotationshastigheten påverkar den linjära hastigheten.

4.3.4 Från hastighet och rotationshastighet till position

Den hastighet och rotationshastighet som beräknas utifrån tillståndsmodellerna används för att beräkna truckens position i x- och y-led och rotation θ i förhållande till dess omgivning. Rotations beräknas genom att utgå från föregående sampel och addera rotationshastigheten $\dot{\theta}$ multiplicerat med samplingstiden T_s . Positionen beräknas med samma metod med tillägg att hastigheten v multipliceras med cosinus respektive sinus av rotationen. Detta ger modellen:

$$\begin{aligned} x[k+1] &= x[k] + v \cos(\theta)T_s \\ y[k+1] &= y[k] + v \sin(\theta)T_s \\ \theta[k+1] &= \theta[k] + \dot{\theta}T_s \end{aligned}$$

där hastigheten v fås genom

$$v = C_{h2h}x_{3-7} - \text{sgn}(C_{h2h}x_{3-7})C_{r2h}x_{15-16}$$

där C_{h2h} -matrisen beskriver överföringen från tillstånden x_{3-7} och C_{r2h} -matrisen överföringen från tillstånden x_{15-16} till den uppmätta hastigheten. Rotationshastigheten $\dot{\theta}$ fås från överföringen mellan rotationshastighetens in- och utsignal

$$\dot{\theta} = C_{r2r}x_{9-14}$$

4.3.5 Hela modellen

När alla olika delmodeller kombineras till en slutgiltig modell för trucken fås med beträckningarna H2H för överföringen från önskad linjär hastighet till uppmätt linjär hastighet, R2R motsvarande för rotationshastighet och R2H för överföringen rotationshastighet till linjär hastighet, följande modell:

$$x[k+1] = \begin{bmatrix} x_1 \\ x_2 \\ x_{3-7} \\ x_8 \\ x_{9-14} \\ x_{15-16} \\ x_{17} \end{bmatrix} = \begin{bmatrix} \text{x-koordinat} \\ \text{y-koordinat} \\ \text{H2H} \\ \text{Rotation} \\ \text{R2R} \\ \text{R2H} \\ \text{Föregående } u_2 \end{bmatrix} = \begin{bmatrix} x_1 + (v \cdot \cos(x_8))T_s \\ x_2 + (v \cdot \sin(x_8))T_s \\ A_{h2h} \cdot x_{3-8} + B_{h2h} \cdot u_1 \\ x_8 + (C_{r2r} \cdot x_{9-14})T_s \\ A_{r2r} \cdot x_{9-14} + B_{r2r} \cdot u_2 \\ A_{r2h} \cdot x_{15-16} + B_{h2h} \cdot \left| \frac{u_2 - x_{17}}{T_s} \right| \\ u_2 \end{bmatrix}$$

där hastigheten v beräknas enligt

$$v = C_{h2h}x_{3-7} - \text{sgn}(C_{h2h}x_{3-7})C_{r2h}x_{15-16}$$

Utsignalerna beror inte direkt av insignalerna. Därför är D-matrisen (som i detta fall är en skalär) noll för alla deltillståndsmodeller och den har inte skrivits ut. Värden i matriserna för överföringen mellan den önskade linjära hastigheten och den uppmätta linjära hastigheten A_{h2h} , B_{h2h} och C_{h2h} är

$$A_{h2h} = \begin{bmatrix} 0.9666 & 0.0146 & 0.0915 & 0.0529 & -0.0853 \\ 0.0973 & -0.1773 & -0.2583 & 0.2669 & 1.0058 \\ -0.0214 & 0.8898 & -0.3705 & 0.2213 & 0.0245 \\ -0.0308 & -0.3481 & -0.4425 & 0.7186 & -0.3022 \\ 0.0650 & -0.0303 & -0.4597 & -0.3848 & -0.2252 \end{bmatrix} \quad B_{h2h} = \begin{bmatrix} -0.0097 \\ -0.1821 \\ 0.3651 \\ 0.0264 \\ -0.0990 \end{bmatrix}$$

$$C_{h2h} = [1.0819 \ 0.0035 \ 0.0227 \ 0.0082 \ -0.0633]$$

och för överföringen mellan önskad rotationshastighet och uppmätt rotationshastighet:

$$A_{r2r} = \begin{bmatrix} 0.9507 & -0.1719 & 0.0118 & 0.0446 & -0.0852 & 0.0195 \\ 0.1665 & 0.5066 & -0.6134 & 0.1540 & 0.3978 & -0.0873 \\ -0.0143 & 0.4195 & 0.4077 & 0.7883 & -0.1192 & -0.1042 \\ 0.0054 & -0.2340 & 0.5329 & -0.0154 & 0.5147 & -0.2810 \\ 0.0155 & 0.1301 & 0.0670 & -0.1826 & -0.4689 & -1.1561 \\ 0.0518 & 0.1554 & 0.1645 & -0.1700 & -0.0431 & 0.3938 \end{bmatrix} \quad B_{r2r} = \begin{bmatrix} 0.0035 \\ -0.0144 \\ -0.0118 \\ 0.0284 \\ -0.0147 \\ -0.0722 \end{bmatrix}$$

$$C_{r2r} = [1.7526 \ -0.1215 \ 0.0362 \ 0.0213 \ -0.0973 \ 0.0370]$$

och slutligen för absolutbeloppet av derivatan av rotationshastighetens påverkan på den linjära hastigheten:

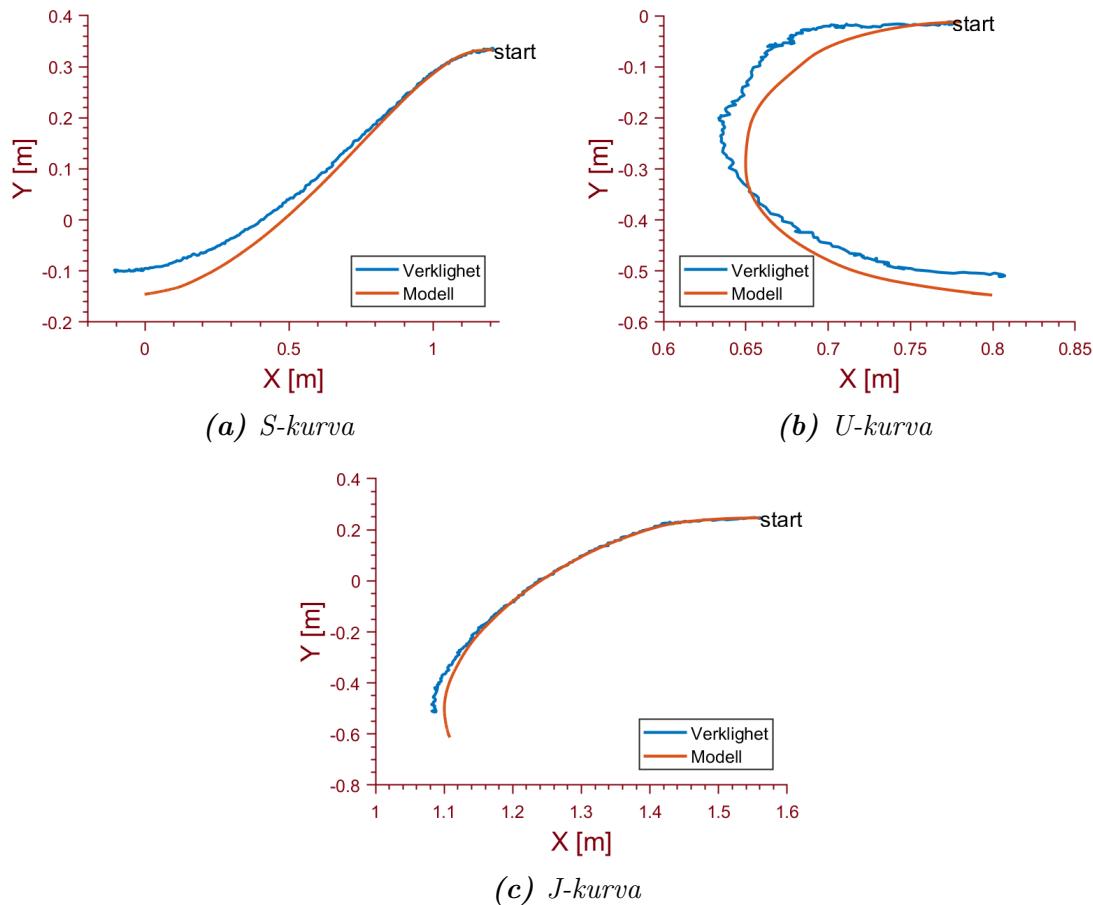
$$A_{r2h} = \begin{bmatrix} 0.9691 & -0.2941 \\ 0.0127 & -0.2568 \end{bmatrix} \quad B_{r2h} = \begin{bmatrix} -95.0116 \\ -399.6000 \end{bmatrix}$$

$$C_{r2h} = [0.00595 \ -0.00139]$$

4.3.6 Finjustering av parametrar

Då positionen i körtesterna med xbox-kontroll jämfördes med positionen från simuleringarna upptäcktes att både den linjära hastigheten och rotationshastigheten blev något för höga. Därför minskas hastigheterna som erhölls från systemidentifiering något genom att multiplicera dem med tal lite mindre än 1. De sluttgiltiga talen i tillståndsmodellen som presenteras i föregående sektion har multiplicerats med dessa tal och är de tal som används till positionsjämförelsen i nästa avsnitt.

Varför hastigheterna blir lägre med samma önskad hastighet i de normala körfallen jämfört med i stegtesterna har inte undersökts. En rimlig förklaring som diskuterades i sektion 4.3.3 är att rotationshastigheten påverkar den linjära hastigheten. Denna påverkan på den linjära hastigheten erhålls inte då ett steg i den linjära hastigheten görs och rotationshastigheten är satt till 0.



Figur 33: Positionen med trucken och med modellen då likadana styrkommandon har användts som insignal. Trucken har styrts med en xbox-kontroll som genererar styrsignaler utifrån hur användaren rör styrspaken.

Tabell 1: Maximala hastigheter för trucken.

Linjär hastighet	0.34 m/s
Rotationshastighet	0.68 rad/s

4.4 Resultat

För att validera att modellen efterliknar trucken användes experiment från den verkliga trucken som bas. Experimenten genomfördes med en xbox-kontroll som användes för att styra trucken. Då xbox-kontrollen användes ändras insignalerna kontinuerligt vilket efterliknar insignalerna då trucken regleras efter en bana. Insignalerna spelades in och användes som insignal till modellen i Simulink där simuleringarna genomfördes. Truckens position i simuleringarna jämfördes med positionen i de experiment som genomfördes i verkligheten. Resultatet kan ses i figur 33.

Värdet på de maximala hastigheterna som trucken kan uppnå ges i tabell 1.

4.5 Analys

I detta avsnitt diskuteras den framtagna modellen. Först jämförs den framtagna modellen med en annan modell med betydligt färre tillstånd. Resultatet visar att det är rimligt att använda den framtagna modellen med 17 tillstånd eftersom modellen med färre tillstånd ger sämre efterlikning av verkligheten. Därefter görs en jämförelse av trucken i simulationsprogrammet Gazebo med trucken i verkligheten. Resultatet visar att det skiljer mycket mellan simuleringen och verkligheten och att modellen därför inte är lämplig att använda i simuleringsprogrammet. Slutligen diskuteras den valda modellen och om den är tillräckligt bra för att använda till en modellbaserad regulator.

4.5.1 Jämförelse med modell av låg ordning

För att avgöra om det är nödvändigt att använda en modell av med 17 tillstånd skapades en modell med enbart sju tillstånd. Tre tillstånd för att beskriva position och rotation samt två tillstånd vardera för att beskriva överföringen mellan önskade hastigheter och uppmätta hastigheter. Modellen tar ingen hänsyn till önskad rotationshastighet påverkar den linjära hastigheten. Modellen ges av

$$x[k+1] = \begin{bmatrix} x_1 \\ x_2 \\ x_{3-4} \\ x_5 \\ x_{6-7} \end{bmatrix} = \begin{bmatrix} \text{x-koordinat} \\ \text{y-koordinat} \\ \text{H2H} \\ \text{Rotation} \\ \text{R2R} \end{bmatrix} = \begin{bmatrix} x_1 + (C_{h2h}x_{3-4} \cdot \cos(x_8))T_s \\ x_2 + (C_{h2h}x_{3-4} \cdot \sin(x_8))T_s \\ A_{h2h} \cdot x_{3-4} + B_{h2h} \cdot u_1 \\ x_5 + (C_{r2r} \cdot x_{6-7})T_s \\ A_{r2r} \cdot x_{6-7} + B_{r2r} \cdot u_2 \end{bmatrix}$$

A-, B- och C-matrisserna för den önskade linjära hastighetens påverkan på den uppmätta linjära hastigheten ges av

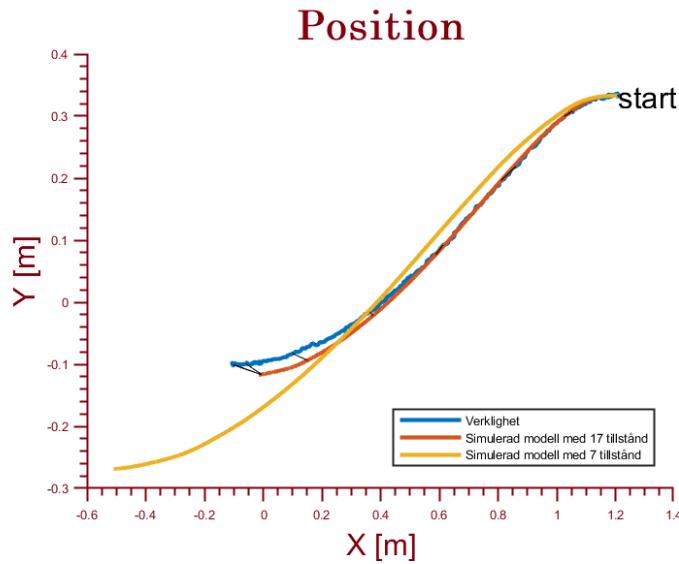
$$A_{h2h} = \begin{bmatrix} 0.9772 & 0.0060 \\ -0.0015 & 0.0666 \end{bmatrix} \quad B_{h2h} = \begin{bmatrix} -0.2351 \\ 37.7167 \end{bmatrix} \quad C_{h2h} = [1.6447 \quad 0.0102]$$

och för den önskade rotationshastighetens påverkan på den uppmätta rotationshastigheten

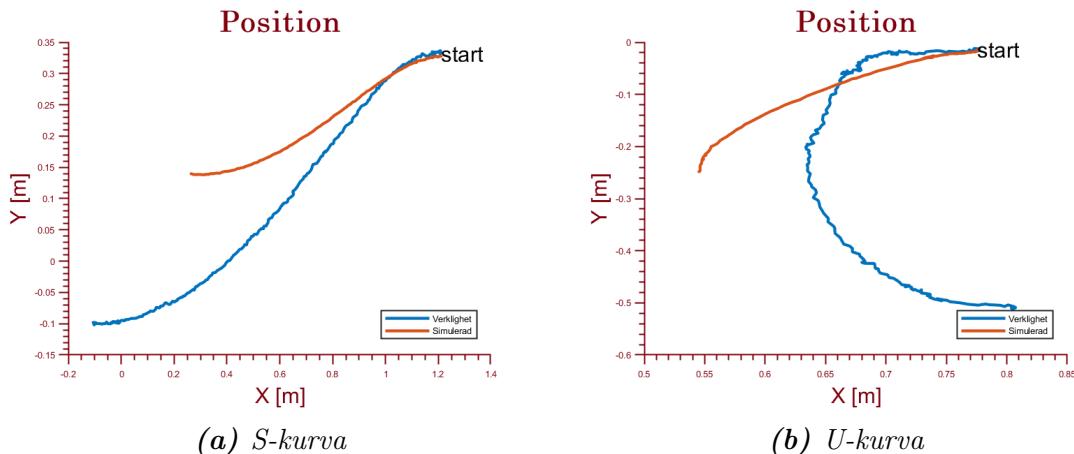
$$A_{r2r} = \begin{bmatrix} 0.9395 & -0.1347 \\ 0.1362 & 0.2757 \end{bmatrix} \quad B_{r2r} = \begin{bmatrix} -0.0895 \\ -0.6343 \end{bmatrix} \quad C_{r2r} = [1.7515 \quad -0.1256]$$

I figur 34 visas resultatet av jämförelsen. Resultatet visar att modellen med många tillstånd är betydligt bättre än modellen med få tillstånd. Det som framförallt skiljer modellerna är att i modellen med få tillstånd åker trucken längre. Huvudorsaken till detta antas vara att ingen hänsyn tas till att den linjära hastigheten minskar då trucken svänger.

Resultatet anses motivera användning av modellen med många tillstånd då detta ger en mycket bättre modell.



Figur 34: Positionen då en modell med 17 tillstånd (röd) och en modell med 7 tillstånd (gul) simulerats. Den blåa kurvan visar positionsmätningar från verkligheten.



Figur 35: Positionen i verkligheten jämfört med i simuleringsprogrammet Gazebo då identisk insignal har använts.

4.5.2 Använda modellen till simulering i Gazebo

All data som används till att bestämma modellen har inhämtats från experiment med den fysiska trucken. Ingen hänsyn har tagits till det simuleringsprogrammet i Gazebo som finns för trucken. För att avgöra om modellen är lämplig att använda även i simuleringsprogrammet undersöktes hur väl simuleringsprogrammet överensstämmer med verkligheten. Samma insignal som användes i verkligheten i två körfall då xbox-kontrollen användes för att styra trucken skickades till trucken i simuleringen. Resultatet i figur 35 visar att simuleringsprogrammet skiljer sig väldigt mycket från verkligheten. I simuleringsprogrammet är trucken mycket långsammare än i verkligheten och kommer därför inte lika långt.

Simuleringen skiljer sig så mycket från verkligheten att den framtagna tillståndsmodellen inte anses lämplig att använda för att beskriva den simulerade trucken. För att kunna använda en modell i simuleringsprogrammet skulle den behöva anpassas efter data från

simuleringen. Huvudorsaken till att simuleringen skiljer sig så mycket från verkligheten antas vara hur trucken är implementerad i simuleringsprogrammet. Det finns många olika parametrar som styr vilka maxhastigheter olika länkar och ledar kan ha. Många av dessa värden har förmodligen uppskattats utan några undersökningar över dess egentliga värden. En mer utförlig beskrivning av begränsningarna med simuleringens efterlikning av verkligheten finns i kapitel 5.

4.6 Slutsats

Modellen med 17 tillstånd som presenterades i avsnitt 4.3.5 anses vara lämpligast att använda för en regulator. De resultat som presenteras i avsnitt 4.4 visar att positionen för modellen i simulering efterliknar positionen från verkligheten mycket väl. Därför är modellen lämplig att använda i en regulator med framkoppling som utnyttjar modellen.

4.7 Vidareutveckling

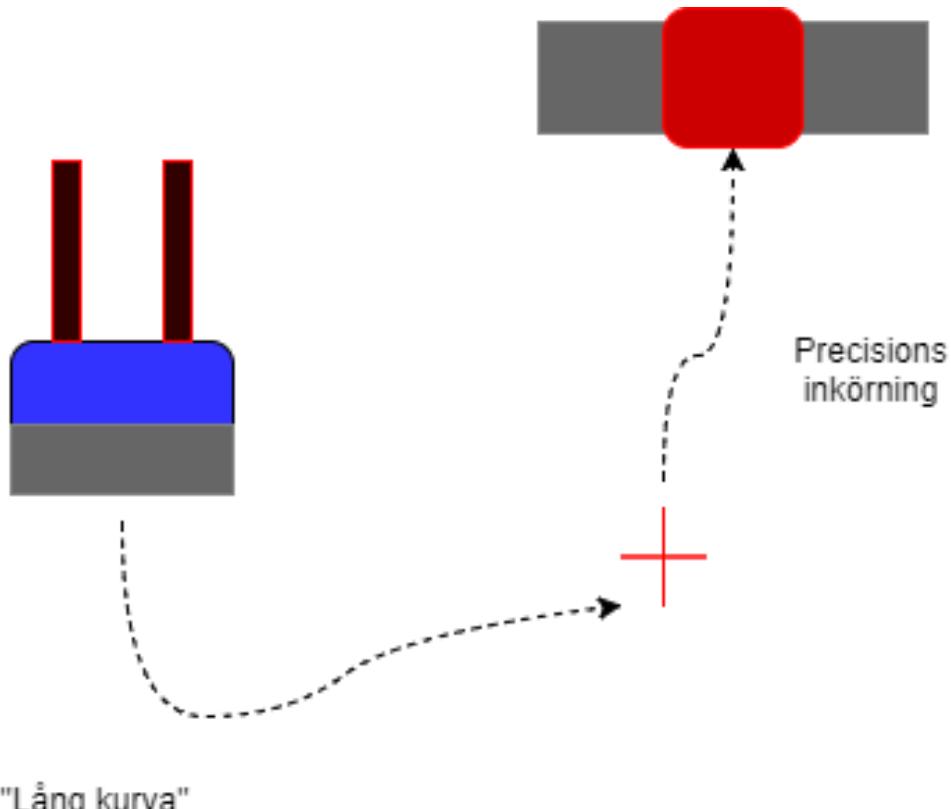
Resultatet då modellen jämfördes med verkligheten är lovande. Modellen efterliknar verkligheten väl och borde således kunna användas för att förbättra regleringen. Framförallt skulle en framkoppling kunna införas i regulatorn som styr precisionsinkörningen som är beskriven i kapitel 5.

Även en modellbaserad prediktions-reglering, *MPC*, hade varit intressant att implementera för att undersöka om den ger en bra reglering. Då denna metod även planerar en trajektoria för trucken fram till pallen skulle den kunna ersätta både nuvarande ruttplanering och regulatorn. Då modellen är olinjär blir tyvärr teorin för att implementera MPC mycket krävande. Därför är det tveksamt om det är möjligt att genomföra i framtida projekt under samma utformning som detta.

Modellen är inte perfekt och arbete med att förbättra denna kan såklart göras. Däremot är det förmodligen svårt att förbättra modellen så mycket att den i slutänden ger upphov till en bättre reglering. Därför skulle en vidareutvecklingen av modellen inte ge så mycket resultat i slutändan.

5 Delområde 3: Precisionsinkörning

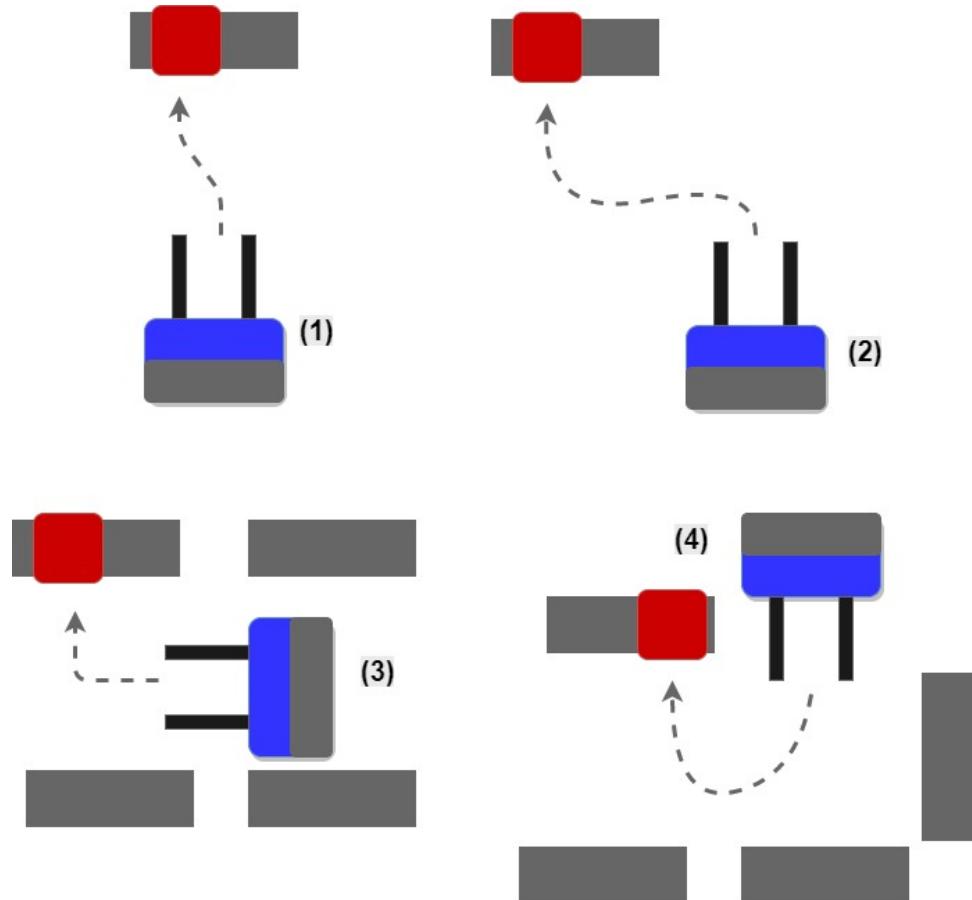
Trucken har som mål att kunna röra sig runt i en lokal och plocka upp och ställa ner pallar på valfri plats. För att lyckas med detta har trucken två banläggare som lägger ut banor beroende på pallens position och omgivningen. Den ena banan, här kallad långa banan används för att trucken ska kunna navigera runt i rummet relativt snabbt. Den andra banan, här kallad precisionsinkörningsbanan används så att trucken kan positionera sig rakt framför en pall innan upphämtning.



Figur 36: Bilden visar precisionsinkörningen som alltså är reglerproblemet

För att trucken ska kunna följa dessa banor har två regulatorer utvecklats. Precisionsinkörningen innebär att trucken kör med gafflarna framåt vilket gör att oscillationer förstärks och därför gör det väldigt svårt att reglera. Precisionskraven är också mycket hårdare då ett maximalt positionsfel på 2.5 cm är tillåtet längst ute på gafflarna. Under detta projekt har regulatorn som följer precisionsinkörningsbanan utvecklats för att bli snabbare och mer robust. Samtidigt som självaste banplaneringen ändrats för att bli snabbare och kunna klara skarpa kurvor mer robust.

I bilden nedan visas fyra exempel på situationer som det önskas att den färdiga trucken ska klara att köra in på. Innan projektet klarade trucken kurva 1 i de flesta fallen, kurva 2,3 och 4 klarade den inte alls att köra in på. Efter projektet klarar trucken fallen 1-3 alltid i simulatorn. Då man valde att ändra kurvplaneringen från en S-kruva till en J-kurva med uppvidring från stillastående i början ser inte kurva 4 likadan ut. Istället för en Usväng gör trucken en skarp J-kurva detta fungerar därför i de flesta fall så länge det finns tillräckligt med utrymme mellan truck och pall för att trucken ska kunna passera. Om man absolut vill att trucken ska köra som ett U går det dock enkelt att ändra på genom att göra ett specialfall där trucken för de fall då vinkeln mellan start- och slutposition är ca 180 grader göra en S-kurva igen(genom att ändra parameter shape_start). J-kurvan innebär också att kurva 2 löses på samma sätt som kurva 3 förutom att trucken vrider 90 grader i början.



Figur 37: Målkurvor som trucken ska kunna plocka upp en pall ifrån

5.1 PD-regulator

Innan detta projekt användes en statisk hastighet och en PD-regulator för att styra truckens vinkelhastighet. Denna regulatorkonfiguration gjorde att banan kunde följas relativt bra men väldigt långsamt. Regulatorn var inte tillräckligt robust och hade svårt att klara vissa fall.

5.2 Hastighetsreglering

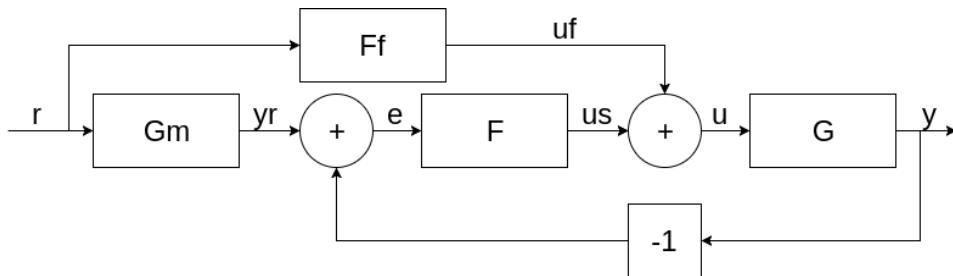
För att minska tiden det tar för regulatorn att genomföra banan implementerades en dynamisk hastighetsregulator. Denna regulator reglerar hastigheten genom att titta på de senaste 10 samplingarna för positionen samt vinkelns och se hur mycket fel dessa var i förhållande till banan. Var dessa fel stora så minskas hastigheten mycket för att regulatorn ska ha tid att justera felet. Regulatorn tittar även på tre punkter på banan 5, 10 och 15 samplingar längre fram. Om det skiljer mycket i vinkel mellan dessa punkter och truckens vinkel så minskas hastigheten. Se ekvation 1

$$v = v_{max} - \sum_{n=-10}^0 |e[n]| - \sum_{m=1}^3 |\phi[5m]| \quad (1)$$

där $e[x]$ är positions- och vinkelfelet och $\phi[x]$ är vinkelfelet mellan trucken och punkten x samplingar fram.

5.3 PD-regulator med framkoppling

För att slippa överställa regulatorn eller ställa ner hastigheten allt för mycket bör en framkoppling implementeras. Detta för att ge trucken information om hur banan ser ut framför sig och därmed kunna öka robusthet och stabilitet hos banföljningen.



Figur 38: Regulatorstruktur för regulator med återkoppling och framkoppling.

5.3.1 Framkoppla ett steg

Första och enklaste implementeringen för att framkoppla regulatorn är att räkna ut kurvaturen hos kurvan på en lämplig punkt längre fram, multiplicera den med truckens nuvarande hastighet och addera denna till styrsignalen. Punkten väljs genom att välja den punkt som ligger sträckan $\tau_d * v$ från trucken. Där τ_d är tidsfördröjningen för vinkelhastighetsförändringen och v är truckens hastighet. Kurvans kurvatur räknas ut på följande sätt

$$\kappa = \frac{\frac{P_{3y}-P_{2y}}{P_{3x}-P_{2x}} - \frac{P_{2y}-P_{1y}}{P_{2x}-P_{1x}}}{\sqrt{(P_{2x}-P_{1x})^2 + (P_{2y}-P_{1y})^2}} \quad (2)$$

Där P är en punkt och P_{nu} är sträckan mellan origo och punkten i u koordinaten med numrering n . Stigande numrering innebär närmare slutet på kurvan.

Problemet med att bara använda ren kurvatur är att man inte fångar systemets dynamik vilket gör det svårt att följa banor ordentligt vilket har indikerats vid lättare testning.

5.3.2 Framkoppla med en första ordningens dynamisk modell

För att försöka fånga en del av systemets dynamik kan man driva framkopplingens del av styrsignalen genom inversen av den uppskattade dynamiska modellen för systemet. Denna modell representeras av blocket F_f i figuren 38. Modellen kommer ha följande utseende

$$F_f = \frac{1 + Ts}{K} * \frac{K_f}{(1 + T_f s)}. \quad (3)$$

Där K och T är systemets uppskattade statiska förstärkning och tidskonstant. K_f och T_f är designparametrar.

Man kan även driva referenssignalen genom en förfiltrering för att matcha referenssignalen med systemets beteende och därmed få en lugnare regulator. Blocket G_m i figur 38 kommer då ha samma utseende som den uppskattade modellen för systemet

$$G_m = \frac{K}{1 + Ts}. \quad (4)$$

5.3.3 Framkoppla med en högre ordningens dynamisk modell

Med den modell som togs fram i delområde 2 får ytterligare information om hur trucken faktiskt kommer bete sig när man vill få ut en viss vinkelhastighet. Det första ordningssystemet som användes ovan kan därför ersättas med modellen som togs fram i delområde 2. Detta görs genom att ersätta G_m med den modell som beskrivs i 4.3.5.

$$F_f = \frac{1}{G_m}. \quad (5)$$

Likt i implementeringen av första ordningenssystemet är lösningen inte alltid proper och därav inte realisbar. Man måste därför se till att gradtalet på täljaren inte är större än nämnaren. Detta kan göras likt tidigare genom att skala F_f med $\frac{K_f}{(1+T_f s)^N}$ och sätta N till gradtalet för modellen G_m . Blir det för beräkningskrävande med den högre ordningens dynamiska modell som framtagits i delområde 2 hänvisas till stycket nedan om dimensionsreducering. Stycket om framkoppling är skrivit enligt [6].

5.4 Dimensionsreducering

Om det under implementering av framkoppling, MPC eller annan modellbaserad regler-teknik uppstår problem med beräkningskapacitet kan detta bero på att den framtagna modellen i delområde 2 har ett stort antal tillstånd. Ett alternativ då är dimensionsreducering, d.v.s. att minska antalet tillstånd men ändå behålla majoriteten av den information som finns i modellen. Detta kan göras på flera olika sätt, en möjlighet är att använda matlab-funktionen balreal för att beräkna en balanserad tillståndsrealisering av modellen och sedan använda matlab-funktionen modred för att ta bort de tillstånd som ger minimal påverkan för systemet. Mindre tillstånd betyder färre beräkningar, vilket i sin tur betyder lägre krav på beräkningskapacitet. Observera dock att detta endast går att göra på de linjära delarna av modellen det vill säga de som ger tillstånden x_{3-17} t.ex. önskad hastighet till faktisk hastighet.

5.5 MPC-regulator

MPC eller model predictive control är likt framkoppling en modellbaserad reglermetod. Denna metod skulle kunna vara en bra ersättning till regulatorn med återkoppling och framkoppling. En av de främsta styrkorna med MPC är att man kan ta hänsyn till bivillkor under regleringen. Detta gör att man skulle kunna implementera saker som höjning av gaffel och utskjutning av torn som bivillkor som alltså optimeras som en del i banföljningen.

Eftersom banföljningen kommer ske i realtid är det extra viktigt att MPC-regulatorn som senare implementeras är beräkningseffektiv och kan hantera korta samplingstider. Eftersom positionen man får ut beror på vinkeln på styrhjulet leder detta till en olinjäritet vilket gör det svårt att använda en linjär MPC för positionsreglering.

Väljer man istället en olinjär MPC är optimeringsproblemet icke konvext och man kan därför inte använda en kvadratisk lösare. Detta gör MPC-designen mycket mer tidskrävande, vilket alltså helst bör undvikas. Anledningen är att man måste iterera igenom flera lösningar och kolla vilken rutt som ger bäst resultat.

5.6 Linjär MPC-regulator

Eftersom den dynamiska modellen från önskad hastighet och vinkelhastighet till position är olinjär, medan en linjär MPC kräver en linjär modell är en möjlighet linjärisering. En linjärisering av den olinjära modellen skulle kunna göras kring några arbetspunkter beroende på drivhjulets vinkel. På så sätt fås några linjära modeller som kan användas vid utvecklingen av en linjära MPC regulator för varje arbetsområde.

Målfunktionen för en grundläggande MPC är:

$$\sum_{j=0}^{N-1} \|z(k+j)\|_{Q_1}^2 + \|u(k+j)\|_{Q_2}^2 = /linjrmmpc/ = X^T \mathcal{M}^T Q_1 \mathcal{M} X + U^T Q_2 U \quad (6)$$

$$= (\mathcal{F}x(k) + \mathcal{G}U)^T \mathcal{M}^T Q_1 \mathcal{M} (\mathcal{F}x(k) + \mathcal{G}U) + U^T Q_2 U$$

I ekvation 6 är Q_1 och Q_2 diagonala viktmatriser där relationen mellan dessa bestämmer om regulatorn ska lägga mest vikt vid att använda små styrsignaler, stort Q_2 , eller ha ett litet reglerfel, Q_1 . Q_1 och Q_2 är blockdiagonala matriser med Q_1 och Q_2 repeterade N gånger. \mathcal{M} är en blockdiagonal matris med M repeterad N gånger.

Målfunktionen för MPC-reglering kan sedan utvecklas med integralverkan och skrivs då enligt ekvation 7.

$$\sum_{j=0}^{N-1} \|z(k+j) - r(k+j)\|_{Q_1}^2 + \|u(k+j) - u(k+j-1)\|_{Q_3}^2 \quad (7)$$

Prestandamåttet kan därför skrivas som i ekvation 8 ur vilket man kan härleda QP-problemet då integralverkan och referensföljning används. Det man bör tänka på vid härledningen av QP-problemet är att det endast är de termer som beror på U som är intressanta då de andra är konstanta.

Omskrivningen av målfunktion med integralverkan för linjär MPC:

$$(\mathcal{M}(\mathcal{F}x(k) + \mathcal{G}U) - R)^T \mathcal{Q}_1 (\mathcal{M}(\mathcal{F}x(k) + \mathcal{G}U) - R) + (\Omega U - \delta)^T \mathcal{Q}_3 (\Omega U - \delta) =>$$

/delar som är beroende av U/ => $U^T \mathcal{G}^T \mathcal{M}^T \mathcal{Q}_1 \mathcal{M} \mathcal{F}x(k) - U^T \mathcal{G}^T \mathcal{M}^T \mathcal{Q}_1 R +$

$$(x(k))^T \mathcal{F}^T \mathcal{M}^T \mathcal{Q}_1 \mathcal{M} \mathcal{G}U - R^T \mathcal{Q}_1 \mathcal{M} \mathcal{G}U - U^T \Omega^T \mathcal{Q}_3 \delta - \delta^T \mathcal{Q}_3 \Omega U + U^T \mathcal{G}^T \mathcal{M}^T \mathcal{Q}_1 \mathcal{M} \mathcal{G}U + U^T \Omega^T \mathcal{Q}_3 \Omega U$$

Nedanför ser man ekvationen för det problem som regulatorn löser för varje samplingsögonblick. Denna gäller såväl för olinjär som för linjär MPC. Om en linjär MPC ska göras implementerar man ovanstående omskrivning av målfunktionen. Om MPC:n skulle bli allt för beräkningskrävande kan man antingen minska prediktionshorisonten eller placera ut en mindre antal punkter längs kurvan att reglera efter.

$$\min_U \text{målfunktion}(x_0, u) \quad (8)$$

$$h(x_0, u) = 0 \quad (9)$$

Där x_0 är initial tillstånd, u insignal och h är valfritt bivillkor. Man kan lägga in så många bivillkor man vill så länge $h(x_0, u) = 0$ är uppfyllt. För att sedan lösa minimeringsproblemet finns för det linjära fallet en funktion quadprog i Matlab. Motsvarighet till dessa eller liknande funktioner finns antagligen också i Python eller C++. Ekvationerna ovan är också härladda ur kurskompendiet för industriell reglerteknik[6] där mer information kan hittas.

5.7 Olinjär MPC-regulator

Om man väljer en olinjär MPC är ett bra sätt att implementera denna genom att använda modellen från önskad hastighet och önskad vinkelhastighet till faktisk position. Detta gör att man kan minimera positionsfelet mellan den position man enligt modellen hamnar i och den position man faktiskt bör vara i vid den punkten.

Eftersom valet av en olinjär MPC medför att optimeringsproblemet blir icke konvext behöver man fundera extra mycket på hur tidskrävande regulatorn blir. Man måste iterera igenom flera lösningar och kolla vilken rutt som ger bäst resultat. Att den framtagna dynamiska modellen har så många tillstånd kan därför bli ett problem. Om det blir problematiskt på grund av alla tillstånd är en möjlighet att kolla på dimensionsreducering för att ändå bevara den mest signifikanta informationen i modellen.

Om man väljer att implementera en olinjär MPC görs detta enligt nedan.

Målfunktionen för en olinjär MPC kan sättas lite som man vill men det påverkar hur lätt problemet är att lösa, ett rimligt alternativ är dock:

$$\sum_{j=0}^{N-1} \|z(k+j)\|_{Q_1}^2 + \|u(k+j)\|_{Q_2}^2 \quad (10)$$

Dock kan inte omskrivningen till ett kvadratiskt problem göras som för det linjära fallet.

I ekvation 10 är Q_1 och Q_2 diagonala viktmatriser där relationen mellan dessa bestämmer om regulatorn ska lägga mest vikt vid att använda små styrsignaler, stort Q_2 , eller ha ett

litet reglerfel, Q_1 . Q_1 och Q_2 är blockdiagonala matriser med Q_1 och Q_2 repeterade N gånger. \mathcal{M} är en blockdiagonal matris med M repeterad N gånger.

Målfunktionen för MPC-reglering kan även för det olinjära fallet utvecklas med integralverkan och skrivs då enligt ekvation 11.

$$\sum_{j=0}^{N-1} \|z(k+j) - r(k+j)\|_{Q_1}^2 + \|u(k+j) - u(k+j-1)\|_{Q_3}^2 \quad (11)$$

Men precis som det nämntes tidigare kan heller inte ekvation 8 skrivas om som ett kvadratiskt problem.

På samma sätt som för det linjära fallet är det ekvationen nedan som är det problem som regulatorn löser för varje samplingsögonblick. Där man för det olinjära fallet implementerar målfunktionen som gavs i ekvation: 11. Det räcker dessutom inte att bara iterera en gång utan man kommer behöva iterera flera gånger och sedan minimera med avseende på U , för att plocka fram den bana som blir bäst. Detta kan dock också göra MPC:n allt för beräkningskrävande. Därför kan det finnas behov att antingen minska prediktionshorisonten eller placera ut en mindre antal punkter längs kurvan att reglera efter.

$$\min_U \text{målfunktion}(x_0, u) \quad (12)$$

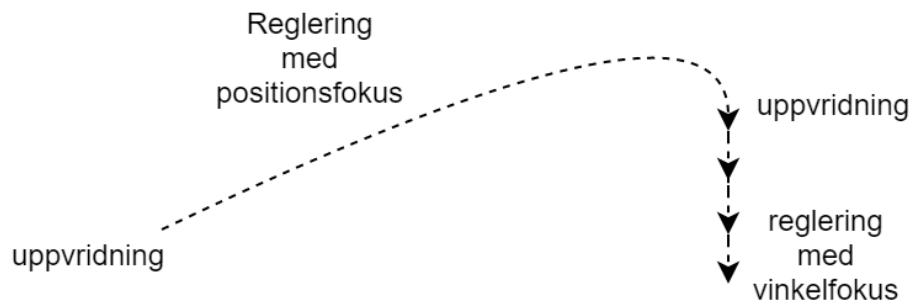
$$h(x_0, u) = 0 \quad (13)$$

Där x_0 är initial tillstånd, u insignal och h är valfritt bivillkor. Man kan lägga in så många bivillkor man vill så länge $h(x_0, u) = 0$ är uppfyllt. För att sedan lösa minimeringsproblemet finns för det olinjära fallet en funktion fmincon i Matlab. Motsvarighet till dessa eller liknande funktioner finns antagligen också i Python eller C++.

Vad man väljer att minimerar är upp till den som implementerar men en metod skulle vara att minimera tiden tills man kommer fram. Man skulle då kunna sätta positions och vinkelmålen samt gaffelhöjning och tornutskjutning som bivillkor. Ekvationerna i stycket om olinjär MPC är skrivit enligt [6].

5.8 Planeraren

För att öka robustheten hos trucken har planeraren förändrats och regulatorn har utvecklats till en tillståndsagent. S-kurvan som tidigare använts har bytts ut mot en J kurva som slutar 15 cm framför pallen för att sedan ha en rak precisionsreglering de sista 15 cm vilket ökar robustheten för skarpa kurvor. De tillstånd som kurvan nu har är: startvinkeljustering (uppvridning), vanlig följdning (reglering med positionsfokus), slutvinkeljustering (uppvridning), 10 cm reglering med precisionsfokus igen för att slutligen de sista 5cm enbart reglera med avseende på vinkel (reglering med vinkelfokus). Se figur 39



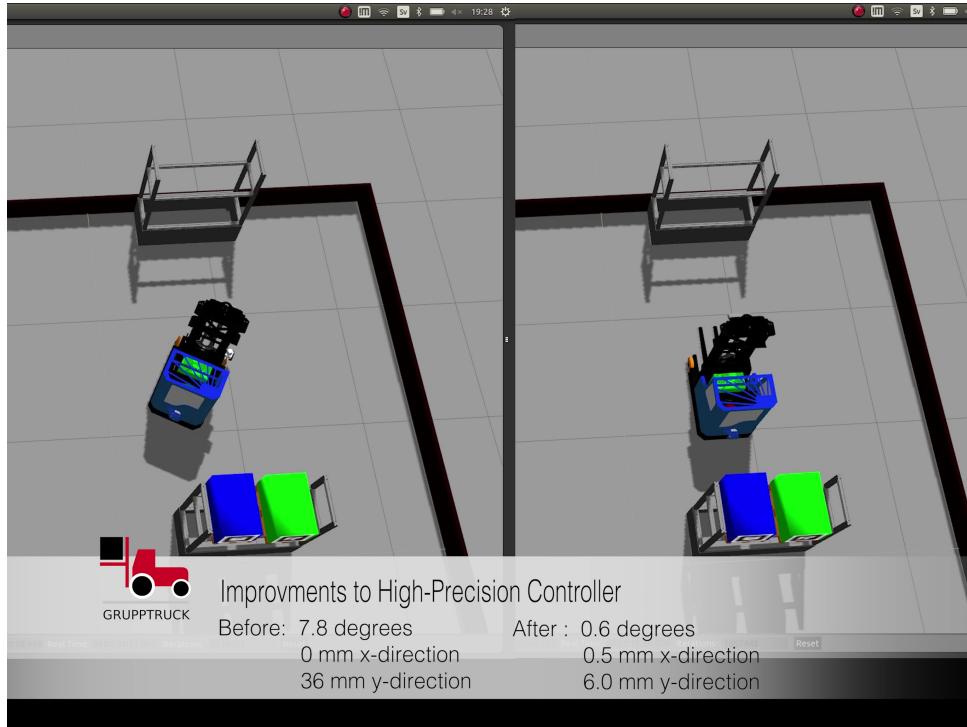
Figur 39: Bilden visar hur kurvan ser ut efter de ändringar som har gjorts.

Trucken startar med att sätta ett delmål med samma orientering som slutmålet och med den positionen att en bana mellan delmålet och målet kommer vara en raksträcka med längden 15cm. Vid startvinkeljustering så roterar trucken till den vinkel som banans första punkt har. När trucken har samma orientering som första punkten så byter trucken tillstånd till vanlig banföljning. I detta tillstånd används truckens hastighet- och vinkelhastighetsreglering som vanligt. När trucken närmar sig delmålet så byter den tillstånd till precisionsinkörning där trucken använder vinkelhastighetsregleringen men byter till en lång statisk hastighet. När trucken tillslut når delmålet byter den till det sista tillståndet sluttvinkeljustering som roterar trucken till vinkelns för delmålet.

När väl delmålet är uppfyllts sätts ett nytt delmål som ligger 5 cm längre fram på raksträckan mot slutmålet. Till detta delmål kommer trucken att använda precisionsföljning. Detta kommer upprepas tills trucken når slutmålet.

5.9 Resultat

Resultatet av projektet är att trucken kan ta betydligt skarpare kurvor i trånga lager men ändå hålla en högre precision vilket man kan se i figur 40 som är tagen efter en skarp insvängning.



Figur 40: Bilden visar hur resultatskillnaden för precisionsinkörning ser ut efter skarp sväng i trångt lager, före och efter projektarbetet

För att undersöka precisionen för regulatorn i simuleringsmiljön så utfördes två olika typer av kurvor med tio testkörningar på vardera kurva. Truckens orientering var i dessa kurvors startposition och slutposition riktad i x-led. I den första kurvan skulle trucken förflytta sig 0.5m i x-led och 0.5m i y-led och såg ut enligt kurva 1 i figur 37. Denna kurva fick resultaten som visas i tabell 2.

I den andra kurvan skulle trucken förflytta sig 1m i y-led och såg ut enligt kurva 2 i figur 37. Denna kurva fick resultaten som visas i tabell 3.

När regulatorn sedan testades på den riktiga trucken var resultatet inte alls lika bra. Trucken var många gånger snabbare både i vinkelhastighet och hastighet, samtidigt som samplingstiderna var olika. Detta gjorde att vinkelhastighetsregulatorn blev överställd samtidigt som hastighetsregulatorn blev underställd. Eftersom tillgången till den riktiga trucken var mycket begränsad fanns det inte tid till att trimma in regulatorn till den verkliga trucken.

Tabell 2: Kurva 1

Vinkelfel [grader]	X-positionsfel [mm]	Y-positionsfel [mm]
0.5	0.7	7.2
0.5	0.3	5.7
0.6	0.6	7.2
0.3	0.5	3.1
0.4	0.9	6.9
0.4	0.7	3.3
0.5	0.2	5.7
0.3	0.2	2.0
0.5	0.2	5.5
0.3	0.3	1.2

Tabell 3: Kurva 2

Vinkelfel [grader]	X-positionsfel [mm]	Y-positionsfel [mm]
0.5	0.6	7.3
0.7	0.3	8.2
0.6	0.2	9.3
0.6	2.2	5.1
0.5	2.2	13.1
0.6	3.0	8.3
0.6	1.6	0.7
0.7	1.0	10.0
0.6	0.6	9.0
0.5	0.6	6.4

5.10 Slutsats

Målet med detta delområde var att ta fram en mer robust och exakt regulator som skulle öka truckens pålitlighet att kunna hämta upp en pall. Efter projektet kan vi se att vi lyckats med detta i simuleringsmiljön, så som den var vid projektets början. Då simuleringsmiljön och verkligheten skiljer sig väldigt mycket fungerar dock inte regulatorn i verkligheten. Regulatorn fungerar även mindre bra i den uppdaterade simuleringsmiljön då den ibland hamnar i ett dödläge på grund av den tillagda dynamiken i simuleringsmiljön. Problemet i både den uppdaterade simuleringsmiljön och verkligheten är truckens beteende är mer aggressivt vilket inte fungerar bra med regulatorn då även den är aggressivt inställd.

För att kunna minska regulatorns aggressivitet men ändå behålla exakthet och robusthet är vårt råd att lägga till någon form av prediktionsförmåga i regulatorn, i form av antingen framkoppling eller en olinjär MPC. Även om man inte implementerar en framkoppling kan man ställa ner aggressiviteten på vinkelhastighetsregulatorn lite grann och göra hastighetsregleringen mer aggressiv detta borde då förbättra den tidigare regulatorn.

Referenser

- [1] *Designspecifikation*, Grupptruck. Sofie Dam, Jasmine Hebib, Johannes Bodin, Daniel Nilsson, Gabriel Fredriksson, Max Antonsson och Emil Relfsson, 2017.
- [2] *System Identification Overview*, Mathworks. 2017. <https://se.mathworks.com/help/ident/gs/about-system-identification.html>
- [3] *What Is Residual Analysis?*, Mathworks. 2017. <https://se.mathworks.com/help/ident/ug/what-is-residual-analysis.html>
- [4] *Robot Operating System (ROS)* <http://www.ros.org/>
- [5] *Physics Engine*, Wikipedia https://en.wikipedia.org/wiki/Physics_engine
- [6] *Industriell reglerteknik kurskompendium*, institutionen för systemteknik vid Linköpings universitet, 2014.