Høgskulen på Vestlandet

Bacheloroppgave Elektro

ING3055

Predefinert informasjon

Startdato:	05-12-2019 09:00	**Termin:**	2019 HØST
Sluttdato:	18-12-2019 14:00	**Vurderingsform:**	Norsk 6-trinns skala (A-F)
Eksamensform:	Bacheloroppgave Elektro med muntlig presentasjon/eksaminasjon		
SIS-kode:	203 ING3055 1 BOPPG 2019 HØST Haugesund		
Intern sensor:	(Anonymisert)		

Deltaker

Kandidatnr.:	203

Informasjon fra deltaker

Tittel *:	Assisterende kjøresystem i trafikken		
Engelsk tittel *:	Driver assistance systems in traffic		
Egenerklæring *:	Ja	**Inneholder besvarelsen konfidensiell materiale?:**	Nei
Jeg bekrefter at jeg har registrert oppgavetittelen på norsk og engelsk i StudentWeb og vet at denne vil stå på vitnemålet mitt *:	Ja		

Gruppe

Gruppenavn:	(Anonymisert)
Gruppenummer:	2
Andre medlemmer i gruppen:	201

Jeg godkjenner avtalen om publisering av bacheloroppgaven min *

Ja

Er bacheloroppgaven skrevet som del av et større forskningsprosjekt ved HVL? *

Nei

Er bacheloroppgaven skrevet ved bedrift/virksomhet i næringsliv eller offentlig sektor? *

Nei

Høgskulen på Vestlandet

BACHELOROPPGAVE

Assisterende kjøresystem i trafikken

Driver assistance systems in traffic

Espen Andreas Vik og Mats Eriksen Kallevig

Bachelor i ingeniørfag, Elektro

Høgskulen på Vestlandet

Veileder: Gisle Yngvar Romslo Kleppe og Alf Jonny Høines

Innleveringsdato: 18.12.2019

Assisterende kjøresystem i trafikken

Høgskulen på Vestlandet - Campus Haugesund
Elektroingeniørfag

Skrevet av:
Mats Eriksen Kallevig
Espen Andreas Vik

Høsten 2019

BACHELORPROSJEKT

Studenten(e)s navn: Espen Andreas Vik og Mats Eriksen Kallevig

---

Linje & studieretning Bachelor i ingeniørfag, Elektro

Oppgavens tittel: Assisterende kjøresystem i trafikken

Oppgavetekst:

Det skal skrives et program som gjør det mulig for biler i trafikken å kommunisere seg imellom for å effektivisere bilkøer og motorveitrafikk.

Målet er at kjøretøyet skal kunne ha en selvkjørende funksjon som skal kunne reagere raskere enn det sjåføren kan, og skal også kunne kommunisere med andre kjøretøy som øker trafikksikkerheten og forbedrer flyten i trafikken.

Programmet skal bli simulert i CodeSys og sammenlignet med observert trafikk for å vurdere programmets effektivitet og funksjonalitet.

Endelig oppgave gitt: *23. Oktober 2019*

Innleveringsfrist: Onsdag 18. desember 2019 kl. 12.00

Intern veileder: Gisle Yngvar Romslo Kleppe

Ekstern veileder: Alf Jonny Høines
emailadresse ekstern veileder: aj-h@outlook.com

Godkjent av studieansvarlig: H. Sporsheim
Dato: 27/11-19

Høgskolen på Vestlandet
Campus Haugesund
Bjørnsonsgt. 45
5528 HAUGESUND

Oppgavens tittel	Rapportnummer
Assisterende kjøresystem i trafikken	*(Fylles ikke ut)*

Utført av

Mats Eriksen Kallevig

Espen Andreas Vik

Linje	Studieretning
Elektro y-vei	Ingeniør

Gradering	Innlevert dato	Veiledere
Åpen	18.12.2019	Gisle Yngvar Romslo Kleppe - Intern Alf Jonny Høines - Ekstern

Ekstrakt

Trafikken i rushtider kan være meget tidkrevende og koste samfunnet samt hver enkelt person tid og penger som skaper irritasjon og misnøye på veiene, noe som gruppen har bemerket seg. I denne sammenheng er det laget et program med assisterende kjørefunksjoner for biler som skal hjelpe sjåføren slik at trafikken kan bli mer effektiv og trafikksikker enn i dag.

Med dette programmet er det laget en simulering med flere biler som ble sammenlignet med observert trafikk for å få en indikasjon om mer automatisering av biler kan effektivisere trafikken

Forord

Denne bachelorrapporten er skrevet av en gruppe på to heltidsstudenter fra elektro Y - vei linjen ved Høgskulen på Vestlandet, campus Haugesund. Begge har erfaring som elektriker med fagbrev og ser frem mot å ferdigstille studiene.

Vi vil meddele takknemlighet til vår veileder Gisle Yngvar Romslo Kleppe fra HVL, som har svart på spørsmål og hjulpet gruppen godt.

Vi vil også takke foreleserne fra Høgskulen på Vestlandet, campus Haugesund som har svart på spørsmål og lært fra seg.

Innholdsfortegnelse

Tabeller og figurer

Forkortelser og begrep

ADAS - Advanced Driver Assistance Systems

ADS - Automated Driver System

GPS - Global Positioning System

PLS - Programmerbare logiske styring

Codesys - Programvare for utvikling og simulering av automatiseringssystemer.

ST - Strukturert tekst. Programmeringsspråk designet for programmerbare logiske styring.

USB - Universal Serial Bus

Fartsholder - Teknologi som automatisk kontrollerer og regulerer et kjøretøys hastighet. Kalles også for Cruisekontroll.

Sammendrag

Med dagens trafikale problemer der en ser den ineffektive rushtrafikken som skaper farer og skaper sikkerhetsrisikoer, samtidig som det senker samfunnets produktivitet har gruppen sett nærmere på et virkemiddel som kan være en av løsningene til dette. Siden i sommer er det blitt laget et program som skal kunne hjelpe en bil i køkjøring der gruppen har laget det slik at det kan brukes i en simulering.

Målet med simuleringen er å vise at en automatisert kjørefunksjon implementert i bilene der de står i kø vil være mer effektivt enn om de var styrt av en sjåfør. For å kunne få en indikasjon om dette er det hentet inn data fra trafikken som viser sjåførenes kjøremønster som kan sammenlignes med de simulerte verdiene. Trafikkdata som er hentet er basert på observasjoner gjort i trafikken i bestemte scenarier: Kø med lyskryss og kø med fotgjengerfelt.

Simuleringens scenario er da laget på denne måten:

Det er programmert inn seks biler som skal kommunisere med hverandre.

De seks bilene er programmert slik at de står i en konstruert kø der de venter på klar bane til å kjøre videre.

Det er programmert inn en hindring som den første bilen må vente på til den er borte, når hindringen er borte kan den første bilen begynne å akselerere som da gir klar vei for de andre bilene bak til å begynne å akselerere.

Det er tatt tiden av den femte og den sjette bilen når de har kommet til hvor den første bilen startet fra. Med disse tidene kan det sammenlignes med den observerte trafikken.

Disse simuleringen kan da gi en indikasjon på om det er raskere for biler med automatiserte kjørefunksjoner i kø å komme seg forbi et bestemt punkt enn om det er en sjåfør som har kontrollen av kjøretøyet alene.

1 Innledning

1.1 Bakgrunn

I perioden der gruppen var på utkikk etter interessante oppgaver så ble gruppen oppmerksomme på de store utfordringene med vegtrafikk og de utfordringene dette gir samfunnet. Det ble interessant å sjekke opp om hvilke løsninger som er tilgjengelige til dette formålet og det ble besluttet å se nærmere på assisterende kjøresystemer. Gruppen bestemte seg for å lage et PLS system som skal kunne hjelpe en bil til å kjøre mer effektivt i køer og trafikk, og det ble også programmert slik at avstander skal kunne holdes automatisk sammen med hastigheter og fartsgrenser.

1.2 Formålet med oppgaven

Formålet med oppgaven er å lage et program som skal hjelpe sjåføren slik at køkjøring blir effektivisert, og da vise at det har en betydning i det store bilde, programmet må også kunne bidra til sikkerheten og hjelpe til med kjøringen slik at køer ikke skapes.

Programmet vil bli simulert i Codesys og sammenlignet med observert trafikk for å vurdere programmets effektivitet og funksjonalitet.

2 Bilen og Trafikken

2.1 Om bilen

Bilen er et sentralt verktøy i dagens samfunn til transport av både personer og utstyr. Med sitt opphav fra Karl Friedrich Benz oppfinnelse i 1883, så har bilen hatt en enorm utvikling siden den gang til i dag. Bilene var ofte tunge, lange og vanskelige å styre som illustreres i Figur 1, gjorde dem usikrede der mange av delene var av dårlig kvalitet. Driftseffektivitet, sikkerhet og komfort har kommet en lang vei siden Benz rullet ut sin oppfinnelse og i dag er bilen blitt noe hverdagslig. Men produsenter ser enda etter bedre løsninger og måter å effektivisere bilene på som gjør det enklere og tryggere for sjåføren og passasjerene (1).

Figur 1: Veteranbiler, Teknisk Museum Praha, Tsjekkia. Foto: Mats Kallevig.

2.2 Trafikk

Trafikk er et stadig økende problem som bringer med seg store utfordringer for fremtiden. I 2014 ble det estimert at amerikanere taper 6,9 milliarder timer og 12 milliarder liter med ekstra drivstoff i løpet av året. På grunn av svinn i trafikken alene er det regnet ut at det koster amerikanere generelt 960 USD i året (2). Man kan trygt si at det å løse opp trafikken er et fremtidsrettet mål.

2.3 Køkjøringen

Køkjøring blir skapt på flere måter, men et kjent tilfelle som forårsaker mange unødige stopp i trafikken er tilfellet der en får små hindringer eller andre grunner som får en sjåfør til å stoppe eller senke ned farten som skaper en bølgeeffekt. Dette skjer på grunn av at hver enkelt sjåfør har en dødtid og vil derfor ikke reagere umiddelbart når det er deres tur til å kjøre, og dette blir da gjeldende for hver av de sjåførene etter (3). Om man da også legger til manglende oversikt og kommunikasjon med sjåførene og bilene imellom vil en få denne effekten forsterket.

2.4 Autonome Kjøre systemer

2.4.1 Hva er autonome biler

Autonome biler er selvkjørende kjøretøy som kan kjøre med minimal, eller null, menneskelig styring. Disse kjøretøyene bruker kunstig intelligens og eksisterende teknologi som adaptive fartsholdere til å automatisere kjøringen.

Selvkjørende kjøretøy kan variere i kompleksitet fra enkle systemer som må konstant overvåkes av en menneskelig sjåfør, til systemer som kan styre seg selv i alle kjøreforhold uten noe menneskelig element i det hele tatt.

2.4.2 Hvordan fungerer autonome biler

Autonome biler bruker en kombinasjon av kunstig intelligens og eksisterende kjøretøysystemer basert på Advanced Driver Assistance Systems (ADAS) for å lage noe som er kjent som Automated Driver System (ADS) (4).

Denne kunstige intelligensen er kjernen i den autonome bilen som behandler innsignaler fra diverse sensorer som er bygd inn i bilen og bruker de signalene til å lage et bilde av verdenen utenfor. Med det bildet, kombinert med kart og Global Positioning System (GPS) data, så kan det autonome kjøretøyet trygt lage en rute gjennom trafikkbildet (4).

Den kunstige intelligensen er koblet til og bruker kjøretøyets gasspedal, bremser og rattstyring til å flytte seg fra A til B. Kjøretøyets sensorer, som kan inkludere alt fra radar til laser, kan detektere fotgjengere eller andre kjøretøy og den kunstige intelligensen gjør korrigering for å tilpasse fart og unngå ulykker.

Autonome kjøretøy er også ofte designet for full menneskelig kontroll og ADS fungerer som en avansert type fartsholder, der sjåføren kan overta kontroll når en måtte ønske.

Noen autonome biler er designet for å kjøre uten det menneskelig element i det hele tatt, men lovligheten av førerløse biler varierer mye i de forskjellige amerikanske delstatene (5) og i Europa. Flere europeiske land (6), inkludert Norge (7), har lover som kun tillater testing av autonome biler mens flere delstater i USA har lover som tillater fullt autonome biler uten menneskelige sjåfører. Vist i Figur 2 er en selvkjørende shuttlebuss som testes i Norge.

Figur 2: Selvkjørende shuttlebuss på Forus i Norge. Kilde: Ina Steen Andersen

2.4.3 Nøkkelsystemer autonome biler er bygd opp av

De viktigste systemene i en autonom bil er kjent som ADAS (4), disse systemene er designet for å gjøre kjøreopplevelsen mer behagelig og tryggere.

Kunstig intelligens gjør autonome biler mulig. Disse kjøretøyene er styrt av programmer som er utviklet og trent gjennom maskinlæring for å kunne lese data fra bilens sensorer til å vurdere og foreta en handling i alle situasjoner.

Elektroniske signalbaserte koblinger har erstattet mekaniske koblinger for styring, bremsepedal og gasspedal over de siste årene. Dette gjør det mye enklere for den kunstige intelligensen og innebygde dataprogrammer til å kontrollere de forskjellige systemene.

Filholder har originalt vært designet for å hjelpe sjåføren å holde seg i sin egen fil i trafikken, men autonome kjøretøy bruker samme sensorene og virkemåtene for å holde seg på veien.

Automatisk bremsing er originalt designet for å forhindre ulykker ved å automatisk bremse i situasjoner der sjåfører har vært for sen til å reagere. Dette blir brukt i en større skala i autonome biler ved hjelp av kunstig intelligens.

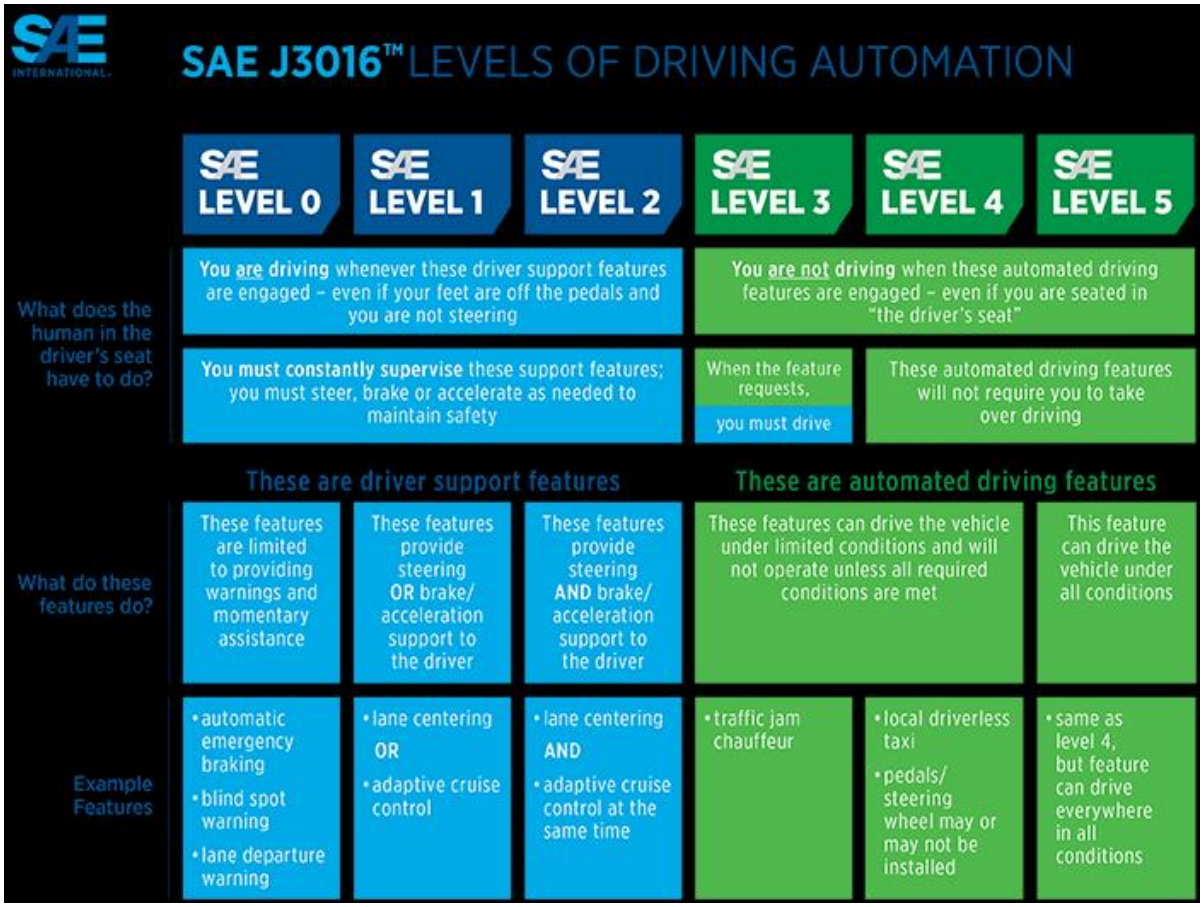
Adaptiv fartsholder er en intelligent type fartsholder som holder ønsket fart og tilpasser seg trafikken og akselererer og bremser automatisk for å holde tritt med trafikken. Opprettholder en trygg avstand mellom egen bil og bilen foran uten at sjåføren må holde foten konstant på gasspedalen.

Lidar, optisk fjernmålingsteknikk, brukes til måling av objekters fysiske posisjon. En moderne Lidar kan registrere mer enn 1 million posisjoner per sekund (8).

Kart og GPS er også sentrale hjelpemidler for at den kunstige intelligensen kan lage en rute fra A til B.

2.4.4 Nivåer av automatisering

The Society of Automotive Engineers (SAE) (9) og US Department of Transportation har utarbeidet beskrivelser (4; 10) av de forskjellige nivåene av automatisering som vist i Figur 3.



Figur 3: Liste over autonome nivåer. Kilde: Society of Automotive Engineers

Nivå 0 involverer ingen automasjon. Dette er kjøretøy som sjåfører må styre selv uten noen hjelpefunksjoner. Disse kjøretøyene kan ha funksjoner som nødbrems, konvensjonelle fartsholder (Cruisekontroll) og advarsel hvis kjøretøyet går utenfor veimerkingen.

Nivå 1 gir sjåførhjelp i form av enten adaptiv fartsholder eller filholder. Dette nivået krever enda at sjåføren styrer selv eller overvåker kjøretøyet og sikrer at systemet fungerer trygt ved bruk av fartsholder og filholder.

Nivå 2 er delvis automasjon. Kjøretøyet har fått noe nivå av autonom kontroll over funksjoner som akselerasjon, bremsing og styring. Sjåføren har enda kontroll og kjøretøyet fungerer ikke uten en menneskelig sjåfør. Kjøretøy som dette har typisk ADAS-lignende automatiserte bremsing, adaptiv fartsholder og filholder.

Nivå 3 er "betinget automasjon", disse kjøretøyene er teknisk sett autonome. De kan navigere fra en plass til en annen, identifisere farer og reagere på dem. En menneskelig sjåfør er enda nødvendig for nødsituasjoner og sjåfører må være forberedt og klar til å ta over kontrollen. Alle systemer på dette nivået må være automatiserte og disse bilene trenger også en omfattende kunstig intelligens kapasitet for å kjøre trygt uten rettelser fra sjåfør.

Nivå 4 er høy automasjon. På dette nivået er kjøretøyet helautomatisert. Det kan trygt navigere fra et sted til et annet under de fleste kjøreforhold. Under tøffere forhold som snø eller kraftig regn kan kjøretøyet trenge endring og korrigeringer fra en menneskelig sjåfør. Denne type autonom bil kan teknisk sett kjøre uten mennesker til stede, men kan inkludere muligheten for at sjåføren tar kontroll over styringen.

Nivå 5 er full automasjon. Kjøretøy på dette nivået av automatisering er virkelig autonome og kan kjøre i førerløs tilstand under alle kjøreforhold. Avhengig av designet så kan sjåføren ta over styringen manuelt, men disse kjøretøyene er ikke laget for å være avhengig av menneskelig styring.

Per 2019, så finnes det ingen kommersielle, tilgjengelige nivå 3 - 5 automatiserte kjøretøy system (10).

2.5 Fordelen med autonome biler

Hovedfordelen med autonome biler, og drivkraften bak utviklingen, er sikkerhet. Mer enn 90 % av alvorlige ulykker er forårsaket av menneskelig feil (4). Den grunnleggende ideen er at hvis det menneskelige elementet blir tatt ut av ligningen, kunne mange liv bli reddet.

Den mer tidsmessige fordelen av autonome biler er at de potensielt kan redusere kødanning og stans i trafikkbildet ved å kjøre mer effektivt. Dette kan resultere i kortere tid brukt til å for eksempel pendling fra og til jobb hver dag.

I helautomatiserte biler kan sjåføren bruke tiden sin på andre ting enn å styre bilen, som for eksempel lesing, oppdatere seg på nyhetene eller å forberede seg til jobben.

En annen fordel er at eldre og funksjonshemmede personer kan få økt mobilitet ved bruk av autonome biler. Siden disse kjøretøyene kan kjøres helt autonomt, så kan de trygt brukes av folk med redusert syn og reaksjonsevne.

3 Programmerbare logiske styringer

Dette kapittelet omhandler Programmerbare logiske styring (PLS) og programvaren Codesys brukt for programmering.

3.1 PLS

3.1.1 PLS-ens Historie

PLS har sin opprinnelse i rele-baserte kontrollsystemer, også kalt trådbundet logikk.

Før PLS-en kom var det vanlig i industrien at en brukte store styringer med releer, kontaktorer, tidsur og tellere. Slike styringer krever mye plass og ekstra arbeid da de kan fylle store skap med elektromagnetiske releer og mye kabling som vist i Figur 4. I slike systemer må elektrikeren montere styringer etter på forhånd utarbeidede releskjema, disse viser hvordan alle brytere, sensorer, motorer, ventiler, releer osv. skal kobles sammen.

Figur 4: Bilde av styreskap med kontaktorer. Kilde: Wallpaperfare

Ulempen med maskinvarestyring er at de tar stor plass og er arbeids- og tidkrevende å implementere og å gjøre endringer på. En relestyring kan ha flere titalls med releer om ikke flere hundre der det vil være ledninger på kryss og tvers. Om da disse styringene skal endres på må hele den fysiske styringen endres og kobles om, og dette vil ta tid og vil derfor være relativt kostbart. Disse styringene har også variert levetid på de forskjellige komponenter slik at det ofte må vedlikeholdsarbeid til for å bytte diverse releer, brytere eller kontaktorer. Dette skaper ekstra muligheter for at deler av anlegget skaper driftsstanser og kostbare vedlikeholdsinngrep (11).

De første PLS-en kom i kommersielle produksjon etter General Motors etterlyste en erstatning for rele styringene. Økt konkurranse og større krav fra kundene krevde større effektivitet, og et naturlig trinn var å få et programvarebasert system som kunne erstatte releene (11).

3.1.2 Oppbyggingen av en PLS

Forenklet kan vi si at en PLS fungerer på samme måte som en datamaskin. Vi kan dele en PLS opp i 6 hoveddeler:

- Innganger
- Strømforsyning
- Sentralenhet (CPU)
- Minne
- Kommunikasjon
- Utganger

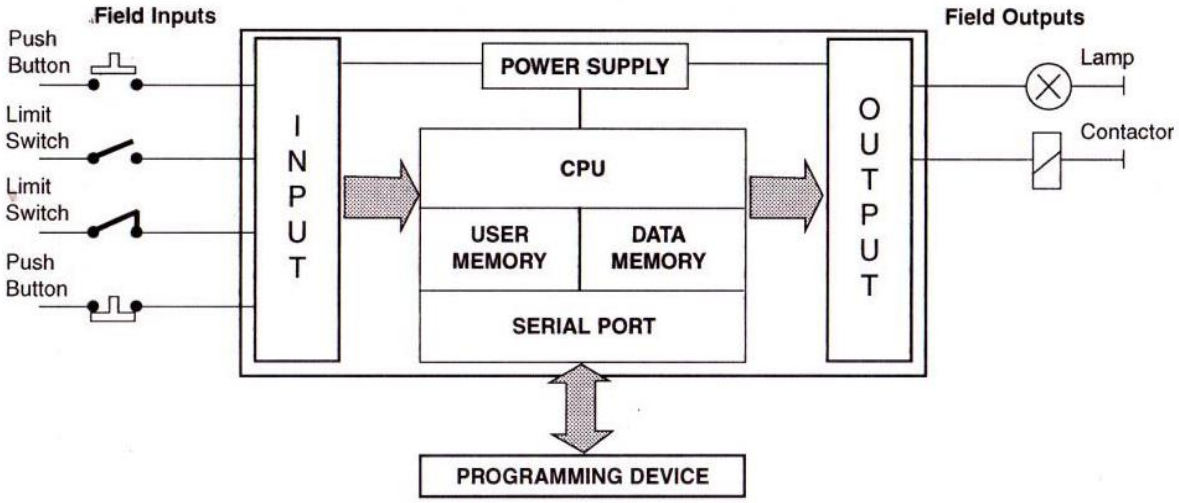
CPU er hjernen i PLS-en. Her utføres alle instruksjonene og beregninger og den styrer informasjonsflyten og hvordan programmene utføres.

Minnets Størrelse vil variere fra et PLS-merke til et annet. Minnet kan ofte utvides ved å legge til ekstra minnekort. I en PLS vil vi vanligvis ha forskjellige sett med hukommelser: ROM for permanent lagring av operativsystemet og systemdata, mens RAM brukes for lagring av program. Siden informasjon i et RAM ikke kan opprettholdes uten strøm, og for at ikke programkoden skal gå tapt ved strømbrudd har PLS-er et batteri.

Kommunikasjonsenheten inneholder blant annet en eller flere protokoller for håndtering av kommunikasjon. Alle PLS-er har en tilkobling for programmeringskabel eller operatørpanel, skrivere eller nettverk. PC og PLS kan fysisk kobles sammen via, for eksempel USB, men i nyere systemer er det også vanlig å gjøre dette gjennom nettverk, Ethernet eller lignende. Noen PLS-er har også innebygde webservere.

Strømforsyning er noe en PLS må ha, dette kan være en utskiftbar modul eller på mindre PLS-er så kan den være integrert del av prosessoren. Forsyningen må ha en driftsspenning og dette kan variere fra 220 volt AC til 24 volt DC. Den siste av disse er gjerne den mest hensiktsmessige å ta i bruk siden en finner gjerne 24 volt DC ute i anlegget og dette er også standard spenning for de fleste sensorer og transmittere (11).

Figur 5 viser en blokkskjematisk oppbygging av PLS.



Figur 5: Blokkskjema av en PLS. Kilde: Pinterest

Hoveddelene er knyttet sammen med et adresserbart system med ledninger/kobberbaner som kalles USB/busser. All kommunikasjon mellom hoveddelene i PLS-en foregår via disse bussene. Vi kaller et system som kommuniserer med forskjellige deler av en struktur der det kan gis kommandoer og sende informasjon via et adressesystem for busser. Kommunikasjonen kan gjøres med en samling av et antall ledere der informasjon overføres binært 1 bit per leder parallelt. Typisk vil en PLS ha 4 busser: Adressebuss, databuss, kontroll buss og systembuss (11).

- Adressebuss brukes for å overføre adressene til hvor i minnet data skal hentes eller sendes. En adressebuss med 16 linjer kan overføre 2^{16}= 65536 ulike adresser.
- Databuss brukes for overføring av data mellom CPU, minne og I/O.
- Kontrollbussen brukes til synkronisering og styring av trafikk retningen.
- Systembussen brukes til I/O-kommunikasjon.

Inn- og Utganger er en PLS sine kontakter med omverdenen. For en modulær PLS inngår alle inn- og utganger i blokker eller moduler som er konstruert for å ta imot ulike typer signaler og å sende ut signaler på ulike format. Det er inngangsblokker for digitale signal, analoge signal, termoelement, enkodere osv. Og følgelig finnes det tilsvarende blokker for utganger.

Hver inngang og utgang har en unik adresse som kan benyttes i programkoden. I/O-modulene sørger for elektrisk isolasjon for å beskytte PLS-en, og har ofte innebygde funksjoner for signalbehandling. Som regel gjør dette det slik at en kan koble direkte opp inn- og utgangene uten å måtte bruke ekstra elektriske kretser.

Digitale inngangssignaler har som regel et potensial på 24 V DC, men den interne spenningen i PLS-en er gjerne på 5 V. For å beskytte elektronikken i PLS-en, benyttes ofte en optokopler i inngangsmodulene. En optokopler består i hovedsak av en LED og en fototransistor.

Når PLS-en leser analoge innsignaler så går disse gjennom en analog til digital omformer, altså en A/D omformer. Omformeren er bygget inn i de analoge inngangsmodulene og et signal blir da kontinuerlig samplet og omgjort til binære verdier. I prinsippet behøves det bare 1 bit for å lagre tilstanden til digitale inngang, benyttes ofte 16 bit for å lagre verdien til en analog inngang (11).

Standard diskrete utganger deles opp i 3 ulike kategorier:

- Rele-utganger som tåler stor belastning og kan tilkobles både DC- og AC-laster, og i tillegg også gir isolasjon mellom PLS og ytre kretser.
- Transistorutganger er mye raskere enn rele-utganger. De er også billigere, men de er mer følsomme overfor overbelastning og feil polaritet. Transistorutgangene kan heller ikke ta imot AC signaler uten at det anvendes et eksternt rele.
- Triacutganger er ikke så vanlige. De benyttes i de tilfellene en trenger hurtig utganger for AC. Slike utganger er også følsomme for overstrøm.

3.2 Programvaren

Programvaren (12) som er brukt i dette forsøket er Codesys utviklet av Smart Software Solutions GmbH. Gruppen har valgt å bruke denne programvaren av flere ulike grunner som funksjonalitet, brukervennlighet, pris og egen erfaring.

Codesys kan simulere mange sett med programkoder og kan derfor bli brukt til å utvikle en ide ved programmering, samt å teste og visualisere simulering av komplekse anlegg på en datamaskin. Dette gjør at programvaren er enkelt å bruke i utviklingsfasen av et prosjekt samt å kjøre programmet ut på fysiske PLS-er. Codesys inneholder alle funksjonene til en PLS og følger standarden IEC 61131-3 i stor grad og er derfor et egnet program til å ta i bruk til dette formålet.

Codesys er ikke utviklet av en PLS fabrikant, som har gjort at det er mange utstyrsprodusenter som har valgt å ta dette i bruk som utviklingsverktøy for sitt utstyr bla. WAGO. Det at Codesys kan lastes ned gratis for ikke-kommersiell bruk, gjør at programvaren er enkelt tilgjengelig og derfor ofte brukt av studenter (11).

3.3 Versjon 3.5 sp 15

Codesys versjon 3.5 sp 15 er den aktuelle versjonen brukt i dette forsøket, denne versjonen har en oppgradert visualiseringsfunksjon som forenkler simuleringen. I tillegg har denne versjonen en oppdatering av designet og mindre funksjonsendringer som skal gjøre programmet mer behagelig, oversiktlig og brukervennlig.

3.4 Strukturert Tekst

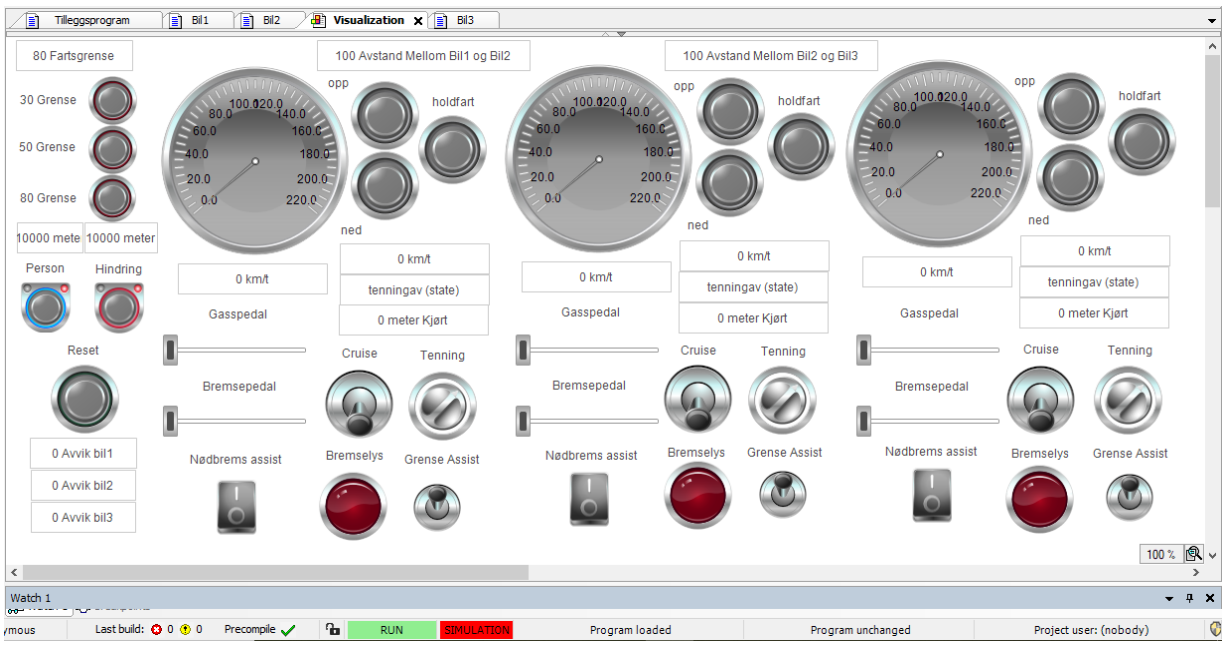
Strukturert tekst (ST) er et programmeringsspråk som blir skrevet som en serie med utsagn som skilles med semikolon, utsagnene bruker forutbestemte betingelser og underrutiner til å endre variabler. Disse variablene kan være definerte verdier, internt lagrede verdier eller innganger og utganger.

En stor fordel med det å bruke ST er at de fleste funksjoner og funksjonsblokker definert i standarden blir programmert i ST av produsentene av utviklingsverktøyet, så da er det en fordel å bruke ST i programmet også. Det er også sånn at i større programmer så er det mye enklere å programmere i ST da siden det er mye mer plass og tidssparende om en er vant med programmering og høynivå språk som ST er.

Ved arbeid med kombinatoriske styringer eller sekvensielle styringer som er basert på logikk, trenges det ikke annet enn å kunne programmere logiske betingelser som OG, ELLER, osv. Håndtering av analoge signaler krever litt tallbehandling og sammenligninger. Dette kan da gjøres med bruk av ulike funksjoner.

3.5 Visualisering

Visualisering er et nyttig verktøy når det kommer til utvikling og testing i sammenheng med simuleringen. Når en kjører simulering gjennom visualiseringsverktøyet vil en få en betydelig bedre oversikt over forsøket, og dette gjør det mye enklere å feilsøke i programmet når en har en strukturert oversikt over alle eventuelle feil som dukker opp under simuleringen. Det er brukt visualisering og simulering som en av nøkkelmetodene for å teste programmet som er utviklet og sammenlignet resultatene med tester og forsøk tatt i sammenheng med oppgaven. Figur 6 er fra visualiseringen under utvikling av programmeringen.



Figur 6: Bilde av visualisering fra under utvikling med tre biler

4 Programmering

4.1 Planlegging

I planleggingsfasen undersøkte gruppen hvilke program som var best egnet til både programmering og simulering. På grunn av erfaring og brukervennlighet, falt valget på Codesys, brukt i faget ING2051 - Styringsteknikk.

Case-struktur i Strukturert tekst ble fort det beste valget på hvordan programmert skulle skrives. Ved Case-struktur kan man lettere bygge opp et utgangspunkt for en bil, som er enklere å utvide og endre på, enn andre programmeringsspråk og metoder som Ladder, Funksjonsblokkdiagram eller Sekvensielle funksjonskart.

Strukturert tekst er også spesielt utviklet for å jobbe med matematiske funksjoner, som er nødvendig for å programmere et bilsystem.

Hver bil kan også bli programmert på separate simulerte applikasjoner som er sammenkoblet, for å simulere biltrafikk der biler kommuniserer med hverandre. Ved å bruke en applikasjon for hver bil, kan man kjøre en tilstandsbasert Case-struktur på hver bil, for å best simulere en bil med mange og avanserte hjelpefunksjoner.

Det ble tatt en vurdering ved hjelp av automatiseringsnivåene hvilket nivå programkoden skulle ligge på. På grunn av at rattstyring er vanskelig å simulere i Codesys samtidig som fartsregulering, så falt oppgaven på nivå 1 der sjåføren får hjelp til kun akselerering og bremsing.

Målet var å programmere en bil som fungerte godt nok for bruk i simulering, for å så sette flere biler inn i en trafikksammenheng.

Før programmeringen startet, ble det laget en oversikt ved å lage funksjonsbeskrivelser og tilstandsdiagrammer som programmet er basert på.

4.2 Begrensing

Programmet er begrenset til en rettlinjet endimensjonal bevegelse i negativ retning langs x-aksen. Programmet styrer kun bilenes hastighet i en rett linje, relativ til andre biler og hindringer, ved bruk av bilfysikk som motorkraft, bremsekraft, rullemotstand og luftmotstand. Ratt og styring av hjul er ikke med i problemstillingen og er dermed ikke programmert eller simulert.

Rullemotstand og luftmotstand er kun aktiv når gasspedalen ikke er trykket inn, slik at ved null gasspådrag, så har bilen en retardasjon som sakker ned farten helt til bilen står stille.

Simuleringen er basert på at det er tørre veiforhold, pluss grader i været og at gode dekk er brukt. Akselerasjonen i simuleringen og observasjoner er så lave, at det er dermed ikke simulert noe glidende friksjon mellom dekk og vei, slik at dekkene har 100 % veigrep under både akselerasjon og retardasjon i simuleringen.

Posisjoner og avstandsmåling mellom biler og hindringer er beregnet på bilenes fart over tid og initial-avstander i simuleringen.

4.3 Tilstandsbasert design

Programmet er designet ved bruk av Case-struktur, som kan beskrives som et sett av unike og distinkte tilstander (states). Fordelen er at kun en tilstand kan være aktivt om gangen. Prosessen i en tilstand fortsetter til et av kriteriene for tilstandsbytte er oppfylt, som for eksempel at en sensor eller bryter aktiveres. Ved bytte til en ny tilstand, kan prosessen endres avhengig av kriteriene til den nye tilstanden.

Fordelen med Case-struktur er at det er mulig å programmere slik at en tilstand kan byttes til hvilken som helst annen tilstand. Dette gir mye fleksibilitet ved utvikling av komplekse programmer der situasjoner konstant endrer seg.

Under er Figur 7, et utdrag av programkoden fra vedlegg D som viser tilstanden Holdfart_cruise som holder farten når fartsholderen er aktivert. En ny tilstand aktiveres dersom kriteriet til hvis (IF og ELSIF) er sant. Kriteriene blir sjekket hvert syklus-intervall på 50 ms. Fra denne tilstanden kan programmet gå over til 5 andre tilstander; Tenning_av, Frikjoring, Aks_cruise, Deaks_cruise og nodbremse_cruise.

```
154
155   Holdfart_cruise:
156
157   Bil_tilstand := 'holdfartcruise';
158
159   IF NOT Tenning THEN
160   State := Tenning_av;
161   ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
162   Set_fart := 0;
163   Cruise := FALSE;
164   State := Frikjoring;
165   ELSIF Avvik > 0.5 THEN
166   State := Aks_cruise;
167   ELSIF Avvik < -0.5 THEN
168   State := Deaks_cruise;
169   ELSIF Nodbrems AND Nodbremse_assist THEN
170   State := Nodbrems_cruise;
171   END_IF
```

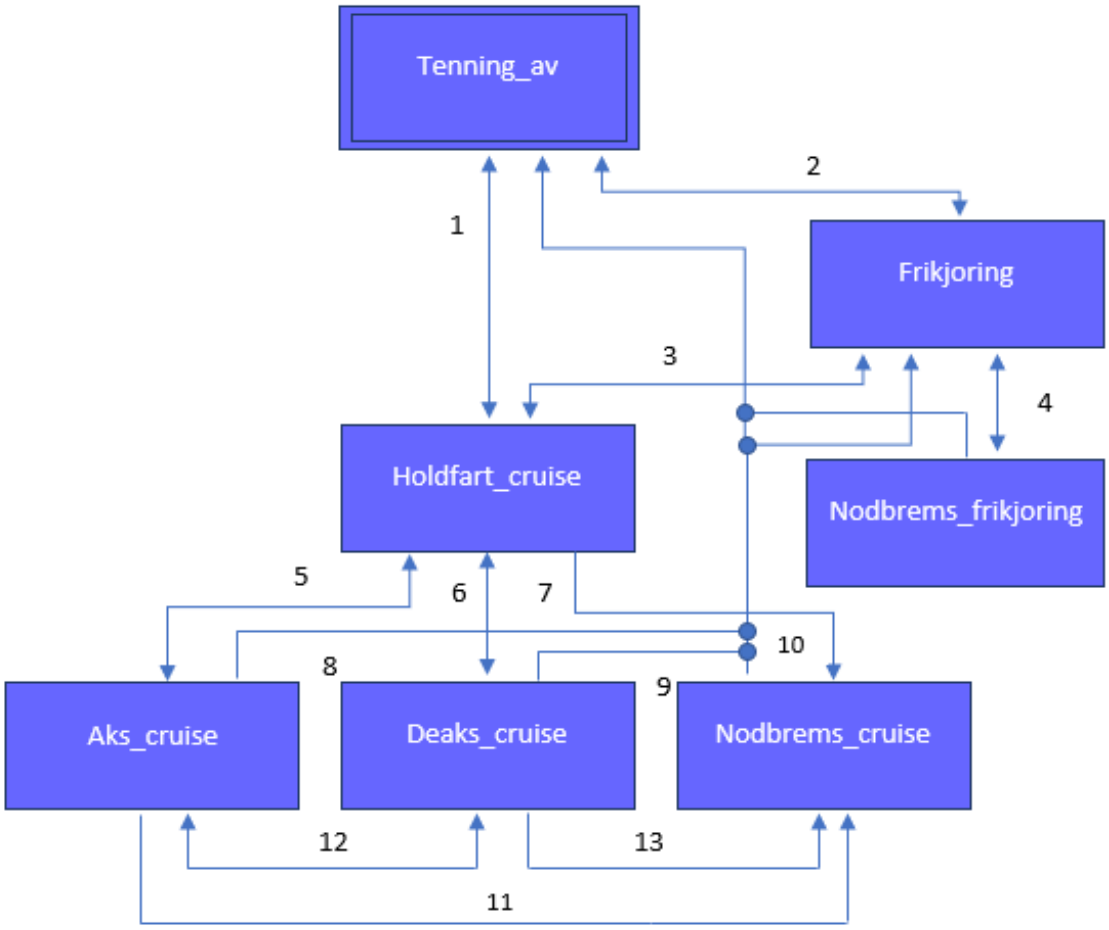
Figur 7: Programkode for Holdfart_cruise tilstand

Denne tilstanden sjekker blant annet det som kalles "Avvik", som er avviket mellom ønsket fart og faktisk fart. Hvis avviket ikke er mellom ønsket intervall, så går programmet over i tilstandene Aks_cruise eller deaks_cruise som øker eller reduserer farten til ønsket fart.

Et annet av tilstandsbyttene er IF NOT Tenning THEN State:= Tenning_av, hvis tenningen er av da blir tilstanden satt til Tenning_av. Bilen er da skrudd av i denne tilstanden.

4.4 Tilstandsdiagram

Programmets kompleksitet og funksjoner, som blant annet krever mange multiple veier inn og ut fra flere tilstander, gjorde at det var mest hensiktsmessig å bruke et større avansert tilstandsdiagram vist i Figur 8, med tilstandsbeskrivelse i Tabell 1, til planlegging av programmets oppbygging.



Figur 8: Tilstandsdiagram for programmering av en bil

4.4.1 Beskrivelse av tilstandene

I Tabell 1 er tilstandene listet med programnavnet i Codesys og med en funksjonsbeskrivelse som kan leses til høyre for tilstandsnavnet.

Tabell 1: Beskrivelse av tilstandene

Navn på Tilstanden	Beskrivelse av Tilstanden
Tenning_av	Start tilstanden I programmet og virker som en simulering av at bilen er skrudd av.
Frikjoring	Denne tilstanden er den standard tilstanden en bil normalt vil være i når en er ute og kjører på landeveien eller i oppkjørselen etc. Her er det ingen hjelpemidler utenom en valgfri nødbremsassist, ellers vil dette være normal drift.
Holdfart_cruise	Når sjåføren ønsker at bilen skal hjelpe mer til for eksempel det å holde farten eller køkjøringsassist så kan en velge Cruise Modus. Når den ønskede farten er oppnådd eksempelvis farten til bil foran, fartsgrense eller en egendefinert fart vil programmet hvile seg i denne tilstanden inntil nye kriterium blir innfridd.
Aks_cruise	Når Bilen er i cruise modus så skal bilen holde en bestemt fart enten om det er farten tilpasset bilen foran, fartsgrensen eller en egendefinert fart, når bilen mister fart eller er under ønsket nivå vil den være i denne tilstanden som setter i gang en prosess som akselererer bilen inntil rett fart er oppnådd.
Deaks_cruise	På samme måte som Aks_cruise skal øke farten når den er for lav så skal Deaks_cruise senke den når den er for stor. I dette programmet blir det satt i gang en prosess som senker farten helt til ønsket fart er oppnådd.
Nodbrems_cruise	Om bilen er i cruise modus så er det en innebygget sikkerhets funksjon i programmet som skal nødbremse bilen om det kommer et objekt eller en bil i vegen som har en for liten hastighet overfor bilens hastighet og avstand. Dette vil også sende bilen til Frikjoring tilstanden.
Nodbrems_frikjoring	Om en kjører i standard tilstand er det også mulighet til å ha sikkerhetsfunksjonen Nødbrems aktiv, da må nødbrems assist settes på og den vil fungere på samme måte som Nødbremsen ellers. Det er verdt å merke seg at dette er en selvbestemt funksjon når en er i frikjoring.

4.4.2 Transisjonsliste for tilstandene

Transisjonslisten i Tabell 2 viser hvordan tilstandsdiagrammet fungerer og hvilket kriterium som må oppfylles for at programmet skal endre sin tilstand.

Tabell 2: Transisjonsliste for tilstandene

Linje nr.	**Retning Alternativ 1**	**Kriterium**	**Retning Alternativ 2**	**Kriterium**
1	Tenning_av >> Holdfart_cruise	Tenning = TRUE AND Cruise = TRUE	Holdfart_cruise >> Tenning_av	Tenning = FALSE
2	Tenning_av >> Frikjoring	Tenning = TRUE	Frikjoring >> Tenning_av	Tenning = FALSE
3	Frikjoring >> Holdfart_cruise	Cruise = TRUE	Holdfart_cruise >> Frkjoring	Cruise = FALSE OR Bremsepedal > 0
4	Frikjoring >> Frikjoring_nodbrems	Nodbrems_assist = TRUE AND Nodbrems = TRUE	Frikjoring_nodbrems >> Frikjoring	Nodbrems_assist = FALSE OR Nodbrems = FALSE
5	Holdfart_cruise >> Aks_cruise	Avvik > 0.5	Aks_cruise >> Holdfart_cruise	Avvik < 0.5
6	Holdfart_cruise >> Deaks_cruise	Avvik < -0.5	Deaks_cruise >> Holdfart_cruise	Avvik > -0.5
7	Holdfart_cruise >> Nodbrems_cruise	Nodbrems = TRUE		
8	Aks_cruise >> Frikjoring	Cruise = FALSE OR Bremspedal > 0	Aks_cruise >> Tenning_av	Tenning = FALSE
9	Deaks_cruise >> Frikjoring	Cruise = FALSE OR Bremsepedal > 0	Deaks_cruise >> Tenning_av	Tenning = FALSE

10	Nodbrems_cruise >> Frikjoring	Nodbrems = FALSE	Nodbrems_cruise >> Tenning_av	Tenning = FALSE
11	Aks_cruise >> Nodbrems_cruise	Nodbrems = TRUE		
12	Aks_cruise >> Deaks_cruise	Avvik < -0.5	Deaks_cruise >> Aks_cruise	Avvik > 0.5
13	Deaks_cruise >> Nodbrems_cruise	Nodbrems = TRUE		

4.5 Oppsett

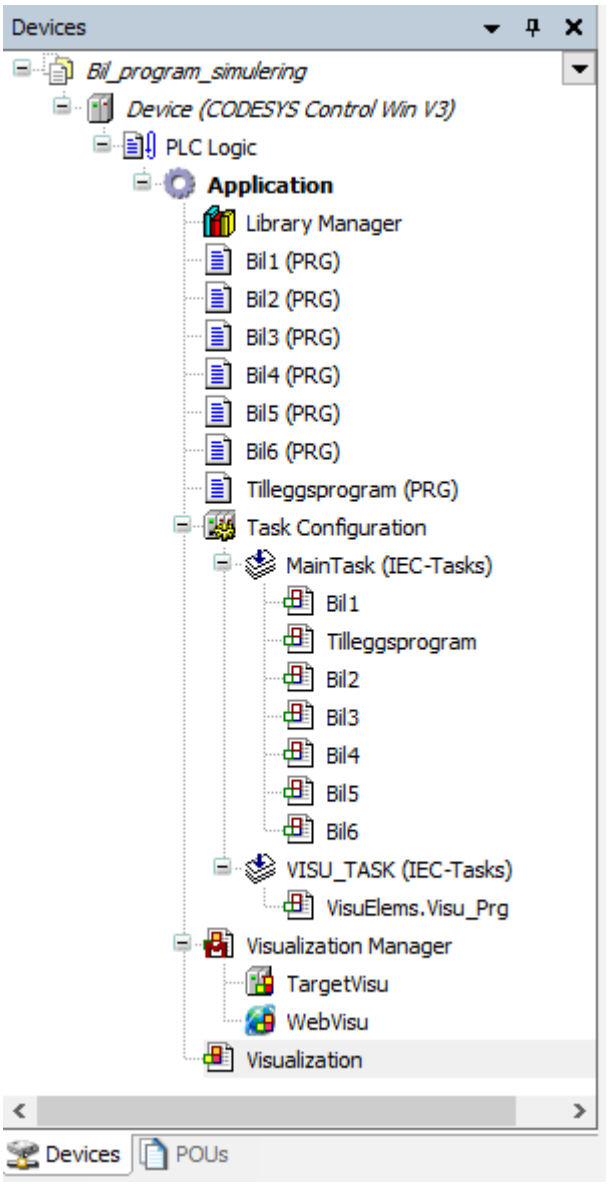
Programmets oppsett er satt i sammen slik at det er flere PLS-programmer, også kalt applikasjoner, i et PLS Device system. Disse programmene er Bil1, Bil2, Bi3, Bil4, Bil5, Bil6 og Tilleggsprogram, som vist i Figur 9, som kan jobbe samtidig uavhengig av hverandre.

Bil1 til Bil6 tilsvarer PLS-programmet som ville vært i hver enkelt bil, og i simuleringen er det meget viktig at disse bilene kan fungere individuelt.

Tilleggsprogram er et hjelpeprogram som skal assistere i simuleringen og gjøre at de seks forskjellige bil-programmene kan samhandle med hverandre og i tillegg så har dette programmet diverse hindringsfunksjoner som tilsvarer forstyrrelser i trafikken.

Applikasjonene kjører på en 50 ms syklisk skannetid, der programmet blir lest igjennom og kan gjøre endringer 20 ganger i sekunder. 50 ms vil da være dødtiden i reaksjonsevnen til de programmerte bilene.

Til slutt så settes dette i sammenheng inn i et visualiseringsprogram som er et verktøy for å få oversikt over simuleringen. I dette programmet så blir det satt inn blant annet speedometere, gass- og bremsepedaler, tenning, cruise modus knapp. i tillegg vil det være knapper som bestemmer fartsgrensen og om det er hindringer i veien. I denne delen av programmet er det viktig at vi kan teste alt samtidig slik at vi får en mest mulig realistisk simulering.



Figur 9: Oversikt av strukturen I Codesys.

4.6 Programmets virkemåte

4.6.1 Bilenes programmerte virkemåte

Programmet inneholder flere variabler som har fått navn som gir en omtrentlig beskrivelse av hva variablene representerer. Med disse variablene bygges det opp en virkemåte for de seks simulerte bilene Bil1 til Bil6 som gjør at det er mulig å simulere dem opp mot hverandre i en simulasjon.

Bilene fungerer ved at de akselererer og retarderer ved en variabel hastighet som kan reguleres med både gasspedal og bremsepedal.

I programkoden står følgende formel for den totale akselerasjonen.

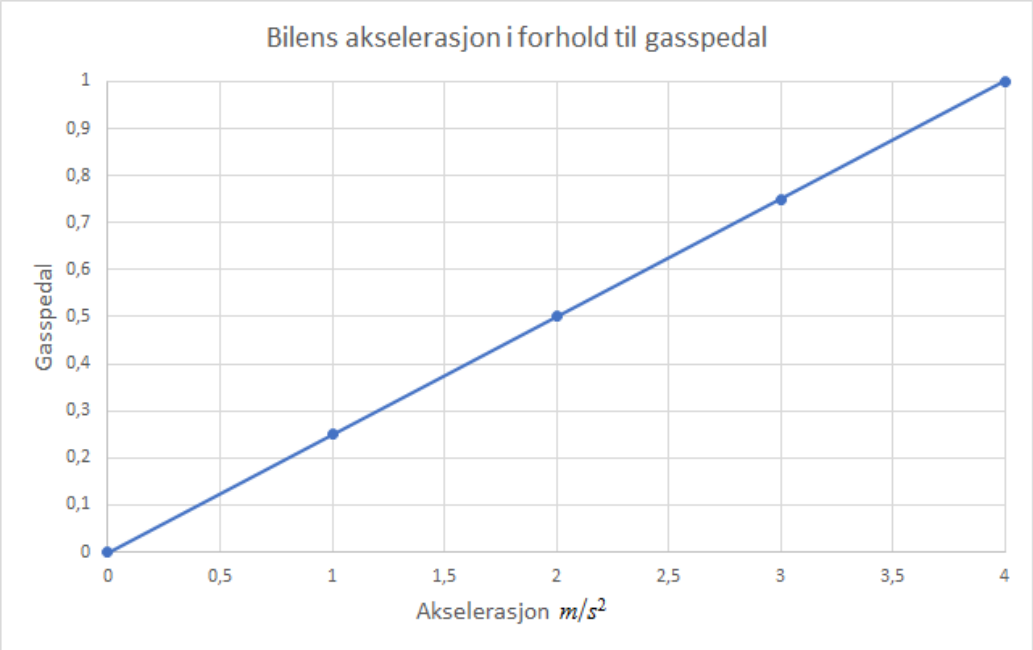
$$\text{Totalakselerasjon} = \text{Aksbil} + \text{retardasjonkonstanter} + \text{Aksbremsing}$$

Hentet fra programkoden i vedlegg D, forenklet uten de leddene for regulering for cruisekontroll. Aksbil er akselerasjonen til bilen, Aksbremsing er retardasjonen under bremsing, Retardasjonskonstanter består av retardasjonen forårsaket av luftmotstand og rullemotstand som fungerer på bilen som får bilen til å sakke ned.

Aksbil er akselerasjonen til bilen som reguleres med gasspedalen. Hentet fra programkoden i vedlegg D.

$$\text{Aksbil} = \text{Motorakselerasjon} * \text{Gasspedal}$$

Motorakselerasjon er en konstant $4\ \mathrm{m/s^2}$og Gasspedal er et variabelt tall mellom 0 og 1 avhengig hvor mye gasspedalen er trykket inn, der 0 er ingen berøring av gasspedal og 1 er fullt inntrykket gasspedal. Dette er et lineært forhold som vist i Figur 10.



Figur 10: Viser bilens akselerasjon i forhold til hvor mye gasspedalen er trykket inn.

Aksbremsing er retardasjon ved bremsing som fungerer på hjulene. Disse kreftene er også avhengig av hvor mye bremsepedalen er trykket inn. Negative krefter som gjør at bilen retarderer.

$$\text{Aksbremsing} = \text{Bremseakselerasjon} * \text{Bremsepedal}$$

Bremseakselerasjon er en konstant $-8\ \mathrm{m/s^2}$ som blir regulert av bremsepedal. Dette er også et lineært forhold.

Luftmotstand resulterer i en konstant $-1\ \mathrm{m/s^2}$ retardasjon av bilen.

Luftfriksjon er en negativ kraft som oppstår når luft glir langs overflaten av bilen, denne er ikke betydelig. Derimot er luftturbulens mye mer betydelig, på grunn av at luften bak bilen blir turbulent, altså at luftstrømmene blir urolige og det skapes trykkforskjeller (13). Biler blir sakket ned når de rører luften for mye, på grunn av virvler og bølger som stjeler energi fra bilen. Kombinasjonen av luftfriksjon og turbulens er kjent som luftmotstand.

Bilfabrikanter har som mål å redusere luftmotstand ved å lage bilene så aerodynamiske som mulig, altså en form som rører opp luften minst mulig, for å blant annet redusere bilens forbruk.

Rullemotstand utgjør en konstant $-0{,}2\ \mathrm{m/s^2}$. Består av kortvarige, elastiske deformasjoner av hjul og underlag i berøringsflater (14) som skaper negative krefter som jobber imot motoren. Kun en liten del av hjulet har kontakt med bakken.

4.6.2 Fartsholders virkemåte og prioriteringer

Når Bilen settes i Cruise modus vil bilen automatisk sette en egendefinert holde-fart, altså en hastighet som bilen vil automatisk justere seg etter. For å oppnå dette må bilen ta hensyn til luft- og rullemotstand samtidig som at det er programmert inn en forsterkningsverdi på hvor fort bilen skal akselerere som avhenger av avviket mellom ønsket hastighet og faktisk hastighet.

Forsterkningen her blir kalt Kp, dette er en verdi som forsterker pådraget med en gitt verdi for at bilen skal oppnå den ønskede hastigheten. Denne Kp-verdien er en konstant i dette programkoden, men kan endres til en variabel for å få forskjellige akselerasjonskurver.

$$\mathrm{u(t)} = \mathrm{u_0} + \mathrm{Kp\ e(t)}$$

$\mathrm{u(t)}$ blir det effektive pådraget, u_0 er det pådrag som trengs for å holde hastigheten, Kp er forsterkningen og $\mathrm{e(t)}$ er avviket.

I tilfellene der hastigheten er over den ønskede verdien vil bilen automatisk senke farten, og her er det også en tilsvarende verdi som forsterker bremsepådraget kalt Kp-brems. Det effektive pådraget vil endre på seg etter hvilket avvik bilen har, om det er langt ifra vil pådraget være høyere og vil gradvis synke jo nærere bilen kommer den ønskede hastigheten og at avviket er lik 0.

I cruise modus kan bilen også rette seg etter andre ønskede hastigheter, om fartsgrenseassisten er på vil bilen ikke overstige fartsgrensen. Bilen kan også innrette seg etter trafikken og bilen foran og vil da holde seg bak bilen automatisk, men for at dette skal fungere sammen må det være et system som avgjør hvor nærme bilen er sin ønskede hastighet og i tillegg så må den kunne prioritere den rette verdien.

For å bestemme hvor langt bilen er fra målet sitt for å oppnå sin hastighet benyttes et avvik. Avviket er forskjellig etter hvilken hastighet som bilen skal prioritere:

- Egendefinert hastighet er en hastighet som sjåføren kan sette etter eget ønske, denne hastigheten kan endres når som helst og justeres. Dette blir kalt Regavvik i programmet, kort for Reguleringsavvik
- Hastighet etter Fartsgrense om fartsgrenseassisten er på, da vil bilen aldri overstige fartsgrensen. Dette blir kalt Grenseavvik i programmet kort for Fartsgrenseavvik.
- Bil foran sin hastighet vil si at bilen setter sin ønskede hastighet lik bilen foran og at bilen skal legge seg en bestemt avstand bak bilen foran. Dette blir kalt for Avvikforan i programmet som da er kort for Avvik for bil foran.

For at disse avvikene skal fungere i praksis så må det være et prioriteringssystem, disse avhenger da følgelig av at fartsgrenseassisten er på, eller om det er en bil foran og hvilken hastighet bilen foran har og i tillegg skal den egendefinerte hastigheten også tas med. Normalt uten bil foran og ingen fartsgrenseassist så vil bilen følge den egendefinerte hastigheten uavhengig hvilken hastighet dette måtte være i programmet følger bilen da Regavvik.

Når det kommer en bil foran med lavere hastighet vil Avvikforan slå inn som det prioriterte avviket, men om denne bilen øker farten og forsvinner vil bilen bak øke farten med bilen foran helt til den egendefinerte hastigheten er lavere enn hastigheten til bilen foran. Når dette skjer vil bilen bak prioritere den egendefinerte hastigheten, altså Regavvik.

I en situasjon der bilen som blir kjørt er bak en bil med økende hastighet og har selv en egendefinert hastighet på 100 km/t så vil bilen følge bilen foran så lenge den ikke akselerer for fort og farten på bil foran er under 100 km/t. Men om bilen bak denne gang har fartsgrenseassisten på vil den prioritere fartsgrense om den er lavere enn både egendefinert og bilen foran sin hastighet. I situasjonen der bilen foran fortsetter å øke hastigheten sin over 100 km/t og egendefinert hastighet er 100 km/t, men med en Fartsgrense på 80 km/t vil fartsgrensen bli prioritert og i programmet blir avviket satt til Grenseavvik.

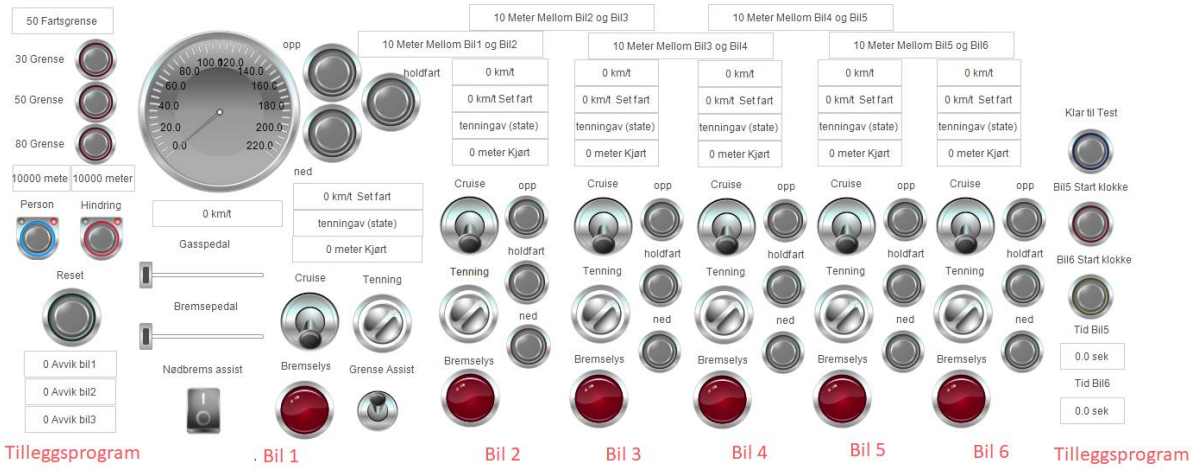
Fartsholderen er programmert til å holde en definert avstand mellom egen bil og bilen foran. Trafikkreglene for avstander mellom biler er ikke konkrete, men vegtrafikkloven § 3 (15) sier at alle har plikt til å vise hensyn og opptre aktsomt i trafikken. Statens vegvesen anbefaler minst tre sekunders mellomrom til bilen foran (16). Blir ofte kalt 3-sekundersregelen. Fartsholderen er da programmert til å holde en avstand mellom 3 og 3,5 sekunders. Hvis bilen foran øker eller senker farten, justerer bilens fartsholder på farten for å opprettholde avstanden.

Ved 50 km/t kjører en bil 13,89 meter per sekund. Fartsholderen vil da holde 3 til 3,5 sekunders mellomrom som utgjør en avstand mellom 41,67 og 46,86 meter. Denne avstanden varierer med hastigheten.

$$\text{Avstand (m)} = \text{Hastighet (m/s)} * \text{tid (s)}$$

4.7 Visualisering

For å få en ordnet oversikt over testingen og simuleringen er det nødvendig med en form for visualisering, og i Codesys brukes Visualization som er en tilleggsfunksjon som gjør dette mulig. Her har vi et utdrag, Figur 11, fra prosjektets visualiseringsprogram.



Figur 11: Oversikt av visualiseringen med seks biler og tilleggsprogrammets funksjoner. Større versjon i vedlegg A.

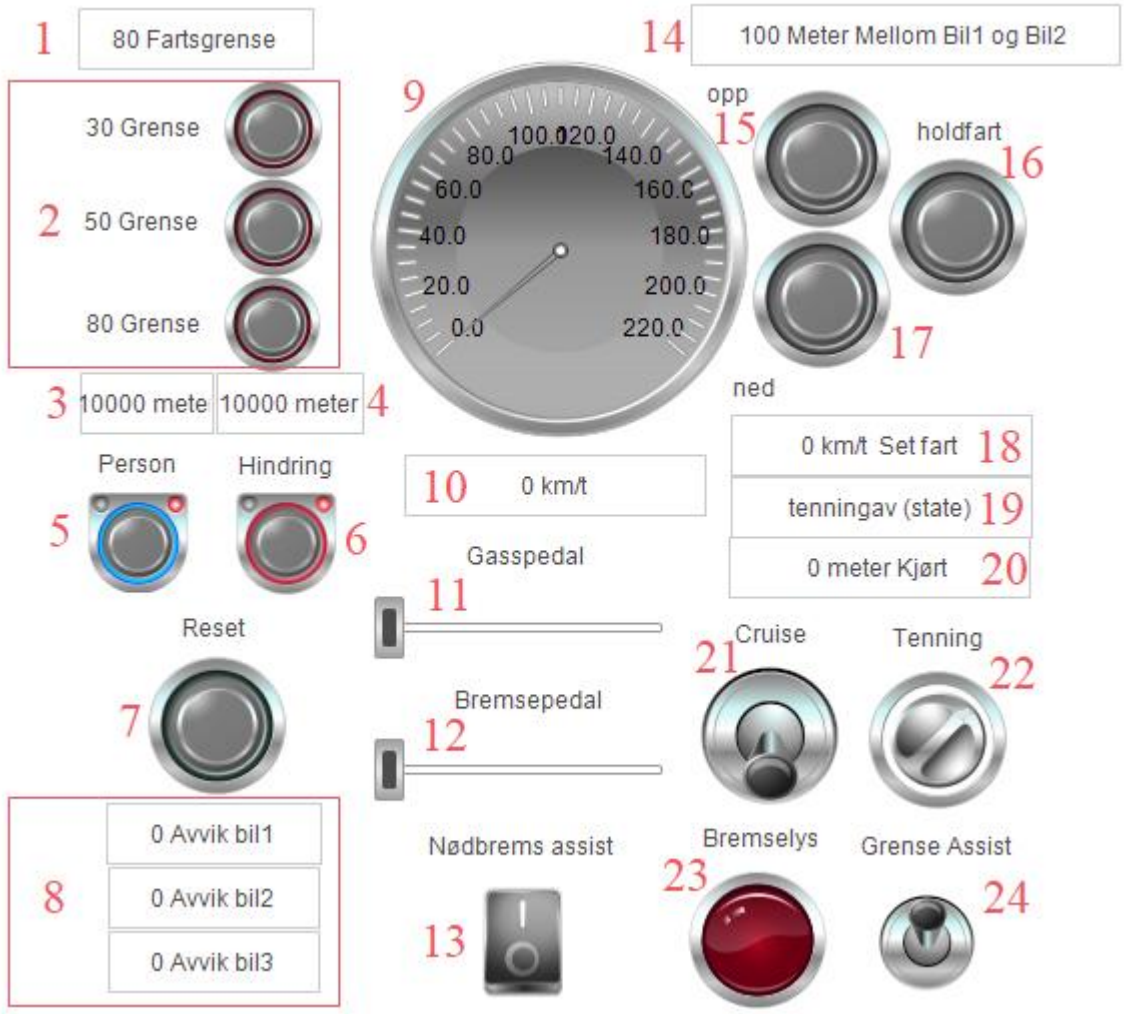
Visualiseringen består av applikasjonene til seks biler samt tilleggsprogrammet. Bil1 er vist med alle programmerte funksjoner som blant annet gasspedal og bremsepedal for kjøring uten fartsholder.

De andre fem bilene, Bil2 til og med Bil6, er kun vist med de nødvendige funksjonene for å sette de i fartsholder-modus for bruk i simuleringen. De er programmert med de samme funksjonene som Bil1, men blant annet gasspedal og bremsepedal blir ikke brukt i simuleringen og er dermed ikke med i visualiseringen.

Tilleggsprogrammet består av funksjonene for å sette hindringer i veibanen, en stoppet bil eller en person, endring av fartsgrenser, reset knapp, testknapp og stoppeklokke for Bil5 og Bil6 for bruk i simuleringen.

4.7.1 Elementoversikt for visualisering.

Figur 12 er en utklippet versjon av visualiseringen som viser nummererte elementer som er beskrevet i Tabell 3. Brytere og tekstblokker fra applikasjonene Tilleggsprogram og Bil1 er her vist.



Figur 12: Elementliste for visualisering, Bil1 og Tilleggsprogram

I visualiseringen har Bil1 alle funksjoner, som blant annet gasspedal og bremsepedal, mens Bil2 til og med Bil6 har kun de mest nødvendige funksjonene for å sette bilene i fartsholder-modus.

Oppbyggelsen av de knappene og tekstblokkene som blir brukt på alle de seks bilene er identiske og eneste forskjell mellom de tilkoblede variablene er oppkallet. For eksempel variabelen Bil1.Bremselys brukt i applikasjonen Bil1, er kalt Bil2.Bremselys i applikasjonen Bil2 og Bil3.Bremselys i applikasjonen Bil3.

Under er Figur 13 som er et utklipp som viser de nummererte elementene til Bil2, disse er lik for Bil3, Bil4, Bil5 og Bil6. Elementene blir beskrevet i Tabell 3 på neste side.

Figur 13: Elementoversikt for visualisering, Bil2.

4.7.2 Elementliste for visualisering.

Tabell 3: Elementliste for nummererte elementer i Figur 12 og Figur 13.

Nummer	Funksjonsbeskrivelse	Tilkoblet variabel
1	Fartsgrensen	Tilleggsprogram.Fartsgrense
2	Knapper for å velge de forskjellige fartsgrensene 30, 50 og 80	Tilleggsprogram.Grense30, Tilleggsprogram.Grense50, Tilleggsprogram.Grense80
3	Viser antall meter mellom bil 1 og eventuell person i veibanen.	Tilleggsprogram.Meterpersonogbil1
4	Viser antall meter mellom bil 1 og eventuell stoppet bil som hindrer veibanen	Tilleggsprogram.Meterhindringbilogbil 1
5	Knapp som setter en person 200 meter foran bil 1 i veibanen for å teste nødbrems og nedbremsing	Tilleggsprogram.Personivei
6	Knapp som setter en stoppet bil i veibanen foran bil 1 for å teste nødbrems og nedbremsing	Tilleggsprogram.Hindringbil
7	Reset knapp for å stoppe alt og sette tilbake i initialbetingelsene	Tilleggsprogram.Reset
8	Viser avviket mellom faktisk og ønsket fart på bil1, bil2 og bil 3	Bil1.Avvik, Bil2.Avvik, Bil3.Avvik
9	Speedometer som viser bilens	Bil1.Fart_bil
10	Bilens fart i vist i tall	Bil1.Fart_bil
11	Gasspedal som kontrollerer bilens akselerasjon	Bil1.Gasspedal
12	Bremsepedal som kontrollerer bilens retardasjon	Bil1.Bremsepedal

13	Knapp for å skru nødbremseassistanse av og på	Bil1.Nodbremse_assist
14	Viser meteravstand mellom bil 1 og bil 2	Tilleggsprogram.Meter1og2
15	Justerer sett-farten i Fartsholder opp med 5 km/t	Bil1.Cruise_fartopp
16	Setter sett-farten lik bilens fart og aktiverer Fartsholder funksjonen	Bil1.Cruise_farthold
17	Justerer sett-farten i Fartsholder ned med 5 km/t	Bil1.Cruise_fartned
18	Viser sett-farten til Fartsholder-funksjonen	Bil1.Set_fart
19	Viser hvilken tilstand bilens PLS-program er i	Bil1.Bil_tilstand
20	Viser hvor lang bil 1 har kjørt siden start av programmet eller det har blitt resatt.	Bil1.Posisjon
21	Aktiverer fartsholder funksjonen og setter også bilen fart lik sett-farten. Kan brukes til å skru av Fartsholder funksjonen.	Bil1.Cruise
22	Tenningsbryter for å slå bilen av og på.	Bil1.Tenning
23	Bremselys for bilen.	Bil1.Bremselys
24	Av og på knapp for bilens fartsgrense assist, hjelper å holde fartsgrensen	Bil1.Fartsgrense_assist

5 Observert trafikk

5.1 Metode for observering

For å få resultater ut av observasjonen trengs en metode som beskriver hvordan en skal få data fra selve observeringen. Først må lokasjon bestemmes, som i dette tilfellet er i Karmsundgaten hvor det er forventet mest trafikk. Tidspunktet må også bestemmes som blir rundt 15:00 - 16:00 da det er det tidspunktet med mest aktivitet.

Målingene vil finne sted på forskjellige steder der det vil være forskjellige forhold og hendelsesforløp. Det vil være målinger med lyskryss der bilene vil stå på rekke, men alle vil ha en mulighet til å se når det er grønt lys og vil derfor ha en ekstra mulighet til å være klar. Det andre scenarioet er med en fotgjengerovergang der det vil være stopp på grunn av en person i veibanen som gjør det vanskelig for bilene bak å se når det er klart for å kjøre. Ved både lyskrysset og fotgjengerovergangen er det minimalt med sideveier slik at forstyrrelser ikke oppstår og fartsgrensen er 50 km/t.

For å få data må resultater fremstilles og her er det bestemt at resultatene vil komme i form av en målt tid med stoppeklokke. Tiden vil måles fra det fremste kjøretøyet beveger seg til et gitt antall kjøretøy bak begynner å kjøre. Alle målingene vil bli notert sammen med forhold, antall biler osv.

5.2 Generelle observasjoner av kjøremønstre

Bilistene har forskjellige kjørestiler og vanemønster som gir store variasjoner av resultat i trafikken. Noen holder lang avstand for bilen foran, andre holder kort avstand. Noen kjører langsomt opp til bilen foran og lager ofte store avstander til bilene foran uavhengig om de står stille eller er i bevegelse. Ved gangfelt så stopper noen biler opp i god tid før en person går over, men noen bilister viser lite hensyn og kjører selv om de kan se at noen vil over gangfeltet.

Bilistene har også forskjellige tilnærminger når det kommer til bevegelse foran eller i kryss og hindringer. Ved overgangsfelt kan det observeres at bilister kan renne inn mot overgangsfeltet og akselerere igjen når det er klar bane. En annen fremgangsmåte i samme situasjon er at bilisten kjører helt opp til overgangsfeltet og står stille der til personen har gått over. Sistnevnte situasjon er den vanligste fremgangsmåten når det kommer til vegkryss og lyskryss, men her er det også tilfeller av at bilene renner inn mot krysset.

Observasjoner gjort ved lyskrysset viser at bilene som står der vil fylle veldig raskt på med høy akselerasjon, til og med når det er kø på andre siden av lyskrysset. Om køen på andre siden er for lang vil bilene i lyskrysset slakke ned før de er ute av lyskrysset, men da så nærme utgangen av krysset som mulig. Selv om det er observert dødtid på nesten hver av bilistene så er det noen som stikker seg ut med en mye lengre reaksjonstid enn andre. I noen tilfeller kan det observeres bilister som glemmer seg ut og står 2 - 3 ganger lenger enn det som er gjennomsnittet.

Det å gjenta den samme målingen flere ganger er vanskelig å få til. Målingene vil alltid variere fra hver gang det er rødt lys eller hver gang en person går over vegen, men målingene blir lagt i sammen slik at kun de målingene som er mest relevante vil ha betydning. For å få de beste og de relativt mest realistiske og sammenlignbare resultatene så ekskluderer vi de målingene som er lengst fra normalen.

5.3 Resultater fra observasjon

I observasjonene gjort ved lyskrysset i Karmsundgata og overgangsfeltet ved meieribygget i Karmsundgata ble det gjort målinger som begge gav resultater som kan sammenlignes med programmet. Bilene ble målt fra trafikklyset skiftet til grønt, frem til 5 - 7 biler hadde akselerert og nådd lyskrysset. I fotgjengerovergangen ble det målt fra bevegelsen i første bil etter fotgjengeren hadde passert, helt frem til 4 - 7 biler hadde nådd frem til fotgjengerfeltet der første bil startet fra.

I programmet er det kun tatt høyde for opptil seks biler så det er bare opp til seks biler som blir medregnet.

5.3.1 Trafikklys

Målingene med trafikklyset viser en økt tidsbruk basert på 20 målinger for å komme seg gjennom krysset desto flere biler som står i kø før grønt lys (kjør). Sjansen for at en eller flere biler glemmer seg ut eller av andre grunner bruker lenger tid enn det som er vanlig, øker også per ekstra bil som står i kø. For hver ekstra bil som står i køen, brukes det i gjennomsnitt 0.1 sekunder mer tid per bil for å komme seg gjennom lyskrysset.

Trafikklyset er rødt i 20 sekunder om gangen som fyller opp en kø med stillestående biler foran lyskrysset som kan sees i Figur 14.

Gjennomsnittslengden fra den sjette bilen og frem til lyskrysset er målt til 35 meter. I Tabell 4 kan gjennomsnittstiden sees fra målingene med fem biler der standardavviket er 2,31 sekunder og seks biler der standardavviket er på 3,56 sekunder i krysset.

Tabell 4: Målinger ved Trafikklys. Full data i vedlegg B.

Antall biler	Bil fem i rekken	Bil seks i rekken
Gjennomsnittstid	13 ± 2 s	17 ± 4 s

Figur 14: Lyskryss, Karmsundgata, Haugesund. Foto: Mats Kallevig

5.3.2 Fotgjengerovergang

Likt som i målingene gjort ved lyskrysset, viser også målingene med gangfeltet at tiden hver bil bruker på å komme seg forbi gangfeltet, økes med 0,1 sekunder for hver ekstra bil.

Ved gangfeltet varierer avstanden mellom stoppede biler mer enn ved lyskrysset, grunnet kortere tid bilistene må stoppe for en fotgjenger som går over. Tiden bilister stopper for en fotgjenger varierte mellom 5 og 8 sekunder som i Figur 15.

De bakerste bilene i rekken, holder ofte store avstander og nærmet bilen foran seg sakte og stoppet ikke opp som observert i lyskrysset.

Gjennomsnittstiden fem og seks biler bruker på å akselerere og nå gangfeltet, kan ses i Tabell 5. der fem biler har et standardavvik på 1,69 sekunder og seks biler har avviket på 2,09 sekunder

Tabell 5: Målinger ved Trafikklys. Full data i vedlegg B.

Antall biler	Bil fem i rekken	Bil seks i rekken
Gjennomsnittstid	12 s $\pm$ 2 s	15 s $\pm$ 2 s

Figur 15: Fotgjengerovergang ved Meieriet i Haugesund. Foto: Espen Andreas Vik

6 Simulering

6.1 Målet med simuleringen

Målet med simuleringen er å sammenligne observert trafikk i bestemte situasjoner opp mot tilsvarende simulerte situasjoner hvor selvkjørende funksjoner fungerer på bilene. Målet blir da at den simulerte trafikken skal være mer effektiv på grunn av den automatiserte kjøre funksjonene.

6.2 Fremgangsmåte

Det første som må på plass etter at grunnlinjen av programmet er laget, er tidsverdiene fra den observerte trafikken. Med disse verdiene er det klart hvor forsøksområdet ligger og programmet kan tilpasses slik at målingene fra simuleringen kan sammenlignes med de observerte verdiene.

I de observerte målingene er det mellom fem og seks biler som er tatt i bruk så da ble det laget tilstrekkelig med biler i programmet som i dette tilfellet er seks biler. Så må bilene settes opp slik som de observerte bilene, og da må programmet programmeres slik at bilene står i kø og venter på en form for gå signal.

Når de simulerte bilene står klar etter hverandre må avstanden mellom bilene beregnes og det må klargjøres et tidsur som måler tiden fra første bil kjører til bil nummer fem, og et annet tidsur som måler fra første bil til bil nummer seks. Når dette er klart kan simuleringen starte og repeteres et gitt antall ganger slik at målingene kan sammenlignes.

Når forsøkene er gjennomført på forutbestemt metode og godkjent av gruppen kan resultatet analyseres. Med de resultatene fra forsøket klargjort kan de sammenlignes med resultatene fra observert trafikk.

Bilene i simuleringen er 4,5 meter lange og det er 2,5 meter mellom hver av bilene som står i kø. Total lengde fra fronten til første bil til fronten av den sjette bilen er målet til å være gjennomsnittlig 35 meter. Disse verdiene er hentet ut fra observasjon av trafikken der en gjennomsnittslengde av bilene, gjennomsnittsavstand mellom bilene ble estimert og total lengde ble målt.

Femte bil måtte kjøre 28 meter og sjette bil må kjøre 35 meter før stoppeklokken tar tiden på hver bil. Bilene holder en variabel avstand mellom seg som er mellom 3 og 3,5 sekunder multiplisert med hastigheten.

6.3 Resultat

Resultatene fra simulasjonen er basert på gjentatte forsøk det er tatt et gjennomsnitt av der tiden starter fra første bil i rekken starter å kjøre til femte bil i rekken kommer forbi den første bilens posisjon, og det samme gjelder sjette bil i rekken. I Tabell 6 brukes 3-sekundersregelen med 3 - 3,5 sekunders tid mellomrom. Det er to målinger med forskjellig akselerasjon på første bil.

Et sett av målingene er med første bil i fartsholder-modus med 50 km/t som ønsket fart. Det andre settet er med konstant 2 m/s^2 akselerasjon opp til 50 km/t på første bil. Siden akselerasjonen med fartsholderen på første bil er raskere enn det som er observert, så er det gjort et ekstra sett med målinger der akselerasjonen er mer lik den akselerasjonen sjåfører har på veiene i dag.

Tabell 6: Resultat fra simulering med standard 3-sekundersregelen. Full data i vedlegg C.

Akselerasjon første bil	Bil fem i rekken	Bil seks i rekken
Fartsholder	9,96 $\pm$ 0,02 s	12,575 $\pm$ 0,03 s
2 m/s^2 akselerasjon	10,96 $\pm$ 0,02 s	13,58 $\pm$ 0,02 s

Ved lyskrysset i Karmsundgata ble det observert at bilene holdt mye kortere avstand til bilene foran under akselerasjon. De brukte ofte lang tid på å reagere, men etter de hadde reagert og akselerert, så holdte de kort avstand til bilen foran på vei gjennom krysset. Simulasjonen ble gjort på nytt med en 2-sekundersregel, dette vises i Tabell 7, med 2 - 2,5 sekunders tids mellomrom som er nærmere den observerte avstanden.

Her er det også gjort to sett med målinger, et sett der fartsholderen akselererer den første bilen automatisk og et annet sett med en konstant 2 m/s^2 akselerasjon.

Tabell 7: Resultat fra simulering med standard 3-sekundersregelen. Full data i vedlegg C.

Akselerasjon første bil	Bil fem i rekken	Bil seks i rekken
Fartsholder	8,00 $\pm$ 0,09 s	9,85 $\pm$ 0,09 s
2 m/s^2 akselerasjon	8,80 $\pm$ 0,00 s	10,68 $\pm$ 0,02 s

Resultatene viser at ved en redusert avstand mellom bilene under akselerasjon, så reduseres tiden bilene bruker gjennom lyskryss og fotgjengeroverganger. Med hurtigere akselerasjon på første bil, så beveger også køen seg hurtigere fra full stopp.

7 Diskusjon

7.1 Behov

Behovet for kjøretøy med autonome kjøreegenskaper kommer til å være økende i en lang årrekke fremover grunnet populasjonsvekst på kloden, men også en økende rikdom og kjøpekraft i flere av landene som har tidligere hatt en ustabil og vanskelig økonomisk situasjon i mange tiår (17).

Denne utviklingen ser vi først og fremst i Asia og da spesielt i Kina, India og deler av Sørøst-Asia, men også store deler av Sør- og Mellom-Amerika har en vekst i økonomi og verdiskaping som gir en bedre situasjon for populasjonen i disse regionene. Flere land har en økende teknologisk fremgang og med dette kommer en raskt økende vekst i økonomien, og når flere land får en stabil og fremgangsrik økonomi vil også deres innbyggere kunne kjøpe og bruke flere kjøretøy (18) som resulterer i en økt trafikkmengde (17).

For å få ned tidsbruken og effektivisere veiene samtidig som et økende miljøfokus gjør at en må få ned forurensning og drivstofforbruk med autonome kjørefunksjoner. Først i de vestlige og asiatiske landene (Kina, Japan og India) (19) som allerede har en stor bilpark, men med mulighet for å ta i bruk nyere og mer avanserte biler. Med dette vil de kunne øke effektiviteten i trafikken som da senker tiden en bruker i kø, men også minske drivstofforbruket vesentlig. Spesielt i store land med en dyr strømpris, uren strømproduksjon og der elektrifiseringen av bilparken er for kostbar, vil det være mer effektivt å få ned forbruket av fossile drivstoff.

I store deler av de vestlige landene er dette allerede oppnåelig, med en stadig økende kjøpekraft i Europa og Nord-Amerika (17) så vil behovet her stige enda mer og vil derfor ha en viktig effekt her også. Selv om landene i Europa spesielt har et økt fokus mot utslippsfrie bilparker (20), vil fortsatt antall biler øke og vil derfor være et enormt behov for å effektivisere trafikken. Siden debatten om mer fritid er blitt et større tema i Norge og de andre vesteuropeiske landene (21) så kan en effektivisert bilpark være en om ikke den fremste løsningen for å frigi noen ekstra minutter i døgnet. Fakta hentet fra Economic Growth, Our World In Data. (17).

7.2 Fremtiden

Om det sees på fremtiden i et lengre perspektiv ser vi at den økte trafikken kommer til å være et økende om ikke i like stor grad som i de siste 20 til 30 årene, men fortsatt stabilt økende. Selv med hjelpefunksjoner som effektiviserer og som hjelper til i trafikken vil det fortsatt være trafikk og en del kø kjøring. For å få full uttelling på de autonome kjøre systemene så vil en del av de store bilprodusentene og aktørene som Nissan, Uber og Google (22) ta i bruk helt selvkjørende biler en gang i fremtiden. Med dette forventer disse aktørene en mer fremkomst effektiv og miljøvennlig endring i bilparken.

7.2.1 Miljø

Med gode løsninger for å effektivisere trafikk og køkjøring så vil dette også bety mindre tid på veiene for hver enkelt bil, og en mer lineær kjøring vil effektivisere drivstofforbruket. Med dette kommer det en redusert forurensing og utslipp fra verdens bilpark som er positivt overfor miljø og lokal luftkvalitet (22).

Med tiden vil bilparken også ha et bedre drivstoff forbruk som gjør bilen mer klimavennlige, og at produksjonen av materialer som stål (23) og aluminium (24) vil bli renere og generelt bilfabrikkene tar nye steg mot bærekraftig produksjon. Nye drivstoff alternativ som elektriske eller hydrogenbiler gjør at utslipp synker og dette nyter klimaet godt av (25).

7.2.2 Mulig økning i Trafikk

Med en effektivisert trafikk der tiden i kø synker, og det er raskere og rimeligere å ta bilen til sin destinasjon er det mulig at denne vinningen ikke får maksimal effekt. Når køene forsvinner blir det for mange mer fristende å ta i bruk bilen eller gå til anskaffelse av en bil enn å bruke andre transportmidler som buss, tog, sykkel etc. Da er det mulig at køene vil få like mye økning etter hvert som den blir effektivisert og vinningen vil gå opp i spinningen (22).

USA har flere titalls år med statistikker som viser at hvis en vei blir utvidet med flere kjørefelt, så blir ikke trafikken forbedret. Ofte beveger trafikken seg senere enn før utvidelsen av motorveien (26). Det er mulig å dra paralleller fra utvidelse av en vei til effektivisering av trafikk grunnet det er mulig at flere vil benytte seg av bil enn andre alternativer.

Dette kan også bety et større miljøutslipp som ikke er den retningen som er ønsket for dagens bilpark. Med tanke på de økte utslippene fra bilene generelt og en økt produksjon av biler som betyr at en må øke utslippene sine for å ta unna en økt etterspørsel gjør dette til en utfordring. Denne ulempen er fortsatt hypotetisk og det er vanskelig å si om dette noen gang kommer til å skje, men er absolutt en aktuell problemstilling en må sette seg inn i til fremtiden.

7.2.3 Forekomst av Autonome biler

I flere år har biler med nivå 2 av autonome biler blitt satt ut i markedet og er derfor mange biler i dag som har slike funksjoner. Tilbake i 2015 ga Tesla ut en oppdatering til sine den gang 60 000 førere som ga dem mulighet til å ta i bruk en nivå 2 funksjon. I dag er det vanlig at nye biler blir utgitt med en slik funksjon, og både Google, Ford og Uber påstår de vil nå et nivå av selvkjørende biler allerede neste år (27).

EUs forskningsråd for transport har estimert at slike biler blir trolig ikke å se på europeiske veier for kommersiell bruk før tidligst i perioden 2026 - 2030 (28). Dette fordi det kreves et utvidet regelverk og kontrollorgan for at slike kjøretøy skal kunne ta i bruk veiene. Teknologien må også være god nok og dette ser ikke EUs forskningsråd for seg er mulig innen den aktuelle tidsperioden. Man ser for eksempel at dagens autonome systemer fungerer kun under gode værforhold og sliter i dårlig vær (29).

7.3 Sikkerhet

7.3.1 Redusere antall ulykker

I 2016 var den årlige dødsraten i trafikken i USA på over 37 000 personer og det ble estimert at 94 % av disse ulykkene skjedde på grunn av menneskelig svikt og uoppmerksomhet. Her har de autonome kjøretøyene avhengig av fremtidige karakteristikker og funksjoner, en mulighet til å redusere alle totale krasj og ulykker med opptil 90 %.

Med slike forbedringer på ulykkesstatistikken kan en regne seg frem til at bare i USA alene vil kunne spare 190 milliarder US dollar i året. I tillegg kan en forvente lavere kostnader når det kommer til helsesektoren, forsikringskostnader og en kan også forvente at bilene kan kreve mindre sikkerhetsutstyr som igjen gjør bilene billigere.

Et problem med disse fordelene er at automatiserte biler fungerer best når det er et større antall av disse kjøretøyene på veien. Autonome biler kan bare fjerne det menneskelige elementet fra ulykker når det ikke er menneskelig sjåfører på veien. Når det gjelder trafikken, så fungerer autonome biler også best hvis de fleste bilene på veien er førerløse (30).

Helt til autonome biler blir normalen, så er hovedfordelen med å bruke en autonom bil kun for å gjøre kjøreturen lettere og mer behagelig for sjåføren og en liten økning av sikkerheten.

7.3.2 Trusler knyttet til nettverket

Det er i dag vanlig at moderne kjøretøy har nettverksfunksjoner for GPS, trafikkvarsler, musikk osv. Man vil se at disse nettverksfunksjonene blir utvidet fremover når det kommer mer avanserte selvkjørende biler som kommer til å ta i bruk kommunikasjonssystemer for å øke effektivitet og sikkerhet på veiene. Men dette kan ta med seg et økt trusselbilde overfor dataangrep som kan bli en utfordring i fremtiden og komme til å skape et behov for større beskyttelse mot slike angrep på individnivå.

Med økt forbruk og etterspørsel for datatrafikk så er det nødvendig med utbygging av nettverk for å øke nettverksrekkevidde og belastningskapasitet. Ved overbelastet nettverk og manglende rekkevidde, kan det skapes trøbbel for biler som kommer til å ha en avhengighet av kommunikasjonssystemer og vil også, selv med et oppgradert nettverk være sårbare for nettverks sammenbrudd. Med flere barrierer rundt dette kan en forhindre større ulykker med nødssituasjoner og hackerangrep, men det må være en økt bevissthet rundt slike temaer om en ikke ønsker store ulykker som kan ha med seg tragiske konsekvenser (29).

7.3.3 Etiske dilemma knyttet til programmeringen

Både internasjonalt og nasjonalt eksisterer det i dag lover som sier at kjøretøy skal ha en fører og at førere skal ha kontroll over kjøretøyet. Flere store aktører har tatt til orde for at datamaskiner skal kunne regnes som sjåfør som skaper et dilemma om hvem som kan regnes som ansvarlig i ulykker. I dag er det noen få steder det er lov med fullt autonome kjøretøy, men flere land aksepterer slike kjøretøy i kontrollerte forhold, eller når det gjelder forhåndsavtalte tester og forsøk.

Hver enkelt fører i dag må ta valg i løpet av millisekunder hele tiden, og i noen tilfeller har disse valgene store konsekvenser. Når en fører er i en farlig situasjon der det er flere alternativ

med tragiske følger f.eks. motkommende kjøretøy der en kan svinge mot fjellvegg eller gangfelt som alternativ. Så tar mennesker disse valgene i dag, og ofte ligger innebygget etisk og moralsk tankegods til grunn basert på hver enkelt sjåførs livssituasjon og erfaring.

Men om en datamaskin skal ta disse valgene, hvordan programmerer man egentlig for ulykker? Hvem skal bilen ta til grunn for å være mer verdt enn andre. Skal det baseres på hvor mange som er i hver enkelt bil og hvor mange som går på fortauet, eller skal dette allerede være forutbestemt at alltid skal bilen ofres eller skal bilen alltid kjøre av veien for å ha høyest mulig sjanse for å redde passasjerene. Det å programmere inn disse alternativene kommer til å skape hodebry for hver produsent, om ikke dette blir politisk bestemt. Da kan livet til personer involvert i ulykker være avhengig av hva politikerne har bestemt, hva produsenten har bestemt eller kanskje bare ren tilfeldighet (29).

7.4 Effektivitet

Effektiviteten av Autonome kjøretøy er ikke enkelt å sette en bestemt grense på, men noen tall kan gis i ulike scenarioer. Det vil for eksempel forekomme helt forskjellige resultat helt etter hva som blir prioritert i løpet av en test, og hvordan sjåføren ønsker at bilen skal oppføre seg.

En kan se for seg at opphopning av trafikken vil redusere drivstofforbruket med opptil 4 % men dette kan gjøre at bilene også kjører lenger. Om en velger et økonomisk kjøresystem kan en forvente en reduksjon på opp mot 20 % i energiforbruk. Om en kjører i rekke med kort avstand til bilen foran vil en kunne spare mellom 3 - 25 % energiforbruk avhengig av bilens karakteristikker.

Om en fokuserer på å øke komfort fremfor bilens ytelser som rask akselerasjon kan en spare drivstofforbruk med 5 - 23 %. Men om fokuset er på å øke effektiviteten i trafikken for å spare mest mulig tid så kan en øke fartsgrensene på motorveien, men dette vil da øke drivstofforbruket mellom 7 - 30 %.

Med mulighet for eldre og funksjonshemmede til å ta i bruk kjøretøyene kan det få en økning på 2 - 10 % av drivstofforbruk. Det er høy sannsynlighet for at flere aktører ønsker å ta i bruk en bildeling eller en form for taxi funksjon på flere biler som da kan senke energiforbruket med opptil 20 %.

Disse tallene er ikke mulig å fastsette helt presist, men fungerer som estimater for det som kan forventes i fremtiden med slik teknologi. Settes dette i sammenheng med alle nummer og tenker seg ut de forskjellige utfallene kan dette bety en reduksjon av totale energiforbruk i trafikken med 40 % og de totale drivhusgass utslippene kan bli redusert med 9 %. Men om en ser for seg et pessimistisk scenario vil dette kunne i verste fall øke det totale energiforbruket med 105 %, da dette ikke er gitt en tilsvarende økning av drivhusgasser etter som det er mulig andre former drivstoff blir gjeldende for fremtidens bilpark. Fakta hentet fra Autonomous Vehicles Factsheet, University of Michigan: Center for sustainable systems (30).

7.5 Usikkerhet i forsøk og observerte verdier

I testen kan man få en indikasjon om hvordan en bil vil reagere om den ble tatt i bruk i trafikken, men det er ikke en garanti for at dette er den eksakte måten noe vil fungere om dette ble implementert i trafikken. Siden det er en simulering av virkemåte kan vi ikke si klart hvordan resultatet blir før det blir testet med biler som har de samme funksjonene.

Målingene som er utført i trafikken viser også bare en liten indikasjon om de faktiske tallene. For å få ut et klarere og mer sammenlignbart resultat ville det i praksis kunne betydd opptil 1000, kanskje flere målinger i egnede forhold. Som i praksis betyr at målingene måtte tas hver dag i flere uker om ikke flere måneder bestemte tider på døgnet, og til og med da vil flere av målingene ikke være helt 100 %. Gruppens målinger viser seg nok til å være en mindre indikasjon av hva de faktiske resultatene kan komme til å være.

Mindre forskjeller per måling vil nok gjøre utslag, men i hvor stor grad er vanskelig å stadfeste. De fleste forskjellene per måling har nok mindre betydning, og vil nok til slutt uansett være et produkt av svikt i det menneskelige ledd. Avstand til bil foran er forskjeller som går igjen, men bilene tar raskt igjen den avstanden når det er klarsignal for å kjøre. Hvilke biler som står i køen er også varierende fra mindre smart biler og motorsykler til busser og semitrailere. Målinger der bilene var av en uvanlig sort ble fjernet fra de tellende målingene.

Alle faktorer i situasjonen rundt en måling vil ha effekt på det endelige resultat, som vær, kjøretøy, uoppmerksomhet, forstyrrelser i og rundt veibanen eller annet. Dette viser nok mer fordelen med et autonomt system, men det er ikke alltid like enkelt å trekke en konklusjon ut ifra målinger der hendelsesforløpet er annerledes gang for gang.

7.6 Tilpasning av programmet for virkelighetens biler og videre utvikling

Et tilpasset program må inneholde flere forskjellige variabler tilknyttet det ekstra utstyret som må tilpasses for at bilen skal kunne opprettholde de programmerte kriteriene. Innganger og utganger må skrives inn i programmet som er tilknyttet bremsene og gasspedal, og i tillegg så er det viktig at clutch og giring er tilkoblet og programmert slik at dette også kan skje automatisk.

For at bilen skal samarbeide og regulere seg i forhold til trafikken, så må den ha rikelig kommunikasjonsutstyr. Det må derfor legges til innganger og utganger for et bestemt antall sensorer som må til for at bilen skal ha en oversikt over biler i nærheten. For kontakt med flere biler i området må bilen ha GPS og internett-utstyr som hjelper bilen med kommunikasjonen.

Etter som simuleringen kun har en vei å kjøre, som forholder seg kun til akselerasjon i forhold til hindringer kan det tenkes at en rattfunksjon vil være aktuell i et program ment for bruk i trafikken. Siden sikkerheten kan økes med å ha en slik funksjon og det vil være nødvendig for å være innenfor en høyere autonomt nivå.

Akselerasjon og retardasjon må omprogrammeres slik at komfort og drivstofforbruk blir forbedret. En mer avansert beregningsmetode for pådragene til fartsholderen er nødvendig slik at fartsendringene blir mer jevne og naturlige.

Programmet er heller ikke justert etter noen bestemte klimaforhold i simuleringen og må da legge til bestemte variabler knyttet til temperaturmålere, sensorer for regn, værmeldinger over nettet eller manuelt bestemte moduser. Programmet vil da oppføre seg annerledes etter hvilke forhold det er på vegen som regn og våt vegbane, snø og snølagt føre, islagt føre eller om det er varmt og pent vær.

8 Konklusjon

Målet med oppgaven var å skape et program med automatiserte kjørefunksjoner der bilen skal kunne kommunisere med andre biler med samme funksjon for å øke sikkerheten, men også kunne reagere raskere enn sjåføren slik at køkjøring går raskere og er mer effektiv.

Med simuleringene tatt i betraktning og sammenlignet opp mot målte verdier fra observert trafikk kan en påstå at det er mer effektivt å ta i bruk automatiserte kjørefunksjoner i de scenariene som simuleringen er sammenlignet mot. Siden trafikken og køkjøringen går raskere med denne funksjonen er det mulig å påstå at bilen reagerer raskere enn det sjåføren gjør med denne funksjonen implementert slik den er i simuleringen.

Det er programmert inn en funksjon som gjør at bilene i simuleringen holder en sikkerhetsavstand til alle tider basert på 3 sekunders regelen. Bilene viser at de ikke bryter med denne sikkerhetsbarrieren i simuleringen, og dermed er det mulig å si at sikkerheten øker med denne programfunksjonen.

Det er også gjort simuleringer der 3 sekunders regelen er omgjort til 2 sekunder for å øke effektiviteten i trafikken, og siden reaksjonstiden til bilene er såpass hurtig så vil de fortsatt kunne holde avstandene sine. Med dette er det mulig å opprettholde en forbedret sikkerhet, og en økt trafikkeffektivitet slik at effekten av å ta i bruk automatiske kjørefunksjoner blir større.

Den første bilens akselerasjon har også stor innvirkning på resten av køen. Det ble gjort to sett med simuleringer med forskjellig akselerasjon på første bil. Et sett der fartsholderen automatisk akselererte den første bilen opp til 50 km/t, med en maksimal akselerasjon på 4 m/s^2. Det andre settet med simuleringer er med en konstant 2 m/s^2 akselerasjon på første bil opp til 50 km/t, dette gir en mer komfortabel fartsøkning og mer lik hvordan observerte sjåfører kjører, men dette er mindre effektivt i forhold til fartsholderens høyere akselerasjonskurve.

Alle forsøkene er gjort i simuleringer, og gir dermed ikke noe avgjørende svar på om automatiske kjørefunksjoner er mer effektive eller ikke. Resultatene fra simuleringen kan brukes som en indikator på at trafikken går raskere unna, men det vil ikke bli bevist inntil forsøket er gjort med biler der disse funksjonene er installert og testet opp mot de samme scenariene.

I diskusjonen er det tatt for seg forskjellige problemstillinger, fordeler og spørsmål til fremtiden som er med på å forme konklusjonen. Fordelene med slike kjørefunksjoner som det er undersøkt på i denne oppgaven er i all hovedsak de store sikkerhets forbedringene det gir til trafikantene som igjen gir finansielle fordeler til samfunnet, både myndigheter, næringsliv og privatpersoner.

Som forsøket illustrerer så kan økt fremkommelighet være en annen større fordel. Biler med automatiske kjørefunksjoner vil kunne få ned tiden hver enkelt bruker på kjøreturen fra A til B som igjen kan føre til reduserte klimautslipp, energi- og drivstofforbruk.

Om det noen gang blir mulig å ta i bruk helt selvkjørende biler, vil det være mulig for alle å ta i bruk bilen. De med nedsatte funksjonsevner, eldre eller barn kan være en del av de som kan få mulighet til å bruke bilen. Dette gir disse samfunnsgruppene større grad av frihet til å leve livene sine slik de ønsker.

Selv om sikkerheten kan bli en av de store gevinstene med et slikt kjøresystem, så kan de ulykkene som finner sted, bli et dilemma. Helautomatiske biler som ender i en eventuell ulykke, med skader eller dødsfall, kan skape problemer med å finne dem som har skyld i at ulykken i det hele tatt tok plass.

Den forbedrede fremkommeligheten som kan gi ekstra goder til miljøet og privatøkonomien kan også vise seg til å skape en motsatt effekt. Med slike forbedringer kan det tenkes at flere vil ta i bruk bilen, som igjen kan gi en økning av trafikken, som igjen skaper flere biler som slipper ut klimagasser. En økning av elbiler i fremtiden kan hindre en mulig økning av klimagasser i trafikken.

Biler som er avhengige av kommunikasjon med hverandre for å øke fremkommeligheten og sikkerheten på veiene, kan være sårbare for hackerangrep. Dette kan resultere i skader, dødsfall og materielle ødeleggelser som betyr at samfunnet må investere i ekstra beskyttelse for private data og driftssikkerhet. I tillegg må myndighetene allerede investere store summer i å utbygge kommunikasjonsnettet slik at bilene kan kommunisere med hverandre uten at kapasiteten skal stå i veien for dette.

Bibliografi

1. **Store Norske Leksikon.** bilens historie. [Internett] 2018. Hentet 13 Desember 2019. https://snl.no/bilens_historie.

2. **Melissa Bopp, Dangaia Sims, Daniel Piatkowski.** Bicycling for Transportation. s.l. : Elsevier, 2018, Chapter 2 - Benefits and Risks of Bicycling.

3. **Craig Cole.** How Do traffic Jams Start. [Internett] Hentet 13 Desember 2019. https://www.autoguide.com/auto-news/2013/12/how-do-traffic-jams-start.html.

4. **Jeremy Laukkonen.** What Are Autonomous Cars. [Internett] Hentet 13 Desember 2019. https://www.lifewire.com/what-are-autonomous-cars-4588893.

5. **Tony Peng.** Global Survey of Autonomous Vehicle Regulations. [Internett] Hentet 13 Desember 2019. https://medium.com/syncedreview/global-survey-of-autonomous-vehicle-regulations-6b8608f205f9.

6. **Autovista Group.** The state of autonomous legislation in Europe. [Internett] Hentet 13 Desember 2019. https://autovistagroup.com/news-and-insights/state-autonomous-legislation-europe.

7. **Ina Andersen.** Nå blir det lov å teste ut selvkjørende kjøretøy på norske veier. *Teknisk Ukeblad.* [Internett] 2017. Hentet 13 Desember 2019. https://www.tu.no/artikler/na-blir-det-lov-a-teste-ut-selvkjorende-kjoretoy-pa-norske-veier/413029.

8. **Norges Geotekniske Institutt.** Geofysikk, fjernmåling og GIS. [Internett] Hentet 13 Desember 2019. https://www.ngi.no/Tjenester/Fagekspertise/Geofysikk-fjernmaaling-og-GIS/LiDAR.

9. **The Society of Automotive Engineers.** Levels of Automation. [Internett] Hentet 13 Desember 2019. https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles.

10. **Swarco.** Veimerkinger og autonome kjøresystemer. [Internett] Hentet 13 Desember 2019. https://www.swarco.com/no/historier/veimerkinger-og-autonome-kjoresystemer.

11. **Hanssen, Dag Håkon.** Programmerbare logiske styringer. s.l. : Fagbokforlaget Vigmostad & Bjørke AS, 2015, Kapittel 1 & 14.

12. **Codesys.** Versjon 3.5 sp 15. [Internett] Smart Software Solutions GmbH, Hentet 13 Desember 2019. https://www.codesys.com/news-events/news/article/release-codesys-v35-sp15-1.html.

13. **Baird, Dr. Christopher S.** Why does air friction affect a car's gas mileage? [Internett] 2012. Hentet 13 Desember 2019. https://wtamu.edu/~cbaird/sq/2012/12/13/why-does-air-friction-affect-a-cars-gas-mileage/.

14. **Store Norske Leksikon.** Friksjon. [Internett] Hentet 13 Desember 2019. https://snl.no/friksjon.

15. **Lovdata.** Lov om vegtrafikken (vegtrafikkloven) . [Internett] Hentet 13 Desember 2019. https://lovdata.no/dokument/NL/lov/1965-06-18-4#KAPITTEL_2.

16. **Statens vegvesen.** Landevegskjøring. [Internett] 2019. https://www.vegvesen.no/s/e-lering/65plussTipsBilforere/Landeveiskjoring06.html.

17. **Roser, Max.** Economic Growth. *Our World In Data.* [Internett] 2019. Hentet 13 Desember 2019. https://ourworldindata.org/economic-growth.

18. **Bekker, Henk.** 2018 (Full Year) International: Worldwide Car Sales and Global Market Analysis. *Best Selling Cars.* [Internett] Hentet 17 Desember 2019. https://www.best-selling-cars.com/global/2018-full-year-international-worldwide-car-sales-and-global-market-analysis/.

19. **World Health Organization.** Registered vehicles. *WHO.* [Internett] Hentet 17 Desember 2019. http://apps.who.int/gho/data/node.main.A995.

20. **European Commission.** Reducing CO2 emissions from passenger cars. [Internett] Hentet 17 Desember 2019. https://ec.europa.eu/clima/policies/transport/vehicles/cars_en.

21. **Noack, Rick.** U.S. workers want more money. Many Europeans just want more free time. *The Washington Post.* [Internett] Hentet 17 Desember 2019. https://www.washingtonpost.com/news/worldviews/wp/2018/02/08/u-s-workers-want-more-money-many-europeans-just-want-more-free-time/.

22. **Alton, Larry.** How Self-Driving Cars Could Impact the Environment. *Blue & Green Tomorrow.* [Internett] Hentet 17 Desember 2019. https://blueandgreentomorrow.com/environment/self-driving-cars-could-impact-environment/.

23. **Norsk Stål.** Grønnere – Stålproduksjon i dag. [Internett] Hentet 17 Desember 2019. https://norskstaal.no/gronnere-stalproduksjon-i-dag/.

24. **Norsk Hydro.** Hydro lanserer grønnere produktmerker. [Internett] Hentet 17 Desember 2019. https://www.hydro.com/no-NO/media/news/2019/hydro-lanserer-gronnere-produktmerker/.

25. **U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy.** Reducing Pollution with Electric Vehicles. [Internett] Hentet 17 Desember 2019. https://www.energy.gov/eere/electricvehicles/reducing-pollution-electric-vehicles.

26. **Stromberg, Joeseph.** Traffic road induced demand. *Vox.* [Internett] Hentet 16 Desember 2019. https://www.vox.com/2014/10/23/6994159/traffic-roads-induced-demand.

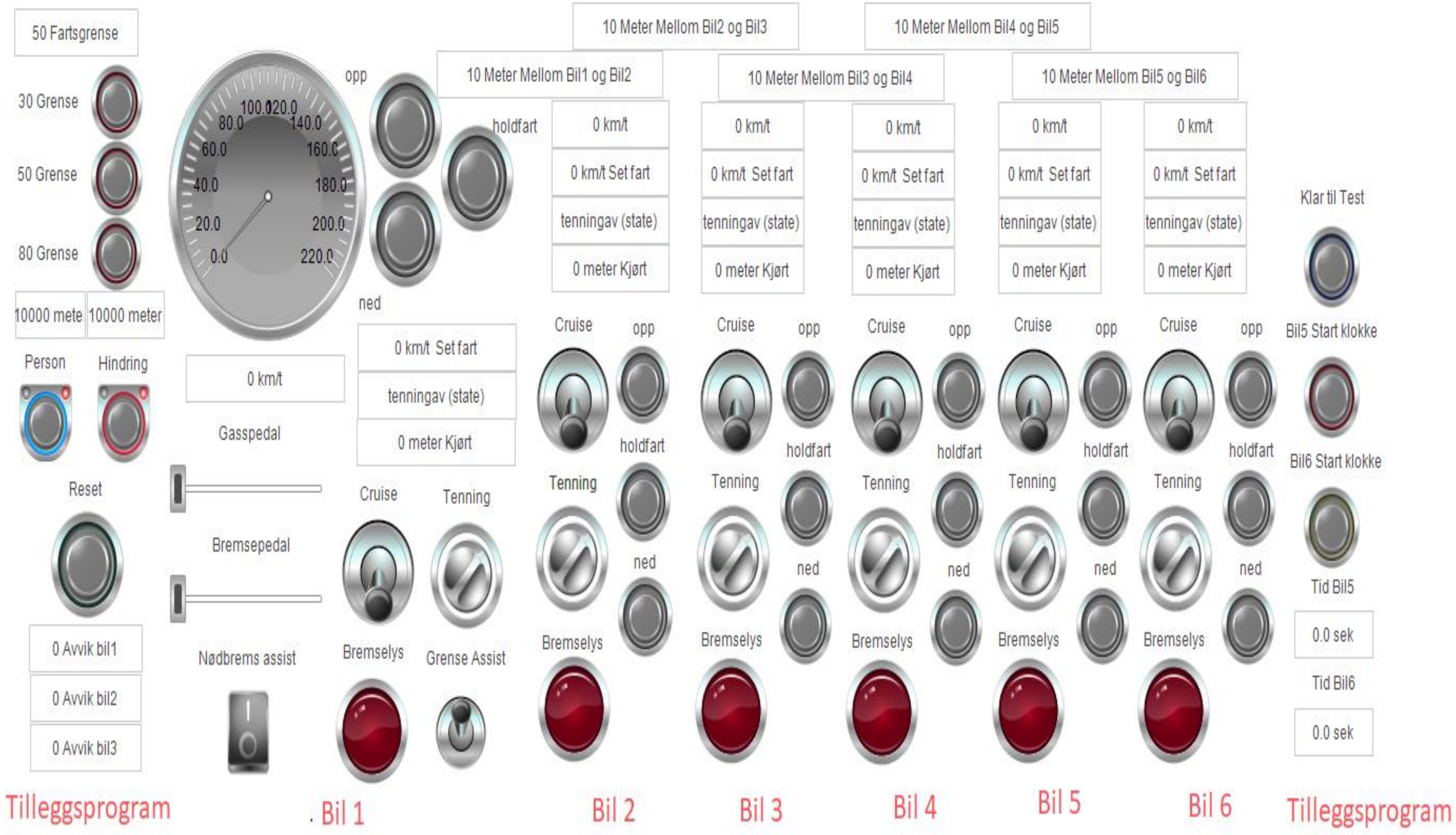
27. **Union of Concerned Scientists.** Self-Driving Cars Explained. [Internett] Hentet 16 Desember 2019. https://www.ucsusa.org/resources/self-driving-cars-101.

28. **ERTRAC.** Automated Driving Roadmap. [Internett] 2015. https://www.ertrac.org/uploads/documentsearch/id38/ERTRAC_Automated-Driving-2015.pdf.

29. **Valevatn, Joakim.** Førerløse biler – behov for politisk styring? *Teknologirådet.* [Internett] 2016. Hentet 13 Desember 2019. https://teknologiradet.no/forerlose-biler-behov-for-politisk-styring/.

30. **University of Michigan: Center for sustainable systems.** Autonomous Vehicles Factsheet. [Internett] 2019. Hentet 13 Desember 2019. http://css.umich.edu/factsheets/autonomous-vehicles-factsheet.

Vedlegg A – Større oversikt av visualisering



Figur 16: En forstørret oversikt over visualiseringen brukt til simulering

Vedlegg B - Målinger fra Observasjon

Tabell 8: Oversikt for målingene fra fotgjengerovergangen.

Forsøk	Bil 1	Bil 2	Bil 3	Bil 4	Bil 5	Bil 6	Bil 7
1	0	1,83	4,76	7,33	9,08	11,49	14,38
2	0	4,45	7,88	12,01	14,12	16,55	18,56
3	0	4,39	6,82	9,82	12,05	14,76	17,58
4	0	4,39	6,96	10,03	12,41	15,55	20,58
5	0	3,4	8,22	10,7	14,3	16,89	19,28
6	0	3,81	6,25	9,66	12,84	15,14	20,76
7	0	4,38	7,43	9,73	13,34	15,44	19,55
8	0	1,44	2,97	4,99	7,25	12,52	14,46
9	0	4,97	7,65	10,63	13,55	16,62	19,75
10	0	4,19	6,51	9,28	11,55	14,53	18,51
11	0	5,11	8,02	9,96	12,77	14,94	18,72
12	0	2,3	6,29	9,34	15,36	22,28	25,36
13	0	5,95	9,1	10,98	13,3	15,67	20,31
14	0	5,57	10,55	13,18	17,87	22,41	26,3
15	0	5,82	7,8	9,92	13,11	15,28	17,35

16	0	4,51	7,49	10,22	12,33	15,4	17,68
17	0	3,15	5,53	9,31	11,79	14,19	16,46
18	0	5,90	9,65	12,58	15,98	26,78	29,87
19	0	4,14	6,13	9	13,49	15,76	19,24
20	0	5,11	8,36	12,75	15,49	18,36	21,94

Tabell 9: Gjennomsnitt og standardavvik for målingene fra fotgjengerovergangen.

	2 Biler	3 Biler	4 Biler	5 Biler	6 Biler	7 Biler
Gjennomsnitt	4,24	7,22	10,07	13,10	16,53	19,83
gjennomsnitt per bil	2,12	2,41	2,52	2,62	2,75	2,83
Største	5,95	10,55	13,18	17,87	26,78	29,87
Minste	1,44	2,97	4,99	7,25	11,49	14,38
Standardavvik				2,31	3,56	

Tabell 10: Oversikt for målingene fra fotgjengerovergangen.

Forsøk	Bil 1	Bil 2	Bil 3	Bil 4	Bil 5	Bil 6	Bil 7
1	0	4,07	5,93	9,01			
2	0	3,93	6,13	9,28	11,91		

3	0	2,69	5,74	8,52	11,06	13,36	15,65
4	0	3,61	6,51				
5	0	2,63	5,45	8,44	10,61	12,64	14,55
6	0	3,02	6,24	8,73	11,75	14,06	
7	0	2,75	4,83	7,03	10,11		
8	0	6,08	10,05	12,68	15,52	17,46	
9	0	4,06	v	9,42	12,13	14,89	
10	0	5,1	8,14	10,93	13,77	17,04	20,09
11	0	3,74	7,7	11,67			
12	0	3,74	5,36	7,82	10,37	13,34	15,15
13	0	5,36	7,82	10,37	13,48	15,15	17,57
14	0	4,71	8,48	10,25	12	14,74	17,7
15	0	3,35	7	9,66	13,41	16,16	20,55
16	0	2,75	5,2	8,2	11,07	14,23	16,85
17	0	4,19	6,66	12,23	15,02	20,16	22,19
18	0	5,89	8,2	10,35	13,94	16,87	18,96

Tabell 11: Gjennomsnitt og standardavvik for målingene fra fotgjengerovergangen.

	2 Biler	3 Biler	4 Biler	5 Biler	6 Biler	7 Biler
Gjennomsnitt	3,98	6,79	9,68	12,41	15,39	17,93

gjennomsnitt per bil	1,99	2,26	2,42	2,48	2,57	2,56
Største	6,08	10,05	12,68	15,52	20,16	22,19
Minste	2,63	4,83	7,03	10,11	12,64	14,55
Standardavvik				1,69	2,09	

Vedlegg C - Målinger fra simulering

Tabell 12: Gjennomsnitt og standardavvik for målingene fra fotgjengerovergangen.

Fartsholder akselerasjon første bil, 2-sekundersregel.			Fartsholder akselerasjon første bil, 3-sekundersregel.		
Forsøk	Bil 5	Bil 6	Forsøk	Bil5	Bil6
1	8,05	9,9	1	10	12,6
2	8,05	9,9	2	9,95	12,55
3	8,00	9,85	3	9,95	12,6
4	8,00	9,85	4	9,95	12,55
5	8,00	9,85	5	9,95	12,55
6	8,15	10	6	9,95	12,6
7	7,85	9,7	7	9,95	12,6
8	8,00	9,85	8	9,95	12,55
9	7,85	9,7	9	10	12,6
10	8,00	9,85	10	9,95	12,55
Gjennomsnitt	8,00	9,85	Gjennomsnitt	9,96	12,58
Standardavvik	0,09	0,09	Standardavvik	0,02	0,025

Tabell 13: Oversikt målinger simulering med konstant 2 $\mathrm{m/s^2}$ *akselerasjon på første bil*

2 $\mathrm{m/s^2}$ konstant akselerasjon på første bil, 2-sekundersregel			2 $\mathrm{m/s^2}$ konstant akselerasjon på første bil, 3-sekundersregel		
Forsøk	Bil5	Bil6	Forsøk	Bil5	Bil6
1	8,8	10,65	1	10,95	13,55
2	8,8	10,65	2	10,95	13,55
3	8,8	10,70	3	10,95	13,60
4	8,8	10,70	4	10,95	13,60
5	8,8	10,65	5	10,95	13,60
6	8,8	10,70	6	10,95	13,55
7	8,8	10,70	7	11,00	13,60
8	8,8	10,70	8	10,95	13,60
9	8,8	10,65	9	10,95	13,55
10	8,8	10,65	10	10,95	13,55
Gjennomsnitt	8,8	10,68	Gjennomsnitt	10,96	13,58
Standardavvik	1,78E-15	0,02	Standardavvik	0,02	0,02

Vedlegg D - Programkode i Codesys ST – Strukturert tekst

Tilleggsprogram

```
PROGRAM Tilleggsprogram
VAR
Grense30, Grense50, Grense80, Reset, Hindringbil, Personivei, TestKnapp, Timerknapp, Timerknapp2 :BOOL;
Intervalrate, Meter1og2, Meter2og3, Meter3og4, Meter4og5, Meter5og6, Meterhindringbilogbil1, Meter1og6kryss, Meter1og5kryss, Timertest, Timertest2, Maalingbil6, Maalingbil5, Fartsgrense, Hindringfart, Meterpersonogbil1 :REAL;
REhindringbil, REpersonivei :R_TRIG;
First_scan: BOOL := TRUE;

END_VAR

Intervalrate := 0.05;

IF First_scan OR Reset THEN
Hindringbil := FALSE;
Hindringfart := 0;
Meter1og2 := 10;
Meter2og3 := 10;
Meter3og4 := 10;
Meter4og5 := 10;
Meter5og6 := 10;
Fartsgrense := 50;
First_scan := FALSE;
ELSIF Grense30 THEN
Fartsgrense := 30;
ELSIF Grense50 THEN
Fartsgrense := 50;
ELSIF Grense80 THEN
Fartsgrense := 80;

END_IF


REhindringbil(CLK:= Hindringbil);


IF REhindringbil.Q THEN

Meterhindringbilogbil1 := 200;

END_IF
```

```
IF Hindringbil THEN

Meterhindringbilogbil1 := Meterhindringbilogbil1 - (Bil1.Hastighet*Intervalrate);
ELSIF NOT Hindringbil THEN
Meterhindringbilogbil1 := 10000;

END_IF

REpersonivei(CLK:= Personivei);

IF REpersonivei.Q THEN
Meterpersonogbil1 := 100;
END_IF

IF Personivei THEN

Meterpersonogbil1:= Meterpersonogbil1 - (Bil1.Hastighet*Intervalrate);
ELSIF NOT Personivei THEN
Meterpersonogbil1 := 10000;

END_IF

Meter1og2 := Meter1og2 + ((Bil1.Hastighet*Intervalrate)-(Bil2.Hastighet*Intervalrate));
Meter2og3 := Meter2og3 + ((Bil2.Hastighet*Intervalrate)-(Bil3.Hastighet*Intervalrate));
Meter3og4 := Meter3og4 + ((Bil3.Hastighet*Intervalrate)-(Bil4.Hastighet*Intervalrate));
Meter4og5 := Meter4og5 + ((Bil4.Hastighet*Intervalrate)-(Bil5.Hastighet*Intervalrate));
Meter5og6 := Meter5og6 + ((Bil5.Hastighet*Intervalrate)-(Bil6.Hastighet*Intervalrate));


 IF TestKnapp THEN
        Meter1og6kryss := Meter1og2 + Meter2og3 + Meter3og4 + Meter4og5 + Meter5og6;
        Meter1og5kryss :=  Meter1og2 + Meter2og3 + Meter3og4 + Meter4og5;
        Maalingbil6 := Bil6.Posisjon;
        Maalingbil5 := Bil5.Posisjon;
        Timertest := 0;
        Timertest2 := 0;
        TestKnapp := FALSE;
END_IF

IF Bil1.Fart_bil > 0 AND Timerknapp THEN
        Timertest := Timertest + Intervalrate;
END_IF


IF Bil6.Posisjon > Meter1og6kryss + Maalingbil6 THEN
        Timerknapp := FALSE;
END_IF

IF Bil1.Fart_bil > 0 AND Timerknapp2 THEN
        Timertest2 := Timertest2 + Intervalrate;
END_IF
```

```
IF Bil5.Posisjon > Meter1og5kryss + Maalingbil5 THEN
        Timerknapp2 := FALSE;
END_IF
```

Bil 1

```
PROGRAM Bil1
VAR
        State : (Tenning_av, Holdfart_cruise, Aks_cruise, Deaks_cruise, Nodbrems_cruise, Frikjoring, Nodbrems_frikjoring);
Tenning, Cruise, Cruise_farthold, Cruise_fartopp, Cruise_fartned, Fartsgrense_assist, Nodbremse_assist, Bremselys, Nodbrems, Bilforan, Bilbak :BOOL;
Intervalrate, Fri, Bremsepedal, Gasspedal, Hastighet, Gassregulering, aksbremsing, aksbil, aksbilregulering, Akselerasjon, Luftmotstand, Rullemotstand, Motorakselerasjon, Retardasjonskonstant, Bremseakselerasjon, Totalakselerasjon, Posisjon, Kpgass, Kpbrems, aksbremsreg, Fart_bil, Set_fart, Avvik, Reg, Bremsreg, Bremsfri, Regavvik, Grenseavvik, Avvikforan :REAL;
REcruise :R_TRIG;

Bil_tilstand: STRING;

First_scan: BOOL := TRUE;

END_VAR

IF First_scan THEN
Tenning := FALSE;
State :=Tenning_av ;
First_scan := FALSE;
END_IF

Intervalrate := Tilleggsprogram.Intervalrate;
Fart_bil :=Hastighet*3.6;
Motorakselerasjon := 4;
Bremseakselerasjon := -8;
Retardasjonskonstant := Luftmotstand + Rullemotstand;
aksbil := Motorakselerasjon * Gasspedal;
aksbilregulering:= Motorakselerasjon * Gassregulering;
aksbremsing := Bremseakselerasjon * Bremsepedal;
aksbremsreg := Bremseakselerasjon * Kpbrems;
Gassregulering := (Kpgass * Avvik);
Gassregulering := LIMIT(0, Gassregulering, 1);
Totalakselerasjon := (aksbil*Fri) + (aksbilregulering*Reg) + Retardasjonskonstant + (aksbremsing*Bremsfri)+(aksbremsreg*Bremsreg);
Akselerasjon := Totalakselerasjon;
Hastighet := Hastighet + (Akselerasjon*Intervalrate);
Hastighet := LIMIT(0, Hastighet, 70);
Posisjon := Posisjon + (Hastighet*Intervalrate);
Regavvik := Set_fart - Fart_bil;
Grenseavvik := Tilleggsprogram.Fartsgrense - Fart_bil;
```

```
Avvikforan := Tilleggsprogram.Hindringfart - Bil1.Fart_bil;

IF (Gasspedal = 0 AND NOT Cruise) OR (Cruise AND Avvik < -1)   THEN
Luftmotstand := -1;
Rullemotstand := -0.2;
ELSIF Gasspedal > 0 OR (Cruise AND Avvik > -1)  THEN
Luftmotstand := 0;
Rullemotstand := 0;
END_IF



IF Tilleggsprogram.Meterhindringbilogbil1 < (Hastighet*8) OR Tilleggsprogram.Meterhindringbilogbil1 < 50 THEN
Bilforan := TRUE;
ELSIF Tilleggsprogram.Meterhindringbilogbil1 > (Hastighet*8) AND Tilleggsprogram.Meterhindringbilogbil1 > 50 THEN
Bilforan := FALSE;
END_IF

IF Tilleggsprogram.Meter1og2 < (Hastighet*8) THEN
Bilbak := TRUE;
ELSIF Tilleggsprogram.Meter1og2 > (Hastighet*8) THEN
Bilbak := FALSE;
END_IF



IF (Tilleggsprogram.Fartsgrense < Set_fart OR Tilleggsprogram.Fartsgrense = Set_fart) AND Fartsgrense_assist AND NOT BILFORAN OR (Bilforan AND Fartsgrense_assist) AND (Tilleggsprogram.Hindringfart > Tilleggsprogram.Fartsgrense) AND (Set_fart > Tilleggsprogram.Fartsgrense OR Tilleggsprogram.Fartsgrense = Set_fart) THEN
Avvik := Grenseavvik;
ELSIF (Fartsgrense_assist AND (Tilleggsprogram.Fartsgrense > Set_fart)) AND (NOT Bilforan OR (Tilleggsprogram.Hindringfart > Set_fart)) OR (NOT Fartsgrense_assist AND NOT Bilforan) THEN
Avvik := Regavvik;
ELSIF Bilforan AND Cruise AND (Tilleggsprogram.Hindringfart < Set_fart) AND (Tilleggsprogram.Meterhindringbilogbil1 < Hastighet*3.5) AND (Fartsgrense_assist AND (Avvikforan < Grenseavvik) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan;
ELSIF Bilforan AND Cruise AND (Tilleggsprogram.Hindringfart < Set_fart) AND (Tilleggsprogram.Meterhindringbilogbil1 > Hastighet*3.5) AND (Fartsgrense_assist AND (Avvikforan < Grenseavvik) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan + 10;
END_IF

IF (Hastighet = 0) AND (Bremsepedal > 0.0) THEN
aksbremsing := 0;
END_IF
```

```
IF Bremsepedal > 0 OR Kpbrems > 0.10 THEN
Bremselys := TRUE;
ELSIF Bremsepedal = 0 AND Kpbrems = 0 THEN
Bremselys := FALSE;
END_IF

REcruise(CLK := Cruise OR Cruise_fartopp OR Cruise_fartned OR Cruise_farthold);

IF REcruise.Q THEN
Cruise := TRUE;
Set_fart := Fart_bil;
Gasspedal := 0;
Cruise_farthold := FALSE;
END_IF

IF Cruise_fartopp AND Set_fart > 0 THEN
Set_fart := Set_fart + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartopp AND Set_fart = 0 THEN
Set_fart := Fart_bil + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartned AND Set_fart > 0 THEN
Set_fart := Set_fart - 5;
Cruise_fartned := FALSE;
ELSIF Cruise_fartned AND Set_fart = 0 THEN
Set_fart := Fart_bil - 5;
Cruise_fartned := FALSE;
END_IF

Set_fart := LIMIT(0, Set_fart, 260);

IF (Bremsepedal > Kpbrems) OR NOT Cruise THEN
Bremsreg :=0;
Bremsfri :=1;
END_IF

IF (Bremsepedal < Kpbrems) AND Cruise THEN
Bremsreg :=1;
Bremsfri :=0;
END_IF

IF (Gasspedal > Gassregulering) OR NOT Cruise OR (Bremsepedal > Kpbrems) THEN
Reg :=0;
Fri :=1;
END_IF


IF (Gasspedal < Gassregulering) AND Cruise THEN
Reg :=1;
Fri :=0;
END_IF
```

```
IF Tilleggsprogram.Meterhindringbilogbil1 < (Hastighet*2.5) OR (Tilleggsprogram.Meterpersonogbil1 < (Hastighet*2.5)) THEN
Nodbrems := TRUE;
END_IF

IF Tilleggsprogram.Reset THEN
Nodbrems := FALSE;
Tenning := FALSE;
Posisjon := 0;
State := Tenning_av;
END_IF


CASE State OF



Tenning_av:
Bil_tilstand := 'tenningav';
Cruise := FALSE;
Bremselys := FALSE;
Nodbremse_assist := TRUE;
Fartsgrense_assist := TRUE;
Set_fart := 0;
Fart_bil := 0;
Hastighet := 0;
Bremsepedal := 0;
Gasspedal := 0;


IF Tenning THEN
State := Frikjoring;
ELSIF Tenning AND Cruise THEN
State := Holdfart_cruise;
END_IF


Holdfart_cruise:

Bil_tilstand := 'holdfartcruise';

IF NOT Tenning THEN
State := Tenning_av;
ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Avvik > 0.5 THEN
```

```
State := Aks_cruise;
ELSIF Avvik < -0.5 THEN
State := Deaks_cruise;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_cruise;
END_IF



Aks_cruise:

Bil_tilstand := 'akscruise';

IF 0.2 < Avvik AND Fart_bil < 30 THEN
Kpgass := 0.08;
ELSIF 0.2 < Avvik AND Fart_bil > 30 THEN
Kpgass := 0.04;
ELSIF Avvik < 0.2  THEN
Kpgass := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpgass := 0;
State := Tenning_av;
ELSIF Avvik < -0.2 THEN
Kpgass := 0;
State := Deaks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF

Deaks_cruise:

Bil_tilstand := 'deakscruise';

IF Tilleggsprogram.Meterhindringbilogbil1 < 7 THEN
Kpbrems := 1;
ELSIF -0.2 > Avvik AND Avvik > -1 AND Tilleggsprogram.Meterhindringbilogbil1 > 7
THEN
Kpbrems := 0.02;
ELSIF -1 > Avvik AND Avvik > -5 AND Tilleggsprogram.Meterhindringbilogbil1 > 7 THEN
Kpbrems := 0.05;
ELSIF -5 > Avvik AND Avvik > -10 AND Tilleggsprogram.Meterhindringbilogbil1 > 7 THEN
```

```
Kpbrems := 0.10;
ELSIF -10 > Avvik AND Avvik > -20 THEN
Kpbrems := 0.15;
ELSIF -20 > Avvik AND Avvik > -30 THEN
Kpbrems := 0.25;
ELSIF Avvik < -30 THEN
Kpbrems := 0.50;
ELSIF Avvik > -0.2 THEN
Kpbrems := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpbrems := 0;
State := Tenning_av;
ELSIF Avvik > 0.2 THEN
Kpbrems := 0;
State := Aks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF

Nodbrems_cruise:
Bil_tilstand := 'nodbremscruise';
IF Tilleggsprogram.Hindringfart < Fart_bil THEN
Bremsepedal := 1;
Set_fart := 0;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Tilleggsprogram.Hindringfart > Fart_bil THEN
Nodbrems := FALSE;
Bremsepedal :=0;
State := frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;

END_IF

Frikjoring:
Bil_tilstand :='frikjoring';
IF NOT Tenning THEN
State := Tenning_av;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_frikjoring;
ELSIF Cruise THEN
```

```
State := Holdfart_cruise;
END_IF

Nodbrems_frikjoring:
Bil_tilstand := 'nodbremsfrikjoring';
IF Hastighet > 0 AND (Tilleggsprogram.Personivei OR Tilleggsprogram.Hindringbil) THEN
Bremsepedal := 1;
Gasspedal := 0;
ELSIF Hastighet = 0 THEN
Nodbrems := FALSE;
Bremsepedal :=0;
END_IF

IF NOT Nodbrems OR NOT Nodbremse_assist THEN
State := Frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;
END_IF


END_CASE
```

Bil 2

```
PROGRAM Bil2
VAR
State : (Tenning_av, Holdfart_cruise, Aks_cruise, Deaks_cruise, Nodbrems_cruise, Frikjoring,
Nodbrems_frikjoring);
Tenning, Cruise, Cruise_farthold, Cruise_fartopp, Cruise_fartned, Fartsgrense_assist,
Nodbremse_assist, Bremselys, Nodbrems, Bilforan, Bilbak :BOOL;
Intervalrate, Fri, Bremsepedal, Gasspedal, Hastighet, Gassregulering, aksbremsing, aksbil,
aksbilregulering, Akselerasjon, Luftmotstand, Rullemotstand, Motorakselerasjon,
Retardasjonskonstant, Bremseakselerasjon, Totalakselerasjon, Posisjon, Kpgass, Kpbrems,
aksbremsreg, Fart_bil, Set_fart, Avvik, Reg, Bremsreg, Bremsfri, Regavvik, Grenseavvik,
Avvikforan :REAL;

REcruise :R_TRIG;

Bil_tilstand: STRING;

First_scan: BOOL := TRUE;
END_VAR


IF First_scan THEN
Tenning := FALSE;
First_scan := FALSE;
State := Tenning_av;
```

```
END_IF

Intervalrate := Tilleggsprogram.Intervalrate;
Fart_bil :=Hastighet*3.6;
Motorakselerasjon := 4;
Bremseakselerasjon := -8;
Retardasjonskonstant := Luftmotstand + Rullemotstand;
aksbil := Motorakselerasjon * Gasspedal;
aksbilregulering:= Motorakselerasjon * Gassregulering;
aksbremsing := Bremseakselerasjon * Bremsepedal;
aksbremsreg := Bremseakselerasjon * Kpbrems;
Gassregulering := (Kpgass * Avvik);
Gassregulering := LIMIT(0, Gassregulering, 1);
Totalakselerasjon := (aksbil*Fri) + (aksbilregulering*Reg) + Retardasjonskonstant + (aksbremsing*Bremsfri)+(aksbremsreg*Bremsreg);
Akselerasjon := Totalakselerasjon;
Hastighet := Hastighet + (Akselerasjon*Intervalrate);
Hastighet := LIMIT(0, Hastighet, 70);
Posisjon := Posisjon + (Hastighet*Intervalrate);
Regavvik := Set_fart - Fart_bil;
Grenseavvik := Tilleggsprogram.Fartsgrense - Fart_bil;
Avvikforan := Bil1.Fart_bil - Bil2.Fart_bil;

IF (Gasspedal = 0 AND NOT Cruise) OR (Cruise AND Avvik < -1)   THEN
Luftmotstand := -1;
Rullemotstand := -0.2;
ELSIF Gasspedal > 0 OR (Cruise AND Avvik > -1)  THEN
Luftmotstand := 0;
Rullemotstand := 0;
END_IF



IF Tilleggsprogram.Meter1og2 < (Hastighet*8) OR (Tilleggsprogram.Meter1og2 < 50) THEN
Bilforan := TRUE;
ELSIF Tilleggsprogram.Meter1og2 > (Hastighet*8) AND (Tilleggsprogram.Meter1og2 > 50) THEN
Bilforan := FALSE;
END_IF

IF Tilleggsprogram.Meter2og3 < (Hastighet*8) THEN
Bilbak := TRUE;
ELSIF Tilleggsprogram.Meter2og3 > (Hastighet*8) THEN
Bilbak := FALSE;
END_IF



IF (Tilleggsprogram.Fartsgrense < Set_fart OR Tilleggsprogram.Fartsgrense = Set_fart) AND Fartsgrense_assist AND NOT BILFORAN OR ( Bilforan AND Fartsgrense_assist AND
```

```
(Bil1.Fart_bil > Tilleggsprogram.Fartsgrense) AND (Set_fart > Tilleggsprogram.Fartsgrense OR Set_fart = Tilleggsprogram.Fartsgrense)) THEN
Avvik := Grenseavvik;
ELSIF (Fartsgrense_assist AND (Tilleggsprogram.Fartsgrense > Set_fart)) AND ((NOT Bilforan) OR (Bil1.Fart_bil > Set_fart)) OR (NOT Fartsgrense_assist AND NOT Bilforan) THEN
Avvik := Regavvik;
ELSIF Bilforan AND Bil1.Fart_bil < Set_fart AND (Tilleggsprogram.Meter1og2 < Hastighet*3.5 AND Tilleggsprogram.Meter1og2 > Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan;
ELSIF Bilforan AND Bil1.Fart_bil < Set_fart AND (Tilleggsprogram.Meter1og2 > Hastighet*3.5) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan + 10;
ELSIF Bilforan AND Bil1.Fart_bil < Set_fart AND (Tilleggsprogram.Meter1og2 < Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan - 10;

END_IF


IF (Hastighet = 0) AND (Bremsepedal > 0.0) THEN
aksbremsing := 0;
END_IF

IF Bremsepedal > 0 OR Kpbrems > 0.10 THEN
Bremselys := TRUE;
ELSIF Bremsepedal = 0 AND Kpbrems = 0 THEN
Bremselys := FALSE;
END_IF

REcruise(CLK := Cruise OR Cruise_fartopp OR Cruise_fartned OR Cruise_farthold);

IF REcruise.Q THEN
Cruise := TRUE;
Set_fart := Fart_bil;
Gasspedal := 0;
Cruise_farthold := FALSE;
END_IF


IF Cruise_fartopp AND Set_fart > 0 THEN
Set_fart := Set_fart + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartopp AND Set_fart = 0 THEN
Set_fart := Fart_bil + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartned AND Set_fart > 0 THEN
Set_fart := Set_fart - 5;
```

```
Cruise_fartned := FALSE;
ELSIF Cruise_fartned AND Set_fart = 0 THEN
Set_fart := Fart_bil - 5;
Cruise_fartned := FALSE;
END_IF

Set_fart := LIMIT(0, Set_fart, 260);

IF (Bremsepedal > Kpbrems) OR NOT Cruise THEN
Bremsreg :=0;
Bremsfri :=1;
END_IF

IF (Bremsepedal < Kpbrems) AND Cruise THEN
Bremsreg :=1;
Bremsfri :=0;
END_IF

IF (Gasspedal > Gassregulering) OR NOT Cruise OR (Bremsepedal > Kpbrems) THEN
Reg :=0;
Fri :=1;
END_IF


IF (Gasspedal < Gassregulering) AND Cruise THEN
Reg :=1;
Fri :=0;
END_IF


IF Tilleggsprogram.Meter1og2 < (Hastighet*2.5) THEN
Nodbrems := TRUE;
END_IF

IF Tilleggsprogram.Reset THEN
Nodbrems := FALSE;
Tenning := FALSE;
Posisjon := 0;
State := Tenning_av;
END_IF


CASE State OF



Tenning_av:
Bil_tilstand := 'tenningav';
Cruise := FALSE;
Bremselys := FALSE;
Nodbremse_assist := TRUE;
```

```
Fartsgrense_assist := TRUE;
Set_fart := 0;
Fart_bil := 0;
Hastighet := 0;
Bremsepedal := 0;
Gasspedal := 0;


IF Tenning THEN
State := Frikjoring;
ELSIF Tenning AND Cruise THEN
State := Holdfart_cruise;
END_IF


Holdfart_cruise:

Bil_tilstand := 'holdfartcruise';

IF NOT Tenning THEN
State := Tenning_av;
ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Avvik > 0.5 THEN
State := Aks_cruise;
ELSIF Avvik < -0.5 THEN
State := Deaks_cruise;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_cruise;
END_IF

Aks_cruise:
Bil_tilstand := 'akscruise';

IF 0.2 < Avvik AND Fart_bil < 30 THEN
Kpgass := 0.08;
ELSIF 0.2 < Avvik AND Fart_bil > 30 THEN
Kpgass := 0.04;
ELSIF Avvik < 0.2  THEN
Kpgass := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpgass := 0;
State := Tenning_av;
ELSIF Avvik < -0.2 THEN
Kpgass := 0;
State := Deaks_cruise;
END_IF
```

```
IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF


Deaks_cruise:
Bil_tilstand := 'deakscruise';

IF Tilleggsprogram.Meter1og2 < 7 THEN
Kpbrems := 1;
ELSIF -0.2 > Avvik AND Avvik > -1 AND Tilleggsprogram.Meter1og2 > 7 THEN
Kpbrems := 0.02;
ELSIF -1 > Avvik AND Avvik > -5 AND Tilleggsprogram.Meter1og2 > 7 THEN
Kpbrems := 0.05;
ELSIF -5 > Avvik AND Avvik > -10 AND Tilleggsprogram.Meter1og2 > 7 THEN
Kpbrems := 0.10;
ELSIF -10 > Avvik AND Avvik > -20 THEN
Kpbrems := 0.15;
ELSIF -20 > Avvik AND Avvik > -30 THEN
Kpbrems := 0.25;
ELSIF Avvik < -30 THEN
Kpbrems := 0.50;
ELSIF Avvik > -0.2 THEN
Kpbrems := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpbrems := 0;
State := Tenning_av;
ELSIF Avvik > 0.2 THEN
Kpbrems := 0;
State := Aks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF
```

```
Nodbrems_cruise:
Bil_tilstand := 'nodbremscruise';
IF Bil1.Hastighet < Hastighet THEN
Bremsepedal := 1;
Set_fart := 0;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil1.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
State := frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;

END_IF

Frikjoring:
Bil_tilstand :='frikjoring';
IF NOT Tenning THEN
State := Tenning_av;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_frikjoring;
ELSIF Cruise THEN
Gasspedal := 0;
State := Holdfart_cruise;
END_IF

Nodbrems_frikjoring:
Bil_tilstand := 'nodbremsfrikjoring';
IF Bil1.Hastighet < Hastighet THEN
Bremsepedal := 1;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil1.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
END_IF

IF NOT Nodbrems OR NOT Nodbremse_assist THEN
State := Frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;
END_IF


END_CASE
```

Bil 3

```
PROGRAM Bil3
VAR
        State : (Tenning_av, Holdfart_cruise, Aks_cruise, Deaks_cruise, Nodbrems_cruise,
Frikjoring, Nodbrems_frikjoring);
```

```
Tenning, Cruise, Cruise_farthold, Cruise_fartopp, Cruise_fartned, Fartsgrense_assist, Nodbremse_assist, Bremselys, Nodbrems, Bilforan, Bilbak :BOOL;
Intervalrate, Fri, Bremsepedal, Gasspedal, Hastighet, Gassregulering, aksbremsing, aksbil, aksbilregulering, Akselerasjon, Luftmotstand, Rullemotstand, Motorakselerasjon, Retardasjonskonstant, Bremseakselerasjon, Totalakselerasjon, Posisjon, Kpgass, Kpbrems, aksbremsreg, Fart_bil, Set_fart, Avvik, Reg, Bremsreg, Bremsfri, Regavvik, Grenseavvik, Avvikforan :REAL;
REcruise :R_TRIG;

Bil_tilstand: STRING;

First_scan: BOOL := TRUE;
END_VAR

IF First_scan THEN
Tenning := FALSE;
First_scan := FALSE;
State := Tenning_av;
END_IF

Intervalrate := Tilleggsprogram.Intervalrate;
Fart_bil :=Hastighet*3.6;
Motorakselerasjon := 4;
Bremseakselerasjon := -8;
Retardasjonskonstant := Luftmotstand + Rullemotstand;
aksbil := Motorakselerasjon * Gasspedal;
aksbilregulering:= Motorakselerasjon * Gassregulering;
aksbremsing := Bremseakselerasjon * Bremsepedal;
aksbremsreg := Bremseakselerasjon * Kpbrems;
Gassregulering := (Kpgass * Avvik);
Gassregulering := LIMIT(0, Gassregulering, 1);
Totalakselerasjon := (aksbil*Fri) + (aksbilregulering*Reg) + Retardasjonskonstant + (aksbremsing*Bremsfri)+(aksbremsreg*Bremsreg);
Akselerasjon := Totalakselerasjon;
Hastighet := Hastighet + (Akselerasjon*Intervalrate);
Hastighet := LIMIT(0, Hastighet, 70);
Posisjon := Posisjon + (Hastighet*Intervalrate);
Regavvik := Set_fart - Fart_bil;
Grenseavvik := Tilleggsprogram.Fartsgrense - Fart_bil;
Avvikforan := Bil2.Fart_bil - Fart_bil;

IF (Gasspedal = 0 AND NOT Cruise) OR (Cruise AND Avvik < -1)   THEN
Luftmotstand := -1;
Rullemotstand := -0.2;
ELSIF Gasspedal > 0 OR (Cruise AND Avvik > -1)  THEN
Luftmotstand := 0;
Rullemotstand := 0;
END_IF
```

```
IF Tilleggsprogram.Meter2og3 < (Hastighet*8) OR (Tilleggsprogram.Meter2og3 < 50) THEN
Bilforan := TRUE;
ELSIF Tilleggsprogram.Meter2og3 > (Hastighet*8) AND (Tilleggsprogram.Meter2og3 > 50)
THEN
Bilforan := FALSE;
END_IF

IF (Tilleggsprogram.Fartsgrense < Set_fart OR Tilleggsprogram.Fartsgrense = Set_fart) AND
Fartsgrense_assist AND NOT BILFORAN OR ( Bilforan AND Fartsgrense_assist AND
(Bil2.Fart_bil > Tilleggsprogram.Fartsgrense) AND (Set_fart > Tilleggsprogram.Fartsgrense
OR Set_fart = Tilleggsprogram.Fartsgrense)) THEN
Avvik := Grenseavvik;
ELSIF  (Fartsgrense_assist AND (Tilleggsprogram.Fartsgrense > Set_fart)) AND ((NOT
Bilforan) OR (Bil2.Fart_bil > Set_fart)) OR (NOT Fartsgrense_assist AND NOT Bilforan)
THEN
Avvik := Regavvik;
ELSIF Bilforan AND Bil2.Fart_bil < Set_fart AND (Tilleggsprogram.Meter2og3 <
Hastighet*3.5 AND Tilleggsprogram.Meter2og3 > Hastighet*3) AND ((Fartsgrense_assist
AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan;
ELSIF Bilforan AND Bil2.Fart_bil < Set_fart AND (Tilleggsprogram.Meter2og3 >
Hastighet*3.5) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan
< Regavvik)) THEN
Avvik := Avvikforan + 10;
ELSIF Bilforan AND Bil2.Fart_bil < Set_fart AND (Tilleggsprogram.Meter2og3 <
Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan <
Regavvik)) THEN
Avvik := Avvikforan - 10;

END_IF


IF (Hastighet = 0) AND (Bremsepedal > 0.0) THEN
aksbremsing := 0;
END_IF

IF Bremsepedal > 0 OR Kpbrems > 0.10 THEN
Bremselys := TRUE;
ELSIF Bremsepedal = 0 AND Kpbrems = 0 THEN
Bremselys := FALSE;
END_IF

REcruise(CLK := Cruise OR Cruise_fartopp OR Cruise_fartned OR Cruise_farthold);

IF REcruise.Q THEN
Cruise := TRUE;
Set_fart := Fart_bil;
Gasspedal := 0;
Cruise_farthold := FALSE;
END_IF
```

```
IF Cruise_fartopp AND Set_fart > 0 THEN
Set_fart := Set_fart + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartopp AND Set_fart = 0 THEN
Set_fart := Fart_bil + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartned AND Set_fart > 0 THEN
Set_fart := Set_fart - 5;
Cruise_fartned := FALSE;
ELSIF Cruise_fartned AND Set_fart = 0 THEN
Set_fart := Fart_bil - 5;
Cruise_fartned := FALSE;
END_IF

Set_fart := LIMIT(0, Set_fart, 260);

IF (Bremsepedal > Kpbrems) OR NOT Cruise THEN
Bremsreg :=0;
Bremsfri :=1;
END_IF

IF (Bremsepedal < Kpbrems) AND Cruise THEN
Bremsreg :=1;
Bremsfri :=0;
END_IF

IF (Gasspedal > Gassregulering) OR NOT Cruise OR (Bremsepedal > Kpbrems) THEN
Reg :=0;
Fri :=1;
END_IF


IF (Gasspedal < Gassregulering) AND Cruise THEN
Reg :=1;
Fri :=0;
END_IF


IF Tilleggsprogram.Meter2og3 < (Hastighet*2.5) THEN
Nodbrems := TRUE;
END_IF

IF Tilleggsprogram.Reset THEN
Nodbrems := FALSE;
Tenning := FALSE;
Posisjon := 0;
State := Tenning_av;
END_IF
```

```
CASE State OF


Tenning_av:
Bil_tilstand := 'tenningav';
Cruise := FALSE;
Bremselys := FALSE;
Nodbremse_assist := TRUE;
Fartsgrense_assist := TRUE;
Set_fart := 0;
Fart_bil := 0;
Hastighet := 0;
Bremsepedal := 0;
Gasspedal := 0;


IF Tenning THEN
State := Frikjoring;
ELSIF Tenning AND Cruise THEN
State := Holdfart_cruise;
END_IF


Holdfart_cruise:

Bil_tilstand := 'holdfartcruise';

IF NOT Tenning THEN
State := Tenning_av;
ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Avvik > 0.5 THEN
State := Aks_cruise;
ELSIF Avvik < -0.5 THEN
State := Deaks_cruise;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_cruise;
END_IF

Aks_cruise:
Bil_tilstand := 'akscruise';

IF 0.2 < Avvik AND Fart_bil < 30 THEN
Kpgass := 0.08;
ELSIF 0.2 < Avvik AND Fart_bil > 30 THEN
Kpgass := 0.04;
ELSIF Avvik < 0.2  THEN
Kpgass := 0;
```

```
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpgass := 0;
State := Tenning_av;
ELSIF Avvik < -0.2 THEN
Kpgass := 0;
State := Deaks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF


Deaks_cruise:
Bil_tilstand := 'deakscruise';

IF Tilleggsprogram.Meter2og3 < 7 THEN
Kpbrems := 1;
ELSIF -0.2 > Avvik AND Avvik > -1 AND Tilleggsprogram.Meter2og3 > 7 THEN
Kpbrems := 0.02;
ELSIF -1 > Avvik AND Avvik > -5 AND Tilleggsprogram.Meter2og3 > 7 THEN
Kpbrems := 0.05;
ELSIF -5 > Avvik AND Avvik > -10 AND Tilleggsprogram.Meter2og3 > 7 THEN
Kpbrems := 0.10;
ELSIF -10 > Avvik AND Avvik > -20 THEN
Kpbrems := 0.15;
ELSIF -20 > Avvik AND Avvik > -30 THEN
Kpbrems := 0.25;
ELSIF Avvik < -30 THEN
Kpbrems := 0.50;
ELSIF Avvik > -0.2 THEN
Kpbrems := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpbrems := 0;
State := Tenning_av;
ELSIF Avvik > 0.2 THEN
Kpbrems := 0;
State := Aks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
```

```
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF

Nodbrems_cruise:
Bil_tilstand := 'nodbremscruise';
IF Bil2.Hastighet < Hastighet THEN
Bremsepedal := 1;
Set_fart := 0;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil2.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
State := frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;

END_IF

Frikjoring:
Bil_tilstand :='frikjoring';
IF NOT Tenning THEN
State := Tenning_av;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_frikjoring;
ELSIF Cruise THEN
Gasspedal := 0;
State := Holdfart_cruise;
END_IF

Nodbrems_frikjoring:
Bil_tilstand := 'nodbremsfrikjoring';
IF Bil2.Hastighet < Hastighet THEN
Bremsepedal := 1;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil2.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
END_IF

IF NOT Nodbrems OR NOT Nodbremse_assist THEN
State := Frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;
END_IF
```

```
END_CASE
```

Bil 4

```
PROGRAM Bil4
VAR
        State : (Tenning_av, Holdfart_cruise, Aks_cruise, Deaks_cruise, Nodbrems_cruise, Frikjoring, Nodbrems_frikjoring);
Tenning, Cruise, Cruise_farthold, Cruise_fartopp, Cruise_fartned, Fartsgrense_assist, Nodbremse_assist, Bremselys, Nodbrems, Bilforan, Bilbak :BOOL;
Intervalrate, Fri, Bremsepedal, Gasspedal, Hastighet, Gassregulering, aksbremsing, aksbil, aksbilregulering, Akselerasjon, Luftmotstand, Rullemotstand, Motorakselerasjon, Retardasjonskonstant, Bremseakselerasjon, Totalakselerasjon, Posisjon, Kpgass, Kpbrems, aksbremsreg, Fart_bil, Set_fart, Avvik, Reg, Bremsreg, Bremsfri, Regavvik, Grenseavvik, Avvikforan :REAL;
REcruise :R_TRIG;

Bil_tilstand: STRING;

First_scan: BOOL := TRUE;
END_VAR

IF First_scan THEN
Tenning := FALSE;
First_scan := FALSE;
State := Tenning_av;
END_IF

Intervalrate := Tilleggsprogram.Intervalrate;
Fart_bil :=Hastighet*3.6;
Motorakselerasjon := 4;
Bremseakselerasjon := -8;
Retardasjonskonstant := Luftmotstand + Rullemotstand;
aksbil := Motorakselerasjon * Gasspedal;
aksbilregulering:= Motorakselerasjon * Gassregulering;
aksbremsing := Bremseakselerasjon * Bremsepedal;
aksbremsreg := Bremseakselerasjon * Kpbrems;
Gassregulering := (Kpgass * Avvik);
Gassregulering := LIMIT(0, Gassregulering, 1);
Totalakselerasjon := (aksbil*Fri) + (aksbilregulering*Reg) + Retardasjonskonstant + (aksbremsing*Bremsfri)+(aksbremsreg*Bremsreg);
Akselerasjon := Totalakselerasjon;
Hastighet := Hastighet + (Akselerasjon*Intervalrate);
Hastighet := LIMIT(0, Hastighet, 70);
Posisjon := Posisjon + (Hastighet*Intervalrate);
Regavvik := Set_fart - Fart_bil;
Grenseavvik := Tilleggsprogram.Fartsgrense - Fart_bil;
Avvikforan := Bil3.Fart_bil - Fart_bil;

IF (Gasspedal = 0 AND NOT Cruise) OR (Cruise AND Avvik < -1)   THEN
```

```
Luftmotstand := -1;
Rullemotstand := -0.2;
ELSIF Gasspedal > 0 OR (Cruise AND Avvik > -1)  THEN
Luftmotstand := 0;
Rullemotstand := 0;
END_IF


IF Tilleggsprogram.Meter3og4 < (Hastighet*8) OR (Tilleggsprogram.Meter3og4 < 50) THEN
Bilforan := TRUE;
ELSIF Tilleggsprogram.Meter3og4 > (Hastighet*8) AND (Tilleggsprogram.Meter3og4 > 50)
THEN
Bilforan := FALSE;
END_IF

IF (Tilleggsprogram.Fartsgrense < Set_fart OR Tilleggsprogram.Fartsgrense = Set_fart) AND
Fartsgrense_assist AND NOT BILFORAN OR ( Bilforan AND Fartsgrense_assist AND
(Bil3.Fart_bil > Tilleggsprogram.Fartsgrense) AND (Set_fart > Tilleggsprogram.Fartsgrense
OR Set_fart = Tilleggsprogram.Fartsgrense)) THEN
Avvik := Grenseavvik;
ELSIF  (Fartsgrense_assist AND (Tilleggsprogram.Fartsgrense > Set_fart)) AND ((NOT
Bilforan) OR (Bil3.Fart_bil > Set_fart)) OR (NOT Fartsgrense_assist AND NOT Bilforan)
THEN
Avvik := Regavvik;
ELSIF Bilforan AND Bil3.Fart_bil < Set_fart AND (Tilleggsprogram.Meter3og4 <
Hastighet*3.5 AND Tilleggsprogram.Meter3og4 > Hastighet*3) AND ((Fartsgrense_assist
AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan;
ELSIF Bilforan AND Bil3.Fart_bil < Set_fart AND (Tilleggsprogram.Meter3og4 >
Hastighet*3.5) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan
< Regavvik)) THEN
Avvik := Avvikforan + 10;
ELSIF Bilforan AND Bil3.Fart_bil < Set_fart AND (Tilleggsprogram.Meter3og4 <
Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan <
Regavvik)) THEN
Avvik := Avvikforan - 10;

END_IF


IF (Hastighet = 0) AND (Bremsepedal > 0.0) THEN
aksbremsing := 0;
END_IF

IF Bremsepedal > 0 OR Kpbrems > 0.10 THEN
Bremselys := TRUE;
ELSIF Bremsepedal = 0 AND Kpbrems = 0 THEN
Bremselys := FALSE;
END_IF
```

```
REcruise(CLK := Cruise OR Cruise_fartopp OR Cruise_fartned OR Cruise_farthold);

IF REcruise.Q THEN
Cruise := TRUE;
Set_fart := Fart_bil;
Gasspedal := 0;
Cruise_farthold := FALSE;
END_IF


IF Cruise_fartopp AND Set_fart > 0 THEN
Set_fart := Set_fart + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartopp AND Set_fart = 0 THEN
Set_fart := Fart_bil + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartned AND Set_fart > 0 THEN
Set_fart := Set_fart - 5;
Cruise_fartned := FALSE;
ELSIF Cruise_fartned AND Set_fart = 0 THEN
Set_fart := Fart_bil - 5;
Cruise_fartned := FALSE;
END_IF

Set_fart := LIMIT(0, Set_fart, 260);

IF (Bremsepedal > Kpbrems) OR NOT Cruise THEN
Bremsreg :=0;
Bremsfri :=1;
END_IF

IF (Bremsepedal < Kpbrems) AND Cruise THEN
Bremsreg :=1;
Bremsfri :=0;
END_IF

IF (Gasspedal > Gassregulering) OR NOT Cruise OR (Bremsepedal > Kpbrems) THEN
Reg :=0;
Fri :=1;
END_IF


IF (Gasspedal < Gassregulering) AND Cruise THEN
Reg :=1;
Fri :=0;
END_IF


IF Tilleggsprogram.Meter3og4 < (Hastighet*2.5) THEN
Nodbrems := TRUE;
END_IF
```

```
IF Tilleggsprogram.Reset THEN
Nodbrems := FALSE;
Tenning := FALSE;
Posisjon := 0;
State := Tenning_av;
END_IF


CASE State OF



Tenning_av:
Bil_tilstand := 'tenningav';
Cruise := FALSE;
Bremselys := FALSE;
Nodbremse_assist := TRUE;
Fartsgrense_assist := TRUE;
Set_fart := 0;
Fart_bil := 0;
Hastighet := 0;
Bremsepedal := 0;
Gasspedal := 0;


IF Tenning THEN
State := Frikjoring;
ELSIF Tenning AND Cruise THEN
State := Holdfart_cruise;
END_IF


Holdfart_cruise:

Bil_tilstand := 'holdfartcruise';

IF NOT Tenning THEN
State := Tenning_av;
ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Avvik > 0.5 THEN
State := Aks_cruise;
ELSIF Avvik < -0.5 THEN
State := Deaks_cruise;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_cruise;
END_IF
```

```
Aks_cruise:
Bil_tilstand := 'akscruise';

IF 0.2 < Avvik AND Fart_bil < 30 THEN
Kpgass := 0.08;
ELSIF 0.2 < Avvik AND Fart_bil > 30 THEN
Kpgass := 0.04;
ELSIF Avvik < 0.2  THEN
Kpgass := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpgass := 0;
State := Tenning_av;
ELSIF Avvik < -0.2 THEN
Kpgass := 0;
State := Deaks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF


Deaks_cruise:
Bil_tilstand := 'deakscruise';

IF Tilleggsprogram.Meter3og4 < 7 THEN
Kpbrems := 1;
ELSIF -0.2 > Avvik AND Avvik > -1 AND Tilleggsprogram.Meter3og4 > 7 THEN
Kpbrems := 0.02;
ELSIF -1 > Avvik AND Avvik > -5 AND Tilleggsprogram.Meter3og4 > 7 THEN
Kpbrems := 0.05;
ELSIF -5 > Avvik AND Avvik > -10 AND Tilleggsprogram.Meter3og4 > 7 THEN
Kpbrems := 0.10;
ELSIF -10 > Avvik AND Avvik > -20 THEN
Kpbrems := 0.15;
ELSIF -20 > Avvik AND Avvik > -30 THEN
Kpbrems := 0.25;
ELSIF Avvik < -30 THEN
Kpbrems := 0.50;
ELSIF Avvik > -0.2 THEN
Kpbrems := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
```

```
Kpbrems := 0;
State := Tenning_av;
ELSIF Avvik > 0.2 THEN
Kpbrems := 0;
State := Aks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF

Nodbrems_cruise:
Bil_tilstand := 'nodbremscruise';
IF Bil3.Hastighet < Hastighet THEN
Bremsepedal := 1;
Set_fart := 0;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil3.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
State := frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;

END_IF

Frikjoring:
Bil_tilstand :='frikjoring';
IF NOT Tenning THEN
State := Tenning_av;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_frikjoring;
ELSIF Cruise THEN
Gasspedal := 0;
State := Holdfart_cruise;
END_IF

Nodbrems_frikjoring:
Bil_tilstand := 'nodbremsfrikjoring';
IF Bil3.Hastighet < Hastighet THEN
Bremsepedal := 1;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil3.Hastighet > Hastighet THEN
Nodbrems := FALSE;
```

```
Bremsepedal :=0;
END_IF

IF NOT Nodbrems OR NOT Nodbremse_assist THEN
State := Frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;
END_IF


END_CASE
```

Bil 5

```
PROGRAM Bil5
VAR
        State : (Tenning_av, Holdfart_cruise, Aks_cruise, Deaks_cruise, Nodbrems_cruise,
Frikjoring, Nodbrems_frikjoring);
Tenning, Cruise, Cruise_farthold, Cruise_fartopp, Cruise_fartned, Fartsgrense_assist,
Nodbremse_assist, Bremselys, Nodbrems, Bilforan, Bilbak :BOOL;
Intervalrate, Fri, Bremsepedal, Gasspedal, Hastighet, Gassregulering, aksbremsing, aksbil,
aksbilregulering, Akselerasjon, Luftmotstand, Rullemotstand, Motorakselerasjon,
Retardasjonskonstant, Bremseakselerasjon, Totalakselerasjon, Posisjon, Kpgass, Kpbrems,
aksbremsreg, Fart_bil, Set_fart, Avvik, Reg, Bremsreg, Bremsfri, Regavvik, Grenseavvik,
Avvikforan :REAL;
REcruise :R_TRIG;

Bil_tilstand: STRING;

First_scan: BOOL := TRUE;
END_VAR

IF First_scan THEN
Tenning := FALSE;
First_scan := FALSE;
State := Tenning_av;
END_IF

Intervalrate := Tilleggsprogram.Intervalrate;
Fart_bil :=Hastighet*3.6;
Motorakselerasjon := 4;
Bremseakselerasjon := -8;
Retardasjonskonstant := Luftmotstand + Rullemotstand;
aksbil := Motorakselerasjon * Gasspedal;
aksbilregulering:= Motorakselerasjon * Gassregulering;
aksbremsing := Bremseakselerasjon * Bremsepedal;
aksbremsreg := Bremseakselerasjon * Kpbrems;
Gassregulering := (Kpgass * Avvik);
Gassregulering := LIMIT(0, Gassregulering, 1);
Totalakselerasjon := (aksbil*Fri) + (aksbilregulering*Reg) + Retardasjonskonstant +
(aksbremsing*Bremsfri)+(aksbremsreg*Bremsreg);
```

```
Akselerasjon := Totalakselerasjon;
Hastighet := Hastighet + (Akselerasjon*Intervalrate);
Hastighet := LIMIT(0, Hastighet, 70);
Posisjon := Posisjon + (Hastighet*Intervalrate);
Regavvik := Set_fart - Fart_bil;
Grenseavvik := Tilleggsprogram.Fartsgrense - Fart_bil;
Avvikforan := Bil4.Fart_bil - Fart_bil;

IF (Gasspedal = 0 AND NOT Cruise) OR (Cruise AND Avvik < -1)   THEN
Luftmotstand := -1;
Rullemotstand := -0.2;
ELSIF Gasspedal > 0 OR (Cruise AND Avvik > -1)  THEN
Luftmotstand := 0;
Rullemotstand := 0;
END_IF



IF Tilleggsprogram.Meter4og5 < (Hastighet*8) OR (Tilleggsprogram.Meter4og5 < 50) THEN
Bilforan := TRUE;
ELSIF Tilleggsprogram.Meter4og5 > (Hastighet*8) AND (Tilleggsprogram.Meter4og5 > 50)
THEN
Bilforan := FALSE;
END_IF

IF (Tilleggsprogram.Fartsgrense < Set_fart OR Tilleggsprogram.Fartsgrense = Set_fart) AND
Fartsgrense_assist AND NOT BILFORAN OR ( Bilforan AND Fartsgrense_assist AND
(Bil4.Fart_bil > Tilleggsprogram.Fartsgrense) AND (Set_fart > Tilleggsprogram.Fartsgrense
OR Set_fart = Tilleggsprogram.Fartsgrense)) THEN
Avvik := Grenseavvik;
ELSIF  (Fartsgrense_assist AND (Tilleggsprogram.Fartsgrense > Set_fart)) AND ((NOT
Bilforan) OR (Bil4.Fart_bil > Set_fart)) OR (NOT Fartsgrense_assist AND NOT Bilforan)
THEN
Avvik := Regavvik;
ELSIF Bilforan AND Bil4.Fart_bil < Set_fart AND (Tilleggsprogram.Meter4og5 <
Hastighet*3.5 AND Tilleggsprogram.Meter4og5 > Hastighet*3) AND ((Fartsgrense_assist
AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan;
ELSIF Bilforan AND Bil4.Fart_bil < Set_fart AND (Tilleggsprogram.Meter4og5 >
Hastighet*3.5) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan
< Regavvik)) THEN
Avvik := Avvikforan + 10;
ELSIF Bilforan AND Bil4.Fart_bil < Set_fart AND (Tilleggsprogram.Meter4og5 <
Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan <
Regavvik)) THEN
Avvik := Avvikforan - 10;

END_IF



IF (Hastighet = 0) AND (Bremsepedal > 0.0) THEN
```

```
aksbremsing := 0;
END_IF

IF Bremsepedal > 0 OR Kpbrems > 0.10 THEN
Bremselys := TRUE;
ELSIF Bremsepedal = 0 AND Kpbrems = 0 THEN
Bremselys := FALSE;
END_IF

REcruise(CLK := Cruise OR Cruise_fartopp OR Cruise_fartned OR Cruise_farthold);

IF REcruise.Q THEN
Cruise := TRUE;
Set_fart := Fart_bil;
Gasspedal := 0;
Cruise_farthold := FALSE;
END_IF


IF Cruise_fartopp AND Set_fart > 0 THEN
Set_fart := Set_fart + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartopp AND Set_fart = 0 THEN
Set_fart := Fart_bil + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartned AND Set_fart > 0 THEN
Set_fart := Set_fart - 5;
Cruise_fartned := FALSE;
ELSIF Cruise_fartned AND Set_fart = 0 THEN
Set_fart := Fart_bil - 5;
Cruise_fartned := FALSE;
END_IF

Set_fart := LIMIT(0, Set_fart, 260);

IF (Bremsepedal > Kpbrems) OR NOT Cruise THEN
Bremsreg :=0;
Bremsfri :=1;
END_IF

IF (Bremsepedal < Kpbrems) AND Cruise THEN
Bremsreg :=1;
Bremsfri :=0;
END_IF

IF (Gasspedal > Gassregulering) OR NOT Cruise OR (Bremsepedal > Kpbrems) THEN
Reg :=0;
Fri :=1;
END_IF
```

```
IF (Gasspedal < Gassregulering) AND Cruise THEN
Reg :=1;
Fri :=0;
END_IF


IF Tilleggsprogram.Meter4og5 < (Hastighet*2.5) THEN
Nodbrems := TRUE;
END_IF

IF Tilleggsprogram.Reset THEN
Nodbrems := FALSE;
Tenning := FALSE;
Posisjon := 0;
State := Tenning_av;
END_IF


CASE State OF



Tenning_av:
Bil_tilstand := 'tenningav';
Cruise := FALSE;
Bremselys := FALSE;
Nodbremse_assist := TRUE;
Fartsgrense_assist := TRUE;
Set_fart := 0;
Fart_bil := 0;
Hastighet := 0;
Bremsepedal := 0;
Gasspedal := 0;


IF Tenning THEN
State := Frikjoring;
ELSIF Tenning AND Cruise THEN
State := Holdfart_cruise;
END_IF


Holdfart_cruise:

Bil_tilstand := 'holdfartcruise';

IF NOT Tenning THEN
State := Tenning_av;
ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
Set_fart := 0;
Cruise := FALSE;
```

```
State := Frikjoring;
ELSIF Avvik > 0.5 THEN
State := Aks_cruise;
ELSIF Avvik < -0.5 THEN
State := Deaks_cruise;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_cruise;
END_IF

Aks_cruise:
Bil_tilstand := 'akscruise';

IF 0.2 < Avvik AND Fart_bil < 30 THEN
Kpgass := 0.08;
ELSIF 0.2 < Avvik AND Fart_bil > 30 THEN
Kpgass := 0.04;
ELSIF Avvik < 0.2  THEN
Kpgass := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpgass := 0;
State := Tenning_av;
ELSIF Avvik < -0.2 THEN
Kpgass := 0;
State := Deaks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF


Deaks_cruise:
Bil_tilstand := 'deakscruise';

IF Tilleggsprogram.Meter4og5 < 7 THEN
Kpbrems := 1;
ELSIF -0.2 > Avvik AND Avvik > -1 AND Tilleggsprogram.Meter4og5 > 7 THEN
Kpbrems := 0.02;
ELSIF -1 > Avvik AND Avvik > -5 AND Tilleggsprogram.Meter4og5 > 7 THEN
Kpbrems := 0.05;
ELSIF -5 > Avvik AND Avvik > -10 AND Tilleggsprogram.Meter4og5 > 7 THEN
Kpbrems := 0.10;
ELSIF -10 > Avvik AND Avvik > -20 THEN
```

```
Kpbrems := 0.15;
ELSIF -20 > Avvik AND Avvik > -30 THEN
Kpbrems := 0.25;
ELSIF Avvik < -30 THEN
Kpbrems := 0.50;
ELSIF Avvik > -0.2 THEN
Kpbrems := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpbrems := 0;
State := Tenning_av;
ELSIF Avvik > 0.2 THEN
Kpbrems := 0;
State := Aks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF

Nodbrems_cruise:
Bil_tilstand := 'nodbremscruise';
IF Bil4.Hastighet < Hastighet THEN
Bremsepedal := 1;
Set_fart := 0;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil4.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
State := frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;

END_IF

Frikjoring:
Bil_tilstand :='frikjoring';
IF NOT Tenning THEN
State := Tenning_av;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_frikjoring;
ELSIF Cruise THEN
Gasspedal := 0;
State := Holdfart_cruise;
```

```
END_IF

Nodbrems_frikjoring:
Bil_tilstand := 'nodbremsfrikjoring';
IF Bil4.Hastighet < Hastighet THEN
Bremsepedal := 1;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil4.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
END_IF

IF NOT Nodbrems OR NOT Nodbremse_assist THEN
State := Frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;
END_IF


END_CASE
```

Bil 6

```
PROGRAM Bil6
VAR
        State : (Tenning_av, Holdfart_cruise, Aks_cruise, Deaks_cruise, Nodbrems_cruise,
Frikjoring, Nodbrems_frikjoring);
Tenning, Cruise, Cruise_farthold, Cruise_fartopp, Cruise_fartned, Fartsgrense_assist,
Nodbremse_assist, Bremselys, Nodbrems, Bilforan, Bilbak :BOOL;
Intervalrate, Fri, Bremsepedal, Gasspedal, Hastighet, Gassregulering, aksbremsing, aksbil,
aksbilregulering, Akselerasjon, Luftmotstand, Rullemotstand, Motorakselerasjon,
Retardasjonskonstant, Bremseakselerasjon, Totalakselerasjon, Posisjon, Kpgass, Kpbrems,
aksbremsreg, Fart_bil, Set_fart, Avvik, Reg, Bremsreg, Bremsfri, Regavvik, Grenseavvik,
Avvikforan :REAL;
REcruise :R_TRIG;

Bil_tilstand: STRING;

First_scan: BOOL := TRUE;
END_VAR

IF First_scan THEN
Tenning := FALSE;
First_scan := FALSE;
State := Tenning_av;
END_IF

Intervalrate := Tilleggsprogram.Intervalrate;
Fart_bil :=Hastighet*3.6;
Motorakselerasjon := 4;
Bremseakselerasjon := -8;
```

```
Retardasjonskonstant := Luftmotstand + Rullemotstand;
aksbil := Motorakselerasjon * Gasspedal;
aksbilregulering:= Motorakselerasjon * Gassregulering;
aksbremsing := Bremseakselerasjon * Bremsepedal;
aksbremsreg := Bremseakselerasjon * Kpbrems;
Gassregulering := (Kpgass * Avvik);
Gassregulering := LIMIT(0, Gassregulering, 1);
Totalakselerasjon := (aksbil*Fri) + (aksbilregulering*Reg) + Retardasjonskonstant + (aksbremsing*Bremsfri)+(aksbremsreg*Bremsreg);
Akselerasjon := Totalakselerasjon;
Hastighet := Hastighet + (Akselerasjon*Intervalrate);
Hastighet := LIMIT(0, Hastighet, 70);
Posisjon := Posisjon + (Hastighet*Intervalrate);
Regavvik := Set_fart - Fart_bil;
Grenseavvik := Tilleggsprogram.Fartsgrense - Fart_bil;
Avvikforan := Bil5.Fart_bil - Fart_bil;

IF (Gasspedal = 0 AND NOT Cruise) OR (Cruise AND Avvik < -1)   THEN
Luftmotstand := -1;
Rullemotstand := -0.2;
ELSIF Gasspedal > 0 OR (Cruise AND Avvik > -1)  THEN
Luftmotstand := 0;
Rullemotstand := 0;
END_IF



IF Tilleggsprogram.Meter5og6 < (Hastighet*8) OR (Tilleggsprogram.Meter5og6 < 50) THEN
Bilforan := TRUE;
ELSIF Tilleggsprogram.Meter5og6 > (Hastighet*8) AND (Tilleggsprogram.Meter5og6 > 50) THEN
Bilforan := FALSE;
END_IF

IF (Tilleggsprogram.Fartsgrense < Set_fart OR Tilleggsprogram.Fartsgrense = Set_fart) AND Fartsgrense_assist AND NOT BILFORAN OR ( Bilforan AND Fartsgrense_assist AND (Bil5.Fart_bil > Tilleggsprogram.Fartsgrense) AND (Set_fart > Tilleggsprogram.Fartsgrense OR Set_fart = Tilleggsprogram.Fartsgrense)) THEN
Avvik := Grenseavvik;
ELSIF  (Fartsgrense_assist AND (Tilleggsprogram.Fartsgrense > Set_fart)) AND ((NOT Bilforan) OR (Bil5.Fart_bil > Set_fart)) OR (NOT Fartsgrense_assist AND NOT Bilforan) THEN
Avvik := Regavvik;
ELSIF Bilforan AND Bil5.Fart_bil < Set_fart AND (Tilleggsprogram.Meter5og6 < Hastighet*3.5 AND Tilleggsprogram.Meter5og6 > Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan;
ELSIF Bilforan AND Bil5.Fart_bil < Set_fart AND (Tilleggsprogram.Meter5og6 > Hastighet*3.5) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan + 10;
```

```
ELSIF Bilforan AND Bil5.Fart_bil < Set_fart AND (Tilleggsprogram.Meter5og6 < Hastighet*3) AND ((Fartsgrense_assist AND (Avvikforan < Grenseavvik)) OR ( Avvikforan < Regavvik)) THEN
Avvik := Avvikforan - 10;

END_IF


IF (Hastighet = 0) AND (Bremsepedal > 0.0) THEN
aksbremsing := 0;
END_IF

IF Bremsepedal > 0 OR Kpbrems > 0.10 THEN
Bremselys := TRUE;
ELSIF Bremsepedal = 0 AND Kpbrems = 0 THEN
Bremselys := FALSE;
END_IF

REcruise(CLK := Cruise OR Cruise_fartopp OR Cruise_fartned OR Cruise_farthold);

IF REcruise.Q THEN
Cruise := TRUE;
Set_fart := Fart_bil;
Gasspedal := 0;
Cruise_farthold := FALSE;
END_IF


IF Cruise_fartopp AND Set_fart > 0 THEN
Set_fart := Set_fart + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartopp AND Set_fart = 0 THEN
Set_fart := Fart_bil + 5;
Cruise_fartopp := FALSE;
ELSIF Cruise_fartned AND Set_fart > 0 THEN
Set_fart := Set_fart - 5;
Cruise_fartned := FALSE;
ELSIF Cruise_fartned AND Set_fart = 0 THEN
Set_fart := Fart_bil - 5;
Cruise_fartned := FALSE;
END_IF

Set_fart := LIMIT(0, Set_fart, 260);

IF (Bremsepedal > Kpbrems) OR NOT Cruise THEN
Bremsreg :=0;
Bremsfri :=1;
END_IF

IF (Bremsepedal < Kpbrems) AND Cruise THEN
Bremsreg :=1;
```

```
Bremsfri :=0;
END_IF

IF (Gasspedal > Gassregulering) OR NOT Cruise OR (Bremsepedal > Kpbrems) THEN
Reg :=0;
Fri :=1;
END_IF


IF (Gasspedal < Gassregulering) AND Cruise THEN
Reg :=1;
Fri :=0;
END_IF


IF Tilleggsprogram.Meter5og6 < (Hastighet*2.5) THEN
Nodbrems := TRUE;
END_IF

IF Tilleggsprogram.Reset THEN
Nodbrems := FALSE;
Tenning := FALSE;
Posisjon := 0;
State := Tenning_av;
END_IF


CASE State OF



Tenning_av:
Bil_tilstand := 'tenningav';
Cruise := FALSE;
Bremselys := FALSE;
Nodbremse_assist := TRUE;
Fartsgrense_assist := TRUE;
Set_fart := 0;
Fart_bil := 0;
Hastighet := 0;
Bremsepedal := 0;
Gasspedal := 0;


IF Tenning THEN
State := Frikjoring;
ELSIF Tenning AND Cruise THEN
State := Holdfart_cruise;
END_IF
```

```
Holdfart_cruise:

Bil_tilstand := 'holdfartcruise';

IF NOT Tenning THEN
State := Tenning_av;
ELSIF (NOT Cruise) OR (Bremsepedal > 0) THEN
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Avvik > 0.5 THEN
State := Aks_cruise;
ELSIF Avvik < -0.5 THEN
State := Deaks_cruise;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_cruise;
END_IF

Aks_cruise:
Bil_tilstand := 'akscruise';

IF 0.2 < Avvik AND Fart_bil < 30 THEN
Kpgass := 0.08;
ELSIF 0.2 < Avvik AND Fart_bil > 30 THEN
Kpgass := 0.04;
ELSIF Avvik < 0.2  THEN
Kpgass := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpgass := 0;
State := Tenning_av;
ELSIF Avvik < -0.2 THEN
Kpgass := 0;
State := Deaks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF


Deaks_cruise:
Bil_tilstand := 'deakscruise';
```

```
IF Tilleggsprogram.Meter5og6 < 7 THEN
Kpbrems := 1;
ELSIF -0.2 > Avvik AND Avvik > -1 AND Tilleggsprogram.Meter5og6 > 7 THEN
Kpbrems := 0.02;
ELSIF -1 > Avvik AND Avvik > -5 AND Tilleggsprogram.Meter5og6 > 7 THEN
Kpbrems := 0.05;
ELSIF -5 > Avvik AND Avvik > -10 AND Tilleggsprogram.Meter5og6 > 7 THEN
Kpbrems := 0.10;
ELSIF -10 > Avvik AND Avvik > -20 THEN
Kpbrems := 0.15;
ELSIF -20 > Avvik AND Avvik > -30 THEN
Kpbrems := 0.25;
ELSIF Avvik < -30 THEN
Kpbrems := 0.50;
ELSIF Avvik > -0.2 THEN
Kpbrems := 0;
State := Holdfart_cruise;
ELSIF NOT Tenning THEN
Kpbrems := 0;
State := Tenning_av;
ELSIF Avvik > 0.2 THEN
Kpbrems := 0;
State := Aks_cruise;
END_IF


IF (NOT Cruise) OR (Bremsepedal > 0) THEN
Kpbrems := 0;
Set_fart := 0;
Cruise := FALSE;
State := Frikjoring;
ELSIF Nodbrems AND Nodbremse_assist THEN
Kpbrems := 0;
State := Nodbrems_cruise;
END_IF

Nodbrems_cruise:
Bil_tilstand := 'nodbremscruise';
IF Bil5.Hastighet < Hastighet THEN
Bremsepedal := 1;
Set_fart := 0;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil5.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
State := frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;

END_IF
```

```
Frikjoring:
Bil_tilstand :='frikjoring';
IF NOT Tenning THEN
State := Tenning_av;
ELSIF Nodbrems AND Nodbremse_assist THEN
State := Nodbrems_frikjoring;
ELSIF Cruise THEN
Gasspedal := 0;
State := Holdfart_cruise;
END_IF

Nodbrems_frikjoring:
Bil_tilstand := 'nodbremsfrikjoring';
IF Bil5.Hastighet < Hastighet THEN
Bremsepedal := 1;
Gasspedal := 0;
ELSIF Hastighet = 0 OR Bil5.Hastighet > Hastighet THEN
Nodbrems := FALSE;
Bremsepedal :=0;
END_IF

IF NOT Nodbrems OR NOT Nodbremse_assist THEN
State := Frikjoring;
ELSIF NOT Tenning THEN
State := Tenning_av;
END_IF


END_CASE
```