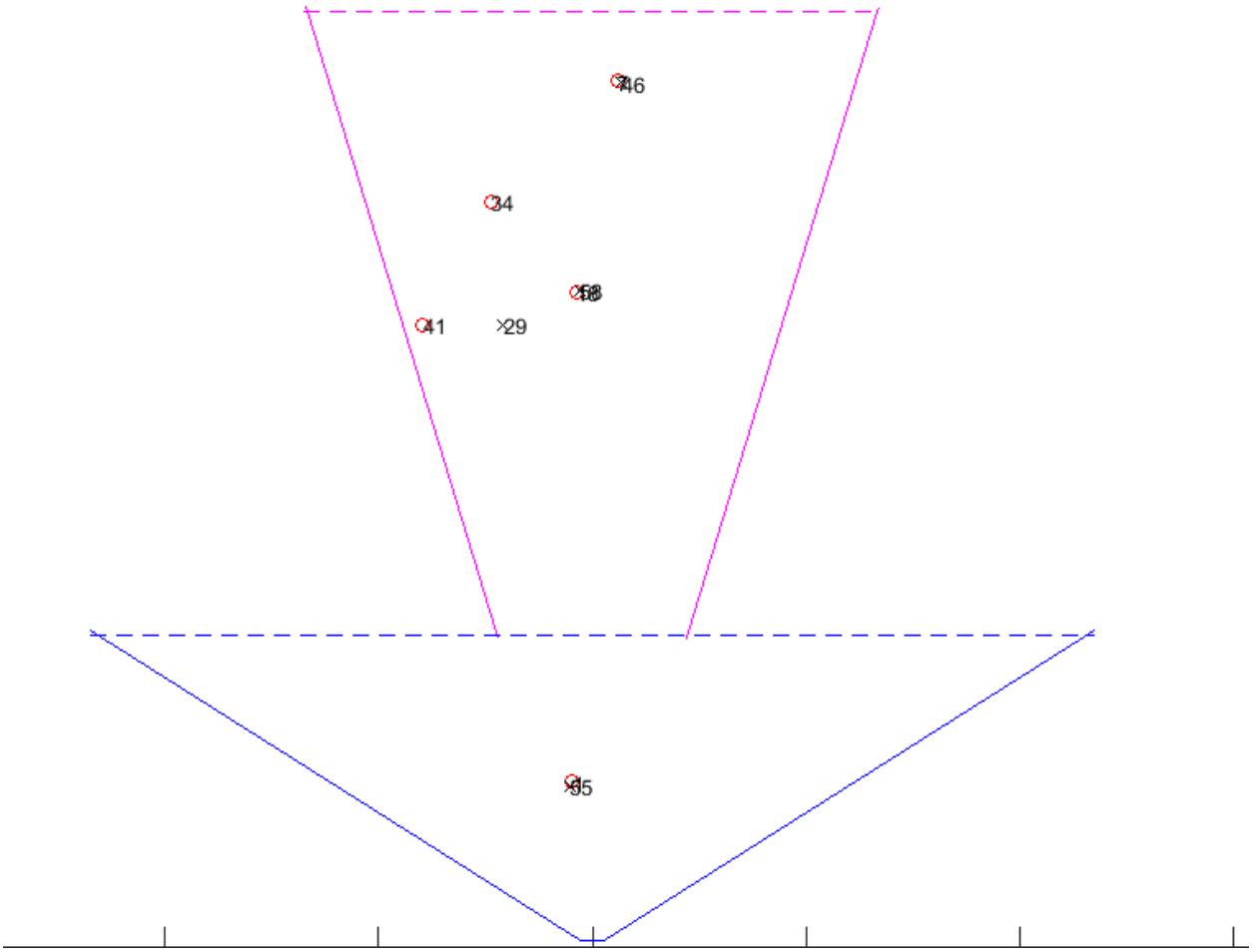
CHALMERS
UNIVERSITY OF TECHNOLOGY



Automatisk Ground Truth av radar- och kamerasystem

Automatic Ground Truth of radar and camera systems

Examensarbete inom högskoleingenjörsprogrammet Mekatronik

Henrik Gunnarsson
John Roth

Institutionen för Elektroteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2018

Automatiserad Ground Truth av radar- och kamerasystem

Automatic Ground Truth of radar and camera systems

Henrik Gunnarsson
John Roth

CHALMERS
UNIVERSITY OF TECHNOLOGY

Institutionen för Elektroteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2018

Förord

Denna rapport beskriver ett avslutande examensarbete för utbildningen Mekatronikingenjör, 180HP, som har utförts av två studenter på Chalmers tekniska högskola. Arbetet omfattar 15HP och har utförts på 70% under en period av tolv veckor på Aptiv. Huvudmålet har varit att automatiskt skapa en sanning för att effektivare undersöka prestanda av ett radar- och kamerasystem, IFV170+MRR2T-systemet, genom att använda ett välutvecklat system som referenssystem, Racam.

Vi vill rikta ett stort tack till Aptiv och främst vår handledare Tommy Majuri. Vi vill också tacka Jian Yang, professor inom Antenner på elektriska institutionen på Chalmers som har agerat examinator.

Henrik Gunnarsson och John Roth

Göteborg, juni 2018

Sammanfattning

För att jämföra hur väl ett auto-detekteringssystem med radar och kamera fungerar så krävs det att resultatet jämförs med någon slags facit. I nuvarande fall krävs det att någon sitter manuellt och markerar detekterade objekt mot en bild. Detta är en väldigt tidskrävande process. Först därefter kan det jämföras mot det autodetekterade flödet av objekt för att se hur väl det fungerar. För att spara tid och pengar vid utveckling av nya system vill Aptiv se en automatiserad lösning för Ground Truth, vilket i korta drag innebär sanningen från direkt observation. För att utveckla detta program har två olika radar- och kamerasystem installerats i en lastbil. Aptivs nyutvecklade radar- och kamerasystem, IFV170+MRR2T, har jämförts mot det redan väl använda och pålitliga systemet Racam. Eftersom att det inte finns någon färdig lösning har även hårdvara och kablage tillverkats och kopplats in. En lastbil har sedan kört runt på motorväg med båda systemen aktiva för att samla in loggfiler. Dessa har sedan synkroniserats och analyserats i ett Matlab-script. Utdata från Matlab ger en tabell med feldetekteringar efter önskad tolerans. I denna tabell kan det tydligt avläsas vilket objekt det handlar om, var den befinner sig i loggfilen och vart någonstans den har detekterats.

Abstract

To see how well an auto detection system with radar and camera works there will need to be a truth to compare it to. Currently a person has to manually mark a picture with a detected object. This is a very time consuming process. After this step, a comparison with the auto detected flow of objects is possible. To save time and money in development of new systems Aptiv wants to see an automated solution for Ground Truth, which in short terms means the truth from direct observation. To develop this new program two radar and camera system have been installed in a truck. Aptiv's newly developed radar and camera system, IFV170+MRR2T, for trucks have been compared to the already trustworthy and utilized integrated radar and camera system Racam. Since there was no solution already completed or developed there has been a need to produce hardware and wiring. A truck has been driving along a highway with both systems active to collect log files. These log files have been synchronized and analyzed in a Matlab script. The out data from Matlab gives a chart with fault detection after the wanted tolerance. This chart contains a clear view of what object is in fault, where in the log file it is located, and where it has been detected.

Innehållsförteckning

Terminologi

RAVA - Ett radar- och kamerasystem som består av både Racam och ett IFV170+MRR2T-system

RKS - Radar- och kamerasystem

UDP - User Datagram Protocol

VDR - Video Data Recorder

ECU - Electronic Control Unit

MCU – Micro Controller Unit

CAN - Controller Area Network

LVDS – Low Voltage Differential Signaling

PLOT – Ett sätt att visa grafer i Matlab

1. Inledning

Allt eftersom mer produkter automatiseras ökas behovet av att jämföra mjukvara och hårdvara snabbt och effektivt. En liten förbättring på effektiviteten av en produkts utförande gör snabbt stor skillnad när en produkt återanvänds flertalet gånger och många produkter ligger ute på marknaden.

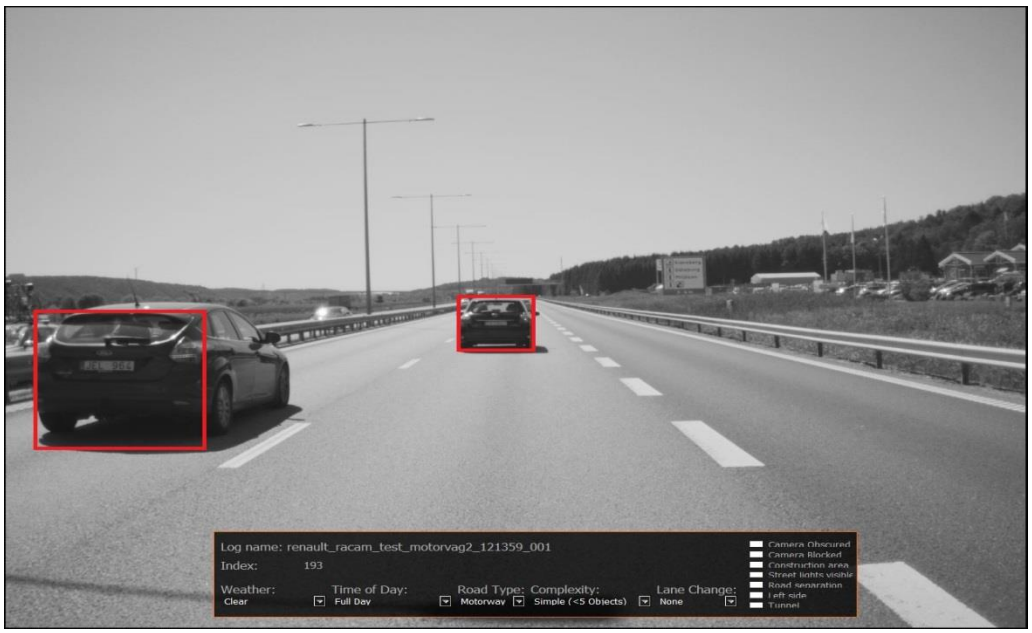
Aptiv är ett företag som bland annat arbetar med mjukvara och hårdvara till bilar. För tillfället är IFV170+MRR2T-systemet under full utveckling. Det är ett nytt radar- och kamerasystem som har anpassats för användning i lastbilar. Systemet handahåller detekteringar framför fordonet. Vilket betyder i korta drag att det känner av vägren, fordon och objekt av intresse framför fordonet. Tidigare har en produkt vid namn Racam (Radar and Camera System) använts vid detektering av objekt framför bil. För att granska ett sådant system används i nuläget många manuella timmars jobb för att analysera och hitta en ground truth för varje enskild bild. Man får aktivt leta upp objektet med eget öga och sedan hitta en radar-avkänning för att skapa en sanning. För att effektivt kunna utveckla IFV170+MRR2T-systemet, och även andra system i framtiden, vill Aptiv utveckla ett referenssystem som enkelt kan urskilja viktiga events i en logg. Detta skulle spara in flertalet mantimmar och framskrida utvecklingen i framtiden.

För att skapa ett sådant system är en lösning att börja med ett referenssystem med hög tillförlitlighet, vilket Racam har visat sig vara. IFV170+MRR2T-system har i nuläget en lägre mognadsgrad och behöver verifieras mot det mer utvecklade systemet. Detta referenssystem hoppas vi kunna automatisera för att spara på värdefull tid och pengar. Enligt önskan från Aptiv har vi utelämnat viss information om hårdvara, mjukvara, produktnamn och datablad.

1.1 Bakgrund

Aptiv är ett globalt teknologiföretag som arbetar inom elektronik och mobility, främst inom bil- och lastbilssektorn. På kontoret på Mölndalsvägen i Göteborg finns, bland annat, ett fokus på aktiv säkerhet. Just nu utvecklas ett radar- och kamerasystem för lastbilar som är nytt för Aptiv. Tidigare har fokus varit på liknande system för personbilar och enklare system enbart med radar eller kamera för lastbilar.

Vid utveckling av system som använder sig av fusion-teknologi brukar Aptiv manuellt sätta en så kallad markering (label) på detektioner på inhämtad data. Vilket går ut på att i ett program rama in objekt på vägen för att finjustera systemets förmåga att korrekt tolka verkligheten. Denna process kan vara mycket tidskrävande. I stora projekt används många tusentals mantimmar för att manuellt skapa referensobjekt på loggfiler. Därför vill Aptiv hitta en automatiserad lösning för att spara tid och pengar. Se exempel på hur en manuell label har placerats på ett objekt i figur 1.



Figur 1. Två fordon har manuellt markerats med varsin label.

1.2 Syfte

Syftet med projektet är att i Matlab skapa ett program som jämför hämtad data från Racam mot data från IFV170+MRR2T-systemet och ge ut en lista med tidpunkter där systemen skiljer sig i sin verklighetstolkning. För att åstadkomma detta krävs det en montering i ett fordon som tillåter insamling av data från två olika radar- och kamerasystem. Detta hoppas underlätta utvecklingen av framtida liknande sammanställningar.

1.3 Avgränsningar

Projektet kommer att utföras i Aptivs lokaler vid Mölndalsvägen, Göteborg. På plats finns hårdvara och olika typer av fordon som kommer att kunna användas under projektets gång. I jämförandet av Racam och IFV170+MRR2T-systemet kommer ett script utföra uträkningar och ge statistik för att se hur pass mycket systemen skiljer sig emellan. Endast avkänningar av fordonsobjekt, som består av personbilar, lastbilar och motorcyklar, kommer att jämföras i projektet. Loggfiler kommer maximalt att ha en samlad tid på två timmar. Det kommer också finnas en avgränsning där inget av systemen anses perfektar, utan det finns utrymme för båda att ha fel. All loggning kommer att ske i en lastbil och en godtagbar loggfil kommer bestå av att båda systemen parallellt är fungerande vid samma körning.

1.4 Preciserad frågeställning

- Kan Racam och IFV170+MRR2T-systemet monteras och installeras i samma fordon?
- Kommer Racam kunna kalibreras för en lastbil?
- Vilka toleranser ska användas för den automatiska jämförelsen av systemen?
- Hur mycket tid kan sparas (per loggad timma data) genom att jämföra de två loggarna automatiskt istället för manuellt?
- Hur precis är IFV170+MRR2T-systemets sensorer och detektioner jämfört med Racam-systemet?

2. Teknisk bakgrund

Under det här kapitlet redogörs den teoretiska bakgrunden för de tekniska komponenterna i projektet.

2.1 Matlab

Matlab är ett programspråk som är utvecklat av Mathworks. Språket är uppbyggt för matematiska tillämpningar då en av Matlabs egenskaper är hur den uttrycker matriser, rad- och kolumnberäkningar direkt i språket.

Matlab används bland annat inom dataanalys, trådlös kommunikation, abstrakta modelleringar, datorseende, signalhantering, robot -och kontrollsystem. [1]

2.2 Radar- och kamerasystem

Radar- och kamerasystem är ett sorts system som används för detektion och spårning av objekt med hjälp av att de båda komponenterna arbetar ihop. Titel för detta sätt att sammanföra data kallas fusion [2]. Vanligtvis används denna typ av system inom larm och säkerhet eller som i detta fall för säkerhetsfunktioner inom transportindustrin och fordon.

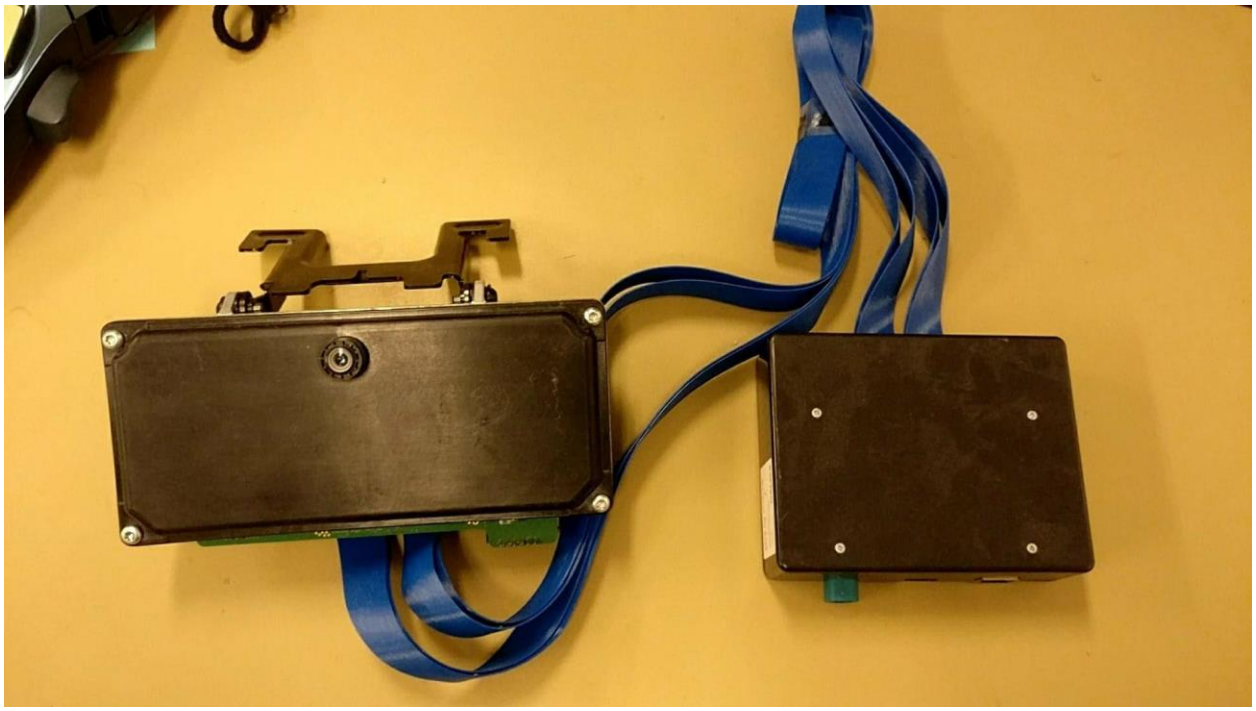
2.3 Ground Truth

Begreppet refererar till sanningen från direkt observation. En detektion från en radar eller ett liknande sensorsystem kan aldrig vara verkligheten, utan är mer korrekt en referens. Ground truth innebär i detta fall att resultatet av en mätning jämförs mot verkligheten, eller en sannolikt rätt bild av verkligheten. Ett exempel från meteorologi är när en storm dyker upp på en Doppler Radar och en meteorolog sedan på plats betraktar stormen med sina egna ögon [3].

2.3 Racam

Racam utvecklades för användning på insidan av vindrutan i en personbil. Systemet använder sig av en Electronic Scanning Radar (ESR), kamera, micro controller unit (MCU) och kretskort samt CPU för att utföra beräkningar och bygga upp en bild av verkligheten [4]. Den kopplas direkt mot bilens kommunikationsbuss för att kommunicera med övriga komponenter samt för att läsa in fordonets hastighet som används av beräkningar i realtid. Figur 2 visar komponenterna som utgör en Racam.

Figur 2, Racam. Kamera och radar i samma enhet till vänster. Instrumentbox till höger.

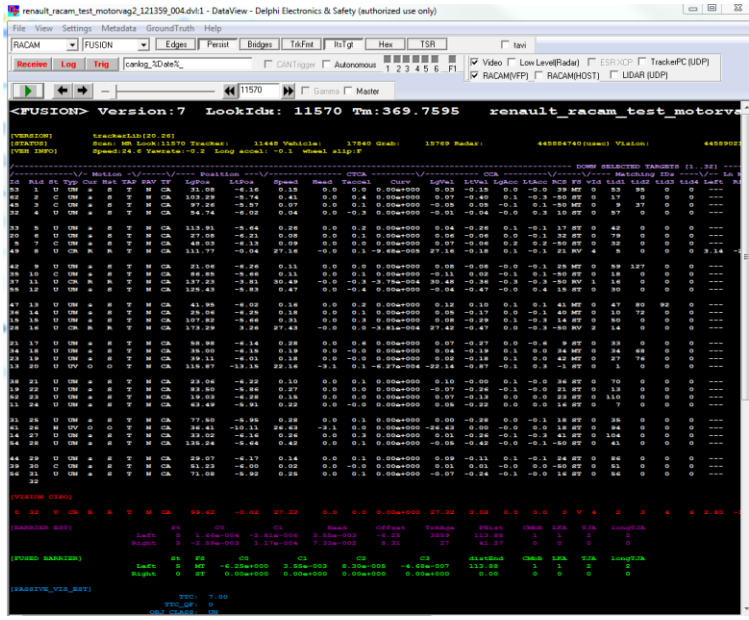


2.4 IFV170+MRR2T

Detta är ett system som just nu utvecklas av Aptiv. IFV170+MRR2T består av en IFV170 kamera och en MRR2T radar. IFV sitter inuti förarhytten och radar är placerad på utsidan nära mitten av kofångaren [Aptiv].

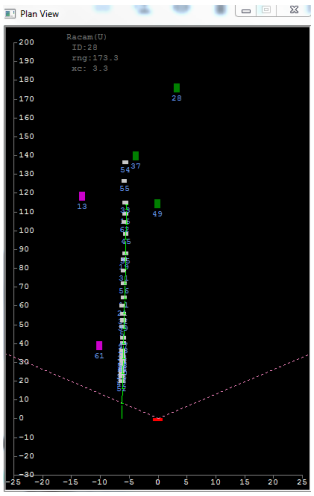
2.5 DVTools

DVTools är ett program och verktyg som används för loggning och återuppspelning av hämtad data från IFV170+MRR2T-systemet och Racam. Den handahåller även tolkningen av bilens CAN-signaler vilket ger hastighet och girgrad (yawrate). DVTools behöver justeras beroende på vilket system den ska kunna spela upp data från. I detta projekt har det använts tre olika versioner då vi har läst upp loggfiler från två olika Racam-konfigurationer och ett IFV170+MRR2T-system. För en vy från uppspelning av loggfil se figur 3 för själva programmets utseende. Figur 4visar Plan View för denna tidsstämpel och figur 5 visar video med radardetektioner.[Aptiv]

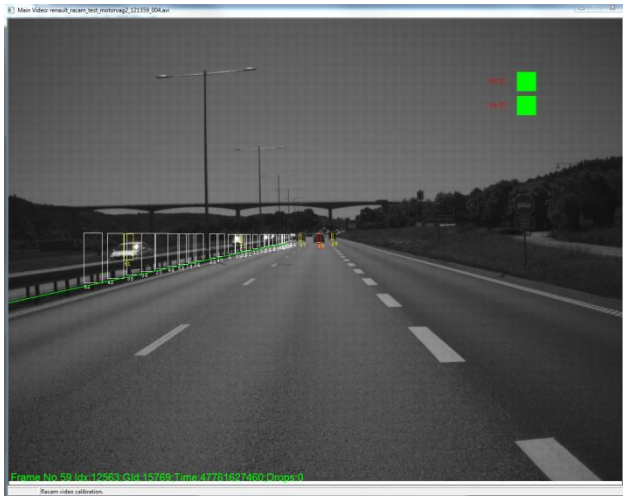


Figur 3, Utseende på interface för DVTools

Figur 4, Plan View från DVTools.



Figur 5, Videouppspelning i DVTools med radartetektion.



2.6 Video data recorder

En VDR är i många sammanhang en applikation som används för att tillåta en dator att agera som en digital videoinspelare. I detta projekt hänvisar det till en dator med ett installerat Windows 7 som operativsystem. Denna dator används främst för att processa och spela upp data som hämtas in genom RKS. Dessa datorer hänvisas som VDR i denna rapport [Aptiv].

2.7 Vector

Vector är ett företag som arbetar med komponenter, vid namn Vectorer, för att koppla ihop exempelvis CAN, ethernet och flexray med en PC [5]. Detta görs genom att Vector i fråga lyssnar på de bussar som skickas mellan systemen och bilen.

2.8 Videograbber

En videograbber används för att fånga videoinspelningar från en analog källa och göra om informationen till digital form. Vilket tillåter uppspelning, strömning eller delning av denna information på en PC [Aptiv].

3. Metod

I detta kapitel hanteras den övergripande struktur och metodik som användes under projektets gång.

3.1 Arbetsgång

För att ge projektet ett välstrukturerat sätt att fortskrida så har huvudmålen delats upp i delmål som avklarades genom arbetets gång. Dessa delmål byggde på frågeställningen som sattes vid projektstart och redogörs för i följande delkapitel. Projektet startades med installation och testning av funktioner i DVTools och Matlab. Parallellt med detta gjordes en draft på komponenter och kretsscheman med hjälp av ingenjörer på Aptiv. Då hårdvaran för insamling av data var konstruerad och införskaffad så installerades systemen i lastbilen. Simultant påbörjades programmering och toleranssättning av koden. Detta arbete fortgick tills projektet nått sitt slutdatum.

3.1.1 Konstruktion

Projektet startades med ett möte på Aptiv med lokala ingenjörer för skapandet av en draft på komponenter och kretsscheman. Det var också dessa ingenjörer som under arbetets gång införskaffade hårdvara och agerade bollplank vid konstruktion av loggningssystemet RAVA. RAVA konstruerades först på en plattform gjord av trä och flyttades senare till lastbilen där majoriteten av kablar kopplades in. Vissa komponenter har tillverkats från grunden och i dessa fall användes freeCAD [6] för att skapa önskad form. Ritningarna har realiserats med hjälp av en 3D-printer och testades i verkligheten. Om formen inte uppfyllde sitt syfte så korrigerades den i CAD och skrevs ut på nytt.

3.1.2 Kalibrering

Det har använts olika verktyg och mjukvaror för att ge rätt parametrar till IFV170+MRR2T och Racam-systemet. Efter att installering och montering i lastbilen blev komplett behövdes information angående position, längd, hjulaxelpositioner noteras och kalibreras in till aktuell hårdvara. Metoden för kalibrering har skiljt sig systemen emellan. IFV170, alltså kameran, har

behövt kalibreras mot en mönstrad duk. Därefter har en automatisk kalibrering vid körning på rak väg utförts.

3.1.3 Loggning

Efter installation och kalibrering så att IFV170+MRR2T och Racam kan köras parallellt så har anordningen varit ute på en körtur. Under denna körning samlade och sparade både systemen data från olika fordon och hinder på och bredvid vägen. Data har sedan använts för strukturering av scriptet som ska lösa problemställningen med jämförelse.

3.1.4 Kodning

Hela programmet som ska ge feldetektering som utdata har programmerats i Matlab. Från våra loggfiler har data sparats undan som sedan har lästs in i scriptet och användes därefter för att spara undan önskat objekt, position och etikett. Med den informationen har jämförelsen fortskridit och objekt har filtrerats ut.

3.2 Data- och informationsinsamlingsmetoder

Under konstruktion och installeringen av RAVA har majoriteten av informationen insamlats från ingenjörer på företaget (Residential Engineers) samt arbetsblad och manualer för aktuell hårdvara. För både programmering och toleranssättning krävdes diskussion med erfarna medarbetare på Aptiv men även från datablad för varje enskilt system. Enligt önskan från Aptiv har vi inte redovisat referenser till dessa datablad [Aptiv].

3.3 Test- och verifieringsmetoder

För att kontrollera inhämtad data från RAVA har det krävts kalibrerade system och testkörningar på både IFV170+MRR2T-systemet och Racam. Därefter undersöktes loggfiler ytterligare med hjälp av DVTools för att kontrollera och justera konfigureringar på systemen.

4. Genomförande

Här beskrivs genomförandet av delmomenten i projektet.

4.1 Konstruktion av RAVA

I detta kapitel redogörs det för konstruerandet och installeringsprocessen av RAVA i fordonet.

4.1.1 Kopplingsschema

Kopplingsschema (bilaga 1) planerades och skapades med stöd från vår handledare samt två ingenjörer på företaget (Residential Engineers). Intervjuer och möten tog rum för att klargöra hur ett kopplingsschema skulle kunna se ut [7][8]. Det slutliga kopplingsschemat [bilaga 1] blev istället följande.

Två radar- och kamerasystem har monterats och installeras sida vid sida. Gemensam faktor är att båda system innefattar följande komponenter:

- VDR
- Hårddisk att spara inspelad loggfil
- Monitor
- Gateway
- Logger
- Strömförsörjning
- Tangentbord med pekdon

Kommunikationsbuss (CAN-signal) kopplas till gateway (se 4.1.2.1) för att processa data från fordonet och sedan spara undan informationen i separat system. Strömförsörjning från lastbilen ger 12V och 24V vilket delas ut till komponenter genom en egenbyggd strömförsörjningslåda (se 4.1.2.2). IFV170 behöver kopplas via en LVDS-kabel till en videograbber, och därefter till VDR via USB3, för att ge korrekt fungerande bildåtergivning. VDR för Racam har ett SIM3-kort installerat och behöver därför ingen enskild videograbber. Se bilaga 1 för kopplingsschema.

4.2 Skapandet av komponenter

Då inte alla komponenter och kablage fanns färdiga att tillgå krävs det för projektets framfart att de skapades.

4.2.1 Gateways

En gateway är en komponent som översätter CAN-signaler från ett standard till en annan. I detta projekt behövs två gateways då CAN-data hämtas från en lastbil och ska översättas till standarden för de båda RKS:en. Lösningen kom från en lokal systemingenjör [9]. Lösningen går ut på att kombinera en standard Teensy 3.6 [10] med en dual CAN-bus adapter. Detta gör det möjligt att programmera mikrokontrollern Teensy att agera gateway för alla CAN-språk. Genom uppkoppling till motsvarande system i lastbilen kunde ett test utföras. Detta utfördes via ett varv runt en parkeringsplats för att kontrollera att rätt värde på speed (hastighet) och yawrate (girgrad) i DVTools togs emot på rätt sätt. Värden i Teensy microcontrollerns kod justerades gradvis med denna metod tills den ansågs tolka variablerna tillräckligt precist.

4.2.2 Strömförsörjningslåda

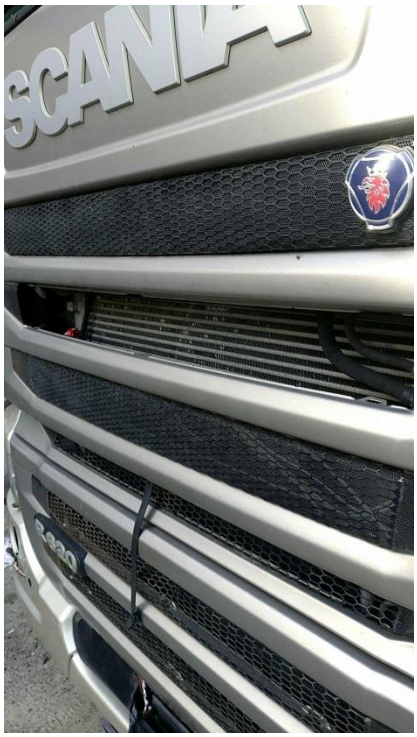
För att få spänning till hårdvaran behövs en bra strömkälla och ett sätt att koppla upp mot källan. Det anses lämpligt att använda sig av banankontakter vid uppkoppling. Det mättes upp att ett banankontaktsuttag kräver ett hål med diametern 8.3 mm. Infästningshål för banankontakter med denna storlek lades till på locket av lådan. Varav uttagen var uppdelade till att det finns tre 24V uttag och två för 12V samt ett större hål för vardera spänningskällan på sidan av boxen. Efter att en sketch skapats i CAD gjordes filen om till en så kallad mesh för att kunna tolkas av programmet som är gjort för användning mot Ultimaker:s 3d- printers [11]. Mesh-filen blir omgjord till ett stdl-format vilket används för att printa ut ritningen. För printning användes Ultimaker 2 extended + och materialet vit polyaktid plast. Efter 15 timmar utskrift blev följande box resultatet (figur 6).

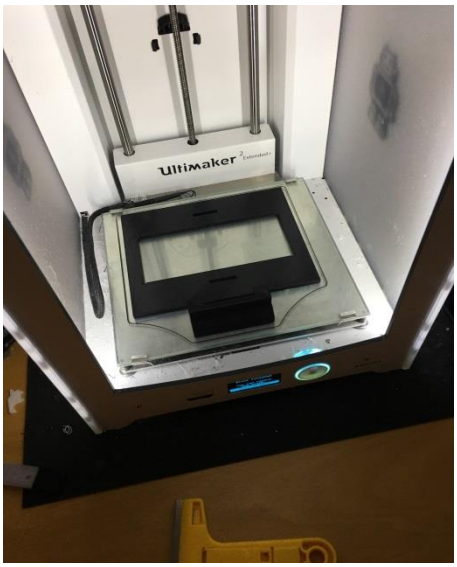
Figur 6. Egentillverkad strömförsörjningslåda från en 3D-utskrift.

4.2.3 Racam fästanordning

Då Racam i ordinarie fall ska sitta bakom rutan på en personbil krävs det en fästanordning för att få den i optimal arbetshöjd i en lastbil. Den valda metoden för att fästa Racam var att ta bort ett galler på fronten av lastbilen vid höjden 1.35m och utnyttja gallrets fästpunkter för att hålla den på plats (figur 7). Här används freeCAD för att skapa en 3D-utskrift till fästanordningen (figur 8). Något som alla prototyper har gemensamt är att de skruvas fast med hjälp av skruvar i kanterna av framsidan på radarn samt fästet på baksidan av Racam. Detta används för att ytterligare fästa systemet till plastramen. Dimensioner justeras tills ramen lätt går att trä över fästpunkterna. Buntband nyttjas på insidan av huven som kilar i fästpunkterna för att hålla ramen på plats (figur 9). Utöver detta används dubbelhäftande tejp på framsidan av ramen för stabilitet och dämpning under körning. Tillhörande instrumenteringsbox fästs med kardborreband mot insidan av motorhuven. Figur 10 visar färdig montering från utsidan.

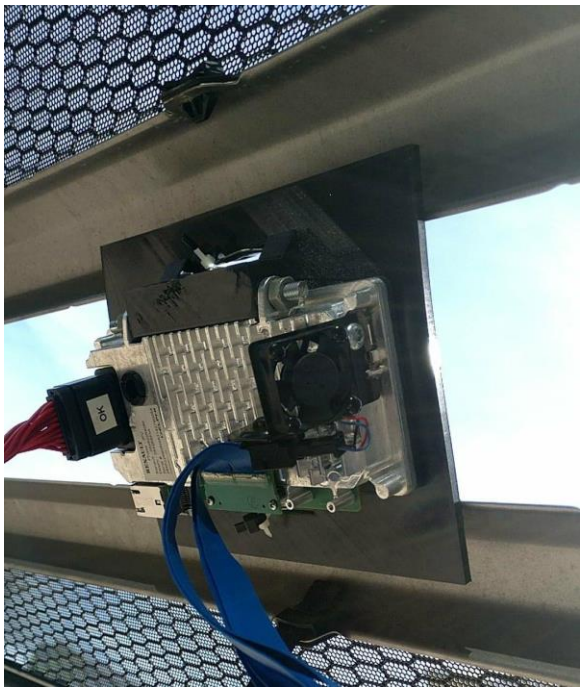
Figur 7. Front på lastbil med bortplockat galler för placering av Racam.





Figur 8. 3D-utskrift av fästanordning för Racam

Figur 9. Racam infäst i lastbil från insidan av frontlucka.



Figur 10. Racam infäst i lastbil från utsidan av frontlucka.

4.2.4 Videograbber

Då IFV170+MRR2T-systemet inte har en inbyggd videograbber i sin VDR och ingen ledig videograbber fanns tillgänglig var alternativet att använda en prototyp vid namnet Latice. Latice är programmerad och byggd på Aptiv [Aptiv].

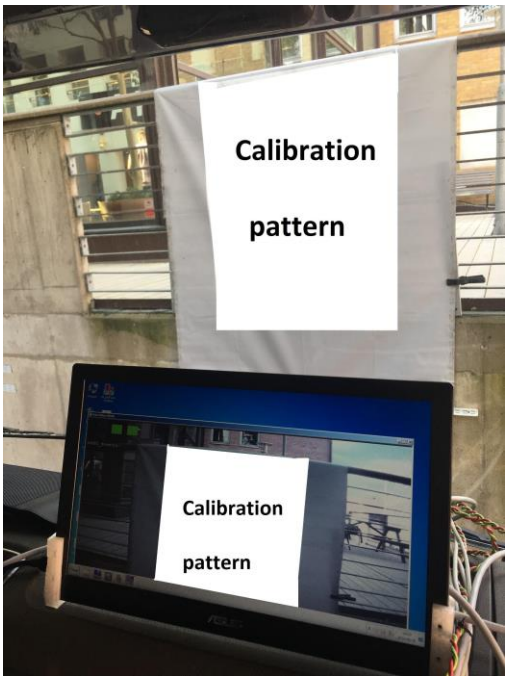
4.3 Kalibrering

I detta avsnitt redogörs för hur kalibrering har gått till väga för båda system i fordonet. Både RACAM och IFV170+MRR2T-systemet är komplicerade och känsliga system. Därför räcker det inte att montera dem på lastbilen utan kalibrering måste göras innan systemen samlar korrekt information.

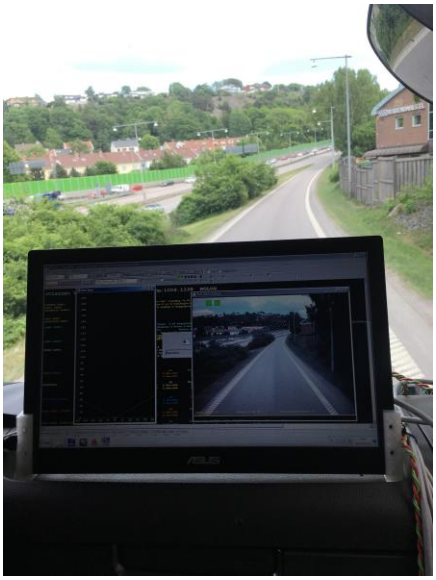
4.3.1 Kalibrering IFV170+MRR2T-system

För att kalibrera IFV170+MRR2T-systemet mäts monterad höjd på aktuell hårdvara, avstånd till främre kofångare, avstånd till bakaxel, bredd på lastbil. Därefter används ett kundspecifikt diagnosverktyg. I detta verktyg uppmäts alla positioner av kamera och radar gentemot fordonets koordinatsystem. Alla toleranser för positionering av radar och kamera samt nödvändiga distanser på lastbilen programmeras in. Efter att exakta mått placeras in i IFV görs en stillastående kalibrering mot ett förbestämt mål, se figur 11. Efter att rätt inställning på kamerans roll ställts in görs en dynamisk kalibrering för att fastställa kamerans och radarns resterande inställningar. En dynamisk kalibrering av systemet utförs sedan genom användning av lastbilens position relativt till väg och väghållning, se figur 12 och figur 13. Efter denna kalibrering är IFV170+MRR2T systemet installerat i lastbilen och fullt fungerande.

Figur 11. Kalibrering av IFV170+MRR2T-system mot förbestämt mönster



Figur 12. Dynamisk kalibrering IFV170+MRR2T-system på motorväg från insidan lastbil.



Figur 13. Dynamisk kalibrering av IFV170+MRR2T-system från DVTools

4.3.2 Kalibrering RACAM

För att kalibrera systemet mäts först höjd på fastmonterad Racam på lastbilen, avstånd från främre kofångare, avstånd från radar till bakaxel, bredd på lastbilen och vinkeln på systemets synfält. Värden programerars in (flashas) över till Racam med hjälp av kundspecifikt diagnosverktyg. Därefter kördes lastbilen på motorväg för en automatisk kalibrering av systemet. För att en automatisk kalibrering skall utföras så krävs det en rak väg där hastighetsgränsen ligger över 50 km/h. Vid alla körningar med RAVA räknades all insamlad data från ett fullt kalibrerat system.

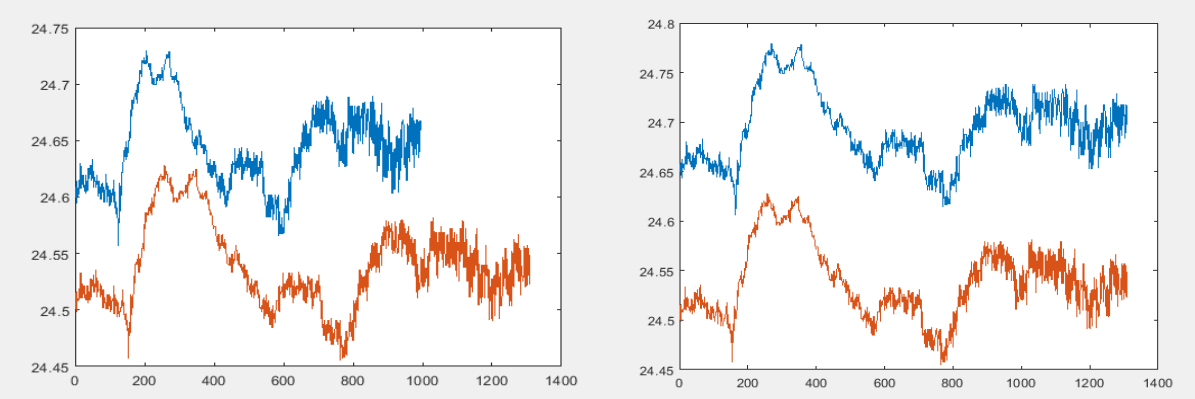
4.4 Programkod

För att använda och systematiskt analysera loggar från båda systemen har programmet Matlab använts. Systemen lagrar information från UDP-strömmen som kan läsas med hjälp av olika funktioner. Dessa funktioner finns i färdiga bibliotek (libraries i Matlab) som är skapade av Aptiv själva [Aptiv]. Det finns bibliotek som är anpassade efter vilket system som ska avläsas och kommer med DVTools. Kod går att följa i bilaga 2.

4.4.1 Avläsning

Loggfilerna är uppbyggda efter olika index så att det ska gå att följa händelserna över tid. I detta projekt har det valts att följa "lookIndex". Vilken helt enkelt är en indexering som går ett steg framåt per radar-avläsning. Genom att gå igenom lookIndex steg för steg kan händelseutvecklingen följas genom tiden. Det som ska jämföras är hur fusion-objekt i Racam och fusion-objekt i IFV170+MRR2T-system matchar varandra. Därför avläses relevant data. Först stegas objekt från IFV170+MRR2T-systemet i ett ögonblick fram och skrivs ut i en plot. Därefter görs samma sak för Racam. Information som longitudposition, latitudposition, objekt-id samlas upp och sparas till nästa sekvens i koden. En annan viktig detalj är att välja vilka objekt det är som ska läsas upp. I denna avläsning har fordon varit av intresse. Därför söker programmet efter detekterade bilar, motorcyklar, lastbilar och oidentifierade fordon. Den utelämnar exempelvis cyklar och fotgängare. Systemen loggar på olika hastigheter och får därför inte samma uppspelningslängd. För att kunna jämföra dem med rätt objekt i varje frame krävs en synkronisering. I denna kod skalas lookIndex för Racam om för att passa in till IFV170+MRR2T-systemets lookIndex. Se figur 14 för att se resultat av synkroniserad bild.

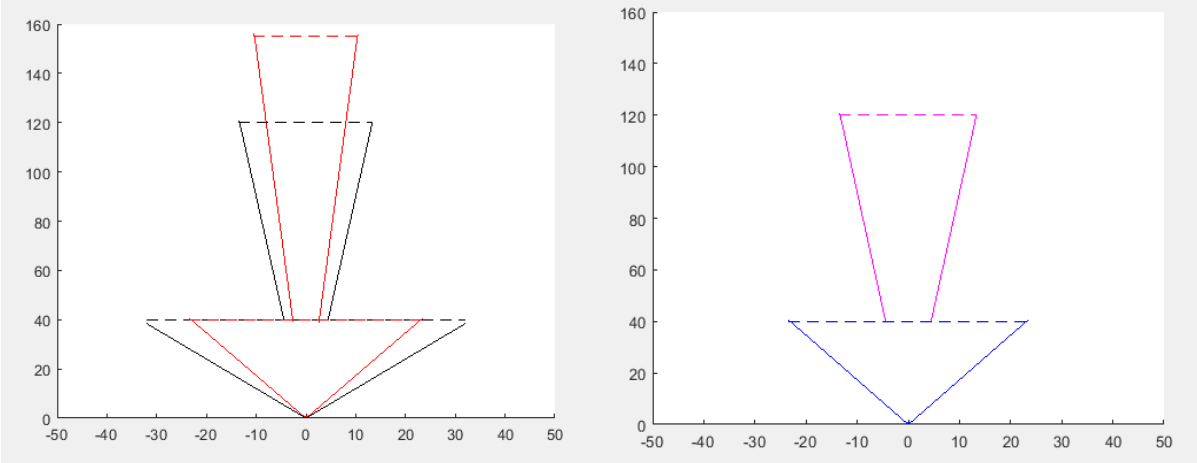
Figur 14. Till vänster en graf före synkronisering. Synkronisering till höger



4.4.2 Begränsningar

I och med att de olika systemen har olika förutsättningar och prestanda behövs begränsningar implementeras för att ge båda en rättvis chans för bedömning och jämförelse. IFV170+MRR2T-systemet har en vidare field of view vid objekt nära lastbilen, medan Racam har en längre räckvidd (Detection Range). Hänsyn får ges till att radar och kamera har olika prestanda och agerar därför inte med samma träffbild. De sitter dessutom monterade på olika nivåer. En maxgräns på 120 meter och en vinkel från lastbilens mitt på 60 grader som zonar ut ointressanta objekt och detektioner finns därför som en avgränsning i koden. För en visuell bild på begränsande zoner i avläsning och jämförelse se figur 15. Värt att notera vid åskådning är hur skalorna skiljer sig mellan longitud och latitud.

Figur 15. Begränsande zoner i jämförelsen för Matlab. Till vänster syns ett exempel på hur olika de olika systemen ser vägen framför fordonet. Till höger är en bild på vart alla objekt kan avläsas på lika villkor.



4.4.3 Jämförelse

Huvudmålet för programmet är att ge en lista med detekterade fel som utdata. Fel i detta fall innebär när objekten har varit på för långt avstånd ifrån givna toleranser eller om det finns olika antal objekt. Alla objekt som hittats bland Racams fusion-logg jämförs med alla fusion-objekt från IFV170+MRR2T-systemet. Eftersom att det är helt olika system finns det ingen medföljande funktion för att para ihop lämpliga detekteringar. Utan det måste göras via en komponerad algoritm i Matlab. Funktionen för att försöka para ihop de båda utgår från att jämföra hypotenusan mot alla objekt. Se figur 16 och figur 17 nedan för en visuell förklaring.

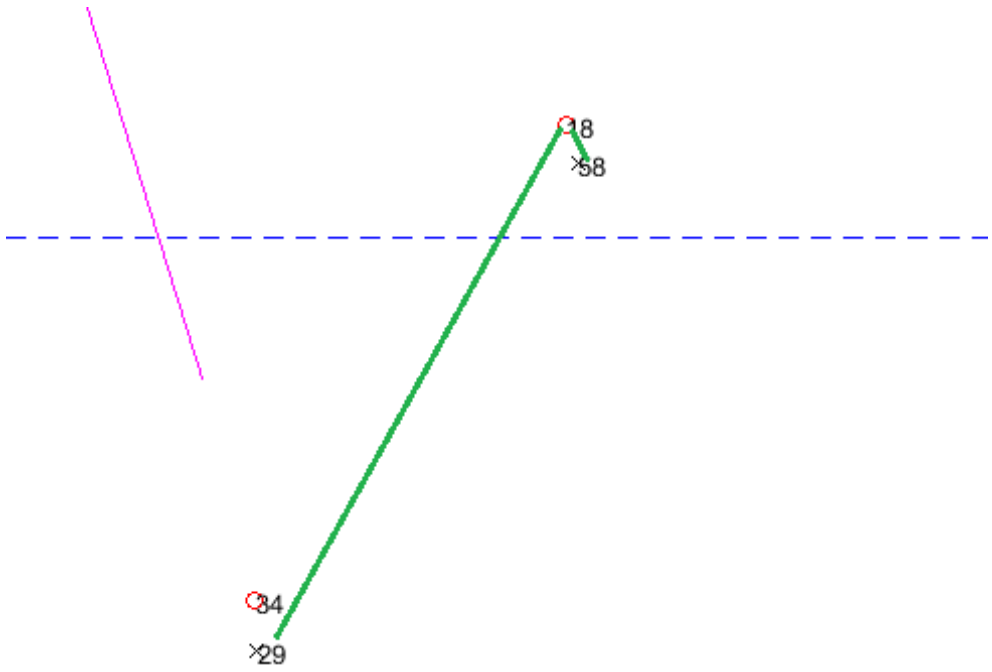
Figur 16. Exempel på hur två objekt får en parning och sorteras i en struct i Matlabscriptet.

cmp_object.h

	1	2	3	4	5	6	7	8	9	10	11
1	1.0640	18	43.1132	-0.3621	3	11000	58	42.0565	-0.2374	2	8534
2	15.0036	18	43.1132	-0.3621	3	11000	29	28.4316	-3.4539	2	8534

Objekt med id 18 är från IFV170+MRR2T. Id 18 får ett avstånd till alla objekt som Racam visar upp. Därefter analyseras tabellen och sorteras efter kortast avstånd. Om man studerar nedan kan man se hur avståndet mellan 18 och 58 är kortare än 18 och 29. Därför paras id 18 med id 58. En matchning mellan två objekt är gjorda och sparas därefter undan för ytterligare jämförelse med andra toleranser. Det är inte en garanti för att det är rätt parning i detta läge. Nästa steg blir att analysera huruvida avståndet är rimligt och godtagbart. Om det är för stort är risken att det saknas eller finns för många objekt med i bilden och det blir helt enkelt en missparning. Det medför en feldetektering som går att följa upp i loggen efteråt. Ett annat exempel är när något av systemen har detekterat med för stor avvikelse mot det andra. Där är det också viktigt att rapportera in felet för vidare undersökning.

Figur 17. Visuell demonstration på hur de två närmaste objekten paras ihop. Objekt id 18 jämför två objekt från det andra systemet, Objekt id 58 och id 29. Den med kortast avstånd sorteras högst upp i parningslistan.



5. Resultat

Följande avsnitt presenterar projektarbetets resultat. Ur ett överskådligt perspektiv har detta projektarbete resulterat i att två stycken olika radar- och kamerasystem, kallat RAVA, har installerats i en lastbil och har färdats på väg med dual loggning. Dessa loggfiler har därefter körts i ett Matlab-script för att uppge en tabell med feldetektering.

5.1 Installation i lastbil

För placering av kamera och radarkomponenter för de båda installerade systemen, se figur 18. Installation av RAVA i lastbilen blev en mycket längre process än väntat. Att samla ihop komponenterna som behövs och rådfråga personal på Aptiv med behövd expertis var tidskrävande. Hårdvara som orsakade fördröjning var speciellt Vectorboxarna (VN8911 och VN1630). Till projektets hjälp tillverkades två stycken gateway:s .

Figur 18. Placering av komponenter från båda RKS från front på lastbil.

5.2 Loggning

IFV170+MRR2T-systemet har blivit kalibrerat och fungerar enligt önskemål. Vid enskild körning blev loggning av data ett lyckat resultat. Även Racam fick en körbar installation. Det som brister är den automatiska vägkalibreringen som har gjorts. Dock har systemetet inte lyckats med sin automatiska sparning av inställningar. Det medför att Racam går att köra och logga data med, men den har bara delvis fullständiga inställningar. Funktionen "Auto Align" har inte gett variabeln "angle" korrekt kalibrerad vinkel. Videograbber slutade fungera. Därmed fick loggfilen från IFV170+MRR2T någon medföljande videoinspelning. Detta inverkar inte på de önskade loggfilerna som innehåller den data som behövs för Matlab-jämförelsen.

5.3 Matlab och jämförelse

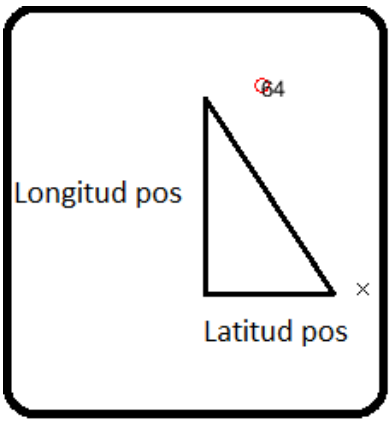
Den fördröjda processen med installation av RAVA i lastbil medförde mindre tid än planerat till att utveckla den jämförande koden. Första steget med att faktiskt använda och ge en plot för två olika system kunde slutföras. Dock gick den sista programmeringsetappen inte att slutföra utan att ha två loggningar från samma tillfälle. Därför saknas viss funktionalitet i önskad jämförelse återigen på grund av tidsbrist. Resultatet ger dock en fungerande tabell med felrapportering. Visningsexempel på resultat finns att se på figur 19.

Figur 19. Resultat från feltabell efter jämförelsekod. Längst till vänster är en indexering över vilket fel i turordning det här. Därefter är det avståndet mellan parade objekt som har uppmäts. I rutorna finns information om vad det är var objekt. Informationen innefattar bland annat ID, longitudinell och latitudnell position.

		IFV170+MRR2T					Racam				
26	15.3380	55	12.8038	5.9849	2	10805	2	10.2263	-9.1349	0	8377
27	6.6218	18	12.3438	-2.4379	3	10806	2	10.5738	-8.8188	0	8378
28	14.9819	55	12.9636	5.9713	2	10806	2	10.5738	-8.8188	0	8378
29	6.6470	18	12.4932	-2.4222	3	10807	2	10.6778	-8.8165	0	8379
30	14.9769	55	13.1221	5.9596	2	10807	2	10.6778	-8.8165	0	8379
31	8.6757	55	13.2813	5.9485	2	10808	58	11.1702	-2.4665	1	8380
32	8.7038	55	13.4386	5.9365	2	10809	58	11.1702	-2.4665	1	8380
33	8.7206	55	13.5972	5.9231	2	10810	58	11.2628	-2.4792	1	8381
34	14.6898	34	4.9163	-2.9648	2	10861	58	19.4847	-1.0797	2	8422
35	15.3771	34	4.9318	-3.0104	2	10865	58	20.1737	-0.9755	2	8426
36	15.4790	34	4.9668	-3.0312	2	10866	58	20.3062	-0.9573	2	8427

5.4 Toleransvärden

Toleranser för att fånga in felaktiga objekt sattes till longitudinell 120 meter. Detta för att kompensera IFV170+MRR2T-systemets kortare räckvidd då objekt längre bort bara skulle uppfattas av Racam och därmed generera onödiga fel. Field of View-vinkel beräknas till 60 grader. Detta för att kompensera hur Racam inte hinner se objekt innan IFV170+MRR2T-systemet och det skulle därmed innebära felaktig felrapportering. Hypotenusan, alltså kortaste vägen mellan två objekt har fått avståndet 2 meter, se figur 20.



Figur 20, hypotenusan är kortaste sträckan mellan två objekt

5.5 Resultat från jämförande script.

Här redovisas resultat på jämförelse och feldetektering.

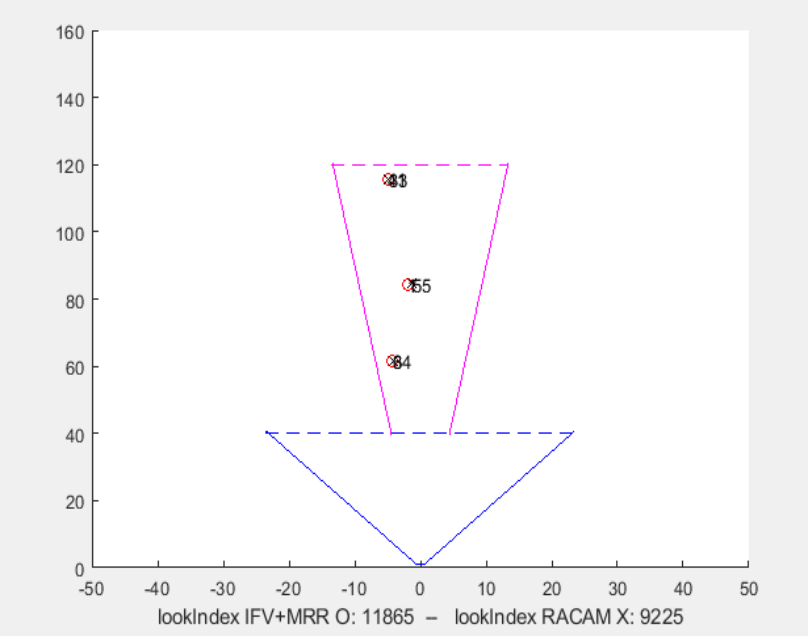
5.5.1 Manuell jämförelse

Tiden det tar för detta Matlab-script att söka igenom två loggfiler tar ungefär 2 minuter. Att sitta manuellt och sätta label för samma loggfil tog två timmar. Då saknas också tiden det tar för att manuellt jämföra inramade objekt. Loggfilernas längd i detta fall var 40 sekunder. Något som innefattar tiden för den automatiska jämförelsen är också tiden det tar för att bygga och installera två system bredvid varandra. I detta projekt tog det för ovana händer ungefär 6 veckor. Om vi lägger till det till själva tiden det tar för att manuellt lägga en label på ett objekt, så skulle en brytpunkt vara vid loggfiler på över en timme. Då skulle det därefter tjänas in tid på att göra det automatiskt istället för manuellt.

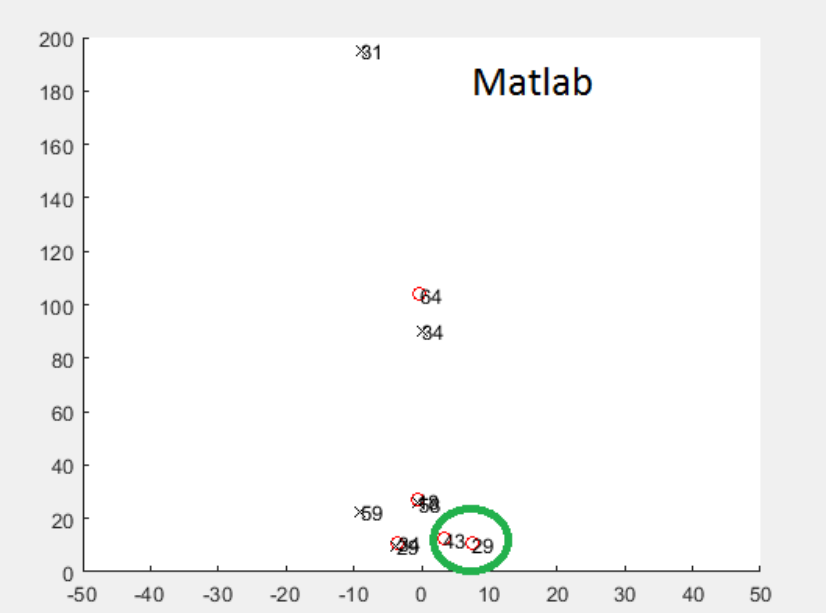
5.5.2 IFV170+MRR2T-system versus Racam

Se figur 21 för ett resultat på hur väl de båda systemen matchar. Förutom några få undantag stämmer de båda överens mycket bra. En feldetektering där IFV170+MRR2T-systemet har registrerat objekt som inte finns enligt Racam går att se på Matlab-plot på avläsning i figur 22, ”plan view” från båda system i figur 23, videoinspelning från Racam i figur 24. Tyvärr saknas videoinspelning från IFV170+MRR2T-system på grund ut av en icke fungerande komponent.

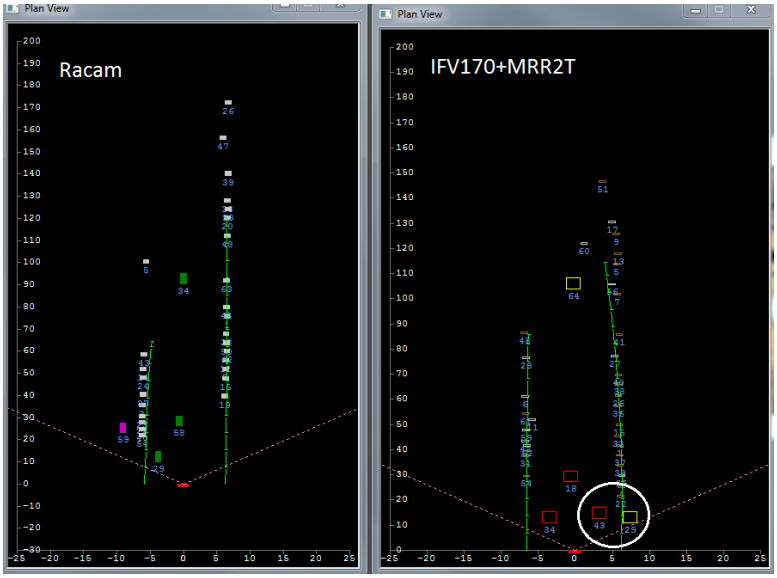
Figur 21. Plot av resultat från objekt från två RKS i RAVA.



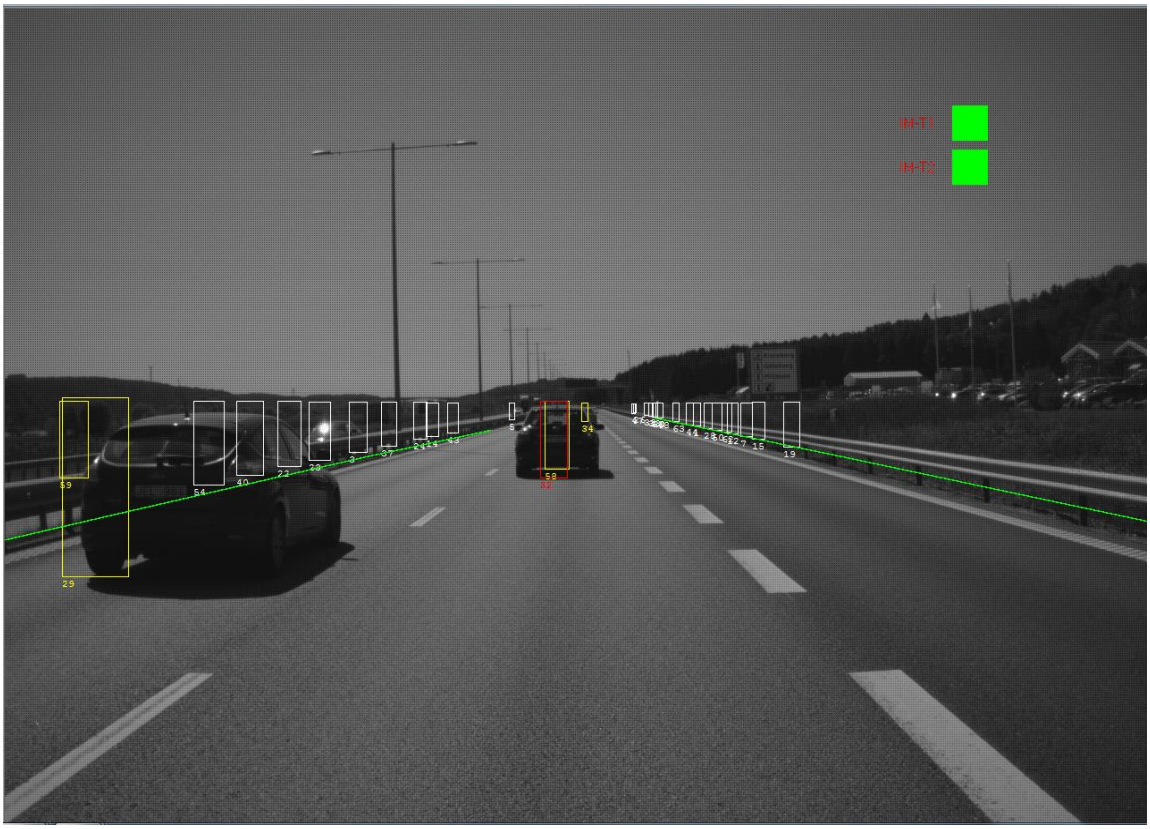
Figur 22. Undersökning av plot i Matlab. Två ensamma objekt kan observeras från IFV170+MRR2T-systemet där Racam inte har någon markering.



Figur 23. Plan view från båda system. Till höger syns att det finns objekt som inte det andra systemet har detekterat.



Figur 24. Videoinspelning från Racam ur loggfil. Till synes utan detektioner på högersidan.



6. Diskussion och slutsats

I detta avsnitt presenteras slutsatser samt diskussioner av resultatet från projektarbetet.

6.1 Hållbarhet

RAVA är en tillfällig konstruktion som byggdes, kopplades och monterades på under tidspress. Detta har lett till att det finns flera svagheter i konstruktionen.

6.2 Sortering och justering av kablar

En av de simplare sakerna som kan förbättras är kabellängd och dragning. När projektet startades var det oklart hur långt de olika kablarna skulle dras. Vilket gjorde att majoriteten av kablar blev för långa vilket mot slutet resulterade i ett virrvarr av sladdar, även när dem var sorterade. Utöver detta så hade det varit fördelaktigt med en noggrannare färgkod på sladdarna.

6.3 Isolering för Racam

Racam är byggd för att sitta inne i en förarhytt där den inte kommer i kontakt med nederbörd och vägdamm, vilket den inte är byggd för att tåla. På lastbilen sitter den oskyddad i fronten vilket hindrar bilen från att köra när det är eller finns risk för nederbörd. En lösning på detta är att bygga en isolerad box som återskapar förhållandet bakom en vindruta, genom att ha samma glas och lutning som på vindrutan i personbilen. Lådan bör vara tät och ha en lättare isolering då systemet endast behöver skyddas från fukt. Eventuellt skulle en vindrutetorkare även behövas vid nederbörd.

6.3.1 Material för ram

Ramen som håller RACAM på plats är gjord av polyaktid vilket är en plastsort som skapas från stärkelse. Polyaktid är en helt nedbrytbar plast ute i naturen och den är inte heller giftig. Nackdelen är att plasten är spröd och kan försvagas kraftigt av värme.

6.4 Komplikationer

Här diskuteras komplikationer under arbetets gång och vad som kunde gjorts bättre.

6.4.1 Videograbber

Vid loggningstillfällen förekom problem med videograbber som verkade bero på glapp i eltillförseln. Detta ledde till att den loggen som användes till att slutföra utvecklingen av programkoden inte har någon videobild från IFV170+MRR2T-systemet. Detta medförde att utvecklingen av kod blev avsevärt svårare och mer tidskrävande. Felsökning och synkronisering av kod och loggfil blev besvärliga att verifiera. Lyckligtvis fanns det loggar när systemen redan hade samma startpunkt. Tack vara dessa kunde projektet gå vidare till nästa steg. Den sista veckan av projektet startade videograbber inte alls och det lät som det fanns en liten lös del i lådan.

6.4.2 Monteringstid

Monteringstiden var planerad att ta två veckor vilket visade sig vara en grov felkalkylering. Då det tog tid att hitta hårdvara, kontakter och personer med rätt kunskap. Monteringen tog istället ungefär fem veckor.

6.5 Vidareutveckling

Idéer för fortsatt arbete på detta projekt.

6.5.1 Automatisk synkning

I nuvarande kod skalas data från Racam:s loggfil upp så att den får samma längd som data från IFV170+MRR2T-systemet. Nästa steg är att istället synkronisera informationen redan vid loggningstillfället genom att dela UDP-signaler till en switch som därefter för med sig dubbla strömmar in i var sin VDR. På det sättet skulle alla loggfilerna från båda system få information från motsvarande system att synkronisera mot i Matlab.

6.5.2 Mindre hårdvara

Det skulle kunna gå att endast använda en modifierad VDR som tar in loggar från båda RKS samtidigt. Detta hade potentiellt kunnat spara utrymme i lastbilen och minskat antalet kablar. Något som troligtvis hade kunnat ordnas med kort varsel vid mer arbete är att skära ner på en monitor så att endast en skärm kvarstår. Detta hade gett mer utrymme och en mer välslipad produkt.

6.5.3 Implementering av labeling i koden

För att utöka feldetekteringens trovärdighet så skulle nästa steg vara att även använda en loggfil med manuella labels. Denna skulle vara tidskrävande att införskaffa. Men skulle dock motsvara en så nära sanning som möjligt. I så fall kan jämförelsen ske mellan tre system.

6.6 Slutsats

Racam har för första gången installerats och fungerat i en lastbil. Samtidigt är det första gången som både Racam och IFV170+MRR2T-systemet har befunnit sig i en simultan loggning. Den jämförande koden i Matlab har visat sig fungera bra men det finns förbättringspotential. Huruvida det är mer tidseffektivt att använda en manuell metod för att hitta en ground truth beror på projektets längd och varaktighet. I ett längre projekt med många timmars loggning kan det absolut vara en bra lösning att använda denna automatiska ground truth för att få en bra överblick och spara tid. Om ändamålet är en kortare logg eller har få körtillfällen så kan den långa byggtiden med att installera två RKS inte vara värt det i slutändan. Detta är definitivt något som borde läggas mer tid på att undersöka och utveckla.

Referenser

[Aptiv] *Enligt önskan från Aptiv har vi utelämnat referenser för vissa produktnamn och från datablad.*

[1]. Mathworks, ”Matlab”, www.mathworks.com 2018 [Online] https://se.mathworks.com/products/matlab.html Hämtad: 2018-03-28.

[2]. NASA, “Radar E-O image fusion”, www.ntrs.nasa.gov 1993 [Online] https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/19940021011.pdf . Hämtad: 2018-05-28.

[3]. Techopedia, ”Ground Truth”, www.techopedia.com 2018 [Online] https://www.techopedia.com/definition/32514/ground-truth Använd (2018-06-13)

[4]. System plus Consulting, www.systemplus.fr 2016 [Online] http://www.systemplus.fr/reverse-costing-reports/delphi-integrated-radar-and-camera-system/ Hämtad: (2018-06-01)

[5]. Vector Informatik, www.vector.com 2018 [Online] https://vector.com/vi_hardware_en.html Hämtad (2018-05-15)

[6]. FreeCAD, https://www.freecadweb.org/ 2018 Använd (2018-05-15)

[7]. Tobias Green, Resident Engineer, Aptiv. Intervju (2018-04-12)

[8]. Mathias Merking, Resident Engineer, Aptiv. Intervju (2018-04-12)

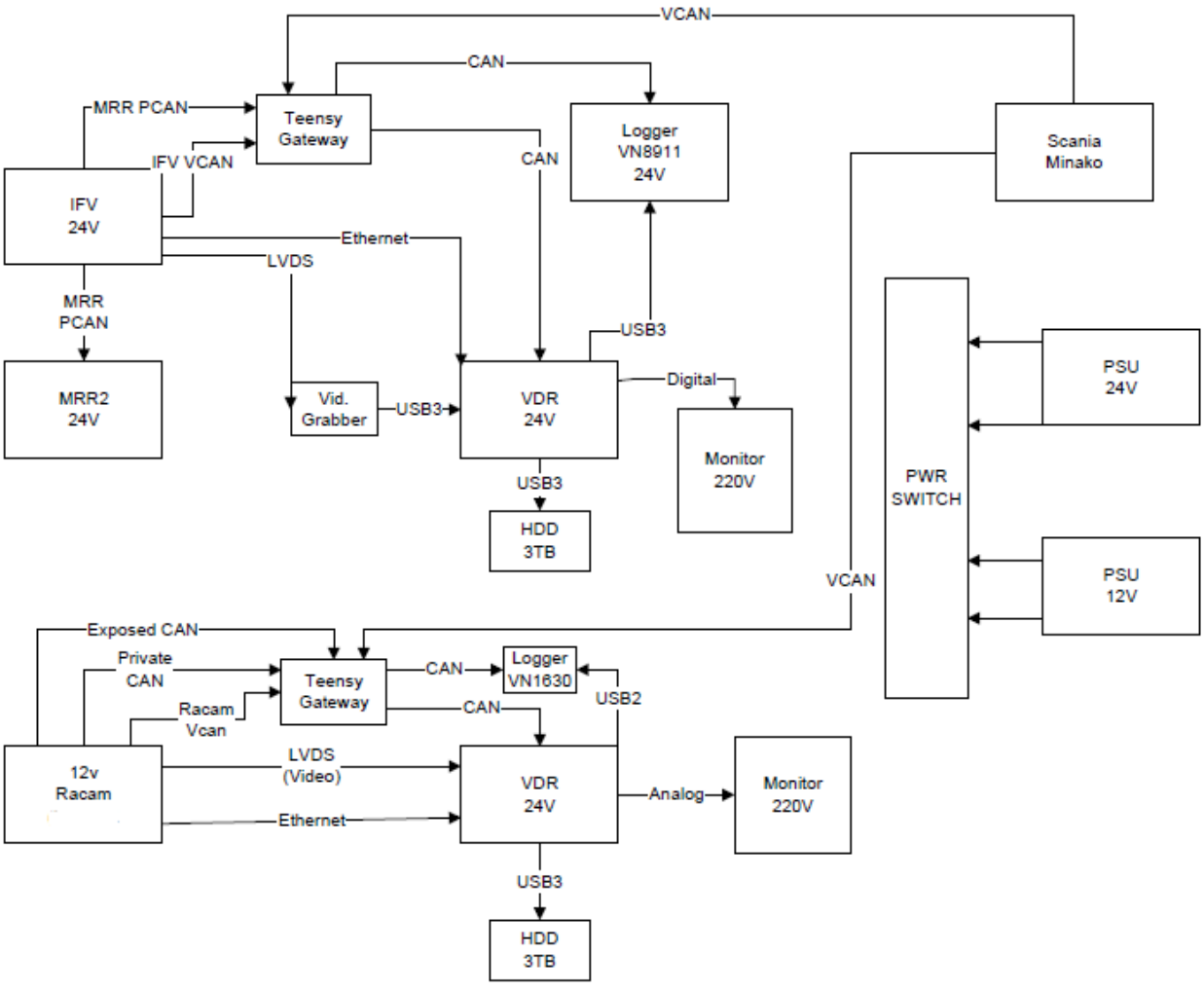
[9]. Martin Larsson System Engineer Aptiv Intervju den (2018-05-28)

[10]. PJRC www.pjrc.com 2018 [Online] https://www.pjrc.com/teensy/td_libs_Ethernet.html Använd (2018-05-31)

[11]. Ultimaker, https://ultimaker.com/en 2011 [Online] https://ultimaker.com/en/products/ultimaker-2-plus/specifications Använd (2018-05-15)

Bilagor

Bilaga 1. Kopplingsschema.



Bilaga 2. Matlabkod med jämförelse-script från loggfiler ur RAVA.

Initiering av adress och dataläsning är inte inkluderad.

```
%initiera plot
figure;
ax = gca;   %Get current axis
hold(ax, 'on')

ax.XLim = [-50 50];     %skala på X-axel
ax.YLim = [0 160];      %skala på Y-axel
values_to_find = [1,10];     %Object av intresse. CR,MC,TR,UV [1,2,3,10]

%time to collision
%cmbbPrimaryConfidence
%240 = 0
%105 = 1
%150 = 2
%195 = 3
cmp_object.TTC = [240 105 150 195];

%Variabelinitiering
ct_miss = 1;
ct_hyp = 1;
%Mainloop för hela RAVA_compare

%V1-----------------------------------------------------------
for iv = 1:length(mudpv.fus.status.lookIndex)
    %Initiering
    delete(ax.Children);
    vvv=1;
    rrr=1;
```

```
cmp_object.v1_pos = [];
cmp_object.racam_pos = [];


%%Plotta ut FoV för radar och kamera----------
x1= -23.5:23.5;
y1 = abs(1.732*x1);
plot(x1,y1,'-b')

x2= -13.4:0.2:-4.4;
y2 = abs(9.0036*x2);
plot(x2,y2,'-m')

x3= 4.4:0.2:13.4;
y3 = abs(9.0036*x3);
plot(x3,y3,'-m')

plot([-23.5 23.5], [40 40], '--b')
plot([-13.5 13.5], [120 120],'--m')
%-----------------------------------------------

%Loop för att stega ingeom varje objekt av intresse
for kv = 1:length(values_to_find)

    %Lägger aktuella objekt i variabel col
    [row,col] = find(mudpv.fus.Fus.fusTracks.object_class(iv,:) == values_to_find(kv));

    %Sorterar bort oncoming vehicles---------------------
    oncv = [];
    rad = 0;
    col_onc = col;
    for oncv = 1:length(col)
        if mudpv.fus.Fus.fusTracks.f_oncoming(iv,col(oncv)) == 1
            col_onc(oncv-rad) = [];
            rad = rad +1;
        end
    end
    col = col_onc;
    %-------------------------------------------------------

    %Sorterar bort objekt med för långt avstånd------------
    dist = [];
    del = 0;
    col_dist = col;
    for dist = 1:length(col)
        if mudpv.fus.Fus.fusTracks.vcs_long_posn(iv,col(dist)) > 120
            col_dist(dist-del) = [];
            del = del +1;
        end
```

```
        end
        col = col_dist;
        %-----------------------------------------------------

        %Sortera bort objekt utanför field of view---------------

        fov=[];
        fov_i = 0;
        fovd=0;
        col_fov=col;
        for fov = 1:length(col)
            if
abs((mudpv.fus.Fus.fusTracks.vcs_long_posn(iv,col(fov)))/(mudpv.fus.Fus.fusTracks.vcs_lat_posn(iv,col(
fov)))) < 1.732
                col_fov(fov-fovd) = [];
                fov_i = fov_i + 1;
                fovd=fovd+1;
            end
        end
        col = col_fov;

        %-----------------------------------------------------

        %Scatter används istället för plot
        scatter(ax,
mudpv.fus.Fus.fusTracks.vcs_lat_posn(iv,col),mudpv.fus.Fus.fusTracks.vcs_long_posn(iv,col), 'Or');
%Scatter för att plotta ut object
        for p=1:length(col)     %Loop för att printa ut alla kolumner i col
            %Behövs för att printa ut ID på object i plot
            text(ax,
mudpv.fus.Fus.fusTracks.vcs_lat_posn(iv,col(p)),mudpv.fus.Fus.fusTracks.vcs_long_posn(iv,col(p)),
num2str(col(p)))
        end

        %Placera object i struct med position och id---------------
        for vv = 1:length(col)
            cmp_object.v1_pos(vvv,1) = mudpv.fus.Fus.fusTracks.vcs_long_posn(iv,col(vv));
            cmp_object.v1_pos(vvv,2) = mudpv.fus.Fus.fusTracks.vcs_lat_posn(iv,col(vv));
            cmp_object.v1_pos(vvv,3) = col(vv);
            cmp_object.v1_pos(vvv,4) = mudpv.fus.Fus.fusTracks.cmbbPrimaryConfidence(iv,(col(vv)));
            for vvvv = 1:length(cmp_object.TTC)
                if mudpv.fus.Fus.fusTracks.cmbbPrimaryConfidence(iv,(col(vv))) ==
cmp_object.TTC(1,vvvv)
                    cmp_object.v1_pos(vvv,5) = vvvv-1;
                end
            end
            vvv=vvv+1;
        end
        %-----------------------------------------------------------
```

```
    end



    %RACAM-----------------------------------------------------

    %För att skala om till samma lookindex
    skala = (length(mudpr.fus.veh.host_speed)) / (length(mudpv.fus.veh.host_speed));
    ir=round(iv*skala);


    %Loop för att stega ingeom varje objekt av intresse
    for kr = 1:length(values_to_find)

        %Lägger aktuella objekt i variabel col
        [row,col] = find(mudpr.fus.Fus.fusTracks.object_class(ir,:) == values_to_find(kr));

        %Sorterar bort oncoming vehicles----------------------
        oncv = [];
        rad = 0;
        col_onc = col;
        for oncv = 1:length(col)
            if mudpr.fus.Fus.fusTracks.f_oncoming(ir,col(oncv)) == 1
                col_onc(oncv-rad) = [];
                rad = rad +1;
            end
        end
        col = col_onc;
        %-----------------------------------------------------

        %Sorterar bort objekt med för långt avstånd------------
        dist = [];
        del = 0;
        col_dist = col;
        for dist = 1:length(col)
            if mudpr.fus.Fus.fusTracks.vcs_long_posn(ir,col(dist)) > 120   %Sätter avstånd på övre
distans av jämförelse
                col_dist(dist-del) = [];
                del = del +1;
            end
        end
        col = col_dist;
        %-----------------------------------------------------

        %Sortera bort objekt utanför field of view---------------

        fov=[];
        fovd=0;
        col_fov=col;
```

```
        for fov = 1:length(col)
            if
abs((mudpr.fus.Fus.fusTracks.vcs_long_posn(ir,col(fov)))/(mudpv.fus.Fus.fusTracks.vcs_lat_posn(ir,col(
fov)))) < 1.6198  %Sätter en vinkelgräns
                col_fov(fov-fovd) = [];
                fovd=fovd+1;
            end
        end
        col = col_fov;

        %-------------------------------------------------------



        %Scatter används istället för plot
        scatter(ax,
mudpr.fus.Fus.fusTracks.vcs_lat_posn(ir,col),mudpr.fus.Fus.fusTracks.vcs_long_posn(ir,col), 'Xk');
%Scatter för att plotta ut object

        for p=1:length(col)     %Loop för att printa ut alla kolumner i col
            %Behövs för att printa ut ID på object i plot
            text(ax,
mudpr.fus.Fus.fusTracks.vcs_lat_posn(ir,col(p)),mudpr.fus.Fus.fusTracks.vcs_long_posn(ir,col(p)),
num2str(col(p)))
        end

        %Placera object i struct med position och id----------
        for rr = 1:length(col)
            cmp_object.racam_pos(rrr,1) = mudpr.fus.Fus.fusTracks.vcs_long_posn(ir,col(rr));
            cmp_object.racam_pos(rrr,2) = mudpr.fus.Fus.fusTracks.vcs_lat_posn(ir,col(rr));
            cmp_object.racam_pos(rrr,3) = col(rr);
            cmp_object.racam_pos(rrr,4) = mudpr.fus.Fus.fusTracks.cmbbPrimaryConfidence(ir,(col(rr)));
            for rrrr = 1:length(cmp_object.TTC)
                if mudpr.fus.Fus.fusTracks.tjaConfidence(ir,(col(rr))) == cmp_object.TTC(1,rrrr)
                    cmp_object.racam_pos(rrr,5) = rrrr-1;
                end
            end

            rrr=rrr+1;
        end
        %-------------------------------------------------------
    end

    %COMPARE

    %Töm structer från föregående
    cmp_object.h = [];
    cmp_object.h2 = [];
    cmp_object.h_list = [];
```

```
    %Om det finns objekt att jämföra
    if cmp_object.v1_pos

        liststepp = 1;
        flagg = 0;
        [vm,vn] = size(cmp_object.v1_pos);
        for hv1 = 1:vm
            %Räkna ut avstånd till alla objekt från racam
            [rm,rn] = size(cmp_object.racam_pos);
            for hracam = 1:rm
                %Pythagoras
                cmp_object.h(hracam,1) = sqrt(((cmp_object.v1_pos(hv1,1)-
cmp_object.racam_pos(hracam,1)).^2)+((cmp_object.v1_pos(hv1,2)-cmp_object.racam_pos(hracam,2)).^2));
                cmp_object.h(hracam,2) = cmp_object.v1_pos(hv1,3);           %Id v1
                cmp_object.h(hracam,3) = cmp_object.v1_pos(hv1,1);           %long v1
                cmp_object.h(hracam,4) = cmp_object.v1_pos(hv1,2);           %lat v1
                cmp_object.h(hracam,5) = cmp_object.v1_pos(hv1,5);           %TTC confidence v1
                cmp_object.h(hracam,6) = mudpv.fus.status.lookIndex(iv);        %lookindex v1
                cmp_object.h(hracam,7) = cmp_object.racam_pos(hracam,3);        %id racam
                cmp_object.h(hracam,8) = cmp_object.racam_pos(hracam,1);        %long racam
                cmp_object.h(hracam,9) = cmp_object.racam_pos(hracam,2);        %lat racam
                cmp_object.h(hracam,10) = cmp_object.racam_pos(hracam,5);       %TTC confidence racam
                cmp_object.h(hracam,11) = mudpr.fus.status.lookIndex(ir);       %lookindex racam
            end
            cmp_object.h = sortrows(cmp_object.h,1);                        %Sortera kortast avstånd
längst upp
            cmp_object.h_list(hv1,:) = cmp_object.h(1,:);                %Spara undan pga mest
lämplig parning


        end

        [cpm,cpn] = size(cmp_object.h_list);
        for cmp1 = 1:cpm
            if cmp_object.h_list(cmp1,1)>2                  %Kollar av lämplig avstånd mellan objekt
                cmp_object.h_fel(ct_hyp,:)=cmp_object.h_list(cmp1,:);   %Om avstånd överstiger så
sparas det objektet undan

                ct_hyp = ct_hyp+1;
            end
        end

    end
    drawnow();
    ax.XLabel.String = ['lookIndex IFV+MRR O: ', num2str(mudpv.fus.status.lookIndex(iv)), '  --
lookIndex RACAM X: ',num2str(mudpr.fus.status.lookIndex(ir))];   %Skriver ut label under plot
end
```