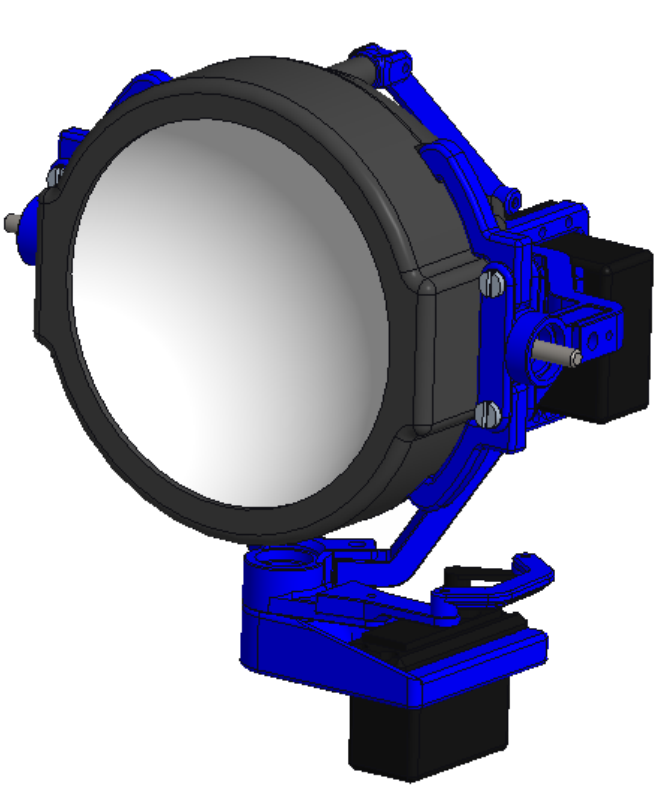
CHALMERS



Adaptivt styrsystem för applicering på extraljus till fordon

Styrsystem med CAN-bussavläsning och gyroskopbehandling som med användarjustering aktivt med hjälp av servomotorer vinklar extraljus.

Examensarbete inom data- och informationsteknik

ANDREAS NYBERG
OLIVER WALLIN

Institutionen för Data och informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2019

EXAMENSARBETE

Adaptivt styrsystem för applicering på extraljus till fordon

Styrsystem med CAN-buss avläsning och gyroskopbehandling som med användarjustering aktivt med hjälp av servomotorer vinklar extraljus.

ANDREAS NYBERG
OLIVER WALLIN

Institutionen för Data och informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg 2019

Adaptivt styrsystem för applicering på extraljus till fordon
Styrsystem med CAN-buss avläsning och gyroskopbehandling som med användarjustering aktivt med hjälp av servomotorer vinklar extraljus.

ANDREAS NYBERG

OLIVER WALLIN

Handledare: Sven Knutsson, *Chalmers tekniska högskola*
Björn Bergholm, *Broccoli Engineering*

Examinator: Peter Lundin, *Chalmers tekniska högskola*

Institutionen för Data- och Informationsteknik
Chalmers Tekniska Högskola
412 96 Göteborg
Telefon: 031-772 1000

Göteborgs Universitet
405 30 Göteborg
Telefon: 031-786 0000

Omslag:
3D-modeller av slutgiltiga prototyper

Institutionen för Data- och Informationsteknik
Göteborg 2019

Sammanfattning:

När man analyserar säkerhetsproblemen i trafiken är ett av de stora problemen synligheten. Föraren behöver god uppsyn över vägbanan och det närliggande området för att kunna se faror såsom dålig vägbana, vilt som är på väg upp på vägen eller människor som befinner sig i en utsatt situation.

I dagsläget använder man om man inte har tillräckligt god sikt extraljus som man placerar på bilen för att få ett större synfält vid mörker. Men användning av extraljus är långt ifrån optimalt, dels upplyses ett onödigt stort område upp, vilket riskerar att störa både närliggande bebyggelse och till större utsträckning blända medtrafikanterna.

För att upprätthålla god sikt utan att ha stora extraljus kan smarta extraljus implementeras istället, genom att använda mindre extraljus med en fokuserad ljuskägla som automatiskt vinklas mot det önskade området kan den goda sikten uppnås utan att behöva upplysa onödigt mycket.

I denna rapport så analyseras den bakomliggande dynamiken mellan fordonets framförsel och önskad ljusfokusering. Som används för att utveckla en användbar prototyp.

Den mekaniska prototypen tas fram och optimeras för att klara av påfrestningar såsom luftströmmen vid färd.

Det bakomliggande systemet tas fram för att ge föraren en användbar ljusbild framför sig vid färd med olika förutsättningar och vägtyper. Systemet använder sig av CAN-bussen och ett gyroskop för att läsa av och räkna ut hur bilen befinner sig på vägen.

Prototypen visar sig under testning fungera på ett bra sätt och följer vägen både i höjdled och sidled. Viss optimering krävs för att öka noggrannheten då det krävs en viss databank för att kunna förutsäga vägens utseende.

Nyckelord: Extraljus, trafiksäkerhet, CAN-buss

Abstract:

When analyzing the safety issues in traffic, one of the major concerns is the visibility. The driver needs a good view of the road surface and the nearby area to be able to see hazards such as poor roadway, wild animals that is walking onto the road or people who are in a vulnerable situation.

At present time, if you do not have enough good view of the road the first solution is to use auxiliary lights that you place on the car to get a larger field of view in the dark. But the use of auxiliary lights is far from optimal, and an unnecessarily large area is illuminated, which risks disturbing both nearby citizens and, to a greater extent, dazzling fellow road users.

In order to maintain good visibility without having large auxiliary lights, smart auxiliary lights can be implemented instead, by using smaller auxiliary lights with a focused light beam which is automatically angled towards the desired area, the good visibility can be achieved without having to illuminate unnecessarily large areas.

The underlying dynamics between the vehicle's driving performance and the desired light focus area are analyzed and used to develop a useful prototype.

The mechanical prototype is developed and optimized to cope with stresses such as the air flow and such during travel.

The underlying control system is developed to give the driver a useful light image in front of the car when traveling in different conditions and road types. The system uses the CAN bus and a gyroscope to read and calculate out how the car is moving on the road.

The prototype turns out to work well during testing and follows the road both vertically and sideways. Some optimization is required to increase the accuracy as it requires a certain database to be able to predict the road in front of the cars appearance.

Keywords: Auxiliary lights, traffic safety, CAN-bus

Förord:

I denna rapport presenteras utvecklingen från idé till prototyp och testning för ett adaptivt styrsystem för applicering på extraljus till fordon. Själva idéen uppenbarade sig på en slingrig grusväg under en mörk vinterkväll, då ljusbrist från fordonet var ett faktum.

Examensarbetet gjordes på Chalmers Tekniska Högskola där Andreas Nyberg och Oliver Wallin har studerat till högskoleingenjörer inom mekatronik. Mekatronikens huvuddelar är elektronik, mekanik, datorteknik, programmering och styrning, i kombination med hållbar utveckling och projektarbete.

Som tack till alla inblandande så vill vi först och främst skänka ett stort tack till Broccoli Engineering AB som åsidosatte tid och resurser så vi kunde verkligställa vår idé. Där Björn Bergholm och medarbetare har bidragit med nya infallsvinklar, information om CAN-buss, material samt goda diskussioner.

Från Chalmers vill vi tacka vår handledare Sven Knutsson som vi haft goda diskussioner med och hjälp oss med rapportskrivningen, Peter Bövik som hjälpte oss med momentberäkningar, klasskamrat Marcus Berg för hjälp när testning på bil skulle göras.

Vi vill även tacka Trafikverket som har bidragit med information om vägar, Transportstyrelsen som har bidragit med information om lagar och regler.

Hoppas ni får en trevlig läsning!
Andreas Nyberg, Oliver Wallin

Innehållsförteckning

Terminologi/Förkortningar:

NFC:

Near Field Communication är en överföringsmetod för att kontaktlöst utbyta data över kortare sträckor. Används i t.ex. hotellrumsnycklar och tunnelbanesystem. [1]

PWM:

PWM(Pulse width modulation) är ett tillvägagångsätt för att få ett analogt resultat med digitala signaler. Genom att modifiera en fyrkantsvåg och justera dess dutycycle kan ett nära analogt resultat fås. [2]

WIFI:

Ett familjenamn för olika radio teknologier som används för trådlösa lokala nätverk. Används flitigt i t.ex. datorer, bilar och drönare. [3]

Bluetooth:

Bluetooth är likt NFC en överföringsmetod för att utbyta data över kortare sträcker via radiobandet. [4]

EEPROM:

Electrically Erasable Programmable Read-Only Memory används för att lagra mindre mängder data i en mikrokontroller. Eeprom kan lagra data även när systemet inte är spänningssatt och kan raderas inte heller vid uppstart. [5]

SPI:

Serial Peripheral Interface är ett interface för att kunna kommunicera mellan en mikrokontroller och dess periferienheter. SPI använder sig av en klocka, en ingång och en utgång för att koppla upp sig mot periferienheterna. Sen har varje periferienhet en signal som sätts hög då den specifika periferienheten ska användas. [6]

I^2C:

I^2C använder sig det bästa från kommunikationsprotokollen SPI och UART. I^2C använder sig endast av en datasignal och en klocksignal för att kommunicera mellan mastern och de olika slaveenheterna. [7]

Mosfet:

Metal-oxide semiconductor field-effect transistor är en speciell typ av transistor som styrs av spännings istället för ström. Ju högre spänning som appliceras på mosfeten desto större ström tillåts passera. [8]

1. Inledning:

1.1 Bakgrund

Det följande examensarbetet görs i samarbete med konsultföretaget Broccoli Engineering AB som erbjuder tjänster inom hård- och mjukvaruutveckling. Företaget har ett 60-tal anställda konsulter utplacerade på olika företag i Sverige. Projektet kommer att utföras på plats på Broccolis kontor i Göteborg. Där kommer projektet kunna få tillgång till kunskap och resurser inom de områdena som speglar sig i projektet.

När fabriksmonterat helljus på fordonet upplevs för svagt så eftermonterar många fordonsägare så kallade extraljus. Funktionen av extraljus är att tillföra mer framåtriktat ljus från fordonet så att föraren kan se bättre när helljuset är påslaget vilket ökar trafiksäkerheten. Där det vanligaste montaget av extraljus är antingen 2–4 stycken runda extraljus eller en avlång så kallad LED-ramp. Dessa placeras på fronten av fordonet.

Projektet avser att utveckla ett optimerat extraljuspaket att placera på fordon. Detta är tänkt fungera genom att först utveckla ett system som läser av bilens CAN-bus och sedan använda den informationen för att bestämma vilken den optimala vinkeln för extraljusen är. Därefter skall extraljusen roteras eller tiltas. Ljuset ska primärt kunna följa vägen i kurvor, över krön och i svackor.

1.2 Syfte/Mål

Arbetets fundamentala idé handlar om att öka säkerheten i trafiken. Extraljusen som kompletterar de befintliga ljuslyktorna ska på ett smart sätt kunna justera sig själva efter vägen, föraren, vilt eller andra trafikanter. Detta genom att avläsa CAN-busen samt eventuellt genom externa sensorer.

Nyare bilar har till viss mån självjusterande lyktor, men deras vinkel samt funktionerna är begränsade. Därför är projektet uppgift att kartlägga och utveckla hur ett optimalt ljus system kan se ut samt bygga en prototyp för det.

Broccoli får då en utvecklingsplattform som underlag för framtida innovationer.

Projektets slutgiltiga mål är att ha en fungerande prototyp som är redo att montera på en bil och testköra. Prototypen ska ha mjukvara som är optimerad för att kunna styra ljuskäglorna på vägen med hänsyn till bilens olika beteenden.

Målen konkretiseras till följande:

Huvudmål:

- Ta fram en fungerande prototyp för ett smart extraljussystem.

Detaljmål:

- Effektivisera elförbrukningen jämfört med traditionella extraljussystem.
- Följa vägen väl efter dess karakteristerik.
- Justera extraljusen efter den avlästa informationen på CAN-bussen samt eventuellt från ytterligare sensorer.
- Användarpanel inne i bilen eller trådlöst till telefon för att ställa in olika lägen för systemet, t.ex. manuellt, auto landsväg, auto motorväg osv.

Optioner:

- Dimma ljuset efter omgivningsljuset.
- Automatiskt blända av vid möte.
- Rikta bort ljuset från skyltar för att förare inte skall bländas.
- Ta fram metod att läsa av flera olika CAN-bus rutiner.

1.3 Frågeställningar

- Hur många extraljus behövs?
- Hur snabb reglering krävs?
- Hur snabb reglering klarar systemet att uppnå?
- Hur påverkar fukt, vind, temperaturdifferenser systemet?
- Hur säkert är systemet för buggar?
- Vad kan förbättras/effektiviseras?
- Går det att göra systemet så det kan appliceras på olika bilar?

1.4 Etik

Problem som framkommer vid genomförandet av detta projekt är vid reglering av extraljusen. Om styrsystemet ger felaktig styrsignal och därmed gör så att extraljusens starka ljus bländar exempelvis mötande fordon så ökar risken för kollision. Samt att större risk finns för att skada någon mer alvarligt vid frontalkollision med människa/djur då extraljusen är monterade på fronten av fordonet. Därmed skall hänsyn tas under genomförandet så att systemet först kan provas i labbmiljö innan det används på väg. När funktionen är säkerställd skall testning på fordon göras på säkra ställen där ingen annan blir påverkad om systemet felar för att minska risken för olyckor. Designen på extraljushållaren skall även designas på så vis att den inte utgör större skada vid kollision än vad traditionellt montage av extraljus gör.

1.5 Avgränsningar

Den stora kunskapsavgränsningen är att CAN-bussavläsning inte är något som tillhandahållits i mekatronikprogrammet däremot är det kunskap som ges genom Broccoli.

CAN-bussavläsningen blir också projektets tekniska begränsning. CAN-busmeddelandena som bilen skickar ut läses av genom att man använder ett identifikations ID för att få fram den önskade informationen. Detta ID skiljer sig ofta mellan bilmodeller och kan vara svårt att ta fram. På grund av detta så kommer prototypen inte att vara skalbar för andra bilmodeller än de som arbetet har hunnit ta fram detta ID för.

Den mekaniska delen av projektet kommer avgränsas genom att ta bort utvecklingen av själva extraljusen och istället köpa färdiga extraljus godkända för allmän väg. Viss hänsyn kommer tas till fordonsreglerande lagar över faktorer som rörelse, ljusstyrka, referenstal mm på lyktorna. Antalet extraljus har begränsats till två stycken för att hålla kostnaderna för projektet låga.

Styrningen kommer baseras på en Arduino plattform för att ta hjälp av redan färdiga kodbibliotek (librarys) för att underlätta programmeringsarbetet.

Begränsad hänsyn kommer tas på prototypen gällande korrosions resistans och påverkan av isbildning som kan begränsa prestandan på systemet.

Beräkningar kommer göras för att identifiera lämpliga reglermotorer för denna prototyp utan hänsyn till massträghet. Hållfasighetsberäkningar på konstruktionen kommer inte beräknas.

2 Teknisk Bakgrund

2.1 Bakgrund extraljus

Figur 1. Extraljus Från [9]. Återgiven med tillstånd.

Extraljus/fjärrljus är eftermonterade lampor som skall tändas då fabriksmonterat helljus på personbil eller lastbil slås på. Syftet med extraljus är att komplettera existerande ljus så att antingen totala längden, totala bredden eller båda blir bättre. Ett extraljus använder sig av en ljuskälla och en reflektor för att rikta ljuset. Ljuskällan kan bestå av antingen halogenlampa, xenonlampa eller LED. Det finns två kategorier på ljusbilden från ett extraljus, pencil och flood. Pencil extraljus ger en längre punktformad ljusbild och flood extraljus ger en bredare och kortare ljusbild. Detta görs genom att reflektorns utseende skiljer.

Extraljusens fysiska utformning är antingen runda eller avlånga. Den runda varianten har oftast en diameter på cirka 175 till 225 mm medan dom avlånga har en höjd på ca 5cm och en längd från ca 10 cm upp till en meter. Där det vanligaste montaget av extraljus är antingen 2–4 stycken runda extraljus eller en avlång så kallad LED-ramp/ljusramp.

2.2 Servo

Servon som används i detta projekt är från radiostyrda bilar till främst styrningen på framhjulen. Dessa servon finns i olika storlekar beroende på hur stor bilen är och hur mycket moment som behövs. [10] Servon har oftast en rotation på cirka 90–180 grader på sin utgående axel. [11]

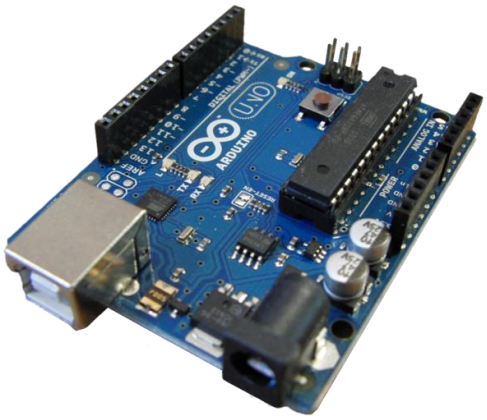
Figur 2. Servo Från [32]. CC BY-SA 2.0

Ett servo i sin helhet består av fyra huvuddelar. En motor, en växellåda, en potentiometer och en styrkrets. Där styrkretsen hantera ingående PWM (pulse width modulation) signal. Motorn är oftast av DC typen med ett högt varvtal och lågt moment. Via en växellåda som har ett antal kuggväxlar kan utväxlingen ändras, så att servots utgående axel roterar många gånger långsammare relativt motorns varvtal vilket gör att momentet ökar. En potentiometer är monterad på utgående axel så att servots styrkrets kan få information var i rotationen servot är i. [12]

Ett servo styrs av en periodisk PWM signal där tiden som signalen är hög bestämmer var servots utgående axel skall befinna sig. Är signalen hög i 1 millisekund så skall servot rotera max åt ett håll, om 2 millisekunder max åt andra hållet och om 1.5 millisekunder skall servot befinna sig i mitten. Dvs när servot beordras att flytta kommer de att flytta till den beräknade positionen och hålla den positionen. Om yttre krafter påverkar servots position så kommer potentiometer tillsammans med styrkretsen hantera denna kraften för att skicka en styrspänning till motorn så beräknade position behålls. [12]

2.3 Utvecklingsplattformen Arduino

Arduino är en utvecklingsplattform som är en mycket populär hos programmerarna på hobbynivå, utbildningssyfte i skolor men även hos experter. Enkelheten gör de möjligt att bygga och koda ihop en prototyp som kopplar samman den fysiska värden med den digitala på kort tid. Utvecklingsplattformen har öppen hårdvara och mjukvara (open source) vilket bidrar med att en tredje part kan använda ritningarna och källkod. PÅ detta sätt kan de bidra med sin egen version på utvecklingskorten och så kallade shields som är externa tilläggsmoduler. Det som kan differentiera mellan olika utvecklingskort är antal in-/ut gångar, processorkapacitet, minne och andra specialfunktioner som Bluetooth, WIFI, PWM, SPI I2C mm. [13] [14] [15]

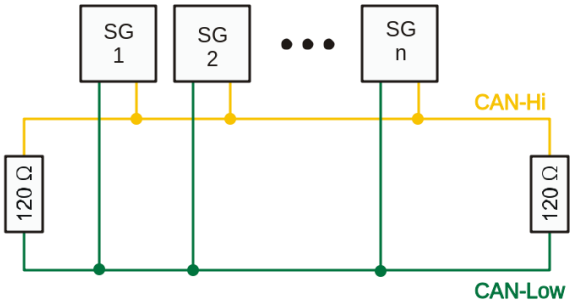


Figur 3. Arduino Uno. Från [33]. CC BY 2.0

För att programmera ett Arduino utevecklingskort så behövs Arduino IDE (Integrated Development Environment). Programmeringsspråket för applikationen är C och C++ och tack vara den öppna hårdvaran och mjukvaran så är utbudet av så kallade libraries (bibliotek) stort. Dessa bibliotek innehåller förprogrammerade funktioner för att uträtta den huvudfunktionen som vill uppnås. [16]

2.4 Controller Area Network - CAN-buss – förklara identifiera – signal – meddelande

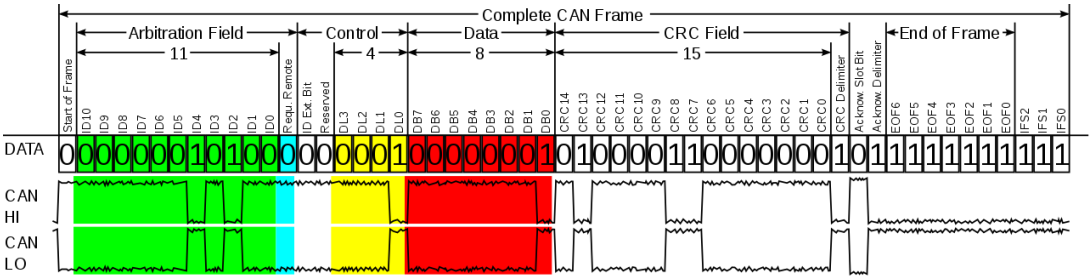
Bosch utvecklade under 1980 talet fram CAN-buss som ett alternativ för att föra över data seriellt mellan noder i ett fordon för att minska på kablage och för att stödja tillväxten av tekniken i fordonen. CAN-buss har i dagsläget blivit de alternativet som fordonstillverkarna oftast väjer för att kommunicera på grund av dess stabilitet, felhantering och överföringshastighet. Maximala hastighet för CAN-buss är 1 Mbps. [17] Fordonstillverkare brukar dela upp kommunikationen i exempelvis två nätverk för motorstyrning och infotainment. Där hastigheterna brukar vara på 500kbps för motorstyrningen och 125 kbps för infotainment. [18]



Figur 4. CAN-buss nätverk. Från [34]. CC BY-SA 3.0

Kommunikationen görs fysiskt seriellt vanligen över en tvinnad parledingen där ledningarna har benämningen CAN-High och CAN-Low. Parledingen har ett termineringsmotstånd på 120 ohm i varje ände på ledningen, detta för att påverkningen från störningar skall vara minimala. Noderna i nätverket som kan vara styrenheter, sensorer osv kopplas på bussen parallellt på parledingen. Se figur 4. [17]

Bitkodnigen är i non-return-to-zero (NRZ) och Carrier Sense Multiple Access/Collision Resolution (CSMA/CR) protokollet skyddar mot sändningar från flera olika noder samtidigt. När en nod skickar ut ett meddelande på bussen så kan alla noder på nätverket läsa meddelandet. De är upp till noderna själva i nätverket att filtrera bland meddelandena som skickas. Dvs noderna har inte specifika adresser. [17] [19]



Figur 5. CAN-buss meddelande. Från [20]. CC BY-SA 3.0

Ett CAN-buss meddelande finns i två olika format, antingen standardformatet vilket har en identifierare på 11-bitar (CAN 2.0A) eller utvidgade formatet vilket har en identifierare på 29-bitar (CAN 2.0B). I ovanstående figur ses ett typexempel på ett CAN-bussmedelande med standardformatet. Men det som är mest intressant för detta projekt är identifieraren (grön markerat i figur 5) samt data (röd markerat i figur 5). Det är även via identifieraren som mottagande noden filtrerar fram den datan noden vill åt. Prioriteten för utsändning av meddelandena görs även via identifieraren.

Datan innehåller informationen som skall skickas över nätverket där storleken på datan kan skifta mellan 0–64 bitar (0–8 bytes), vilket inte är mycket. I vissa fall kan de ske att en nod skickar ut data i flera meddelanden om noden skall skicka ut mer information. [17] [19]

Vissa meddelanden är standardiserat att de skall skickas ut på CAN-bussen. Dessa meddelandena går under SAE J1979 standarden och berör meddelanden som har med och kan påverka emissioner (avgaser) dvs motor relaterad information. [21] Resterande av meddelanden kan varje fordonstillverkare göra som de vill med vilket gör att hitta rätt identifierare och där efter avkoda datan svår om ingen databas finns. En databas gör så data kan översättas till värden som kan hanteras. Denna databas är oftast modellspecifik och kan även skifta mellan olika modellår på bilarna. [18]

2.5 Reglerande lagar för extraljus, helljus med mera.

Transportstyrelsen tillsammans med EU reglerar de lagar som berör fordon och där med extraljus. Transportstyrelsen definierar helljusstrålkastare som:

> ”Helljusstrålkastare - strålkastare som avger helljus och är avsedd att belysa vägen en lång sträcka framför fordonet; i begreppet ingår även kurvstrålkastare och fjärrstrålkastare”. [22]

I EU finns de begränsade lagar som kan läsas i ECE-reglemente 48 gällande maximalt antal extraljus, ljusstyrka och referenstal. [23] Dessa reglerna behövs inte uppfyllas i Sverige. Det som krävs är att extraljuset och ljuskällan skall uppfylla typgodkännande som gör enheterna E-märkta. [22]

I nedanstående paragrafer framgår ett axplock av installationskrav gällande extraljusen från TSFS2013:63 kapitel 21:

> *”20 § Bil som tas i bruk den 1 januari 2005 eller senare ska ha två eller fyra helljusstrålkastare framtill som avger vitt ljus.”* [22]
>
> *”21 § Bil som tagits i bruk före den 1 januari 2005, ska ha minst två helljusstrålkastare framtill som avger vitt eller gult ljus och som under 118 TSFS 2013:63 mörker och vid klar sikt kan belysa vägen på en sträcka av minst 100 m framför fordonet.”* [22]
>
> *”23 § Bil får dessutom ha ytterligare helljusstrålkastare som består av fjärr eller kurvstrålkastare.”* [22]
>
> *”25 § Helljusstrålkastare och ljuskälla till sådan strålkastare får även vara typgodkänd enligt ECE-reglemente 98 (se 3 kap. 1 §) och ECE-reglemente 99 (se 3 kap. 1 §).”* [22]
>
> *”28§ Helljusstrålkastare ska vara justerbar.”* [22]
>
> *”30§ Helljusstrålkastare ska vara så inkopplad i fordonets elektriska system att den omedelbart slocknar vid omkoppling från helljus till halvljus. ”* [22]

Samt från förordningen 2001:650 om vägtrafikregister fås information om registreringsskylten:

> *”8§ Registreringsskyltarna ska vara väl synliga och hållas i sådant skick att de lätt kan avläsas. Under färd får last eller annat inte placeras så att skyltarna inte går att avläsa.”* [24]

3 Metodbeskrivning

Projektet inleds med en idégenerering gällande den grundläggande designen på hållaren till extraljusen, lämplig hårdvara, val av bilar att analysera för att bygga upp en kunskapsbas. Kontakt med transportstyrelsen krävs för att få exakt data på vilka avgränsningar som finns från deras sida. Därefter tas ett beslut över vilka komponenter som krävs och en exakt detaljplan sätts. Komponenter beställs.

Projektet fortsätter med att göra en datainsamling på bilar för att kunna utreda vilka data som kan användas i projektet. CAN-busen loggas samt spelas in för att i testmiljö kunna spela upp datan för systemet så att prototypen ska kunna testas i labbmiljö.

Prototypens mekanik ritas upp i CAD för att lägga en grund för beräkningar angående, materialval, hållfasthet, styrutrustning mm. Därefter byggs den ihop enligt designen.

Mjukvaran utvecklas så den går att skala upp med tiden.

När enskilda delar i systemet är färdiga så görs en testkörning av komponenten.

Slutligen när allt är klart görs en grundlig testning och kontroll för att se så allt fungerar som önskat.

3.1 Användarpanel

För att den slutgiltiga produkten ska underlätta för användaren måste gränssnittet mellan människa och maskin fungera på ett smart och smidigt sätt. Därför är det viktigt att se till att användarpanelen uppfyller alla krav som användaren kan tänkas ha. Dessa behov kan innebära allt från tillförlitligheten på produkten, lagom komplexitet vilket i sin tur innebär en lättanvänd produkt samt produktens estetik.

Under projektets inledande projektering togs ett antal alternativ fram som möjliga typer av användarpaneler:

- **16x2 teckens LCD display:**
  LCD displayen är tänkt att monteras inom räckhåll för föraren i bilen och ska ge användaren möjlighet att med en rotary encoder ”scrolla” igenom menyer för att justera inställningarna på ljus systemet. Denna lösning kräver att projektet konstruerar och bygger en inbyggnad av skärmen.
- **4,7’’ touchskärm:**
  Touchskärm ger större möjligheter för att anpassa skärmen till det yttersta. Säkerhetsaspekten kommer in här då lösningen inte får dra användarens ögon från vägen utan måste var tillräckligt intuitivt för att ge möjligheten att göra små justeringar snabbt och enkelt. Denna lösning är ur en produktionssynvinkel dyrare än LCD displayen samt kräver att projektet konstruerar och bygger en inbyggnad av skärmen.
- **Mobiltelefon applikation:**
  Genom att använda användarens mobiltelefon drar man ner på komponentkostnader då man inte behöver bygga in en skärm utan endast en Bluetooth, NFC eller WiFi mottagare. Denna lösningen kräver att projektet konstruerar och bygger applikation.

Alternativen behöver nu jämföras och klassificeras jämtemot varandra, för att göra detta måste de olika inställningarna som användaren har att ändra tas fram. För att vara på säkra sidan så används alla möjliga inställningar som faktorer vid klassificeringen. Därför är det inte säkert att alla av de kommande inställningarna kommer att finnas i den slutgiltiga produkten.

- **Mode:**
  Här ska användare kunna ställa in vilket läge systemet ska vara i. Detta kan påverka känsligheten på lamporna eller liknande. Man ska också kunna ha ett manuellt läge där man själv ställer in statiska vinklar på lamporna.
- **Manuell justering:**
  Som tidigare nämnt i ”mode” så ska användaren ha möjlighet att ställa in en statisk vinkel som när inställningen är gjord förflyttar lyktorna till den angivna vinkeln.
- **Bilmodell:**
  När CAN-busen läses av så fungerar det inte att använda samma identifierare för två olika typer av bilar. Därför måste användaren välja vilken bil som används så systemet använder rätt algoritm för att få in den önskade datan. Detta kommer vara en inställning som troligtvis bara behövs göras en gång.
- **Avbländning:**
  Ger användaren möjlighet att blända av lyktorna, men detta kommer troligtvis tas från CAN-bussen så det är inte en högprioriterad inställning.

Återigen för att göra den säkraste lösningen så ska inte användaren behöva använda panelen under färd. Alternativt ska systemet aktivt motverka föraren att använda systemet. Samtidigt är nästan ingen av inställningarna av den typ som behövs justeras regelbundet, därför är det svårt att motivera valet av både LCD och touchskärm som alternativ till användarpanel. Båda alternativen kräver även mer resurser under utvecklingsfasen och det leder till att man måste ha extern utrustning i kupén som knappt används.

Därför finns det inget annat alternativ som motiverar användarpanelen att vara konstruerad på något annat än en mobilapplikation.

3.2 Infästning till fordonet

För att montera extraljusen finns två alternativ, på fronten eller på taket av fordonet. Där hänsyn skall tas till att montage är möjlig för olika bilmodeller, bra aerodynamik, estetik och etik.

- **Montage på takräckena på bilen**

Att montera på takräckena på bilen kan vara fördelaktigt på så vis att de lyser från en hög punkt vilket därmed ger att ljuset kan sträcka sig längre. Nackdelar är att montaget blir modellspecifikt vilket gör att en fästanordning måste konstrueras för varje bilmodell vilket gör anordningen dyr och komplex. En annan nackdel är aerodynamiken försämras när extraljusen monteras på taket vilket i sin tur ökar bilens bränsleförbrukning.

- **Montage på fronten på bilen**

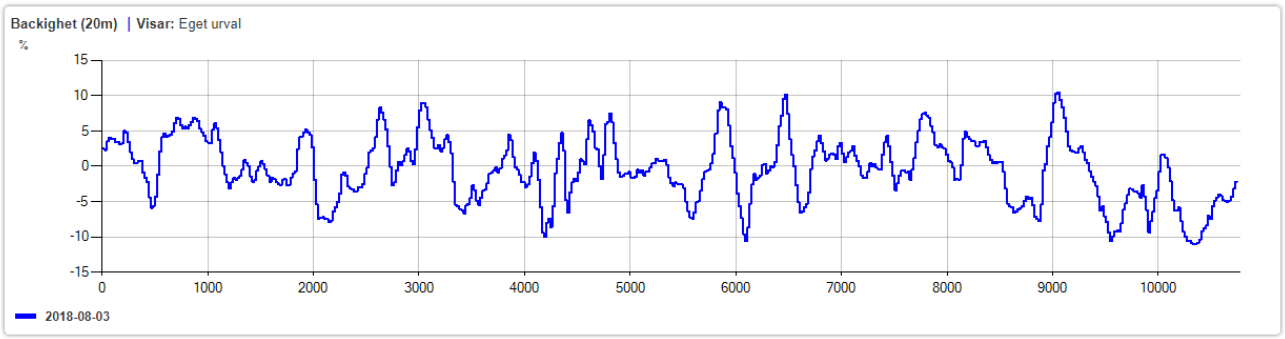
Med montage i fronten på bilen så behövs mindre hänsyn tas till att göra konstruktionen specifik för varje bilmodell. Därmed kan en universalmodell användas för att täcka att stort antal av olika bilmodeller vilket även gör de billigare om systemet skulle tas i produktion. Ur aerodynamisk synpunkt finns även fördelar då luftmotståndet blir lägre då extraljusen sitter mer tätt monterat mot bilen jämfört med takmontaget. Detta alternativet är också mer vanligt att se vid montage av extraljus. Vilket gör att fordonsägarna är mer vana att montera på detta sätt.

Under utvecklingsarbetet är det tänkt att montera extraljusen på taket på bilen via takräcken. Detta för att slippa borra och skruva i bilens stötfångare eftersom ändringar kan ske under projektets gång. Att ha extraljusen på taket gör det även lättare att se hur extraljusen rör sig via bilens taklucka.

Prototypen skall vara anpassad för att kunna montera extraljusen på fronten av fordonet. Då denna monteringspunkt är mer en populär position att montera extraljusen på samt ur aerodynamisk synpunkt är detta bättre.

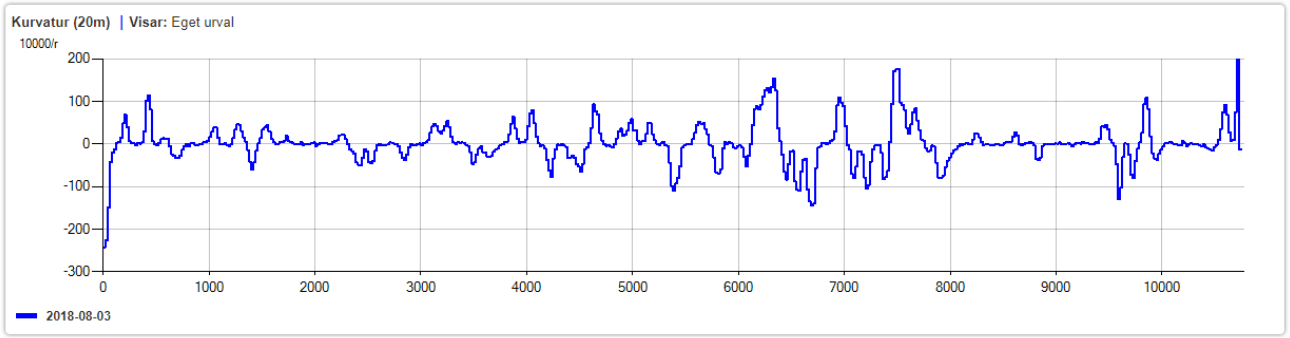
3.3 Mekaniken för extraljusen

För att beräkna vilken maxvinkel som ljusen behöver rotera samt tilta så gjordes en analys av Härskogsvägen vägnummer 523 i Härryda kommun med hjälp av Trafikverkets verktyg PMSV3 som i grunden är information om Sveriges belagda allmänna vägar. [25] Denna väg valdes då den har en hög brackighet och hög kurvatur vilket gör den optimal för att ge extremvärden till detta projekt.



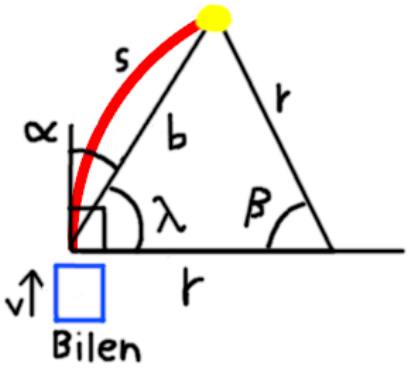
Figur 6. Backighet Härskogsvägen. Från [26].Återgiven med tillstånd.

Som kan ses i figur 6 så har Härskogsvägen en maximal brackighet på cirka 10%, omräknat till grader cirka 6 grader (positiva värden uppförsbacke, negativa värden nedförsbacke). För lite extra marginal så valdes så att tiltningen på extraljusen skall klara av minst 10° åt varje håll.



Figur 7. Kurvatur Härskogsvägen. Från [27]. Återgiven med tillstånd.

Som kan ses i figur 7 så är Härskogsvägen en kurvatur med extremvärden på cirka 170 enheter i grafen (positiva värden vänsterkurva, negativa värden högerkurva), vilket omräknat blir en radie på cirka 58 meter. Skyltad hastighet på Härskogsvägen är 50-70km/h. Via ekvationen nedan så fås en uppskattning av rotationsvinkel som extraljusen behöver.



Figur 8. Rotationsvinkel

I figuren ovanför så representeras bilen den blå rektangel, vägen den röda linjen och var ljuset skall placeras av den gula punkten. För att ge en uppskattning av rotationsvinkeln som behövs så antas att ljuset skall befinna sig 3 sekunder före bilen. 3 sekunder har valts då detta ger ett säkert avstånd att stanna på om hinder på vägen skulle framkomma.

$$S = v * t = \frac{70}{3{,}6} * 3 \approx 58\, meter$$

$$Procentuell\ andelen\ av\ halvcrikeln = \frac{S}{\frac{2r * \pi}{2}} \approx \frac{58}{182} \approx 0{,}31$$

$$Där\ med\ blir\ \beta \approx 180° * 0{,}31 \approx 55°$$

Sträckan b beräknas via cosinussatsen:

$$b^2 = r^2 + r^2 - 2 * r * r * \cos\beta \Rightarrow\ b \approx 53\, meter$$

Vinkeln α tas fram via vinkeln λ genom sinussatsen:

$$\frac{\sin\lambda}{r} = \frac{\sin\beta}{b} \Rightarrow \sin\lambda = \frac{\mathrm{r} * \sin\beta}{b} \Rightarrow \lambda \approx 74° \Rightarrow\ \alpha \approx 90° - \lambda \approx 90° - 63° \approx 27° \approx 30°$$

Där med kommer 10 grader för tiltningsvinkel och 30 grader för rotationsvinkel användas som utgångspunkt. Vidare tester på verklig hårdvara skall utföras för att undersöka om dessa värden behöver ändras.

3.4 Reglermotorer

För att ljuskäglan ska hamna där elektroniken avser att rika det så behövs en mekanik som är stadig samt har hög pression. Mekaniken skall inte ska uppta för mycket plats, inte vara för komplex samt inte kräva överdrivet starka reglermotorer.

Som alternativ som reglermotor finns tre alternativ som kan stå upp till kraven. Till valet av reglermotor så behöver hänsyn behövs tas till att extraljusen i sig skall roteras 30 grader samt tiltas 10 grader.

- **Stegmotor**

  Stegmotorns fördelar är att den är väldigt precisa i sin rotation men inte har så mycket moment. Vilket leder till att en nedväxling med en växellåda/kuggväxel krävs för att få upp styrkan. Stegmotorer saknar någon typ av positionering vilket i så fall behövs kompletteras med för att tillföra positioneringen som krävs i detta projekt. Därmed behövs även en kompletterande styrkrets för att tillföra positioneringen. En stegmotor har inga märkbara förslitningar om den används under lång tid.

- **DC motor med inbyggd växellåda**

  En DC motor med sin växellåda har fördelen att den har mer styrka än en stegmotor. Men även denna motor har samma nackdel då den saknar inbyggd positionering och kräver en styrkrets. En DC motor är dock inte lika precis i sin rotation som en stegmotor samt kan slitas ut vid lång användning då kolen i motorn kan ta slut.

- **Servo**

  Ett servo har sin fördel att den är starka men även har positionering inbyggt samt att den inte kräver någon styrkrets då den regleras via en PWM signal. Rotationen är begränsad i servot till totalt cirka 180 graders rörelse vilket gör att ändpunkts hantering inte behövs. Utseendet av konstruktionen måste dock ta i hänsyn då den begränsade rotationsrörelsen från servot.

Ovannämnda tre alternativ anses var fördelaktiga på sina sätt beroende på hur konstruktionen av hållaren bestäms till.

3.5 Konceptgenerering och avgränsning av koncept

Med tanke på avgränsningen av antalet extraljus är två stycken kan maximalt två olika koncept användas, en till vänster och en till höger.

För att få fram ett bra koncept för extraljusen ur estetik och funktionsmässig synpunkt så börjas det med en brainstorming, i denna fas är inga idéer dåliga idéer. Brainstormingen skall göras till minst sex koncept finns.

Därefter görs en viktad Pughs matris för att poängsätta koncepten vilket därmed eliminerar de alternativen som är mindre lämpade. I Pughs matrisen så används ett koncept som referens och resterande koncepten jämförs med denna referens utifrån kriterium. Helst så skall referensen väljas till de konceptet som anses vara det mellersta alternativet, så att en objektiv jämförelse kan göras. [28] Kriterium som alternativen jämförs mot har en viktning från 1 till 10, där 1 är ”inte viktig” och 10 är ”mycket viktig”. Vid jämförelsen så kan något av ”+/-/0” väljas. Därefter summeras poängen och den med högst poäng väljs till att gå vidare med. Se figuren nedan för ett exempel på viktad Pughs matris.

	Viktning	Referens Koncept 1	Koncept 2	Koncept 3
Kriterium 1	1	0	+	0
Kriterium 2	10	0	0	+
Kriterium 3	4	0	-	-
Summa +		0	1	10
Summa -		0	4	4
Summa		0	-3	6
Gå vidare		Nej	Nej	Ja

Figur 9. Exempel Pughs matris

Vinner ett alternativ överlägset så skall detta koncept konstrueras för båda extraljusen. Om två koncept får poängmässigt nästan identiskt resultat så skall dessa två alternativ konstrueras. Om fler än två alternativ får samma resultat så skall en ytterligera Pughs matris göras, där dåligt presterande koncept skall tas bort och en ny referens skall användas.

3.6 Analysera CAN-bussen

För valet av analyseringsmetod av informationen som finns på CAN-bussen krävs en bra och smidig metod. Alternativen är många men eftersom tidsåtgången och verifieringsmöjligheterna måste tas till hänsyn så begränsas alternativen. De alternativen som återstår är antingen att koppla upp sig med en dator med ett CAN-bussinterface och ett datorprogram som läser av eller att bygga ihop en egen konstruktion bestående av en Arduino med tillhörda CAN-buss komponenter.

- **Analyseringsprogram och färdigt CAN-buss interface**
  Denna analysmetod är färdigprogrammerad och hårdvaran finns att köpa färdig. Fördelarna hos detta alternativ är att loggning samt uppspelning av loggar är möjlig samt ett gränssnitt finns för att rita grafer så att analys av datan kan göras visuellt. Största fördelen är att det är lätt att med en tillhörde databas hitta rätt identifierare som behövs under projektet. En databas gör så att data kan översättas till värden som lättare kan hanteras. Möjligheten att verifiera datan är stor, då programmet och hårdvaran anses vara säker eftersom det är professionella komponenter.

- **Arduino som grundhårdvara**
  Att använda en Arduino som grundhårdvara är fördelaktigt för de ger kunskap om hur systemet skall programmeras i senare i projektet skede. Analysmetoden går ut på att testa sig fram, då möjligheten att applicera en databas är svår. Nackdelen är att det kan vara tidskrävande att skriva koden samt risken finns att datan hanteras fel. Utan någon databas blir verifiering av datan är svår att göra.

För avläsning av CAN-bussen valdes alternativet med en dator och ett analyseringsprogram samt ett färdigt CAN-bussinterface. Då tillförlitligheten är hög, en databas kan lätt kopplas till och den är mindre tidskrävande än Arduino alternativet. Datorprogrammet som valdes är CANalyser och ett interface från Kvaser. Då Broccoli redan hade detta program och hårdvara som kunde lånas under projektets gång. Övriga analyseringsprogram föll bort då dessa program är väldigt dyra vid inköp om den ska ha samma funktioner som CANalyser. Personalen på Broccoli har även goda kunskaper om CANalyser samt har en databas till den bilen som lånades för att genomföra detta projekt.

Inkopplingen till bilens CAN-buss nätverk görs via ett uttag som finns vid pedalerna på förarsidan av bilen. Via detta uttag så skall CAN-bussen undersökas genom att koppla upp datorn med CAN-buss interfacet. En aktivitet skall göras med bilen som exempelvis rotera på ratten, därefter undersöks aktiviteten med datorn efter den identifierare som har med positioneringen av ratten. Loggar görs genom att utföra en testkörning, där all data som skickas på CAN-bussen spelas in. Denna logg kan därefter spelas upp (skickas) till Arduinon i labbmiljö.

3.7 Programmering

Projektets fundamentala huvudfunktioner ligger i programmeringen. Det är kodningen som knyter ihop alla inkommande data såsom t.ex. CAN-bussmeddelandena och gyroskopets signaler för att styra ljusen på ett önskvärt sätt.

Sedan tidigare har Arduino valts som plattform för att utveckla projektet vilket innebär att projektet kodas i programmeringsspråket C. Arduinon erbjuder också en mängd så kallade ”librarys” eller bibliotek med redan skriven kod som underlättar projektets programmering. Konkret så undviks alla djupa registerskrivningar och istället behöver man endast fokusera på programmets funktion.

Programmeringserfarenhet för programmeringsspråket C finns inom gruppen sedan tidigare, projektets inledande fas gick ut på att bygga upp ett kodskelett som tillkommande komponenter och funktioner enkelt kan appliceras på. Initialt i projektet ska också en form av diagnostikfunktion utformas för att underlätta för felsökning under projektet, funktionen ska innehålla flaggsättningar, rådata från periferienheterna samt användarens indata.

3.8 Miljöpåverkan

Inom ramen för miljöpåverkan så finns de tre påverkande faktorer: tillverkning, användning och återvinning. Där systemet skall undersökas om minskad energiåtgång vid användning är möjlig jämfört med traditionellt montage av extraljus. Detta via att systemet har effektivare placering av ljuskäglorna med anseende på vägens karakteristik. För att undersöka om systemet med två extraljus är mer energieffektivt skall systemet jämföras med ett antal stationära extraljus. Där antalet stationära extraljus väljs till ett antal som ger samma effektiva ljus som systemet med rörliga.

Miljöpåverkan vid tillverkning och hur möjligheten till återvinning är största del beroende på produktval/materialval och eftersom syftet med detta projekt är att undersöka teorin inte massproducera, så överlämnas faktorerna tillverkning och återvinning till framtida arbeten. Prototyperna tillverkas av PLA plast med hjälp av 3D-skrivare. PLA är till stor del återvinningsbar, genom att ”mala” ner plasten går det att återigen smälta ner det till den form där den återigen kan 3d-printas.

4 Val av komponenter

4.1 Användarpanel:

Som nämnt i rapportens sektion 3.1 bestäms det att en mobilapplikation ska vara kontakten mellan användaren och systemet.

4.1.1 Bluetooth modul

För att etablera kontakten mellan mobilen och Arduino som systemet ska baseras på behöver en modul komplettera Arduino. Valet att använda Bluetooth som överföringsplattform kommer naturligt då både NFC och WiFi inte anses lämpliga för systemet.

På grund av projektets begränsade tidsram utvecklas inte Bluetooth kretsen utan köps in som en färdig modul. Modulen som anses vara mest lämplig är **HC-06** som finns tillgänglig i svenska elektronikbutiker. Modulen har även som har stort litterärt stöd samt att det finns Arduino bibliotek vilket underlättar under programmeringsprocessen.

4.1.2 App Development Environment

Projektgruppen har sedan tidigare erfarenhet av att bygga applikationer till mobiltelefoner. För att utveckla denna app väljs därför **”MIT App Inventor 2”** som erfarenheten finns i. MIT App Inventor 2 är programmeringsmiljö där man dels utseendemässigt komponerar skärmens utseende samt genom blockprogrammering programmerar vad applikationen ska göra. För detta projektet är applikationens huvuduppgift att skicka användarens data över Bluetooth till Arduinon.

4.2 Val av utvecklingsplattform samt tillhörande komponenter

4.2.1 Arduino modell

Arduinon valdes till modellen Uno R3. Då utbudet av färdiga tilläggsmoduler (shield) är stort vilket gör att utvecklingsarbetet går snabbt och smidigt. Ardinon Uno använder sig av en ATmega328 processor vilket är kraftfullt nog för detta projekt, har tillräckligt med in/ut-gångar och klarar av SPI samt I^2C kommunikation som används i detta projekt.

4.2.2 Canbus modul

En Canbus modul i shield-format från Sparkfun valdes då denna modul går lätt att kopplas samman med Aruinon samt stöder CAN-busskommunikation upp till 1Mb/s. [29] Detta kort kommer hantera kommunikationen från CAN-bussen i bilen till Arduinon. Modulen kommunicerar med SPI med Arduinon

4.2.3 Gyroskop

Då bilen som används i detta projekt inte har en signal för lutningen på bilen jämfört med horisontell nivå så krävs en ytterligare modul. Det behövs ett gyroskop för att få fram lutningen. Gyroskoptet som ansågs mest lämpligt var en MCP-6050 som komuniserar med I^2C med Arduinon. Beroende på att gruppen har erfarenhet av detta gyroskop sedan tidigare.

4.3 Val av extraljus

Till projekt valdes extraljus på 176 mm i diameter från Biltema, artikelnummer 42-330. Extraljuset valdes då ljusbilden är punktformad samt att två alternativa infästningar är möjliga, antingen i foten eller på baksida. Samt att linsen går att vända på så extraljuset går att monteras upp och ner. Detta gör att den är en mycket bra kandidat för att representera större mängd extraljus bland de extraljusmodeller som finns på marknaden idag.

4.4 Val av reglermotorer

Till valet av reglermotorer ansågs servon vara mest lämpad. Då dessa har fördelen att kunna leverera mycket moment med tanke på sin kompakta storlek samt att rotationen på cirka 180 grader som passar konstruktionsdesignen som koncept 5 och 6 har, se sektion 5.4.1 och 5.4.2. För varje extraljus kommer två servon användas. Huvudsakliga anledningarna till att stegmotorer och DC motor med växellåda valdes bort som alternativ är att styrkretsar behövs vilket ökar komplexiteten.

Största påverkande faktorn för reglermotorerna är luftmotståndet, där med väljs maximal hastighet till 70km/h (ungefär 20 m/s) då denna protypen inte skall köras under höghastighets förhållande.

Via ekvationerna i bilaga 2 fås maximalt momentet som krävs för att rotera samt tilta extraljusen:

- Koncept 5: 3 Nm respektive 1 Nm
- Koncept 6: 1 Nm respektive 1 Nm.

Med avseende på resulterande moment och konstruktionsdesignen så valdes servot till Turnigy TGY-S8166M [30]. Då servot har ett utgående moment på 3 Nm och har små yttre dimensioner. Servots extra moment används som säkerhetsmarginal eftersom massträgheten hos extraljusen och konstruktionen inte togs till hänsyn vid momentberäkningarna.

4.5 Val av diverse resterande komponenter

Ett universal fäste som ser ut som en L-balk för två extraljus valdes att köpas färdigt. Då den bidrar till en bra grund att fästa extraljusen.

Till lagrade leder valdes radiallager av typen 6702-2RS (15x21x4 mm). Dessa lager valdes då dom kommer funka väl till den 3D utskrivna delarna och har ett bra pris. Optimalt vore att välja axiallager, men kostnaden för dessa gjorde att alternativet fall bort.

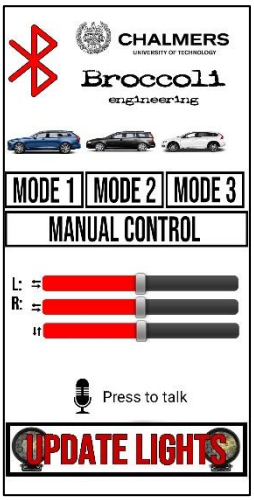
Skruvar, muttrar, brickor med mera valdes utefter konstruktionsdesignen krävde. Hänsyn togs till att de skulle hålla, därmed överdimensionerades dimensionerna på många punkter.

5 Konstruktion/Systemkonstruktion

5.5 Användarpanel/Applikation

5.5.1 MIT App Inventor 2

Applikationen konstruerades i MIT App Inventor 2 som är en online baserad tjänst som tillåter en att både programmera och designa applikationen genom ”drag’n’drop” tekniker. Stor vikt läggs för att göra appen lättanvänd och intuitiv, detta görs med enkla färger och tydliga knappar.



Figur 10. App hemskärm

Appen ska ge användaren möjlighet att över Bluetooth välja vilken bilmodell systemet är applicerat på, vilket ”mode” som ska användas samt justera inställningarna för vinklarna vid manuell styrning av ljusen. En option för appen är att göra den röststyrd.

Designmässigt utvecklades appen runt ett enkelt tema. Där istället det visuella fokuset lades på att göra det tydligt för användaren vilka val som var gjorda i appen genom att visa det med en tydlig kontrastfärg.

Därefter programmerades inlagda knappar och visuella reglage för att utföra den funktion som önskas. I **bilaga 1** återfinns den kompletta programmeringen för applikationen.

Den gröna rektangeln i **bilaga 1** visar initieringsdelen av appen. Här initieras dels variablerna och arrayer nollställs. Dessutom finns upprättningen av bluetoothkommunikationen här. När Bluetooth ikonen trycks hämtar mobilen en lista på alla tillgängliga Bluetooth apparater i närheten, därefter får användaren välja i listan vilken dom vill ansluta till. Därefter skapar appen en kommunikation med modulen på Arduinon och ikonen ändrar färg från röd till grön.

Vidare så innehåller den gula rektangeln kommandona för vad som sker när de olika knapparna trycks. Det som händer på vid aktivering av olika knappar är att en funktion kallas på, dessa funktionerna kommer förklaras senare i texten.

I den nästa röda rektangeln hittar man röststyrningen. När mikrofonen trycks läser mobilen in det du sagt och jämför det de inlagda kommandona som återfinns i kodblocken. Därefter uppdateras texten på skärmen vad som sagts och rätt funktion kallas på.

Majoriteten av funktinorna finns i den mörkblå rektangeln. Funktionerna ändrar bild så att den knapp man tryckt blir markerad i den kategorien den ingår i. Dessutom så ändras variabelvärdet. Här finns också funktionen *update_array* som lägger in alla variablerna samt positionen som dragaxlarna har i en array.

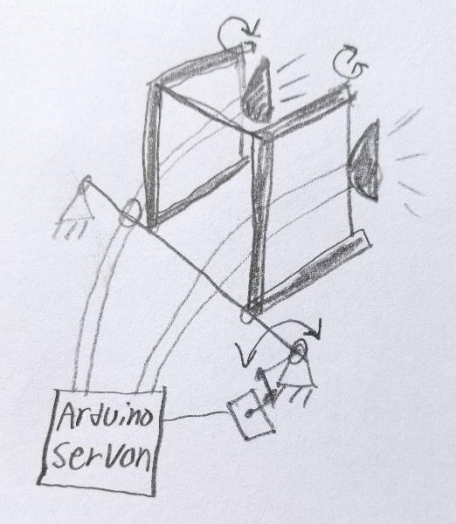
Sist är den ljusblåa rektangeln. Funktionen *update_lights* räknar först ut accelerationen som mobilen utsätts för, detta föra att förhindra användning av mobilen vid färd i bilen. Om accelerationsberäkningen ger att positivt resultat utförs den tidigare nämnda funktionen som uppdaterar arrayen. Sedan om bluetooth har kontakt så skickas arrayen som en text vilket ger ett utseende som följande ”{1 1 125 125 125}”, alltså skiljs varje variabel med ett mellanslag och sändningen har en tydlig början och ett tydligt slut vilket underlättar vid inläsningen. Om ingen bluetooth finns så säger mobilen att ingen bluetooth finns. Accelerationsberäkningen är ett väldigt smalt säkerhetssystem via som om mobilen är i rörelse stoppar användaren från att använda systemet under färd.

Bluetoothmodulens konstruktion kommer beskrivas under kodbeskrivningen.

5.6 Alternativa koncept

Utefter metoden som beskrivs i sektion 3.5 Konceptgenerering och avgränsning av koncept. Så gjordes en brainstorming session för att generera alternativa koncept. Dessa konceptskisser skall sedan rangordnas i en Pughs matris för att hitta lämplig lösning/lösningar se i sektion 5.3. Därefter skall vinnande konceptskisser förfinas genom att ritas upp i CAD miljö, se sektion 5.4.

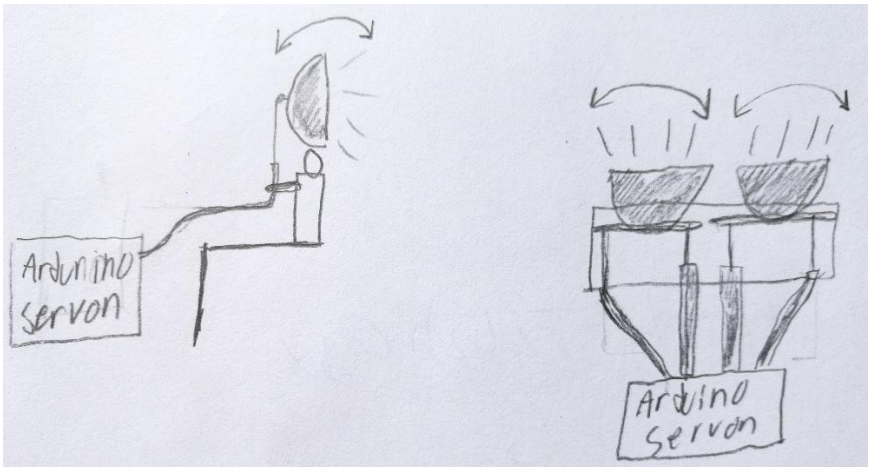
5.2.1 Koncept 1 – ”Kuben”



Figur 11. Koncept 1 - "Kuben"

På koncept 1 så sitter extraljusen monterat i ett robust ramverk. Lösningen möjliggör rotation och tiltning via vajrar som är styrda av servomotorer se figur 11. Utöver ramverket så finns en låda där mikrodatorn, servona, vajrarna och resterande komponenter monteras. Fördelen att ha en låda som är kopplad till ramverket är att lådans position är flexibel samt att lådan kan placeras där framkomsten av väta mm är minimal. Nackdelar är att plats för denna låda på fordon kan vara begränsat samt ramverket i sig anses vara en del som är något icke universell då den tar upp mycket plats på fordonets front.

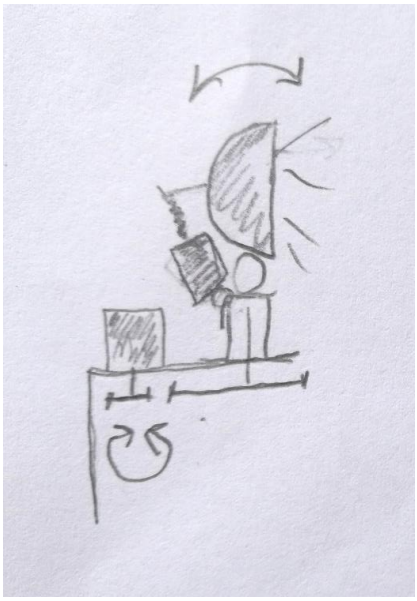
5.2.2 Koncept 2 – ”Vajrarna”



Figur 12. Koncept 2 - "Vajrarna"

Koncept 2 har idéer från koncept 1 där skillnaden är en ny idé kring infästningen till fordonet. Ramverket har bytts ut till en L-balk se figur 12 för att utöka montersmöjligheten på fordonet. Koncept 2 har samma låda med styrvajrar som koncept 1. Därmed så kvarstår fördelarna och nackdelarna hos lådan. Tiltningen görs i ”foten” av extraljuset vilket inte är optimalt då rotationscentrum är en bit ifrån tyngdpunkten och medför att starka styrmotorer behövs.

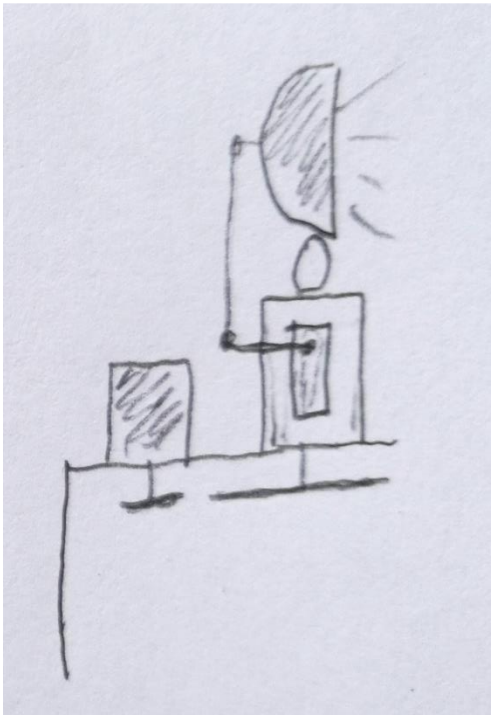
5.2.3 Koncept 3 – ”Ledskruven”



Figur 13. Koncept 3 - "Ledskruven"

Koncept 3 använder sig av samma L-balk för infästning på fordonet som koncept 2. Servona har bytts ut mot stegmotorer för rotation och tiltning. Rotation görs via en stegmotor som via en kuggväxel med nedväxling roterar konstruktionen. Tiltingen görs via en stegmotor med en ledskruv som är monterad på baksidan an extraljuset se figur 13. För att konstruktionen skall tåla väta, smuts mm så krävs inkapsling av dreven samt för ledskruven. Inkapslingen av ledskruven blir den mest problematiska då inkapslingen blir komplex och stor. Även momentcentrum för extraljusen hamnar i foten på extraljuset vilket gör att en kraftig stegmotor behövs för tiltingen.

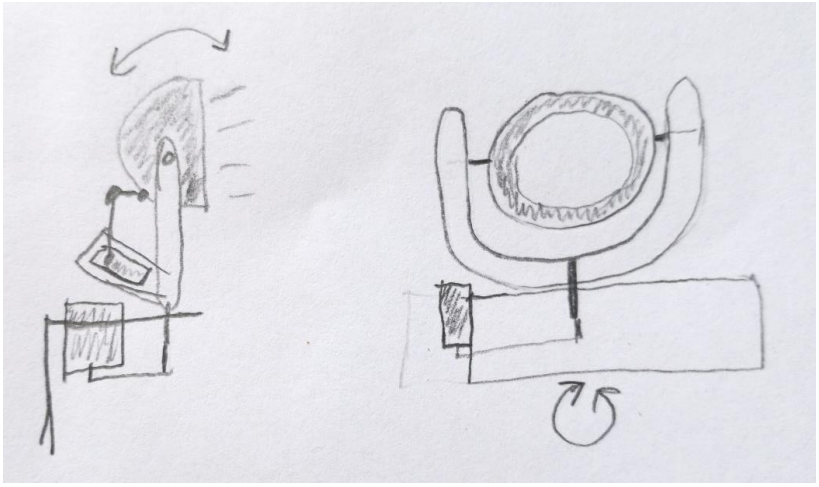
5.2.4 Koncept 4 – ”Blandningen”



Figur 14. Koncept 4 - "Blandningen"

Koncept 4 är mycket lik koncept 3, där skillnaden är att stegmotorn för tiltningen har bytts ut till ett servo se figur 14. Vilket gör att inkapsling inte behövs då en ledarm kan användas som tål väta, smuts mm. Koncept 4 möjliggör att en billigare styrmotor kan användas som levererar samma resulterande moment, då servon är billigare än stegmotorer. Nackdelen kvarstår att rotationscentrum för tiltningen ligger i foten.

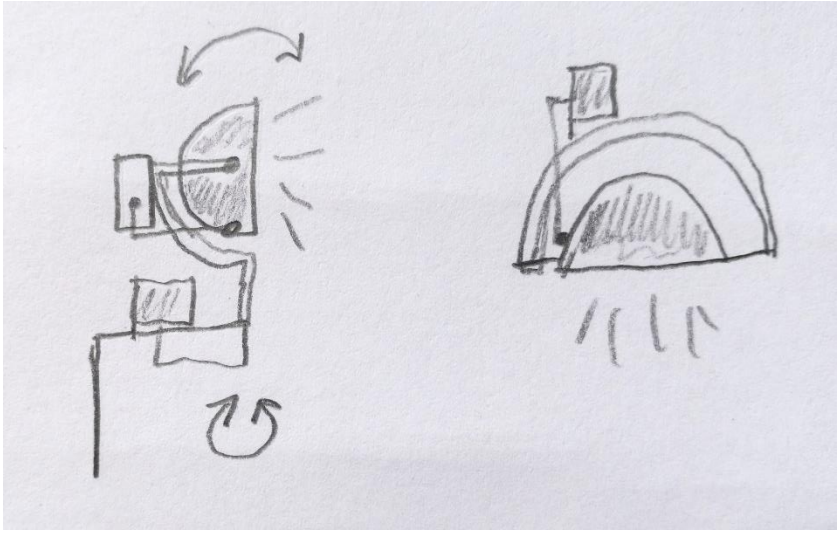
5.2.5 Koncept 5 – ”U-profilen”



Figur 15. Koncept 12 - "Glada munnen"

Koncept 5 använder sig av en U-formad del som huvuddel se figur 15. Detta för att placera rotationscentrum för både rotation och tiltningen så nära extraljusets tyngdpunkt som möjligt. U-delen är lagrad i mot L-balken för att medföra rotationen via servo. I de övre ändarna av den U-delen är även lagrad för att kunna tilta extraljuset kring dess tyngdpunkt se figur 15. Vilket medför att momentet som krävs för tiltningen minskar jämfört med koncept 2,3 och 4 då rotationscentrum befinner sig i linje med tyngdpunkten. Eftersom ledarmar används till båda axlarna så tål även konstruktionen väta, smuts mm.

5.2.6 Koncept 6 – ”Bordslampan”



Figur 16. Koncept 6 - "Bordslampan"

Koncept 6 är en vidareutveckling av koncept 5. Idé gick mot hur man skulle kunna göra för att förbättra konstruktionen med avseende på att ta mindre plats, minska luftmotståndet och vara mer anpassningsbar till andra extraljus. Därmed placerades mycket av konstruktionen på baksidan se figur 16.

5.7 Eliminering av koncept

Elimineringen görs via en Pughs matris som förklaras i sektion 3.5. För att kunna rangordna de olika koncepten så krävs olika kriterium. Dessa kriterier är tagna från målen som har nämnts i rapporten inledande del.

Kriterium 1: Klara av 30 graders vinkling i rotationsled och 10 grader tiltning.
Kriterium 2: Vara så noggrann i sina rörelser samt inte kräva för kraftiga reglermotorer.
Kriterium 3: Inte vara svårare att montera än ett traditionellt montage av extraljus.
Kriterium 4: Anpassningsbar på en mängd olika bilar.
Kriterium 5: Anpassningsbar till liknade extraljus.
Kriterium 6: Mekaniken skall tåla vatten, is, snö, smuts.
Kriterium 7: Miljöpåverkan för systemet.
Kriterium 8: Lågt luftmotstånd.

Referensen för denna viktade Pughs matris är koncept 2. Då koncept 2 ansågs vara de mellersta av alternativen.

	Viktning	Referens Koncept 2	Koncept 1	Koncept 3	Koncept 4	Koncept 5	Koncept 6
Kriterium 1	10	0	0	+	+	+	+
Kriterium 2	7	0	-	0	0	+	+
Kriterium 3	1	0	-	0	0	+	+
Kriterium 4	9	0	0	0	0	+	+
Kriterium 5	4	0	-	+	+	-	-
Kriterium 6	3	0	+	-	0	+	+
Kriterium 7	5	0	-	0	0	0	0
Kriterium 8	5	0	-	0	0	0	+
Summa +		0	3	14	14	30	35
Summa -		0	22	3	0	4	4
Summa		0	-19	11	14	26	31
Gå vidare		Nej	Nej	Nej	Nej	Ja	Ja

Figur 17. Pughs matris - eliminering av koncept.

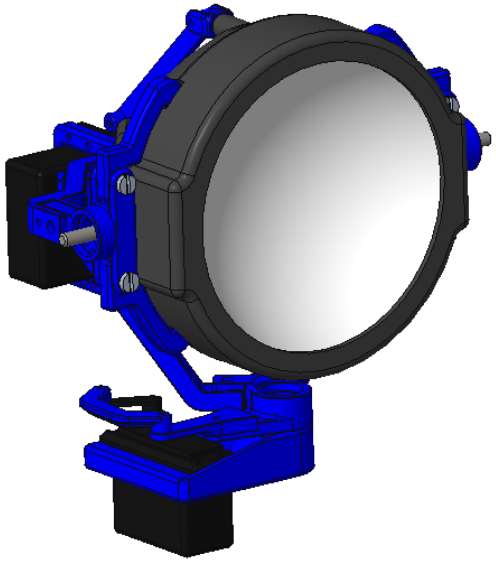
5.8 Vidareutveckling av koncept

Som kan ses i sektion 5.3 så presterade både koncept 5 och 6 bra i Pughs matrisen. Dom kräver inte så kraftiga reglermotorer samt använder samma typ reglermotorer. Därmed anses båda var lämpliga att gå vidare med. Koncepten ritades upp i CAD miljö med avseende på målen och fysiska mått på komponenterna. Dessa konstruktioner skall skrivas ut på en 3D-skrivare om färdiga komponenter inte går att köpa.

5.8.1 Koncept 5 CAD

Figur 18. Koncept 5 CAD

5.8.2 Koncept 6 CAD



Figur 19. Koncept 6 CAD

5.9 Beräkningsalgoritmer för ljuspunkt

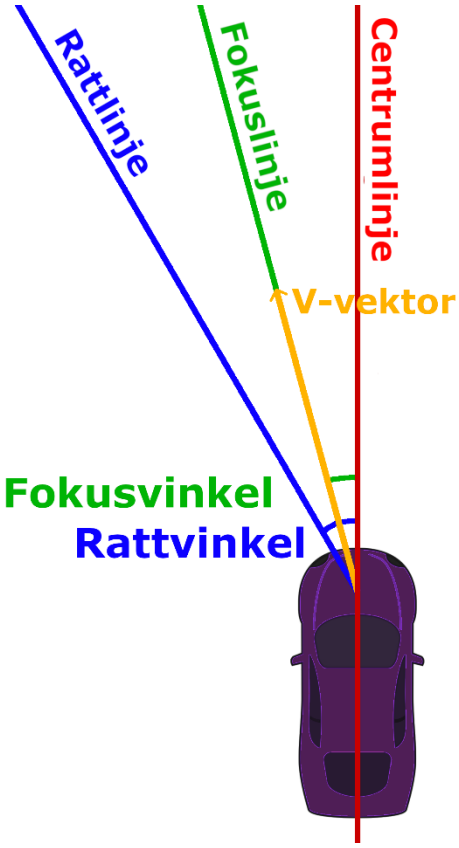
5.9.1 Fokuslinje

Vid färd har fordonet en rattlinje som kommer från rattvinkelns utslag, en centrumlinje som går genom bilens faktiska centrumlinje och en hastighetsvektor som ligger mellan centrumlinjen och rattlinjen. Det som vill åstadkommas med denna algoritmen är att bestämma vilken vinkel som fokuslinjen har, fokuslinjen kan ses som en approximation av vinkeln som hastighetsvektorn är i. Detta anses vara den punkt dit det är optimalt att rikta ljuskäglorna.

För att räkna ut och för att kunna justera och kalibrera funktionen så används variabler i funktionen.

Fokusvinkeln är en funktion som beror på primärt hjulvinkeln, som läses från CAN-bussen och är rattutslagsvinkeln, denna varierar någonstans mellan ±400°. Rattvinkeln multipliceras med variabel k_3 för att omvandla den till att motsvara fokusvinkeln när bilen är stillastående.

Därefter kommer den sekundära fordonshastigheten in. Dess funktion är att linjärt minska rattvinkelns påverkan på fokusvinkeln ju större hastighet fordonet har. $Vehicle_{speed}$ avläses också från CAN-bussen och är i enheten km/h. Hastigheten multipliceras med k_1 som är innan justering ställd så att vid hastigheten 120 km/h motverka och sätta fokusvinkeln till 0.



Figur 20. Beskrivning fokuslinje

Alltså arbetar funktionen med att sätta rattvinkeln till fokuslinjen men att minska rattens påverkan på fokusvinkeln ju högre hastighet bilen framförs med.

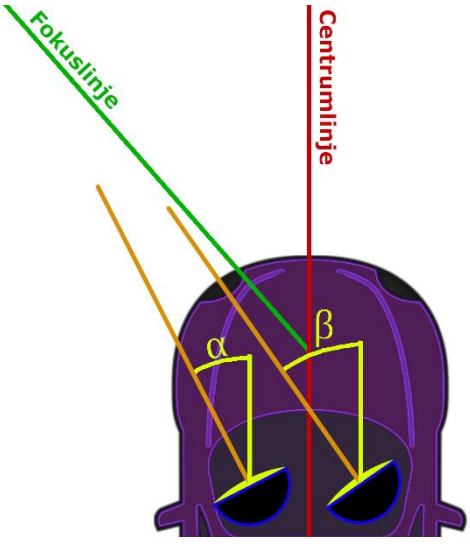
$$fok_{vinkel} = wheel_{angle} \times k_3 - vehicle_{speed} \times k_1$$

5.9.2 Vinkelsynkning

När fokusvinkeln är bestämd behöver de individuella strålkastarnas vinkel bestämmas. Detta för att kunna matcha höger och vänster ljuskägla mot fokuslinjen.

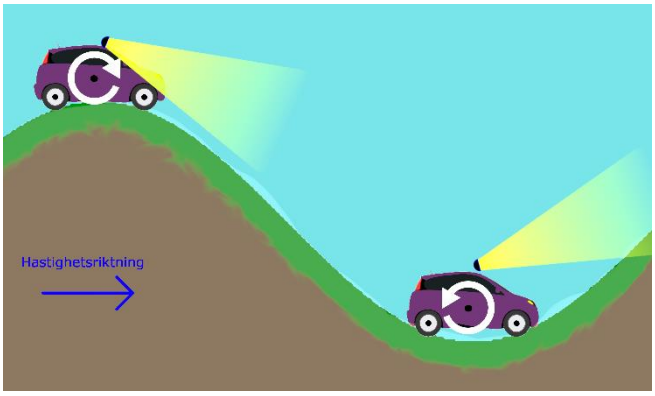
$$Vänstersväng: \alpha = fok_{vinkel}\,; \beta = fok_{vinkel} * k_2$$
$$Högersväng: \alpha = fok_{vinkel} * k_2\,; \beta = fok_{vinkel}$$

Vid vänstersväng kommer den alfavinkeln sättas till vinkeln för fokuslinjen och betavinkeln kommer sättas till fokuslinjens vinkel multiplicerat med en faktor för att vinkla motståendes sidans extraljus lite mer.



Figur 21. Vinkelstyrning

5.9.3 Höjdledsberäkning



Figur 22. Höjdledsjustering

Algoritmen för att beräkna vinklingen i höjdled av ljusen är baserat endast på gyroskopets indata. Detta är inte en optimal lösning och behöver för att fungera korrekt i alla användningsfall kompletteras med ytterligare givare/sensorer.

Gyroskopet begränsas i koden för att endast ge rotationsvinkeln runt en axel, detta kräver att dosan som gyroskopet är inbyggt i sitter rätt placerat så rätt axel avläsas. Vid fortsatt arbete skulle detta kunna kodas om så att man med hjälp av alla de tre axlarna kan beräkna bilens vinkling.

Teorin bakom vilken vinkel extraljusen ska ha är något mer komplicerad än tidigare algoritmer. Den vänstra bilen i figur 26 är påväg över kullen, detta ger att derivatan runt bilens centrumpunkt är negativ. Vid negativ derivata på bilens lutning så sänks ljusen för att kunna se vägen nedanför bilen.

För den högra bilen i figur 25 som är påväg upp i kullen så har den bilen en positiv derivata. Vid positiv derivata så lyfts ljusen något för att kunna se upp i backen. Figuren ovan är såklart överdramatiserad för att visa teorin.

Med denna algoritmen så kommer ljusen stabiliseras i neutralt läge när bilen kör i uppför eller nedförsbacke.

$$Light_{angle} = Gyro_{derivata} \times k_4$$

Där k_4 är en faktor för hur mycket derivatan ska påverka. $Light_{angle}$ läggs antingen till eller ifrån motorernas neutrala position beroende på om derivatan är positiv eller negativ.

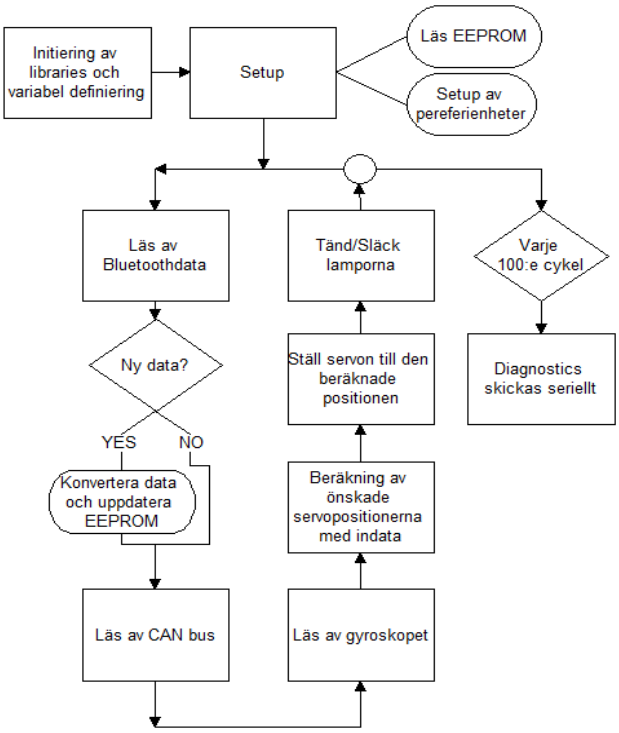
5.6 Systemets mjukvara

5.6.1 Programmeringsgenomgång

Kontrollsystemet programmeras för att fungera tillsammans på ett optimalt och effektivt vis. För att uppnå den önskade funktionen konstrueras koden efter flödesschemat i figur 20.

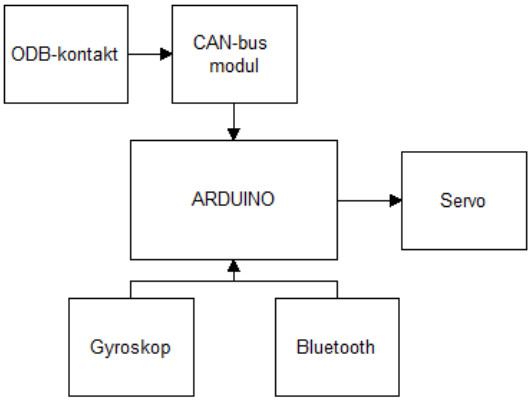
Flödesschemat initierar först de kodbibliotek som projektet använder sig av och definierar de variabler som kommer användas globalt i koden. Därefter går koden in i setup där olika periferienheter till mikrokontroller både på och utanför Arduinon konfigureras för önskad funktion. I setup läses också eeprom in som sparar användarens inställning från föregående session. Detta för att användaren inte ska behöva koppla upp sig med Bluetooth varje gång som systemet ska användas.

Därefter går programmet in i loopen som cyklar kontinuerligt under drift. Funktionerna i loopen läggs till fortlöpande under projektets gång samtidigt som fler komponenter tillkommer.



Figur 23. Programsekvens

Först kollar koden om Bluetoothmodulen väntar på att ta emot data, detta sker endast då användaren skickar data via mobilapplikationen till Bluetoothmodulen. Om så är fallet så läses den inkommande datan in och behandlas tills datan istället kan läggas in i variabler. Då detta nu är användarens senaste uppdatering så uppdateras också eepromen. Om det inte kommer nya data till Bluetooth modulen så uppdateras inga tillhörande variabler, inte heller uppdateras eeprom, detta för att minimera tidskrävande funktioner i loopen.



Figur 24 Systemförklaring Arduino

Sedan fortsätter programmet med att läsa av CAN-bussen, där Arduinon använder sig av CAN-buss modulen som används då Arduinon saknar nödvändiga periferikomponenter för att klara av att läsa av datan. CAN-bussen läses av med en funktion som filtrerar ut de sökta ID indexen, se sektion 2.4 för mer information om CAN-buss. Datan från CAN-bussen omvandlas sen från HEX format till decimalt och läggs också in i variabler.

Gyroskopet kommunicerar med Arduinon via kommunikations gränssnittet I^2C. Gyroskopet ställs i setup in för att endast visa ena axelns rotation. Detta för att minimera överförda data över I^2C samt att minimera faktisk kod och beräkningstid.

När nu alla indata till systemet bestämts så ska datan passera beräkningsalgoritmerna för att ta fram rätt utsignal till styrmotorerna samt lamporna. Detta finns i ”5.6 Beräkningsalgoritmer för ljuspunkt”.

Sedan ska styrmotorernas position uppdateras med de nyligen beräknade värdena. Strax efteråt ska lamporna uppdateras beroende på om helljuset är på eller inte.

Denna loopen körs kontinuerligt med en cykeltid på ungefär 3 ms, var 100:e cykel skrivs också diagnostiken ut, detta för att kunna kontrollera systemets funktion.

5.6.2 Detaljgenomgång av viktiga koddelar.

5.6.2.1 Loop

Loopen i koden ser ut som i figur 22.

```
void loop(){
   if(Serial.available() > 0){
      bluetooth_read = Serial.readString();
      bluetooth_read.toCharArray(raw_data, 15);
      raw_data_converter();
      diagnostics[17] = 1;
   }
   read_canbus();
   mpu6050.update();
   diagnostics[20]=1;

   move_servo();
   control_lights();

   if(cycle==100){
    update_diagnostics();
    view_diagnostics();
    cycle=0;
   }else{
    cycle++;
   }
}
```

Figur 25. Loopen

Den inleds med att kolla om det finns inkommande data från Bluetooth modulen. Om så är fallet läses datan in som datatypen ”string” för att sedan omvandlas till en array av datatypen ”char”. Därefter omvandlar funktionen ”raw_data_converter()” den råa indatan till individuella variabler för de olika indata parametrarna. Diagnostiken uppdateras för att berätta att systemet tagit emot data från Bluetooth modulen.

Därefter eller om det inte fanns data från Bluetooth modulen så läses CAN-bussen av vilket kommer förklaras i nästa delkapitel.

Gyroskopets värde läses av och variabeln för dess värde uppdateras.

Algoritmerna för servonas position utförs i funktionen ”move_servo()” där sedan servonas nya position skrivs till dom. Direkt efter det så uppdateras lampornas tillstånd av systemet.

Diagnostiken skrivs sedan ut var 100:e cykel för att minska på fördröjningen som den seriella utskriften orsakar.

5.6.2.2 CAN-buss inläsning

CAN-buss inläsningen är något lik den för Bluetooth. Om data från CAN-bussen är tillgänglig så läses datan in till en buffert. Därefter filtreras de önskade id ut för att endast visa den nyttiga datan för att minimera cykeltiden för funktionen.

```
if(CAN_MSGAVAIL == CAN.checkReceive()){
    CAN.readMsgBuf(&len, buf);
    unsigned long canId = CAN.getCanId();

    if(CAN.getCanId() == 0x496){
      for(int i=0;i<len;i++){
        can_2[i] = buf[i];
      }
      main_beam_indicator = (can_2[2]/16)/8;
    }
}
```

Figur 26. CAN-buss inläsning

I figur 23 visas endast ett exempel över hur helljusavläsningen görs. Helljuset har sedan tidigare lokaliserats till att ligga under id 0x496. Datan under denna id läggs in i en array för att sedan kunna behandlas. För att få ut rätt bit görs en liten algoritm. Vilken bit som motsvarar helljuset finns under bilaga 3.

5.6.3 Diagnostik

Diagnostiken ser ut som i figur 24. Det första som skrivs ut är inputen som antingen kommer från Bluetoothen eller från eepromen. Först siffran motsvarar vilket ”mode” bilen är i, andra siffran är vilken bilmodell som valts. Därefter kommer datan för om man manuellt vill styra ljusen i höjdled och sidled.

```
Diagnostics:
Input:
1 0 153 97 248
CAN-bus:
16 1 0 71 0 0 0 0
Servo:
5 7 2
Flags:
1 0 1 1 1 0 0 0 0 0
```

Figur 27. Diagnostik exempel

Sedan skrivs CAN-bussen ut, i ordningen rattvinkel, höger/vänster, helljus och bilens hastighet. Efterföljande nollor är till för att i framtiden kunna lägga till fler CAN-buss avläsningar.

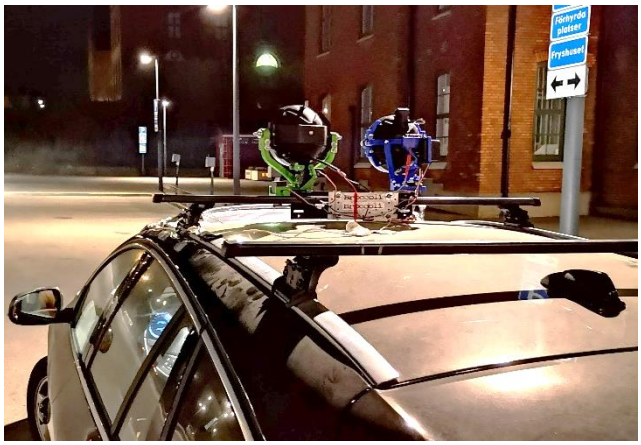
Servonas position skrivs sedan ut i vinkel i ordningen: vänster sidled, höger sidled och bådas vinkling i höjdled.

Flaggorna skrivs sist ut i ordningen: setup färdig, Bluetooth har fått data, korrekt data från eeprom/bluetooth, CAN-buss initierad, gyroskop kalibrerat och sist om helljuset är på eller av.

Detta skrivs som tidigare nämnt ut var 100:e cykel vilket motsvarar ungefär var 300 ms

5.7 Testning på bil

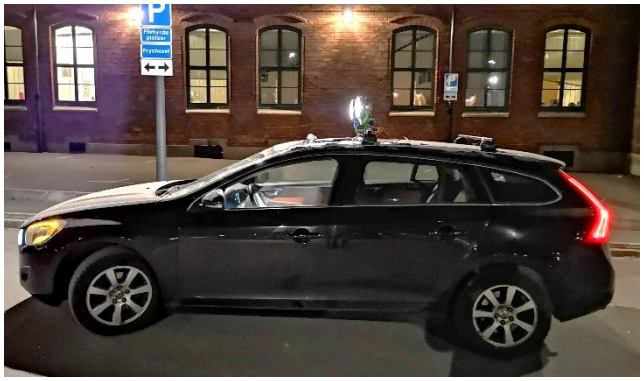
När prototypen ansågs redo för testning togs en testplan fram över hur man bäst kan få svar på huruvida teorin fungerar i praktiken eller inte. Prototyperna blev monterade på ett standardfäste för att sätta extraljus vid registreringsplåten i fronten. Det var dock inte ett alternativ att bulta fast prototypriggen i fronten, dels för att det krävde modifiering av bilens front som inte kan återställas efter testning samt att det kräver en del arbete för att montera upp eller plocka ner systemet mellan testen.



Figur 28. Testning på bil.

Därför monterades systemet på det främre takräcket, detta gör det enkelt att ta ner hela takräcket utan att behöva montera av det utvecklade systemet. Dock krävs justeringar för ljusens neutralläge pga. monteringshöjden jämfört med om de vore placerade i fronten.

Elsystemet kopplades ihop med bilens batteri för att ge matning till båda extraljusen och styrsystemet. Extraljusens tändning och släckning sker egentligen automatiskt av systemet men förbikopplades vid testningen. Passageraren sitter istället med kontakter och tänder och släcker dessa manuellt. Detta för att inte riskera felfall där lamporna lyser konstant, samt för att kunna koppla bort lamporna när inte testen är aktiva.



Figur 29. Testning på bil.

Testet genomfördes på en stor parkeringsplats för att kunna köra runt och testa systemets funktion till det yttersta. Inga hastigheter över 30 km/h förekom under testet.

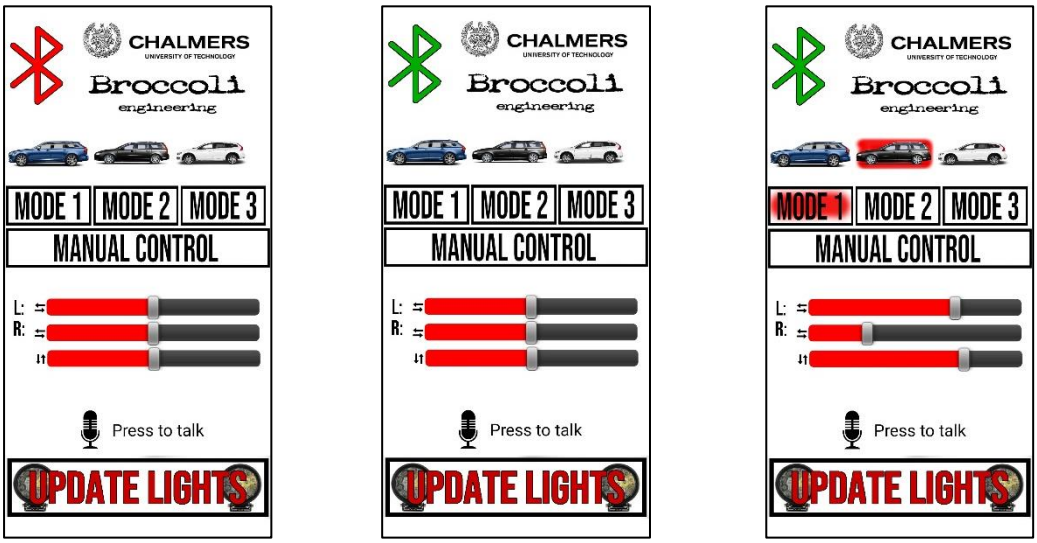
6 Resultat

6.5 Användarpanel/App

Tidigare under 5.1 förklarades kodningen och utformningen av applikationen grundligt.

Det slutliga resultatet är en app som ger användaren möjlighet att koppla upp sig till systemets bluetooth. Användaren har möjlighet att göra ett enkelt och tydligt val för vilken bilmodell som används.

Det går också att välja vilket ”mode” man vill köra med, vilket t.ex. kommer ändra vilken aggressivitet ljusen har vid sväng. Vid manuell kontroll av vinkeln har användaren möjlighet att justera detta med de tre dragaxlarna. Detta ger användaren en möjlighet att ha statiska vi klar på ljusen, t.ex. om man i parkerat tillstånd vill lysa upp något speciellt på vägen eller om man vill använda systemet som ”vanliga” extraljus. Användaren kan också välja att köra appen helt röststyrd.



Figur 30. Utseende på färdig applikation

Målet inledningsvis var att ha en användarpanel där användaren kan finjustera och ändra systemets funktioner på ett enkelt och smidigt sätt, vilket applikationen uppfyller.

6.6 Programmering

Den slutgiltiga koden utför de uppgifter som önskas.

Vid uppstart avläses mikrokontrollerns eeprom för att använda systemets tidigare sessions förinställningar. Detta medför att användaren inte behöver koppla upp sig mot systemet vid varje driftsättning.

Periferienheten för Bluetooth läses av och möjliggör användaren att justera funktioner i systemet aktivt när systemet är i drift. Användaren har också tillgång till en fungerande mobilapplikation för att på ett enkelt och säkert sätt justera rätt parametrar i systemet.

CAN-bussen fungerar på ett önskvärt sätt. Projektet har analyserat fram vart i CAN-bussen som nyttig information finns. Därefter kompletterades koden i Arduinon med denna informationen för att kontinuerligt kunna läsa av den önskade datan.

Gyroskopet läser in rotationsvinkeln runt ena axeln och skickar datan till Arduinon över I2C interfacet. Datan kompletterar med data som inte hittades på CAN-bussen. Den kompletterade datan gör det möjligt att vinkla lamporna i höjdled på ett önskat sätt.

De framtagna algoritmerna beräknas i programmet med hjälp av den tillgängliga indata, algoritmerna är vid projektets slutfas någorlunda kalibrerade men kräver ytterligare testning och optimering för att fungera på bästa sätt.

Motorstyrningens styrsignal beräknas varje cykel och skrivs till servomotorerna. Styrsignalen hade vid fortsatt arbete behövt någon form av filtrering för att försöka få bort de små vibrationerna som gyroskopet läser av. Utöver det sker motorstyrningen på det önskade sättet.

Tändning/släckning av extraljusen görs genom en mosfet som i sin tur styrs av Arduinon. Datan om huruvida lamporna ska vara på eller av läses från CAN-bussen, förarens helljusreglage är direkt kopplat till lampornas ljustillstånd.

Diagnostiken ger var 100:e cykel en seriell utskrift med nyttig information för att ge programmeraren den information som behövs för att felsöka systemet.

Den slutgiltiga cykeltiden för systemet ligger på ungefär 3 ms. Vilket innebär att vi har en uppdateringsfrekvens på lamporna på runt 300 ggr/sekund.

Den färdiga kommenterade koden ligger i bilaga 3 om ytterligare genomgång önskas, under 5.5 förklaras de viktigaste delarna i koden.

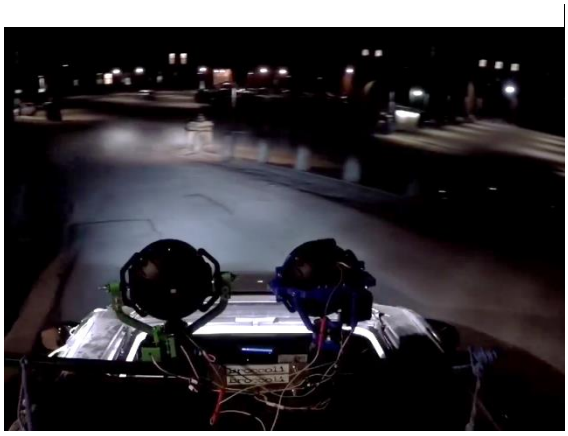
6.7 Testning på bil

Testkörningen i låg hastighet fungerade smärtfritt och visade på båda svagheter i systemet och konstruktionen. Koncept 6, vilket är det högra extraljuset i figurerna nedan vibrerar något mer än den andra under färd. Detta är både ett konstruktions och materialproblem.

Samtidigt påvisades att systemet fungerar och gör vad som önskas. Som man ser i figur 31 när bilen är i neutral rattposition och stillastående så är ljusen centrerade ett par meter framför bilen. Man kan också se att ljusen inte är helt synkade i höjdled. Detta beror dels på små mekaniska skiljaktigheter mellan de två prototyperna.

Figur 31. Ljusbild rakt framåt.

I figur 32 är bilen på väg ner i en nedförsbacke samtidigt som vägen svänger lätt åt vänster. Som bilden visar så följer ljusen vägen på ett bra sätt. Ljusen svänger efter vägens kurva och sänks ner för att kompensera för vägens lutning. Det som skulle behöva förbättras är ljusens position relativt varandra, på bilden syns det att de är något mer särade på jämfört med neutralläget som visades i föregående bild. Ljusen kan komma lite mer åt vänster för att ligga optimalt, vinklingen i höjdled ligger på ett bra ställe och ljusens synkning behöver justeras i parametrarna.



Figur 32. Ljusbild nerförsbacke och vänstersväng.

I figur 33 är bilen på väg uppför en backe samtidigt som vägen svänger lätt åt vänster. Bilden visar hur ljusen inte riktigt följer vägens kurva. Detta visar en av de större begränsningarna i systemet, att endast bilens nuvarande data används i algoritmerna som räknar ut lampornas position. Detta gör att när bilen är påväg uppför en backe innan föraren fysiskt börjat svänga så vet inte bilen att det är en backe eller sväng påväg. Systemet behöver få en mer sofistikerade beräkning för att prognostisera vägens utseende framför bilen. Däremot fungerar återigen vinklingen i höjdled på ett fungerande vis.

Figur 33. Ljusbild uppförsbacke och högersväng

Vid testkörningen så noterades även energiförbrukningen för hela systemet. Då konstaterades att hela systemet drar totalt 8–10 amper. Där 1–3 amper är för regler motorerna samt styrelektroniken och resterande ström utgörs av de två extraljusen på 3.5 amper styck.

7 Slutsatser/Analyser och Diskussion

När projektet är avslutat är det dags att gå igenom målen för att se och resonera kring hur dessa uppnåtts.

Huvudmålet var att ta fram en fungerande prototyp för ett smart extraljussystem, vilket är till stor del uppnått. Projektet resulterade i en prototyp som uppför sig till stor del som önskat. Där finns en del återstående arbete för att göra upplevelsen för föraren optimal och helt smärtfri. Projektet uteslöt den noggranna korrigeringen och optimeringen av parametrarna för att få ljuskäglorna att träffa exakt där det önskas. Tidsramen för projektet tog slut så arbetet med detta avslutades innan arbetet kunde utföras, då det inte påverkade det objektiva resultatet lades istället tiden på att dokumentera systemet och förbättra den slutgiltiga rapporten.

För att gå in på detaljmålen så vill projektet effektivisera elförbrukningen jämfört med traditionella extraljussystem. Detta kan uppnås men är något svårare att konkret säga ja eller nej på. Systemet i sig drar en del ström som ungefär motsvarar en till extraljus. Däremot så bör funktionen av systemet göra det möjligt för bilägaren att välja 2 extraljus istället för 3 om dom är utrustade med detta systemet. För att ytterligare bekräfta att detaljmålet uppnåtts behöver tid läggas för att minimera elförbrukningen på systemet, vilket känns helt möjligt då systemet i dagsläget utvecklats för att testa teorin och inte för att vara så energisnålt som möjligt.

Systemet skulle justera ljuskäglorna efter data från CAN-bussen. Arbetet med att analysera CAN-bussen gick smidigt och den mest kritiska informationen som hastighet, rattvinkel samt helljusets tillstånd hittades. Däremot hittades inte data angående bilens vinkling för att för att se om bilen är i upp eller nerförsbacke, samt datan från ljussensorn bakom backspegeln saknades. Detta resulterade i att ett gyroskop fick läggas till för att alla kritiska in datan kunde läsas in samt att optionen för automatisk avbländning valdes bort.

Med denna inkommande datan var det inledande målet att kunna omvandla den för att skicka till styrdonen på ljusen. Detta lyckades till viss del då det är denna del som kräver viss optimering för att fullständigt anses fungerande. Parametrarna i algoritmerna behöver alltså förbättras.

Användarpanelen är konstruerad och implementerad. Mobiltelefonsapplikationen är fungerande och skickar ut den data som önskas. Samtidigt läser Arduinon in datan som kommer över Bluetooth rätt och lägger den i rätt variabler. Applikationen ger användaren möjlighet att justera alla parametrar som nämns i detaljmålen. Däremot används inte denna datan till så stor utsträckning i programmet. Projektet utfördes endast på en bil vilket innebär att inställningen i appen för vilken bil man har är onödig för tillfället. De olika ”mode” lägena är också inte implementerade, detta därför att projektet än så länge inte kan styra ljuskäglorna exakt dit som önskas, därför lades inga resurser på att försöka göra detta snabbare eller på annorlunda sätt. Dessa funktioner har därför ingen betydelse för systemet och därför lades tiden på att optimera koden för att testa teorin istället.

För att blicka tillbaka på syftet med projektet, att öka säkerheten i trafiken och analysera samt utreda huruvida teorin bakom systemet fungerar för att öka säkerheten. Det konkreta svaret på det är ja, teorin har en praktisk funktion i bilindustrin för att öka säkerheten. När systemet är korrekt kalibrerat kommer önskade områden på vägen belysas och öka förarens sikt på vägbanans riskområden. Detta samtidigt som bländningen av människor och/eller medtrafikanter minimeras.

Projektgruppen är nöjd med arbetets resultat och anser att genomförandet utförts på ett effektivt och målinriktat vis. Resultatet ger en bild av hur en praktisk applikation hade funkat och ger samtidigt en ide över vilka ytterligare funktioner som kan implementeras.

8 Förslag till fortsatt arbete

Att fortsatt arbete behövs är tydligt. Systemet är fungerande i dagsläget men det finns delar som behöver förbättras för att ge användaren en mer tillfredsställande användning. Nedan kommer projektgruppens idéer över vad ett fortsatt arbete skulle fokusera på.

Optimera parametrar i algoritmer:

Parametrarna i de framtagna algoritmerna är det som kommer ge mest resultat för minst arbete. Algoritmerna kräver justering för att bilens rörelse på vägen ska motsvara lyktornas rörelse. Mer tid i bil med systemet uppkopplat behövs för att kunna testa och finjustera dessa parametrar.

Ny tillverkningsmetod:

I nuvarande prototyp är fästena 3D-printade, vilket är en jättebra lösning vid prototyptillverkning men det leder också till lite ”flex” i prototyperna. Materialet är inte optimalt för applikationen. Vid fortsatt arbete borde mekaniken modifieras för att kunna tillverkas ur aluminium eller liknande, för att få en robustare prototyp.

Använda bilen som gyroskop:

Som tidigare nämnt så används ett externt gyroskop för att beräkna bilens vinkel. Detta för att datan för just detta inte hittades i CAN-bussen. Därför borde fortsatt arbete gå ut på att antingen leta reda på om den datan finns gömd någonstans eller alternativ utveckla en beräkning av bilens vinkel i de andra axlarna, som hittats i CAN-bussen och använda dessa för att räkna ut bilens vinkel.

Filtrering av styrsignal:

Styrsignalerna som skickas till servona är väldigt noggranna, vilket inte måste vara positivt. Prototypernas material i kombination med servonas små snabba korrigeringar ger en onödig vibration i ljusen. Detta kommer förbättras om prototyperna tillverkas i aluminium. Men samtidigt borde styrsignalerna filtreras för att ta bort de små justeringarna som endast orsakar vibrationer.

Styrdon i extraljusen:

En stor faktor till varför systemet inte är så energieffektiv som initialt önskats är på grund av de stora servona som krävs. För att minska de krävda styrdonen så finns en ide att istället för att vinkla hela extraljuset bara justera den inbyggda lampan och dess reflektor. Om detta kan åstadkommas tas luftmotståndets påverkan tas bort och det krävs endast styrdon för att rotera/tilta ljusen inuti de nuvarande höljena.

Ljusramp:

Som en variant av att vinkla extraljusen finns också ett alternativ där man istället använder en ljusramp, möjligen med en radie där man istället tänder och släcker individuella leds. Då krävs det inga styrdon utan endast elektrisk styrning för alla lamporna. Dock försvinner då möjligheten att vinkla i höjdled.

Dimma ljuset vid möte:

När CAN-bussen analyserades hittades ingen data för värdet på ljussensorn som sitter bakom backspegeln. Med den datan skulle man kunna implementera automatisk avbländning om man får möte på vägen. Samt dimma ljusen beroende på den omgivnings ljusstyrkan vid det specifika tillfället, detta minskar energiförbrukningen.

Videokamera:

För att kraftigt göra systemet mer sofistikerat kan en kamera implementeras. Med detta kan videon analyseras för att bekräfta vägens karaktär för att optimera och bekräfta algoritmernas beräkningar. Videon kan visa om bilen får möte så ljusen ska stängas av. Samt se om bilen möter vägskyltar, då kan ljusen riktas bort något för att inte blända föraren.

Flera CAN-buss:

För tillfället läser systemet endast av CAN-bussen i den bil systemet är utvecklad runt, Volvo V60. Detta är långt ifrån optimalt om systemet ska produceras till en färdig produkt. Därför behöver någon form av rutin tas fram för att kunna hitta rätt index på CAN-bussen för olika biltyper.

9 Referenser

[1] ”Wikipedia NFC,” [Online]. Available: https://en.wikipedia.org/wiki/Near-field_communication. [Använd 19 Maj 2019].

[2] Arduino , ”PWM,” [Online]. Available: https://www.arduino.cc/en/Tutorial/PWM. [Använd 19 Maj 2019].

[3] ”Wikipedia Wi-Fi,” [Online]. Available: https://en.wikipedia.org/wiki/Wi-Fi. [Använd 19 Maj 2019].

[4] ”Wikipedia Bluetooth,” [Online]. Available: https://en.wikipedia.org/wiki/Bluetooth. [Använd 19 Maj 2019].

[5] J. Tyson, ”How Rom works,” 29 Augusti 2000. [Online]. Available: https://computer.howstuffworks.com/rom5.htm. [Använd 14 Maj 2019].

[6] M. Grusin, ”Sparkfun Serial Peripheral Interface (SPI),” [Online]. Available: https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all. [Använd 14 Maj 2019].

[7] Circuit Basics, ”Basics of the I2C cummunication protocol,” [Online]. Available: http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/. [Använd 14 Maj 2019].

[8] O. Liang, ”How to use mosfet,” [Online]. Available: https://oscarliang.com/how-to-use-mosfet-beginner-tutorial/. [Använd 14 Maj 2019].

[9] Biltema, ”Extraljus 42-330,” 22 April 2014. [Online]. Available: http://productimages.biltema.com/v1/image/imagebyfilename/42-330_xxl_1.jpg. [Använd 20 Maj 2019].

[10] Wikipedia, ”Wikipedia,” [Online]. Available: https://sv.wikipedia.org/wiki/Servo. [Använd 15 Maj 2019].

[11] servo database, ”servo database,” [Online]. Available: https://servodatabase.com/. [Använd 15 Maj 2019].

[12] Jameco, ”How servo motors work,” [Online]. Available: https://www.jameco.com/jameco/workshop/howitworks/how-servo-motors-work.html. [Använd 15 Maj 2019].

[13] Arduino , ”Products Compare,” [Online]. Available: https://www.arduino.cc/en/. [Använd 15 Maj 2019].

[14] Arduino, ”Arduino Uno R3,” [Online]. Available: https://store.arduino.cc/arduino-uno-rev3. [Använd 15 Maj 2019].

[15] Arduino, ”Arduino Mega 2560 R3,” [Online]. Available: https://store.arduino.cc/mega-2560-r3. [Använd 15 Maj 2019].

[16] Arduino, ”Guide introduction,” [Online]. Available: https://www.arduino.cc/en/. [Använd 15 Maj 2019].

[17] KTH, ”Can bus,” [Online]. Available: https://www.kth.se/social/upload/526eab8ef2765479ddbd9131/CAN. [Använd 20 Maj 2019].

[18] B. Bergholm, Interviewee, *CAN-buss.* [Intervju]. 20 Februari 2019.

[19] ”CAN bus,” Wikipedia, 6 Maj 2019. [Online]. Available: https://en.wikipedia.org/wiki/CAN_bus. [Använd 20 Maj 2019].

[20] Erniotti, ”CAN-Bus-frame in base format without stuffbits,” 12 Mars 2014. [Online]. Available: https://commons.wikimedia.org/wiki/File:CAN-Bus-frame_in_base_format_without_stuffbits.png. [Använd 20 Maj 2019].

[21] Wikipedia, ”OBD-II PIDs,” [Online]. Available: https://en.wikipedia.org/wiki/OBD-II_PIDs. [Använd 15 Maj 2019].

[22] Transportstyrelsen, ”Transportstyrelsen,” 2013. [Online]. Available: https://www.transportstyrelsen.se/TSFS/TSFS%202013_63k.pdf.

[23] United Nations, ”Addendum 47: Regulation No.48,” 16 Oktober 2014. [Online]. Available: https://www.unece.org/fileadmin/DAM/trans/main/wp29/wp29regs/2015/R048r12e.pdf. [Använd 14 Maj 2019].

[24] Regeringskansliets, ”Infrastrukturdepartementet RST TM,” 23 Augusti 2001. [Online]. Available: http://rkrattsbaser.gov.se/sfst?bet=2001:650. [Använd 14 Maj 2019].

[25] Trafikverket, ”Information om belagda vägar, verktyget PMSV3,” 19 Oktober 2018. [Online]. Available: https://www.trafikverket.se/resa-och-trafik/vag/Sveriges-vagnat/Information-om-belagda-vagar-verktyget-PMSv3/. [Använd 20 Maj 2019].

[26] Trafikverkaet, ”PMSV3 - Analysera sträcka - Backighet,” 3 Augusti 2018. [Online]. Available: https://pmsv3.trafikverket.se/Pages/StrackaAnalys/chartview.ashx?start=0&len=10775&width=778&height=250&id=c893f8c1-e6ad-46e0-a70c-a74b13c4fd55&ymin=&ymax=&type=1&variabelid=050518a9-8c15-4f09-b0dd-43cec83aad0f&lankod=14&vagnr=523.00&riktning=1&korfalt=10. [Använd 20 Maj 2019].

[27] Trafikverket, ”PMSV3 - Analysera sträcka -Kurvatur,” 3 Augusti 2018. [Online]. Available: https://pmsv3.trafikverket.se/Pages/StrackaAnalys/chartview.ashx?start=0&len=10775&width=778&height=250&id=5e78342b-c228-43a8-b8bd-c25c39ff52f1&ymin=&ymax=&type=1&variabelid=55d54a75-37eb-4396-9e39-0569577c3019&lankod=14&vagnr=523.00&riktning=1&korfalt=10. [Använd 20 Maj 2019].

[28] S. P. D. N. S. David, ”Technique 40. Pugh Matrix,” 2017. [Online]. Available: https://app.knovel.com/hotlink/pdf/rcid:kpITTPSO01/id:kt011C2824/innovators-toolkit-50/technique-40-pugh-matrix?kpromoter=federation. [Använd 20 Maj 2019].

[29] Sparkfun, ”CAN-BUS Shield,” [Online]. Available: https://www.sparkfun.com/products/13262. [Använd 20 Maj 2019].

[30] Hobbyking, ”Turnigy S8166M,” [Online]. Available: https://hobbyking.com/en_us/turnigytm-s8166m-high-torque-servo-33kg-0-21sec-154g.html?___store=en_us. [Använd 20 Maj 2019].

[31] J. R. Y. Cengel, Fundamentals of Thermal-Fluid Sciences, 5 red., Penn Plaza, New York: Mc Graw Hill, 2017.

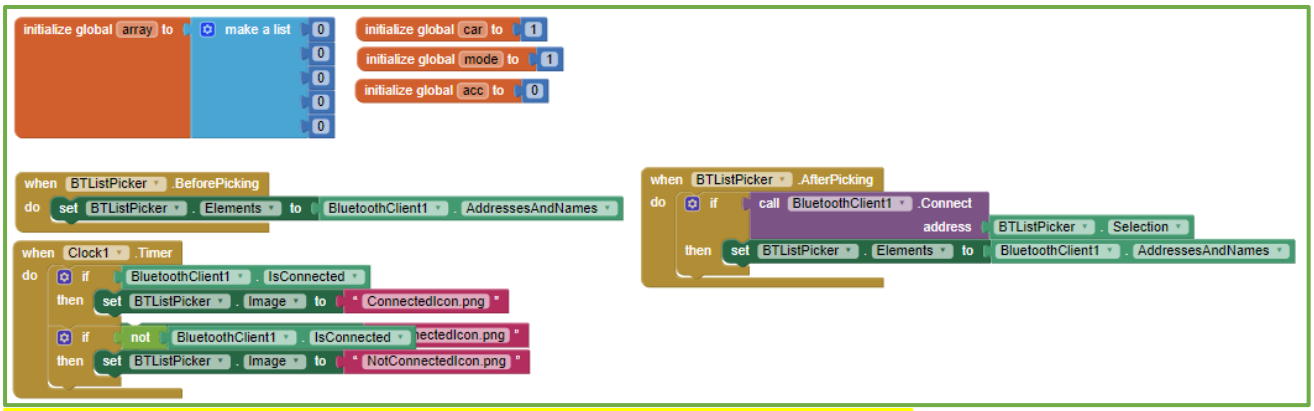
[32] oomlout, ”Internal mechanism of a continuous rotation servo,” 22 April 2009. [Online]. Available: https://en.wikipedia.org/wiki/Servo_(radio_control)#/media/File:Exploded_Servo.jpg. [Använd 15 Maj 2019].

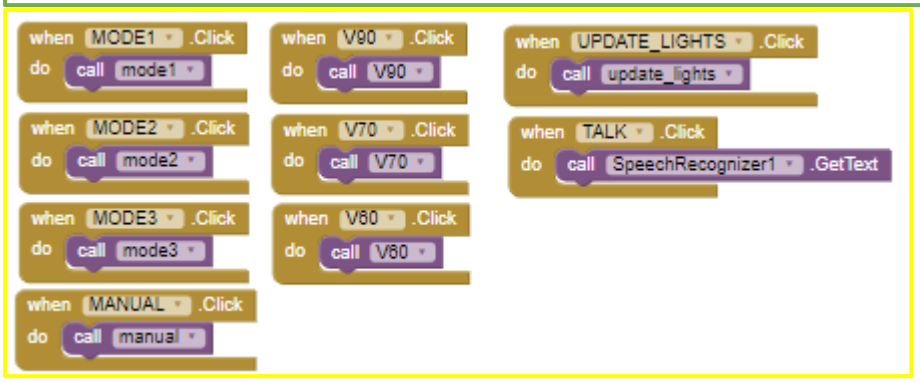
[33] JotaCartas, ”A Arduino Uno board,” 18 April 2011. [Online]. Available: https://commons.wikimedia.org/wiki/File:Arduino-uno-perspective-transparent.png. [Använd 15 Maj 2019].

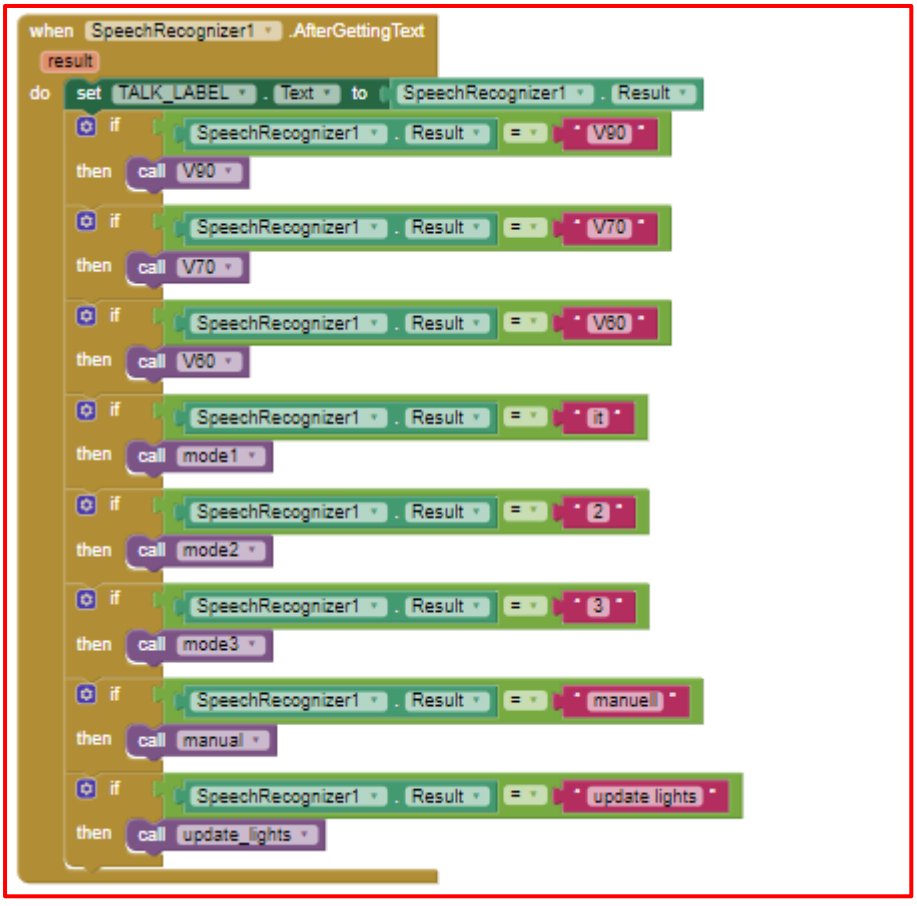
[34] Stefan-Xp, ”CAN-Bus Topology (Electronic Two Wire Connection),” 24 Februari 2008. [Online]. Available: https://sv.wikipedia.org/wiki/Controller_Area_Network#/media/File:CAN-Bus_Elektrische_Zweidrahtleitung.svg. [Använd 20 Maj 2019].

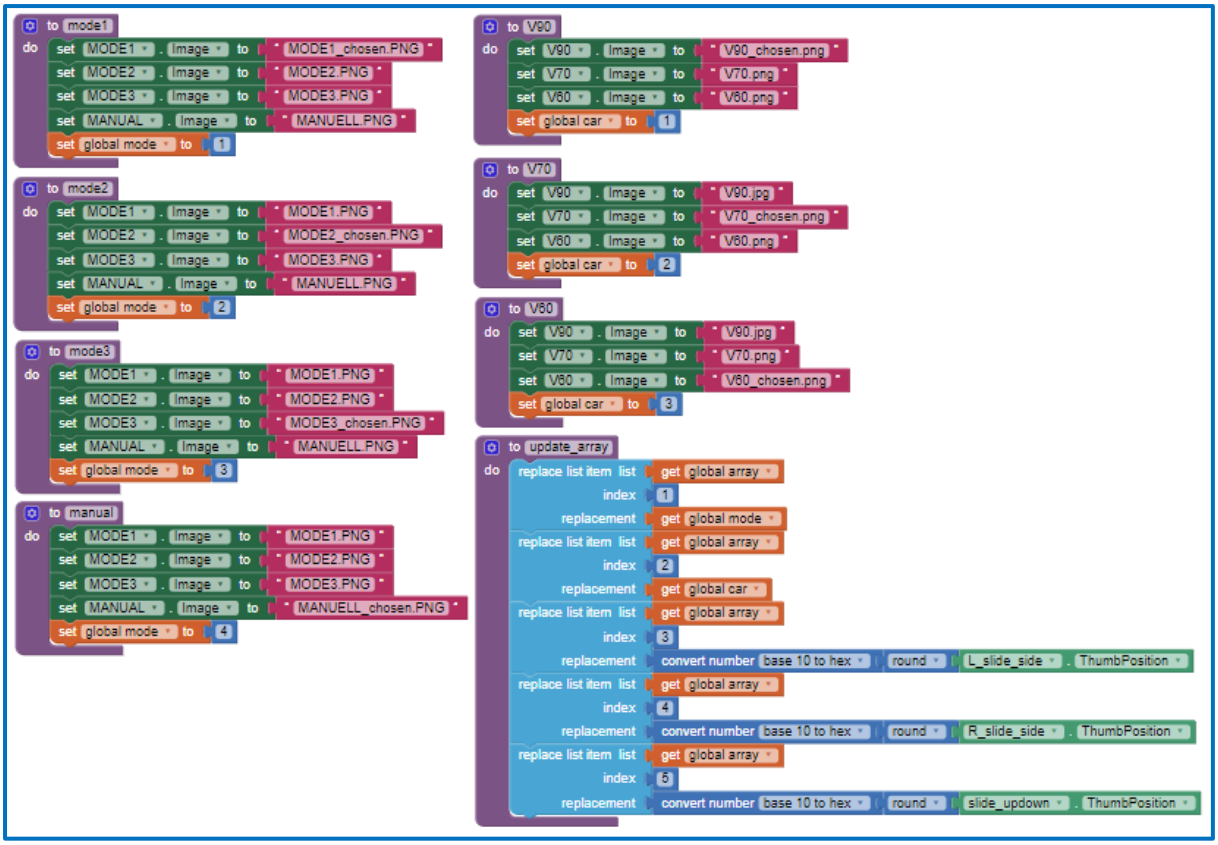
Bilagor:

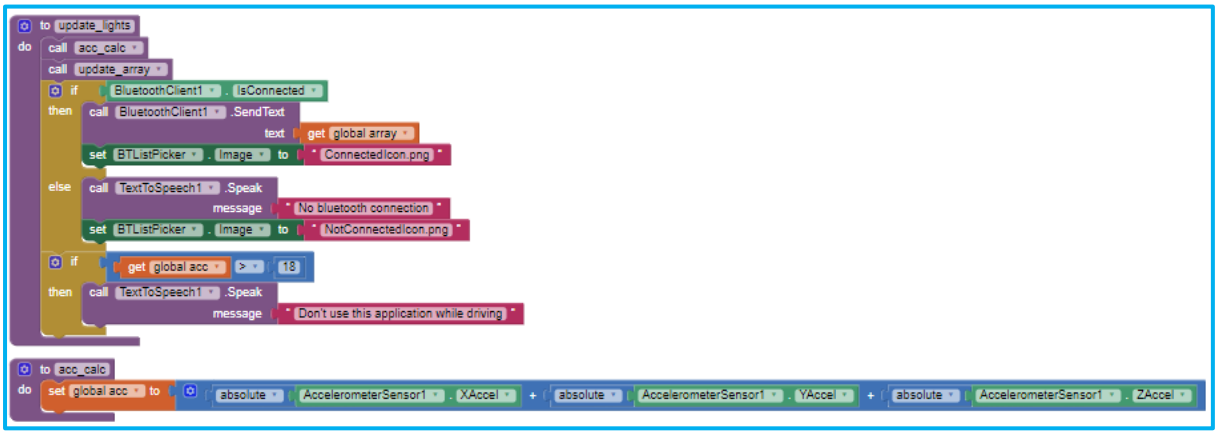
Bilaga 1: Kod till mobilapplikation








to mode1
do set MODE1 . Image to "MODE1_chosen.PNG"
set MODE2 . Image to "MODE2.PNG"
set MODE3 . Image to "MODE3.PNG"
set MANUAL . Image to "MANUELL.PNG"
set global mode to 1
to mode2
do set MODE1 . Image to "MODE1.PNG"
set MODE2 . Image to "MODE2_chosen.PNG"
set MODE3 . Image to "MODE3.PNG"
set MANUAL . Image to "MANUELL.PNG"
set global mode to 2
to mode3
do set MODE1 . Image to "MODE1.PNG"
set MODE2 . Image to "MODE2.PNG"
set MODE3 . Image to "MODE3_chosen.PNG"
set MANUAL . Image to "MANUELL.PNG"
set global mode to 3
to manual
do set MODE1 . Image to "MODE1.PNG"
set MODE2 . Image to "MODE2.PNG"
set MODE3 . Image to "MODE3.PNG"
set MANUAL . Image to "MANUELL_chosen.PNG"
set global mode to 4
to V90
do set V90 . Image to "V90_chosen.png"
set V70 . Image to "V70.png"
set V60 . Image to "V60.png"
set global car to 1
to V70
do set V90 . Image to "V90.jpg"
set V70 . Image to "V70_chosen.png"
set V60 . Image to "V60.png"
set global car to 2
to V60
do set V90 . Image to "V90.jpg"
set V70 . Image to "V70.png"
set V60 . Image to "V60_chosen.png"
set global car to 3
to update_array
do replace list item list get global array
index 1
replacement get global mode
replace list item list get global array
index 2
replacement get global car
replace list item list get global array
index 3
replacement convert number base 10 to hex round L_slide_side . ThumbPosition
replace list item list get global array
index 4
replacement convert number base 10 to hex round R_slide_side . ThumbPosition
replace list item list get global array
index 5
replacement convert number base 10 to hex round slide_updown . ThumbPosition


to update_lights
do call acc_calc
call update_array
if BluetoothClient1 . IsConnected
then call BluetoothClient1 .SendText
text get global array
set BTListPicker . Image to "ConnectedIcon.png"
else call TextToSpeech1 .Speak
message "No bluetooth connection"
set BTListPicker . Image to "NotConnectedIcon.png"
if get global acc > 18
then call TextToSpeech1 .Speak
message "Don't use this application while driving"
to acc_calc
do set global acc to absolute AccelerometerSensor1 . XAccel + absolute AccelerometerSensor1 . YAccel + absolute AccelerometerSensor1 . ZAccel

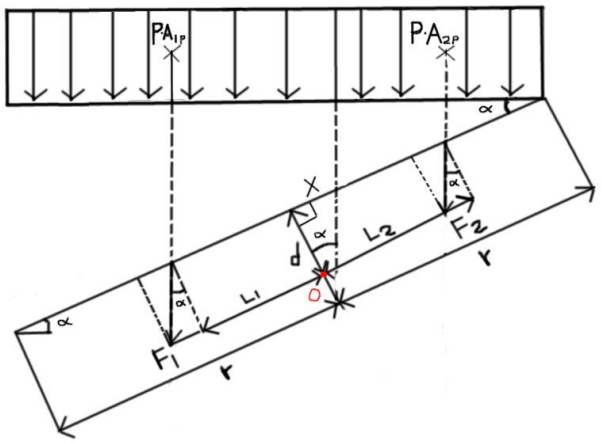
Bilaga 2: Momentberäkningar rotationscentrum på extraljus samt servomoment

$$Formel\ luftmotståndskraft\text{:}\ \ F = \frac{1}{2} C_d \rho_{luft} A v^2 \ \Rightarrow\ \mathrm{P} = \frac{F}{A} = \frac{1}{2} C_d \rho_{luft} v^2$$

Densiteten på luft valdes vid 20 grader Celsius och 1 atmosfärstryck, luftmotståndskoefficienten C_d approximerades till koefficienten för ett halvklot. Dessa fås ur läroboken, Fundamentals of Thermal-Fluid Sciences [31].

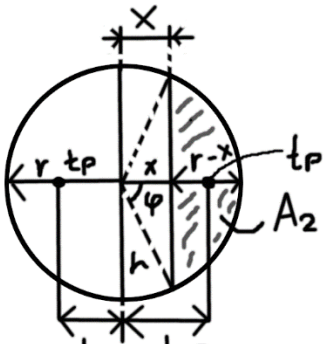
Diametern på extraljuset är 176 mm och tyngdpunkten sitter 40 mm in från främre kanten.

$$\rho_{luft\ 1\ atm\ 20°C} = 1.204\ \left[\frac{kg}{m^3}\right], C_d = 1.2,\ \mathrm{v} = 20\ \left[\frac{m}{s}\right], \mathrm{r} = 88\mathrm{mm}\,, d = 40mm,$$



Summan av moment kring centrum O moturs:

$$M_o = (F_1 \sin\alpha + F_2 \sin\alpha) * \mathrm{d} + F_1 \cos\alpha\, L_1 - F_2 \cos\alpha\, L_2 = (F_1 + F_2) * \sin\alpha + (F_1 L_1 - F_1 L_1) * \cos\alpha$$



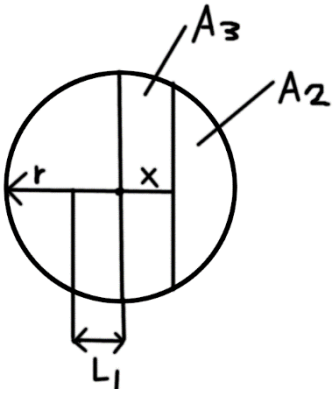
$$F_1 = P \times A_{1P} = P \times A_1 \times \cos\alpha \qquad F_2 = P \times A_{2P} = P * A_2 \times \cos\alpha$$

$$A_1 = A - A_2 = \pi r^2 - A_2$$

$$\cos\varphi = \frac{d}{r} = \frac{d}{r} \times \tan\alpha \Rightarrow \varphi(\alpha) = \cos^{-1}\left(\frac{d}{r} \times \tan\alpha\right)\ \varphi\ i\ radianer$$

$$A_2 = \frac{r^2}{2(2\varphi - \sin(2\varphi))}$$

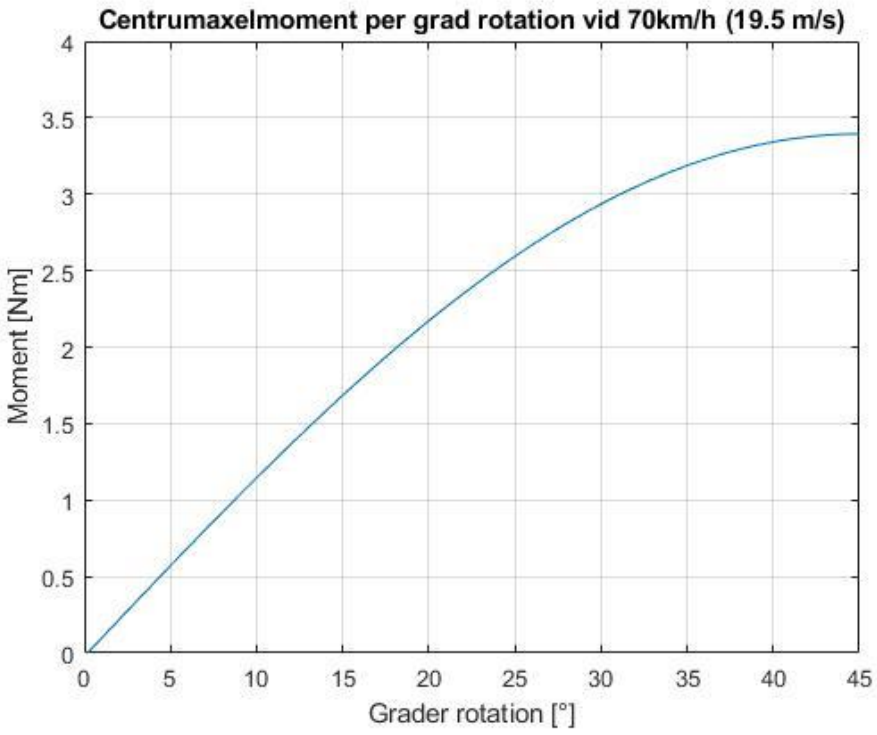
$$L_2 = \frac{\frac{2}{3} r^3 \times (\sin\varphi)^3}{A_2}$$



$$A_3 = \frac{\pi r^2}{2} - A_2$$

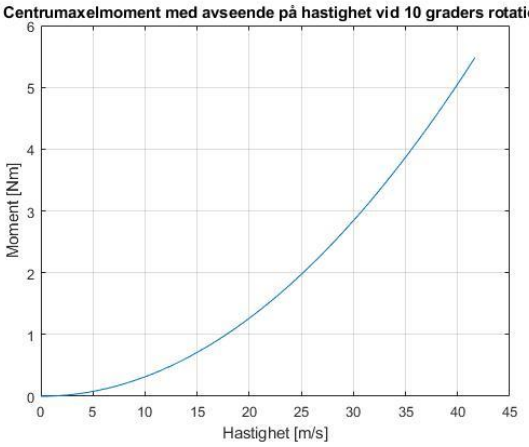
$$L_1 \approx \frac{\frac{\pi r^2}{2} \times \frac{4r}{3\pi} + A_3 \times \left(-\frac{x}{2}\right)}{\frac{\pi r^2}{2} + A_3} = \frac{\frac{2r^3}{3} - \left(\frac{\pi r^2}{2} - A_2\right) \times \frac{x}{2}}{\pi r^2 - A_2} \quad \text{Där } \mathrm{x} = \mathrm{d} \tan\alpha$$

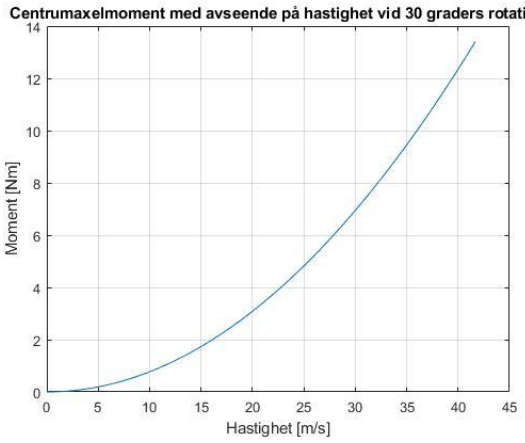
Instoppning av överstående ekvationer i momentekvationen ger summan av moment kring M_0:



För koncept 5 och 6 befinner tiltningsaxeln och rotationsaxeln genom tyngdpunkten för extraljuset. Där med har gravitationen ingen påverkan och ovanstående graf för momenten giltig för båda rotation och tiltning.

I nedanstående grafer kan summan av moment runt M_0 ses med avseende på hastighet vid 10 och 30 graders rotation.

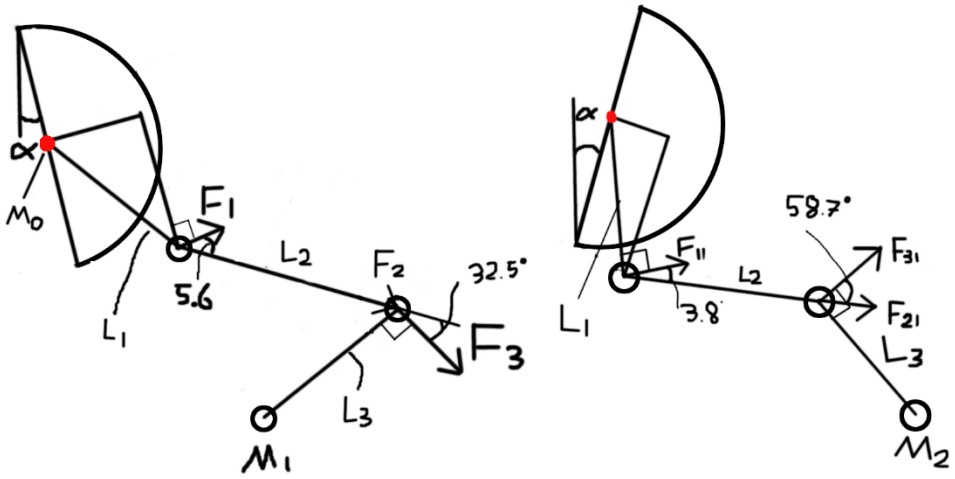




Koncept 5:

Rotationsmekaniken har en ”ett till ett” förhållande och där med blir momentet för servot enligt grafen ”Centrumaxelmoment per grad rotation vid 70km/h (19.5m/s)”.

Tiltningsmekaniken har en mer komplex mekanik där med valdes ändlägena att analyseras dvs 10 grader åt vadera håll.

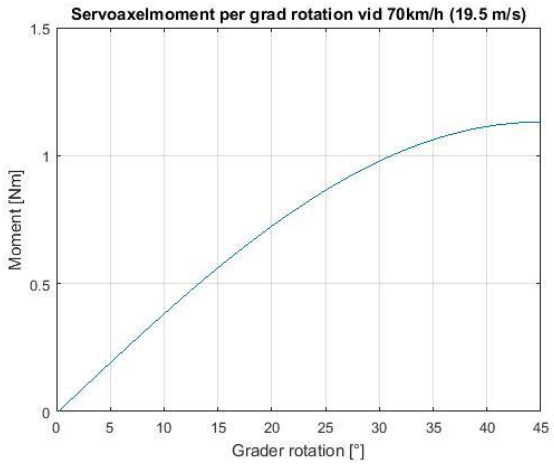


$$M_1 = L_3 * F_3 = L_3 * \frac{F_2}{\cos 32.5°} = L_3 * \frac{F_1 * \cos 5.6°}{\cos 32.5°} = L_3 * \frac{\frac{M_0}{L_1} * \cos 5.6°}{\cos 32.5°} \approx 0.37\ [Nm]$$

$$M_2 = L_3 * F_{31} = L_3 * \frac{F_{21}}{\cos 58.7°} = L_3 * \frac{F_{11} * \cos 3.8°}{\cos 58.7°} = L_3 * \frac{\frac{M_0}{L_1} * \cos 3.8°}{\cos 58.7°} \approx 0.59\ [Nm]$$

Koncept 6

Mekaniken för rotationen har ett 1:3 förhållande med M_0 där med blir momentet för servoaxeln som grafen till vänster nedanför. Tilningsmekaniken har ett 1:3.7 förhållande med M_0 vilket gör att momentet för servoaxeln blir enligt grafen till höger nedanför.



Bilaga 3: Arduino kod

```
//Libraries-------------------------------------------------
#include <EEPROM.h>

#include <FastLED.h>
#define DATA_PIN 7          //Datapin för ledstrip
CRGB leds[8];               //Ledstrip data array
#include <Servo.h>
#include <SPI.h>
#include "mcp_can.h"        //Lib för Can-bus

#include <MPU6050_tockn.h>
#include <Wire.h>
MPU6050 mpu6050(Wire);
float prev_pitch=0, pitch, pitch_der;
int ang;

//Variables and other initiations---------------------------------------------------------
const int SPI_CS_PIN = 10;  // Can-bus CS pin är inte default så korrigeras här
MCP_CAN CAN(SPI_CS_PIN);
unsigned char len = 0;
unsigned char buf[8];
unsigned char can_1[10];    //Array för invärdet från CAN-bus
unsigned char can_2[10];    //Array för invärdet från CAN-bus
unsigned char can_3[10];    //Array för invärdet från CAN-bus
unsigned char can_4[10];    //Array för invärdet från CAN-bus

Servo rot_left;             //Servo för sidledsrotation
Servo rot_right;            //Servo för sidledsrotation
Servo ud_left;              //Servo för höjdledsrotation
Servo ud_right;             //Servo för höjdledsrotation

unsigned int steering_angle;
float steering_angle_factor = 0.04395;

unsigned int steering_angle_sign;
int wheel_angle;

int alfa_angle,beta_angle,fok_line;             //Variabler för ljusvinkelreglering
double k1=0.2, k2=1.5, k3=1.2;                          //Dessa justeras för att ändra systemets funktion

unsigned int main_beam_indicator;

unsigned int vehicle_speed;
float vehicle_speed_factor = 0.01;

unsigned int light_sensor = 0;

String bluetooth_read, prev_bluetooth_read;
char raw_data[15];                              // (0 0 255 255 255)
int input[5];                                   // 0:mode 1:car 2:R_side 3:L_side 4:B_updown
int prev_input[5];                              // 0:mode 1:car 2:R_side 3:L_side 4:B_updown

int diagnostics[26];                            // Diagnosticsarray, indexering finns i bilaga
int cycle = 0, counter_pitch=0;

void setup(){
    diagnostics[16] = 0;                        //Setup not complete
    Serial.begin(38400);                        //Sets the baud for serial data transmission
    Wire.begin();
    mpu6050.begin();
    mpu6050.calcGyroOffsets(true);
    canbus_init();
    FastLED.addLeds<WS2811,DATA_PIN,GRB>(leds, 8 ).setCorrection(TypicalLEDStrip);
    FastLED.setBrightness(20);

    read_eeprom();
    init_led_show();
    init_servo();

    diagnostics[17] = 0;      //bluetooth not recieved data
    diagnostics[16] = 1;      //Setup complete
}
```

```
void loop(){
   if(Serial.available() > 0){                    //Om data kommer från bluetooth modulen
      bluetooth_read = Serial.readString();       //Så läggs all data in i en String
      bluetooth_read.toCharArray(raw_data, 15);   //Stringen omvandlas sen till en chararray med 15 positioner
      raw_data_converter();                       //Arrayen behandlas sen för att ta ut rätt värde
      diagnostics[17] = 1;                        //diagnostiken uppdateras med att bluetooth har mottagits
   }
   read_canbus();
   mpu6050.update();
   diagnostics[20]=1;

   move_servo();
   control_lights();

   if(cycle==100){
    update_diagnostics();
    view_diagnostics();
    cycle=0;
   }else{
    cycle++;
   }
}

void canbus_init(){
  if(CAN_OK == CAN.begin(CAN_500KBPS)){
    diagnostics[19] = 1;
  }else{
    diagnostics[19] = 0;
  }

    CAN.init_Mask(0, 0, 0xfff);           // there are 2 mask in mcp2515, you need to set both of them
    CAN.init_Mask(1, 0, 0xfff);

    CAN.init_Filt(0, 0, 0x496);           // main beam indicator
    CAN.init_Filt(1, 0, 0x10);            // Steering_angle
    CAN.init_Filt(2, 0, 0x148);           // vechel_speed
    CAN.init_Filt(3, 0, 0x20);            // light_sensor
   }

void init_servo(){
  rot_left.attach(3);                     // attaches the servo on pin 9 to the servo object
  rot_right.attach(5);                    // attaches the servo on pin 9 to the servo object
  ud_left.attach(6);                      // attaches the servo on pin 9 to the servo object
  ud_right.attach(9);                     // attaches the servo on pin 9 to the servo object
}
```

```
void raw_data_converter(){
  int num_var2, num_var3, int_flag=0;

  input[0] = raw_data[1]-'0';             //Första två variablerna är ensiffriga,
  input[1] = raw_data[3]-'0';             //eftersom första positionen är "{" ligger dom på 1 och 3 i raw_data
                                          //Dessa sparas till input 0 och 1. "-'0'" görs för att ta bort asciivärdet

  if(raw_data[6]-'0' == -16){             //Sen kan de resterande va antingen 1 eller 2 variabler i hex,
    num_var2 = 1;                         //om position 6 är ett mellanslag "-16" så finns bara en variabel
    input[2] = char_to_int(raw_data[5]-'0');  //och input[2] blir endast datan som finns omvandlat från char till int
  }else{
    num_var2 = 2;                             //Om det inte är ett mellanslag så sätts variabeln till 2
    input[2] = char_to_int(raw_data[5]-'0')*16+char_to_int(raw_data[6]-'0'); //och input[2] beräknas av båda positionerna
  }

  if(raw_data[7+num_var2]-'0' == -16){    //för att kolla på nästa del så behöver man veta om föreg. hade 1 eller 2 var
    num_var3 = 1;                         //med den informationen kolla algoritmen om indata nummer 4 har en eller två pos
    input[3] = char_to_int(raw_data[6+num_var2]-'0');    //likadant som tidigare påverkas avläsningen
  }else{
    num_var3 = 2;
    input[3] = char_to_int(raw_data[6+num_var2]-'0')*16+char_to_int(raw_data[7+num_var2]-'0');
  }

  if(raw_data[7+num_var2+num_var3+1]-'0' == -7){   //för den sista indata var. används de föreg. antal pos.
                                                   //för att räkna ut värdet på samma sätt som tidigare.
    input[4] = char_to_int(raw_data[7+num_var2+num_var3]-'0');
  }else{
    input[4] = char_to_int(raw_data[7+num_var2+num_var3]-'0')*16+char_to_int(raw_data[8+num_var2+num_var3]-'0');
  }

  for(int i=0;i<5;i++){                 //värdet på input arrayen jämförs med vad det va tidigare
    if(input[i]!=prev_input[i]){        //om värdet inte är samma så sätts flaggan
      int_flag=1;
    }
  }
  if(int_flag){                         //det nya värdet uppdateras till eepromen
    write_eeprom();
    for(int i=0;i<5;i++){
        prev_input[i] = input[i];       //sen uppdateras det tidigare värdet
    }
  }
}

int char_to_int(int in){
  if(in <= 9){
    return in;
  }else if(in > 9){
    in = in - 7;
    return in;
  }
}

void write_eeprom(){
  EEPROM.write(1,input[0]);           //skriver värdet i input[0] till plats 1 i eeprom
  EEPROM.write(2,input[1]);
  EEPROM.write(3,input[2]);
  EEPROM.write(4,input[3]);
  EEPROM.write(5,input[4]);
}

void read_eeprom(){
  input[0] = EEPROM.read(1);          //skriver värdet till input[0] som finns på plats 1 i eeprom
  input[1] = EEPROM.read(2);
  input[2] = EEPROM.read(3);
  input[3] = EEPROM.read(4);
  input[4] = EEPROM.read(5);
}
```

```
void read_canbus(){
  if(CAN_MSGAVAIL == CAN.checkReceive()){
      CAN.readMsgBuf(&len, buf);
      unsigned long canId = CAN.getCanId();

      if(CAN.getCanId() == 0x496){
        for(int i=0;i<len;i++){
          can_2[i] = buf[i];
        }
        main_beam_indicator = (can_2[2]/16)/8;
      }

      if(CAN.getCanId() == 0x10){
        for(int i=0;i<len;i++){
          can_1[i] = buf[i];
        }
        if(can_1[0] != 0 && can_1[1] != 0){
          steering_angle = ((int)can_1[7] + (can_1[6]/16)*4096 + (can_1[6]%16)*256) * steering_angle_factor;
          if(can_1[0]>16){
            steering_angle_sign=1;
            wheel_angle = steering_angle;
          }else{
            steering_angle_sign=0;
            wheel_angle = -steering_angle;
          }
        }
      }

      if(CAN.getCanId() == 0x148){
        for(int i=0;i<len;i++){
          can_3[i] = buf[i];
        }
        if(can_3[0] != 0 && can_3[1] != 0){
          vehicle_speed = ((int)can_3[7] + (can_3[6]/16)*4096 + (can_3[6]%16)*256) * vehicle_speed_factor;
        }
      }

      if(CAN.getCanId() == 0x20){ // ljussensor
        for(int i=0;i<len;i++){
          can_4[i] = buf[i];
        }
        if(can_4[0] != 0 && can_4[1] != 0){
          light_sensor = ((can_4[0]& B00000011)<<8) | can_4[1];
        }
      }
    }
}
```

```
void move_servo(){
  //Sideways---------------------------------------
  if(wheel_angle>=0){
    fok_line=wheel_angle*k3; //- vehicle_speed * k1;
    if(fok_line<0){
      fok_line=0;
    }
    alfa_angle=fok_line*k2;
    beta_angle=fok_line;
  }else if(wheel_angle<0){
    fok_line=wheel_angle; // + vehicle_speed * k1;
    if(fok_line>0){
      fok_line=0;
    }
    alfa_angle=fok_line;
    beta_angle=fok_line*k2;
  }

  if(alfa_angle>=250){
    alfa_angle=250;
  }else if(alfa_angle<=-250){
    alfa_angle=-250;
  }

  if(beta_angle>=250){
    beta_angle=250;
  }else if(beta_angle<=-250){
    beta_angle=-250;
  }

  //Up-down----------------------------------------
  if(counter_pitch==5){
    pitch = mpu6050.getAngleZ();
    pitch_der=pitch-prev_pitch;
    ang = pitch_der*200;
    if(ang>400){
      ang=400;
    }else if(ang<-400){
      ang=-400;
    }
    prev_pitch = pitch;
    counter_pitch=0;
  }else{
    counter_pitch++;
  }

  rot_left.writeMicroseconds(round(1500-alfa_angle*0.5));
  rot_right.writeMicroseconds(1500+beta_angle);
  ud_left.writeMicroseconds(1500-ang);
  ud_right.writeMicroseconds(1500+ang);
}

void control_lights(){
  if (main_beam_indicator == 1){
    digitalWrite(8, HIGH); // sets the digital pin 13 on
  }else{
    digitalWrite(8, LOW);  // sets the digital pin 13 off
  }
}
```

```
void update_diagnostics(){
  diagnostics[0] = input[0];    //Värdena läggs in i diagnostics
  diagnostics[1] = input[1];
  diagnostics[2] = input[2];
  diagnostics[3] = input[3];
  diagnostics[4] = input[4];

  diagnostics[5] = steering_angle;  //CAN-bus
  diagnostics[6] = steering_angle_sign;  //CAN-bus
  diagnostics[7] = main_beam_indicator;  //CAN-bus
  diagnostics[8] = vehicle_speed;  //CAN-bus
  diagnostics[9] = 0;  //CAN-bus
  diagnostics[10] = 0;  //CAN-bus
  diagnostics[11] = 0;  //CAN-bus
  diagnostics[12] = 0;  //CAN-bus

  diagnostics[13] = 0;  //L_servo side
  diagnostics[14] = 0;  //R_servo side
  diagnostics[15] = 0;  //B_servo updown

  //Setup complete
  if(diagnostics[16]){
    leds[0]=CRGB(0,255,0);
  }else{
    leds[0]=CRGB(255,0,0);
  }

  //Bluetooth recieved data
  if(diagnostics[17]){
    leds[1]=CRGB(0,0,255);
  }else{
    leds[1]=CRGB(255,0,0);
  }

  //Correct values
  if(input[0]==0 || input[1]==0 || input[2]==0 || input[3]==0 || input[4]==0){
    diagnostics[18] = 0;
    leds[2]=CRGB(255,0,0);
  }else{
    diagnostics[18] = 1;
    leds[2]=CRGB(0,255,0);
  }
  FastLED.show();
}


void shift_led(int end_point){
  for(int i=end_point;i>0;i--){
    leds[i] = leds[i-1];
  }
}

void shift_led_delete(int end_point){
  for(int i=end_point;i>0;i--){
    leds[i] = leds[i-1];
    leds[i-1] = CRGB(0,0,0);
  }
}
```

```
void view_diagnostics(){
  Serial.println("Diagnostics:");

  Serial.println("Input:");
  for(int i=0;i<5;i++){
    Serial.print(diagnostics[i]);
    Serial.print(" ");
  }
  Serial.println();

  Serial.println("CAN-bus:");
  for(int i=5;i<13;i++){
    Serial.print(diagnostics[i]);
    Serial.print(" ");
  }
  Serial.println();

  Serial.println("Servo:");
  for(int i=13;i<16;i++){
    Serial.print(diagnostics[i]);
    Serial.print(" ");
  }
  Serial.println();

  Serial.println("Flags:");
  for(int i=16;i<26;i++){
    Serial.print(diagnostics[i]);
    Serial.print(" ");
  }
  Serial.println();
  Serial.println();
  Serial.println();
}

void init_led_show(){
  int count=8;
  int order[8] = {2,7,0,1,5,3,6,4};
  for(int t=0;t<3;t++){
    for(int i=0;i<=13;i++){
      leds[0] = CHSV(i*20,255,255);
      shift_led(7);
      FastLED.show();
      delay(40);
    }
  }

  for(int i=0;i<8;i++){
    leds[order[i]]=CRGB(0,0,0);
    FastLED.show();
    delay(100);
  }

  for(int i=7;i>0;i--){
    leds[0]=CRGB(255,0,0);
    FastLED.show();
    delay(30+(7-i)*5);
    for(int j=0;j<i;j++){
      shift_led_delete(i);
      FastLED.show();
      delay(30+(7-i)*5);
    }
  }
  leds[0]=CRGB(255,0,0);
  FastLED.show();
  delay(100);
}
```

Bilaga 4: CAN-buss id

Namn	**Namn i CANalyse**	**Funktion**	**ID nr (hex)**	**Skickas från**	**Bitkodning**	**Längd (bitar)**	**Datatyp**	**Värde innan beräkning**	**Faktor**	**Offset**	**Efter beräkning**	**Ytterligare information**
Styrvinkel	SteeringAngle	Rattvnikel i grader från centrum	0x10	SAS	Motorola	15	Unsigned	0–32767	0,04395	0	0–1440,10965	Max rattutslag är ungefär 440
Rattutslag vänster/höger	SteeringAngleSign	Rattutslag höger eller vänster	0x10	SAS	Motorola	1	Unsigned	0–1	1	0	0–1	1 = höger 0 = vänster
Helljus	MainBeamIndicator	Visar om helljus är på eller av	0x496	CEM	Motorola	1	Unsigned	0–1	1	0	0–1	1 = helljus på 0 = helljus av
Hastighet	VehicleSpeed	Hastighet i km/h	0x148	BCM	Motorola	16	Unsigned	0–65535	0,01	0	0–320	

SAS = Steering Angle Sensor, CEM = Central Electronic Module, BCM = Brake Control Modul