CHALMERS

Robot Setup Generator

Examensarbete inom Data- och Informationsteknik
Mekatronikingenjör

Henrik Hjelm
Arvid Pearson

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET
Göteborg, Sverige 2018

EXAMENSARBETE

Robot Setup Generator

Henrik Hjelm
Arvid Pearson

Institutionen för Data- och Informationsteknik
CHALMERS TEKNISKA HÖGSKOLA
GÖTEBORGS UNIVERSITET

Göteborg 2018

Robot Setup Generator

Henrik Hjelm
Arvid Pearson

Examinator: Jonas Duregård
Handledare: Sven Knutsson

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

Omslag:
Körrobot som styrs av RC8.

Institutionen för Data- och Informationsteknik
Göteborg 2018

SAMMANFATTNING

Denna rapport beskriver utvecklingen av Robot Setup Generator. Ett verktyg för att underlätta utformandet av körtest för robotstyrda bilar som används vid objektiv aktiv säkerhetsverifiering. Syftet var att eliminera eventuella fel som kan uppstå när flera körscenarion skall läggas in i ett test. Målet med projektet var att skapa ett program som automatiserar delar av arbetet under en konfiguration. Projektet resulterade i att ett program som tar lastfall från simuleringsmiljön och automatiskt skapar robotkonfigurationen med önskade inställningar för roboten. Detta program underlättar arbetet under konfigurationsfasen då de eventuella mänskliga felen elimineras och frigör tid som annars lades på repetitivt arbete. Arbetet har utförts i Volvo Cars anläggning på provbanan i Hällered.

ABSTRACT

This report describes the development of Robot Setup Generator. A tool to facilitate the setup of driving tests for robot controlled cars used for objective active safety verification. The purpose was to eliminate any errors that may occur when multiple driving scenarios are to be added. The goal of the project was to create a program that automates parts of the work during a setup. The project resulted in a program that takes files from the simulation environment and automatically creates the robot configuration with desired settings for the robot. This program facilitates work during the setup phase, eliminating the possible human errors and freeing time that was otherwise put on repetitive work. The work has been carried out in the Volvo Cars facility on the Hällered test track.

FÖRORD

Vi vill tacka Volvo Cars, Albert Lawenius, Per Hesslund samt all personal på avdelningen för möjligheten att utföra vårt examensarbete på faciliteterna i Hällered. Vi vill även passa på att tacka vår handledare från Chalmers Sven Knutsson.

Innehåll

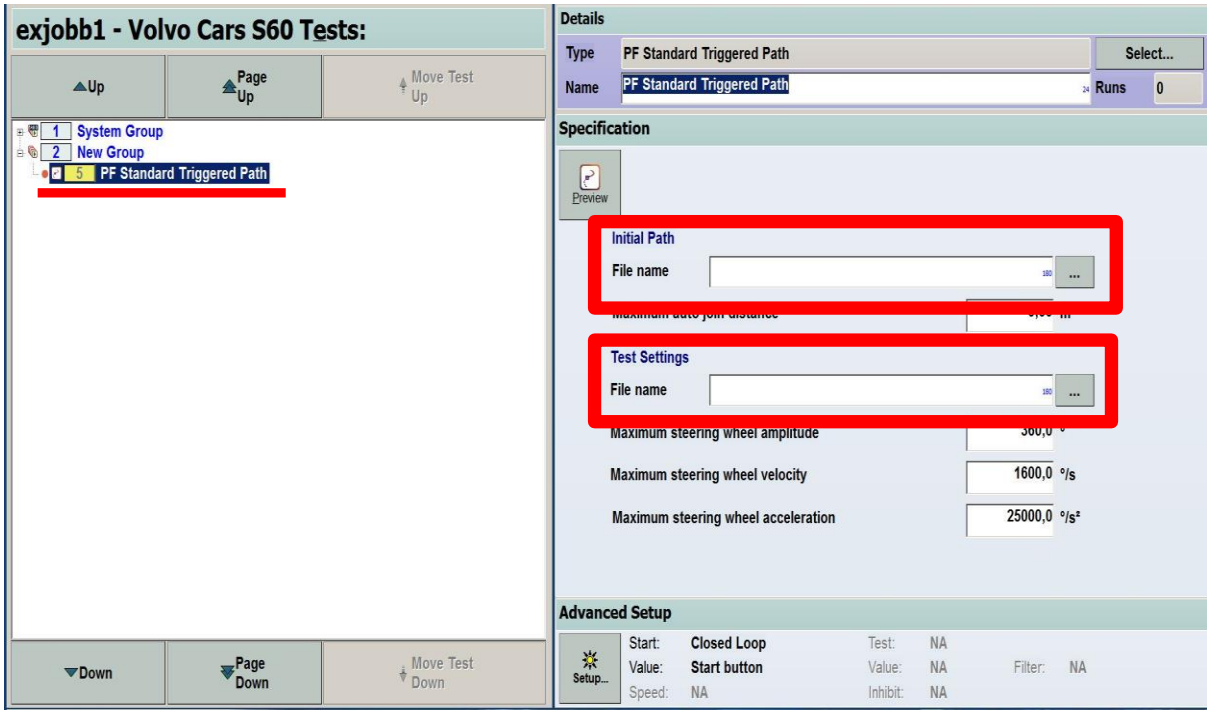
1 Inledning

Avdelningen Vehicle and Rig Testing på Volvo Cars testar olika funktioner i bilar på en provbana. Bland testerna ingår test av aktiv säkerhet. Aktiv säkerhet är åtgärder innan kollision som minimerar skador och krockvåld. Det är nödvändigt att testa funktionerna i verkligheten för att garantera önskat resultat. Olika scenarion testas med hjälp av de riggar och den utrustning som avdelningen jobbar med. Scenarierna avser att simulera verkliga trafiksituationer på provbanan.

Vid körprover använder man sig av en körrobot för att få noggrannhet och repeterbarhet, denna kör bilen på det sätt som tidigare bestämts i en simuleringsmiljö. För att tala om för roboten hur den skall köra kan digitala körfiler skapas som den skall följa. Betydelsen av körfiler förtydligas i teknisk bakgrund på sidan 6. När ett test utförs måste rätt test peka på rätt körfil, detta görs idag manuellt, vilket är både tidskrävande och felkänsligt.

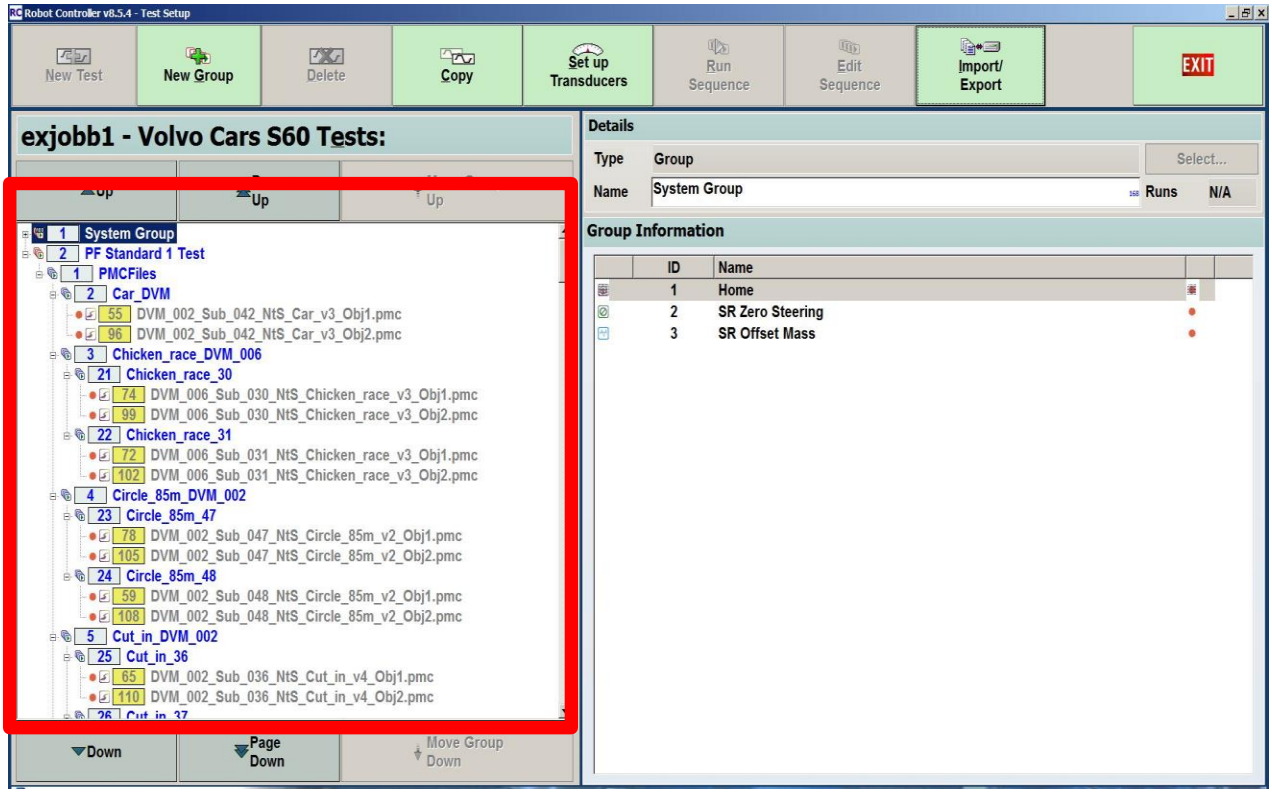
1.1 Syfte

När ett körtest med flera olika körfiler skall utföras vill man säkerställa att det inte kan uppstå några fel i konfigurationsfasen. När körfiler ska sättas upp manuellt kan den mänskliga faktorn spela in och det missas att skriva rätt inställningar till programmet. Detta gör i sin tur att det testscenario som körs på testbanan inte blir som planerat. Kostnaderna för att göra tester på provbanan är i många fall höga och därför kan missar i konfigurationsfasen bli väldigt kostsamma. Figur 1 visar ett exempel på hur körfiler kan läggas in.



Figur 1, Till vänster ligger testet och till höger visas var filer kan läggas in.

När ett körprov bestående av flera olika test skall sättas upp kan det se ut enligt figur 2. Vid detta scenario finns risken att peka på fel fil samt att missa körfiler.



Figur 2, Visar hur det kan se ut när flera test är inkluderade.

Ytterligare ett problem med denna konfigurationsmetod är att den är väldigt tidskrävande och monoton. Utifrån ovan nämnda problem är syftet att skapa en programgenerator för roboten som utför vissa delar av konfigurationen för att minimera risken för fel. Genom att automatisera de önskade delarna av inställningarna försvinner mycket repetitivt arbete och tiden kan läggas på annat.

1.2 Mål

Målet med projektet är att skapa ett program som utifrån en EXG-fil och obestämt antal körfiler skapar en ny EXG-fil med samtliga körfiler i. Förklaring av vad en EXG-fil är finns under teknisk bakgrund på sidorna 5 och 6. Följande mål behöver uppfyllas för att nå ett lyckat resultat.

- ✓ Skapa kod som genererar önskat antal test utifrån ett test med önskade egenskaper. Koden skall utifrån en mapp med körfiler lägga in ett nytt test för varje körfil. Den skall hantera alla olika typer av test. För närvarande kan RC8 (förklaras på sidan 5) skapa 50 olika varianter av tester. Koden skall hantera övervakningsfunktionerna som finns i RC8 och för detta lägga in erforderliga egenskaper i varje test. Övervakning sker under ett test där olika parametrar kontrolleras. Den ska också hantera om det finns eventuella undermappar i mappen med körfilerna. Om det finns undermappar skall samma struktur efter generering synas i RC8.
- ✓ Skapa ett GUI.

- ✓ Implementera programkoden i koden för GUI.

1.3 Avgränsningar

Arbetet kommer endast utföras på programvaran för generatorn. Programvaran för simulationen och riggen kommer ej förändras. Fysiska egenskaper så som bandimensioner och accelerationer kommer ej tas i beaktande. Denna data finns i körfilerna vilka programkoden ej kontrollerar. Det kan finnas fall där körvägar utanför bandimensionerna av misstag används eller att accelerationen sker från 0 till 100 på en meter. Dessa problem kontrolleras inte i programvaran för generatorn.

2 Metod

Kapitlet beskriver de olika moment och arbetsuppgifter som skall utföras. Hur projektets utveckling planeras att ske från start till mål.

Arbetet med projektet påbörjas genom att läsa in sig i manualer som finns tillgängliga från robotleverantören och även få problemformulering tydligt förklarad från Volvo. Samtidigt lära känna robotprogrammet som hädanefter i rapporten kallas RC8 och finns beskrivet i teknisk bakgrund på sidan 5. Viktigt är att förstå hur grupper, testfiler samt inställningar skapas. Då RC8 har väldigt många olika inställningar samt olika funktioner och övervakningsverktyg fortsätter denna process under hela projektet.

Utifrån en första kännedom om RC8 påbörjas arbetet genom att problemen presenteras utförligt av Volvo Cars. Genom en grundlig förståelse för vad som behöver göras är en viktig del i arbetet att tolka källkoden i exg_d, exg_l och exg_u filerna (dessa filer ingår i EXG-filen och finns förklarade under teknisk bakgrund på sidorna 5 och 6). Även eventuella befintliga lösningar studeras.

För att förstå källkoden i de olika filerna, byts olika egenskaper ut i olika tester och exporteras till olika EXG filer för att jämföras. En textjämförare används för att lätt kunna detektera olikheter i filerna. Eftersom Winzip finns installerat på Volvodatorer används det för att öppna EXG-filen och anteckningar används för att öppna och redigera i exg_d, exg_l och exg_u. EXG-filen är en komprimerad fil där ändelsen är ändrad från zip till EXG detta gör att den går att öppna med Winzip men det finns även en mängd andra program som kan öppna denna fil.

Sedan testades att byta ut egenskaper i exg_d filen genom att använda programmet Anteckningar. Denna fil komprimeras sedan tillbaka in i exg-format för att importeras in i RC8. Detta utförs för att kontrollera att det går att ändra manuellt i textfilerna för att se ändringar i RC8. Efter denna kontroll tolkas källfilerna för att se mönster och gemensamma delar som finns i olika typer av testscenarion. Dessa mönster får ligga till grund för hur programkoden skapas och hur Robot Setup Generator byggs upp. Robot Setup Generator är alltså namnet på den planerade generatorn och benämns hädanefter som RSG i rapporten.

Programmeringen görs i Matlab. Att valet föll på just detta språk grundar sig i att de mötte företagets önskan och dess enkelhet gällande texteditering. Först testas alla

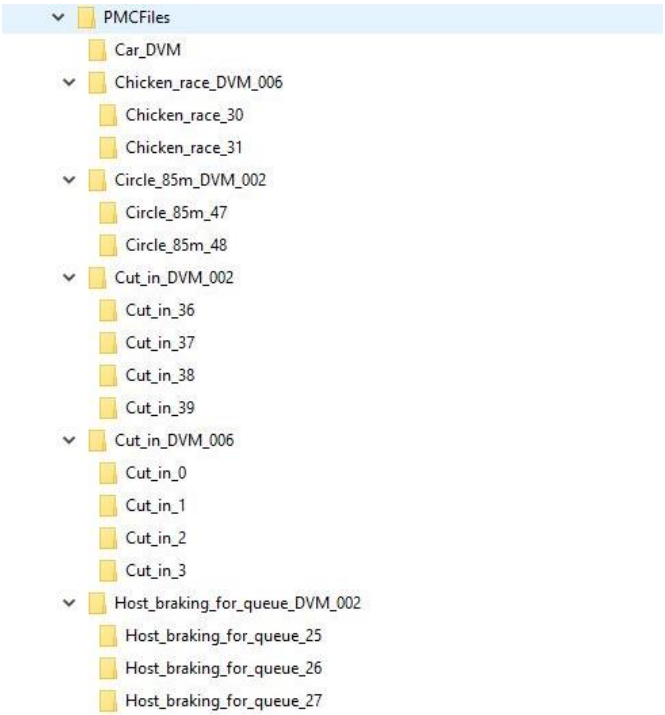
grundfunktioner för att kontrollera om de saker som ska göras är möjliga och tillräckliga. Grundfunktionerna som programmet ska innehålla är möjlighet att komprimera och packa upp, läsa och skriva till en textfil samt ändra namn i filändelsen.

Efter dessa grundfunktioner kontrollerats kan ett övergripande flödesschema skrivas. Från att ladda in EXG-filen till att skriva ut och komprimera en ny EXG-fil.

Från de tänkta grundfunktionerna samt flödesschemat börjar RSG skapas genom att kopiera och skriva rätt text i exg_d-filen. Utifrån tidigare tolkning av textfiler för olika tester kopieras rätt delar av textfilen. Eftersom det finns 50 olika tester i RC8 kopieras vissa detaljer olika för dessa tester. Olika delar av de kopierade detaljerna skall också ändras beroende på vilket av dessa test som använts. Dessa detaljer är viktiga bland annat för att koppla rätt test till rätt körfil. En exg_d-fil med endast ett test i består som minst av 70 000 tecken och då måste RSG kunna identifiera de delar som behövs för varje test och multiplicera detta med antal körfiler. Men också ändra i dessa delar så de kopplas till olika test.

Nästa del i programmeringen är att implementera kod för att hantera och skriva ut rätt text i exg_l-filen. Denna fil fungerar likt en pekare där sökväg, testid:n och filtyp måste representeras. För varje körtest som ska finnas skall det också vara representerat i exg_l-filen för att RC8 skall kunna identifiera det som ett test.

Programmet skall också hantera mappsystem för körfilerna där undermappar måste hanteras på ett korrekt sätt. Beroende på hur dessa mappstrukturer ser ut skall RSG hantera och kunna skapa testgrupper efter undermappar för implementering i RC8. I (figur 3) finns ett exempel på hur en mappstruktur skulle kunna se ut. I detta exempel skall RSG skapa testgrupper i flera nivåer. Först en huvudgrupp PMCFiles sedan skapas testgrupper i den ordning som mapparna ligger i filsystemet. Denna struktur skall sedan finnas i RC8 efter importering av EXG-filen.



Figur 3, Exempel mappstruktur, under varje mapp i denna struktur finns körfiler. Dessa körfiler skall organiseras under en testgrupp med samma namn som sin mapp.

När programflödet är klart skall ett GUI skapas. Det skall vara utformat så att användaren kan peka på en mapp med körfiler och peka på en EXG-fil. Sedan skall programmet jobba för att skapa korrekta exg_d samt exg_l filer och slutligen samla ihop alla filer till en EXG-fil där alla körfiler från mappen finns med och fungerar som önskat.

3 Teknisk bakgrund

Detta avsnitt avser att beskriva de olika tekniska komponenter och begrepp som används i rapporten.

3.1 AB Dynamics RC v8.04

En programvara som styr och kontrollerar AB Dynamics olika robotar. Den kan styra en styrrobot, bromsrobot, accelerationsrobot m.m. Men kan också styra dessa olika robotar i kombination och göra kombinationstester [1]. Programmet kan sätta upp olika scenarion, övervaka dessa under tiden som de körs och logga resultat samt olika mätdata. AB Dynamics RC v8.04 förkortas i rapporten som RC8.

3.2 EXG-fil

EXG-filen kommer ifrån RC8. Denna fil skapas när RC8 exporterar en grupp av tester för att till exempel lägga över testerna till en robot eller annan dator. Den består av tre XML-filer exg_d, exg_l och exg_u. XML är standarden för att ge dokument innehållsstruktur [2].

Exg_d är en textfil som innehåller alla olika typer av inställningar som ställts in i RC8. Inställningar innehåller av bl.a. vilken grupp eller undergrupp testet tillhör, alla egenskaper som testet skall innehålla och vilka övervakningar som skall ske under testet.

Exg_l innehåller deklarationer till de olika körfiler som finns i testerna. För varje körfil finns det kolumner som visar vilket fordon som används, vilket testid som är kopplat till de olika testerna, sökväg, filnamn på körfilerna och en siffra för vilken typ av fil det är. Beroende på hur körfilerna används i RC8 ges de olika siffror. T.ex. om de används som PMCReturn som är en körväg tillbaka från ett test får den siffran 2. Eller PMCInitial som är en initierande körväg tills ett villkor uppfylls till exempel en knapptryckning tillskriver RC8 den siffran 6.

Exg_u innehåller en lista över användardefinierade körnamn för att länka till fordon/ID. Ointressant för RSG.

3.3 Körfiler

Körfiler är de filer som beskriver körvägen för roboten. Utifrån ett lokalt koordinatsystem läggs en körväg upp genom antingen RC8 eller genom ett simuleringsprogram. Körfilerna som skapas i simuleringsprogrammet blir filer med ändelsen pmc och när körfiler genereras med RC8 fås två filer med ändelserna pmc och tem. Finns det tem filer använder RC8 sig av dessa då den processar dem snabbare.

3.4 Matlab

Matlab är ett högnivåspråk för datavisualisering, algoritmutveckling och matematiska kalkyleringar [3]. Det finns inbyggda toolboxar som är till för exempelvis att skapa GUI:n och fristående program.

3.5 Matlab GUI

Verktyg för att skapa användargränssnitt i Matlab. Här går det att skapa olika knappar, textboxar, kontrollfönster o.s.v. allt enligt personligt tycke [4].

3.6 Deploytool

Detta verktyg gör det möjligt för användaren att kompilera olika program och funktioner för att skapa fristående applikationer [5].

3.7 Matlab Runtime

Matlab Runtime är ett fristående Matlabprogram som gör att program som är kompilerade i Matlab går att köra på datorer utan Matlablicens [6]. Användare måste ha den version av Matlab Runtime som är kopplad till den Matlab-version som de kompilerade programmet är skapat i. Matlab Runtime har till skillnad från Matlab inget användargränssnitt. Det går heller inte att ändra i Matlabkod i Matlab Runtime utan detta kan endast ske genom att använda Matlab.

3.8 Winzip

Winzip är ett program som används för att packa upp och komprimera en eller flera filer [7]. Det kan jobba med många olika typer av filer t.ex. zip, rar och bz2.

4 Genomförande

Kapitlet beskriver projektets tillvägagångssätt från start till mål. Här tas även samtliga problem och deras lösningar upp.

Uppstarten fungerade så att en genomgång av företagets verksamhet beskrevs för att ge en bakgrund till projektets syfte även manualer till RC8 studerades. De gav ge en förståelse om projektets avsikt, fördelar och krav. En introduktion till de verktyg som företaget använder gav en grundläggande insikt i hur det specifika arbetet med RC8 fungerade.

Förstudier för att försöka ta reda på om det fanns några befintliga lösningar på problemet gav inga resultat. Varken simuleringsmiljön som genererar körfiler eller RC8 har öppen källkod. Det blev därför snabbt klart att egen programvara skulle utvecklas.

4.1 Tolkning

Viktigt var att bekanta sig med programmet för att kunna konstruera körfiler och grupper med olika antal test i. Detta skedde genom manualläsning, tester och kunskap från företagets anställda.

4.1.1 Tolkning av programmets två huvudfiler

När en grupp exporteras komprimerar RC8 denna till en EXG-fil, det är en fil av zip-format med en annan ändelse. En Zip-fil utnyttjar sig av icke förstörande komprimering för att minska storleken på desamma [8]. När EXG-filen packas upp innehåller den minst tre filer exg_d, exg_l och exg_u. Beroende på hur många körfiler gruppen innehåller, ligger här även pmc- och tem-filer. Filen som namnges exg_u har ingen betydelse för RSG och kommer ej tas i beaktande. Ovanstående kontrollerades genom att vid upprepade försök skapa grupper med olika antal test för att studera avvikelser i EXG-filen. Därigenom kunde fastslås att antal test var korrelerat med antal pmc- och tem-filer i EXG-filen.

4.1.1.1 exg_d

Exg_d filen är den största filen RC8 genererar och definierar testets struktur och samtliga inställningar. Genom tester med olika inställningar kunde skillnader i koden detekteras. Det testades att ändras manuellt i exg_d-filen för att se ändringar i RC8 efter inläsning. Det bekräftade antagandet om att inställningarna behandlades i exg_d-filen.

Därefter utfördes kontroller om hur exg_d-filen påverkas när en gruppkonfiguration med fler än ett test skapas. Förändringarna jämfördes med en grupp som bara innehöll ett test. Detta gjorde det tydligt med vilka delar av exg_d-filen som förändrades och vilka som ej gjorde det. Fler kontroller med varierande grupper testades och ett mönster om hur flera test påverkar exg_d-filen kunde bestämmas. Det avgjorde även under exakt vilka rubriker som förändringarna skedde.

Koden i exg_d-filen är indelad i stycken under rubriker, det var av vikt att finna lämpliga ställen för att navigera i koden. Vid studier av koden blev det klart att etiketten ROWDATA skulle vara lämplig, för att hitta mellan stycken i koden där förändringar har

skett. Fyrtio stycken ROWDATA finns i alla typer av test, de hjälper till at hitta tydliga avgränsningar i koden där just förändringar sker när fler test görs.

Det finns flera variabler i koden som måste tas i beaktande. De variablerna kopplar körfiler till rätt test samt namnger och anger testets struktur (Bilaga B).

4.1.1.2 exg_l

Filen med ändelsen exg_l är en lista med pekare som anger den lokala destiantionen till varje körfil. Det visade sig att exg_l- filen var tom när inga körfiler var kopplade till gruppen, men när körfiler var kopplade till en grupp skrevs kod till exg_l-filen. Det konstaterades att koden som skrevs i exg_l-filen var pekare till körfilerna.

Senare visade det sig att antagandet om att de bara var en eller flera pekare i denna fil inte stämde, ett felmeddelande uppstod vid inläsning i RC8. De framkom att det var nödvändigt att koppla rätt körfil och rätt fordon till rätt testsetup i exg_d-filen.

När olika typer av körfiler används ges olika suffix till pekaren i filen. Används pmc, tem, PMCReturn eller PMCInitial måste rätt ändelse kopplas till rätt körfil. Ändelsen antar ett siffervärde.

4.1.2 Kombinationstester

I RC8 finns det möjlighet att köra över 50 olika tester. Samtliga analyserades för att hitta skillnader. Alla tester var i huvudsak lika förutom några test nämligen kombinationstesterna. Ett kombinationstest kan ha flera olika test i ett test, därför skiljer sig koden drastiskt gentemot övriga test.

4.2 Programmering

Programmeringen inleddes genom att kontrollera de funktioner som planerades att användas i Matlab. Dessa funktioner inkluderade möjligheten att packa upp en fil, kopiera och skriva i en textfil, komprimera en fil samt att ändra den komprimerade filens ändelse till EXG. Ändelsen behövde ändras då RC8 bara läser in filer med ändelsen EXG.

4.2.1 Inläsning och mappgenomgång

RSG:s grundfunktion utgörs av att välja ett mappsystem innehållande körfiler, välja EXG-fil och slutligen en slutmapp. Det var därmed viktigt att tidigt få de funktionerna att fungera för att lägga grund till RSG:s utformning.

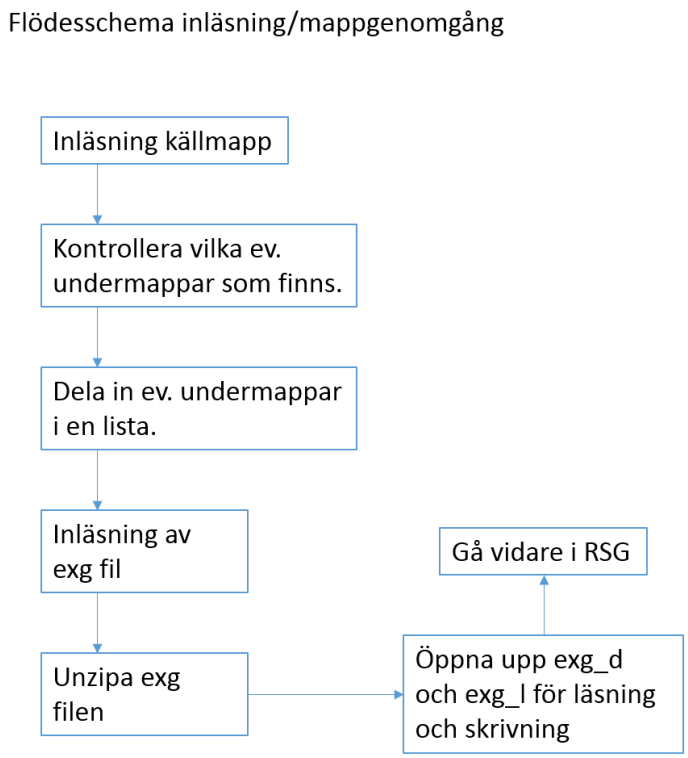
4.2.1.1 Inläsning

För inläsningen av rätt mappar och EXG-fil användes inbyggda funktioner i Matlab, `uigetdir` respektive `uigetfile`. Båda funktionerna kallar på ett popupfönster där man kan välja mapp eller fil likt ett vanligt Windows program. EXG-filen packas upp och sedan tillåts skrivtillstånd.

4.2.1.2 Mappgenomgång

För att RSG ska kunna veta hur den skall bygga upp de önskade filerna var programmet tvunget att studera den källmapp där körfilerna ligger med undermappar. Utifrån hur

denna mapp ser ut kan programmet lägga rätt filer under rätt mappar och skapa rätt struktur där önskade filer är kopplade till varandra. Se figur 4.



Figur 4, flödesschema inläsning/mappgenomgång.

4.2.1.3 Destinationsmapp

När RSG arbetat klart med EXG-filen läggs denna i en mapp som användaren väljer. Detta görs på samma sätt som när inläsning av mapp sker. `uigetdir` lägger en sökväg till en variabel. För att kontrollera att det inte finns filer med samma namn som på den skapade filen läggs allt innehåll i den tänkta destionationsmappen till en variabel. Denna variabel jämförs sedan med namnet som valts. Finns det en fil med samma namn kommer en varningsruta upp. Om ersättning av filen önskas så går RSG vidare annars avbryts exekveringen och nytt namn kan väljas.

4.2.2 Identifiering och extrahering

Det är viktigt att identifiera flertalet variabler och plocka ut textsträngar ur exg_d-filen som skall användas för att skapa nya test.

4.2.2.1 Identifiering av variabler

Samtliga variabler är antingen namn eller tal och kan bland annat definiera om det är ett kombinationstest som körs eller ej. Detta detekteras genom att söka i exg_d-filen med ord som ligger positionen framför och positionen efter variabeln. En Matlab funktion vars namn är `extractBetween` används för att hitta strängen mellan de valda positionerna. Representerar strängen ett tal konverteras denna till en double genom funktionen `str2double`. Det är för att sedan kunna ändra variabelns värde, det är nödvändigt när nya test skall göras.

4.2.2.2 Extrahera alla originalsträngar

Från den ursprungliga EXG-filen extraheras de områden i exg_d-filen som förändras när fler test läggs till. För att säkerställa att samma inställningar alltid kommer med tas delar av koden ut och sparas i en sträng. Sedan raderas koden i den ursprungliga exg_d-filen för att sedan ersättas med ny kod för respektive test.

4.2.3 SUBGROUPS

För att definiera subgrupper måste samtliga valda körfilsmappar genomsökas, för att kunna avgöra vilka körfiler som skulle vara kopplade till vilka grupper. Genom att söka igenom ett mappsystem skall RSG konstruera ett system med subgrupper identiskt med det valda mappsystemet.

En del komplikationer uppstod i denna fas av arbetet. Svårigheterna låg i att få rätt mapp att kopplas till rätt subgrupp. Sedermera valdes en lösning som läste igenom två mappar samtidigt, den aktuella mappen och den eventuella undermappen till den aktuella. Då var det möjligt för RSG att avgöra om den aktuella mappen hade ytterligare en undermapp eller om den var på den lägsta mappnivån. På så vis kunde RSG avgöra vilken mapp som hade vilka undermappar och kunde då avgöra hur subgruppsstrukturen skulle te sig.

4.2.4 Genomgång av körfilsmappens innehåll

I genomgången av körfilmappens innehåll sker en inläsning av mappens innehåll och även all skrivning av kod till filerna och kopiering av körfiler. Inläsningen repeteras så länge det finns mappar kvar att genomsöka i mappsystemet. Samtliga underrubriker till denna rubrik omfattas av denna repetition.

4.2.4.1 Kopiera körfiler

En tillfällig mapp som kallas Robot_Setup_Generator skapas, körfiler kopieras sedan till denna. Funktionen `copyfile` användes och kopierar samtliga filer i varje mapp med ändelserna pmc och tem till den skapade mappen. När programmet sedan är helt klart raderas sedan hela mappen.

4.2.4.2 Exg_l-fil

I varje mapp kontrolleras samtliga körfiler för att sedan skrivas ut som pekare i exg_l-filen. Viktigt här var att få funktionen att koppla rätt körfil till rätt test.

Enkel kod skrevs först för funktionen men blev senare ofullständig då tem- och pmc-filer inte alltid kom i par. Tidiga gjorda antaganden utgick från att det endast kunde vara tem- och pmc-filer i par vilket var fel. Pmc-filer kan komma ensamma men också körfiler såsom PMCReturn eller PMCInitial finnas. De två senare filerna kan kopplas till ett och samma körprov. För att skilja de filerna åt från vanliga körfiler ges de ändelsen _Return respektive _TrigIp. Funktionen fick skrivas om utifrån de ändrade förutsättningarna.

När ett kombinationstest har identifierats krävs det stora förändringar i funktionen. Då koden skiljde sig så mycket var det nödvändigt att verkligen tolka koden för kombinationstester ordentligt. En separat funktion fick utvecklas för varje gång ett kombinationstest körs. En del problem uppstod under detta moment med att koppla

rätt körfil till rätt test. Detta löstes senare med att införa fler villkor för att vara säker på att rätt körfil kopplas till rätt test.

4.2.4.3 Exg_d-fil

Under all skrivning till exg_d-filen så upprepas samtliga segment så länge de finns fler filer i den aktuella mappen. I de stycken där koden ändras för varje nytt test, måste kod skrivas till varje stycke för varje nytt test.

Som start beslöts att skriva ett testprogram för att få kopieringen att fungera ordentligt. Problem uppstod då variabler som var beroende av testets utformning fick samma värde i samtliga test när stycken kod bara hade kopierats rakt av. RC8 hittade då ej rätt körfiler kopplade till testet. Det visade sig då viktigt att kunna byta värden på variablerna, därför fick de identifieras för att sedan kunna räknas upp. Därför sattes det in olika räknare som gavs möjlighet att räkna upp när vissa villkor uppfylls. Det löste de inläsningsproblem som RC8 tidigare visade. Senare gjordes även funktioner som namnsatte varje test med den specifika testfilens namn.

När ett kombinationstest körs måste här utvecklas separata funktioner för att handskas med de stora skillnaderna gentemot ett vanligt test. Speciella villkor för skrivning till exg_d-filen avgör hur testen skall kopplas ihop med varandra.

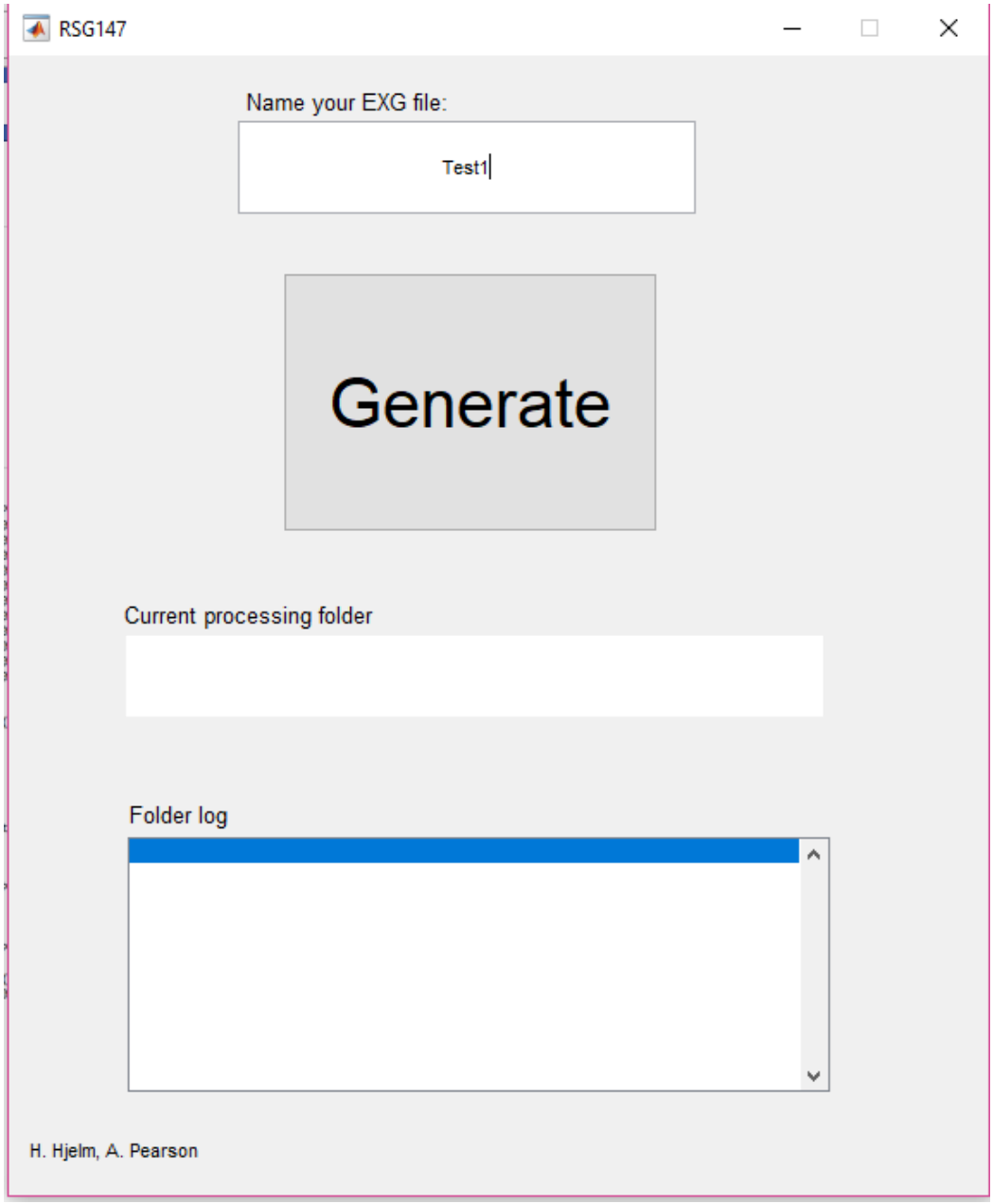
4.2.5 Paketera

Slutligen skall samtliga filer paketeras som en EXG-fil. För att packa filerna användes Matlabs inbyggda kommando `zip`. Detta kommando används för att komprimera de filer som önskas. Genom att tidigare i programmet kopiera över alla önskade filer till samma mapp tar `zip`-funktionen filerna och komprimerar dessa till en zip-fil med ändelsen EXG. Även den ursprungliga EXG-filen med dess modifierade filer exg_d, exg_l och exg_u följer med i zip-filen.

Eftersom RC8 endast läser in filer med ändelsen EXG ändras ändelsen genom funktionen `movefile`. `movefile` tog namnet från den komprimerade filen och ändrade den till önskat namn med ändelsen EXG. Efter att `zip`-funktionen utförts raderades alla överflödiga filer med funktionen `delete` för att slippa onödiga filer på datorns hårddisk.

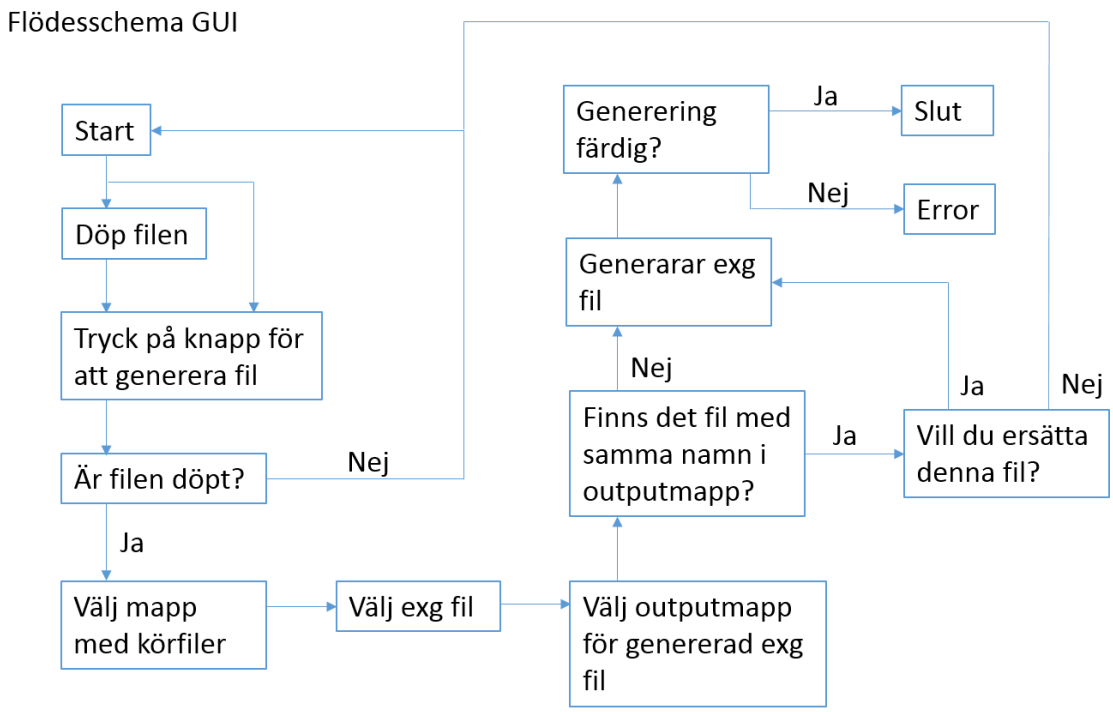
4.2.6 GUI

Till ett fristående program behövdes ett GUI utvecklas. Inbyggt i Matlab finns ett verktyg som heter Matlab GUI som användes för detta ändamål. I Matlab GUI valdes att ha ett textfält för att kunna döpa den genererade filen till önskat namn och sedan en knapp för att generera filen. Det lades även in två fält där ett av dessa visar vilken mapp RSG arbetar i och det andra fältet visar en logg där alla mappar som arbetats igenom går att se. Det slutliga utseendet visas i figur 5.



Figur 5, GUI, programmets slutgiltiga utförande.

För att kombinera den tidigare programmerade koden med GUI:t genererades ett nytt Matlabprogram där egenskaperna för GUI:t fanns. I denna kod lades sedan den tidigare koden in under funktionen för knappen. Alltså exekveras denna kod då användaren trycker på knappen. Vidare programmerades det in funktioner för att kontrollera att användaren lagt till ett namn i textfältet. Samt att varningsruta kommer upp om det finns en fil med samma namn i destinationsmapp. Beroende på hur källfilen ser ut lades det också till felfunktioner med pop-up fönster för att meddela att det inte fungerar i RSG. Hur funktionen för GUI:t ser ut går att utläsa i figur 6.



Figur 6, Flödesschema GUI.

4.2.7 Fristående program

Genom att skapa en fristående app kan RSG distribueras till de personer som har nytta av det. För att skapa denna app användes Deploytool. Genom detta verktyg lades alla nödvändiga filer ihop till en app som gick att installera på andra datorer. Eftersom Matlab Runtime behövs vid körning av RSG på datorer som inte har Matlab eller har fel version av Matlab inkluderades detta i RSG:s programpaket.

4.3 Verifiering

För att säkerställa RSG:s funktion var en övergripande verifiering nödvändig. Verifieringen utfördes löpande under projektets gång, för att i stunden avgöra om RSG fungerade som det var tänkt. När RSG ansågs vara klar utfördes en slutgiltig verifiering med en rad olika körprov för att garantera funktionen för RSG:n. Verifiering fungerade på det sättet att företaget hade skapat olika tester som programmet var tvunget att hantera.

När denna var utförd kunde RSG sedan utlämnas till företaget för användning. För att säkerställa att RSG tar med alla önskade egenskaper och utför önskade uppgifter korrekt hölls kontinuerlig kontakt med AB Dynamics.

5 Resultat

De slutgiltliga Resultaten av examensarbetet blev följande:

- Fristående program för att skapa en grupp med flera olika körfiler har tagits fram.
- Programmet läser igenom vald mappstruktur.
- GUI har skapats för programmet.

•Tester för att garantera ovanstående har gjorts genom att använda samtliga olika utformningar på tester. Handledare från Volvo har även testat och godkänt funktionen.

6 Slutstats/diskussion

Projektet får anses som färdigställt, då målen är uppfyllda och RSG uppfyller samtliga krav. RSG kan även hantera ytterligare två körfiler, PMCReturn och PMCInitial. Genomförandet har fortlöpt enligt angiven tidsplan(se bilaga C). Verifiering har skett under hela programmeringsfasen för att kontrollera önskad funktion. Under projektets gång har nya problem tillkommit och ytterligare tolkning blivit nödvändig. En slutgiltig verifiering utfördes med blandade test för att säkerställa RSG:s funktion. RSG utlämnades sedan till företaget för användning.

6.1 Kritisk diskussion

En begränsning som finns i programmet är att varje grupp som RSG genererar är av samma testtyp. Det finns alltså ingen möjlighet till att generera en grupp med flera testtyper än den som är "grundprovet".

När arbetet startade fanns det många angreppspunkter för att tackla problemen. Beslut togs att nästintill omgående påbörja programmeringen, detta trots en otillräcklig kunskap om hur RC8 fungerade. Det mynnade ut i att det var nödvändigt att flera gånger tolka koden på nytt för att lösa de nya problem som uppstod. I efterhand skulle tid sparas på att från början göra en grundlig analys av allt som skulle göras, och hur det skulle angripas.

6.2 Vidareutveckling

Det skulle vara möjligt att genom att endast peka på en mapp med körfiler, vilka är namnsatta med en viss ändelse för att kunna bestämma vilket test som just den körfilen skall ha. Det skulle innebära att programmet skulle kunna på helt egen hand ta hand om körfiler utan att man behöver sätta upp ett "grundprov". Nackdelar med denna lösning är att man sedan själv får sätta upp vilka inställningar man vill ha. RSG blir även känsligt för uppdateringar i RC8, då RSG i sådana fall måste innehålla strängar från nuvarande version. Med nuvarande lösning är RSG mer kompatibelt med uppdateringar i RC8, även finns det möjlighet att ge rätt inställningar.

Ett program likt RSG skulle kunna användas av flera företag än just Volvo. ABD som tillverkar alla riggar/robotar samt det mesta av utrustningen som Volvo använder sig av levererar även mycket till andra bilföretag. Alla de 25 största biltillverkarna i världen använder ABDs produkter [9]. Programvaran som ABD levererar ser i stort sätt likandan ut för alla bilföretag därför skulle ett generellt program kopplat till RC8 kunna skrivas och distruberas av ABD. Genom detta skulle en fortsatt utveckling för RSG kunna vara att implementera verktyget I RC8. Som det ser ut I dagens läge är RSG endast utvecklat för Volvo men genom att istället arbeta direkt med ABD skulle en plugin kunna göras. Denna kan sedan inkluderas I programpaketet för RC8 för att användas av alla kunder till ABD om så önskas.

Problemet med att koppla RSG till ABDs RC8 är att RSG är utvecklad för att koppla filer mellan Volvos simuleringsmiljö och RC8. Denna lösning har endast efterfrågats av Volvo

och inte någon av ABDs andra stora kunder. Så med detta kommer frågan om andra biltillverkare har problemet att överföra filer från simuleringsmiljöer till RC8? Antingen så finns inte problemet hos andra tillverkare eller så har dem inte funderat på om det finns smidagare sätt att kunfigurera sina testningar. Det finns en övervägande chans att andra billtilverkare har en simuleringsmiljö som inte genererar några körfiler. Därför finns inte ett behov av RSG. Men in framtiden när dessa tillverkare utvecklar en funktion för att generera körfiler till RC8 finns möjligheten att RSG kommer till användning även för dessa tillverkare. Volvo anser att en produkt som RSG hjälper dem att bibehålla en ledande position inom säkerhet. Om problemet skulle uppkomma hos andra tillverkare så är självklart RSG en produkt som skulle kunna anpassas till en bredare användning.

ABD levererar även produkter och tjänster som används inom andra områden. Kvalitetskontroll är en av dessa tjänster där det kontrolleras hur bilen klarar vibrationer samt oväsen [10]. Dem kan då leverara allt från test robot till sensorer och program som presenterar resultat. När test konfigureras i dessa typer av robotar används inte RC8. Front-end och back-end ser liknande ut som RC8 men det har annan funktion. Det går därför inte att använda RSG då programmet är skrivet mot hur RC8 skriver ut xml kod. RC8 är också specifikt skapat för att användas mot alla typer av körrobotar/riggar samt målbärare. Detta gör att även RSG blir specifiserat mot dessa körrobotar/riggar samt målbärare.

ABD har I dagens läge ett antal konkurrenter som fokuserar på olika delar. ASI är en av dessa och har störst fokus på automatisering. Företaget har flera olika områden som dem koncentrerar sig på bland annat gruvindustri, lantbruk, bilindustri och militärbruk [11]. Programvaran som motsvarar RC8 kallas Modulus och där konfiguraras tester som skall köras [12]. Utifrån informationen på ASIs hemsida har dem ingen simulationsmiljö kopplat till Modulus. Beroende på om det går att importera filer till Modulus kan RSG ha en funktion. Men RSG måste dock skrivas om för att koppla ihop Volvos simuleringsmiljö till Modulus.

Det andra kunkurrerande företaget är Vehico som riktar sig mot bilindustrin [13]. Detta företag verkar inte heller ha någon simuleringsmiljö. Återigen så måste alltså deras program och olika filer tolkas för att se om en lösning likt RSG går att implementera. Ett tredje konkurrerande företag DSD som har störst fokus på aktiv säkerhet [14]. DSD har ett program som inte fungerar på samma sätt som RC8. Deras programvara är mer inriktat på hur krockar ser ut och hur dem analyseras. Vilket leder till att RSG inte är applicerbart på denna programvara.

6.3 Hållbarhet och etik

Ur ett hållbarhetsperspektiv får RSG ses som att den ej skadar miljön i någon utsträckning. Ingen materialåtgång krävdes under utveckling eller under användning av RSG. RSG är tidssparande för användaren vilket kan bidra till att datorn används mindre och energibesparingar är ett faktum. Rent etiskt finns det inga nackdelar med att använda RSG, då repetitivt arbete begränsas kan detta möjligtvis bidra till en bättre arbetssituation för företagets anställda.

7 Referenser

[1] Anthony Best Dynamics Ltd. AB Dynamics RC Software Manual [e-bok]. Wiltshire: 2017.

[2] XML. Malmö: Nationalencyklopedin; 2018 [citerad 7 juni 2018] Tillgänglig från: https://www.ne.se/uppslagsverk/encyklopedi/l%C3%A5ng/xml

[3] MATLAB Product Description. Natick: The MathWorks Inc; 2018 [citerad 16 maj 2018]. Tillgänglig från: https://se.mathworks.com/help/matlab/learn_matlab/product-description.html

[4] MATLAB GUI. Natick: The MathWorks Inc; 2018 [citerad 16 maj 2018]. Tillgänglig från: https://se.mathworks.com/discovery/matlab-gui.html

[5] MATLAB Deploy tool. Natick: The MathWorks Inc; 2018 [citerad 16 maj 2018]. Tillgänglig från: https://se.mathworks.com/help/compiler/deploytool.html?s_tid=doc_ta

[6] MATLAB Runtime. Natick: The MathWorks Inc; 2018 [citerad 16 maj 2018]. Tillgänglig från: https://www.mathworks.com/help//compiler/about-the-matlab-runtime.html

[7] WinZip. Mansfield: WinZip Computing; 2018 [citerad 18 maj 2018]. Tillgänglig från: http://www.winzip.com/win/se/prodpagewz.html

[8] Kodek. Malmö: Nationalencyklopedin; 2018 [citerad 7 juni 2018] Tillgänglig från: https://www.ne.se/uppslagsverk/encyklopedi/l%C3%A5ng/kodek

[9]AB Dynamics. Bradford on Avon. Anthony Best Dynamics Ltd; 2018 [citerad 10 Juli 2018]. Tillgänglig från: https://www.abdynamics.com/

[10]PLATO. Bradford on Avon: Anthony Best Dynamics Ltd; 2018 [citerad 10 Juli 2018]. Tillgänglig från: https://www.abdynamics.com/en/products/nvh-test-systems/product-quality-testing-plato

[11]ASI Robots. West Mendon: Autonomous Solutions, Inc.; 2018 [citerad 10 Juli 2018]. Tillgänglig från: https://www.asirobots.com/

[12] Automotive. 990 North 8000 West Mendon, UT 84325: Autonomous Solutions, Inc.; 2018 [citerad 10 Juli 2018]. Tillgänglig från: https://www.asirobots.com/automotive/

[13] Vehico. Braunschweig: VEHICO GmbH; 2018 [citerad 10 Juli 2018]. Tillgänglig från: http://www.vehico.com/index.php/en/

[14] DSD. Linz: Dr. Steffan Datentechnik Ges.m.b.H; 2018 [citerad 10 Juli 2018]. Tillgänglig från: http://www.dsd.at/index.php?lang=en

8 Bilagor

A. Manual

Robot Setup Generator

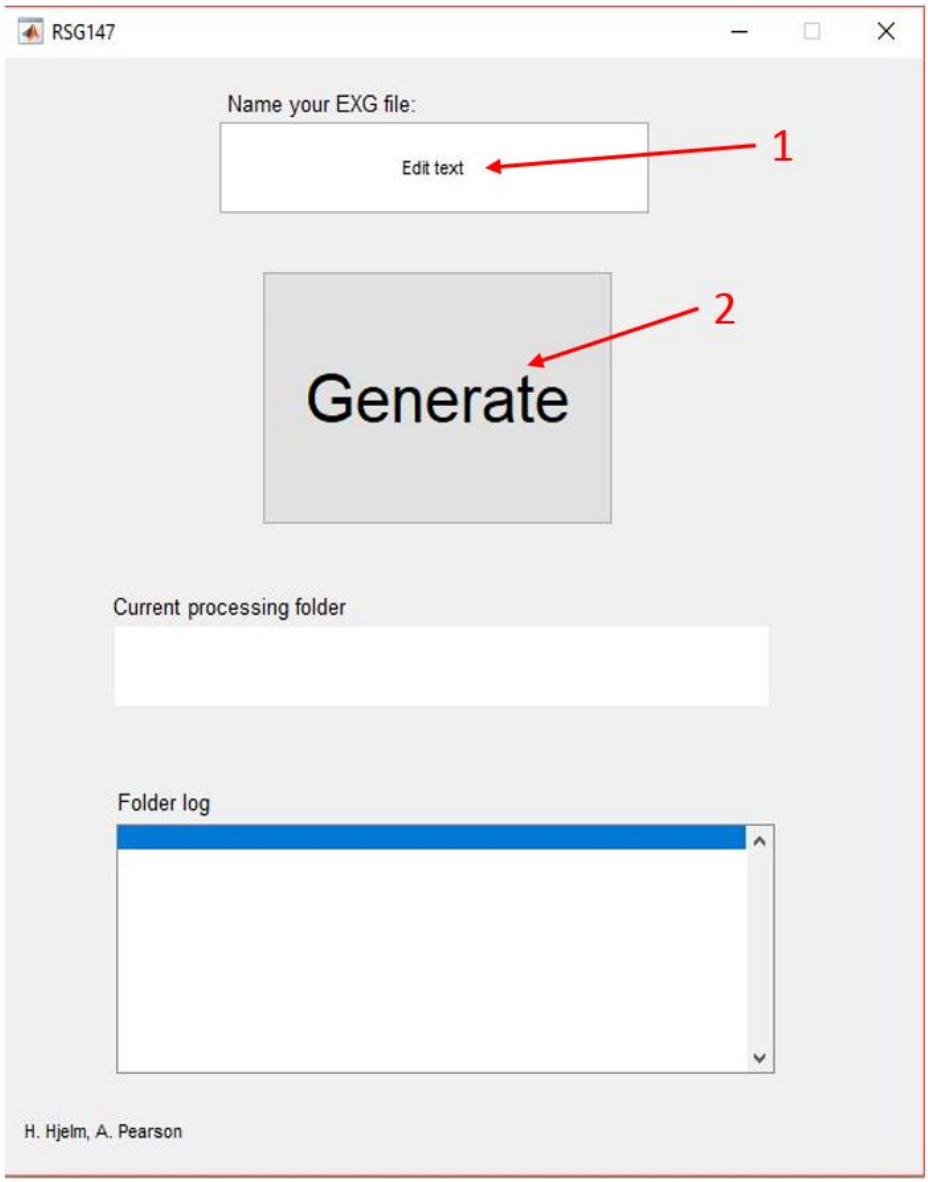
Manual

This is a manual for usage of Robot Setup Generator. The manual will provide information about the steps you need to generate an EXG file and give you information about what you can do in the program.

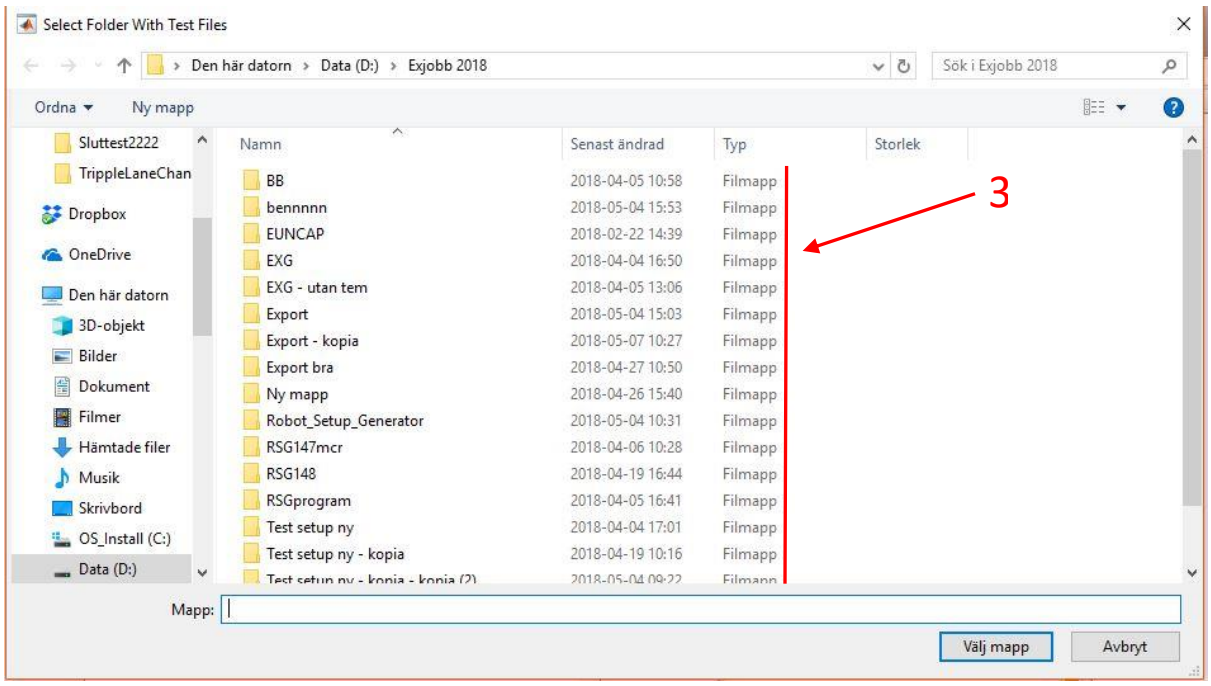
Clarifications/Explanations/Limitations
During the generating process you can follow what folder the programme is processing in the window “Current processing folder”. After and during the generating process you can also see what folders the programme has processed in the folder log.
If you want to generate a SR/BR/AR test you cannot copy previous tests when you create your original EXG. You have to create it from scratch otherwise you will get an error message.
If you want to have tests with return paths you need to create a return path file for every run file. This return path file need to have the same name as the run file it is intended to connect to with an ending of _Return.
If you want to generate PF Standard Triggered Path tests with initial path files. You need to name these files the same as the run file that it is intended to connect to with an ending of _TrigIp.
You cannot create several tests in the original EXG file. If you do this the generated EXG file will be broken. Combination tests do work apart from Initial paths in the test PF Standard Triggered Path.

Instructions

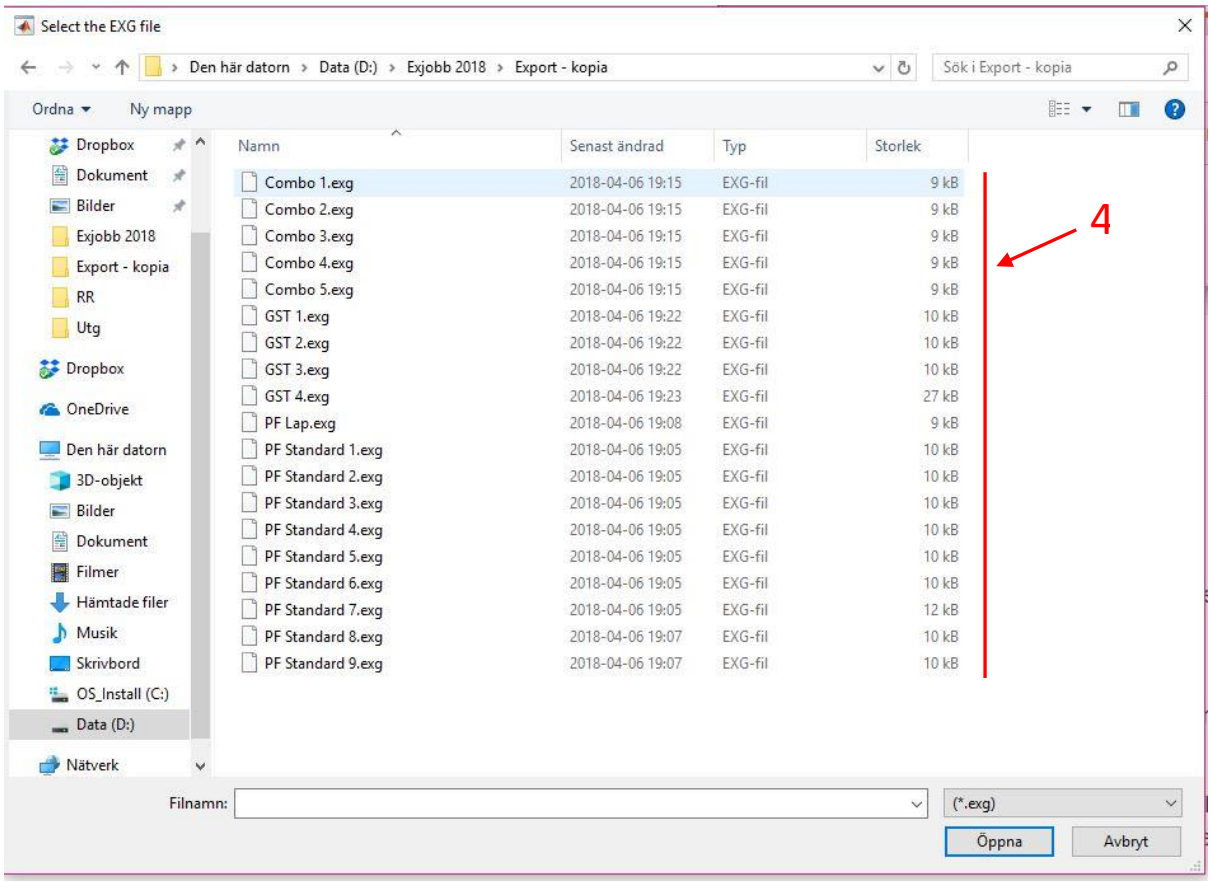
1. Name your file.
2. Push Generate button.



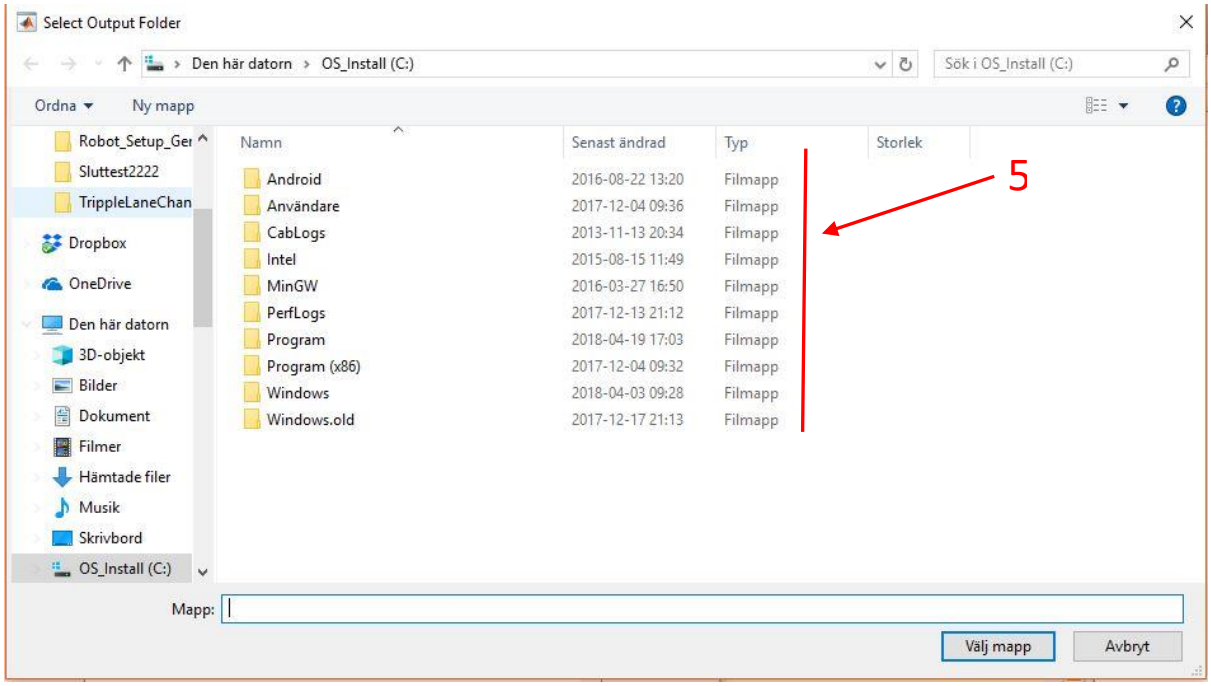
3. Choose folder with run files.



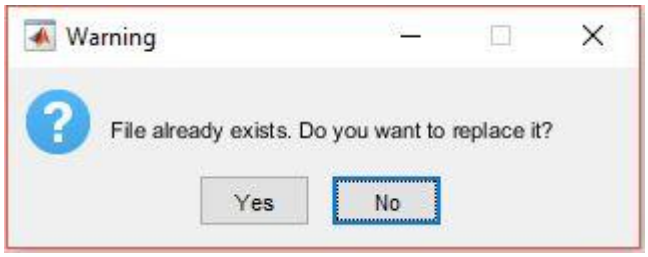
4. Choose your EXG file.



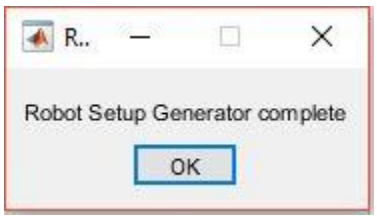
5. Choose output folder.



6. If the file name you have chosen already exists in the output file a warning pops up. Press no and return to point one. Press yes to replace previous file.



7. Complete



B. Struktur över EXG_d-filen

GROUPS
Rubriken GROUPS definierar hur gruppen skall se ut. Den enda viktiga parametern för projektet utgörs av variabeln VIEWGROUPLEVEL, vilken talar om ifall testet skall övervakas på gruppnivå eller ej. När den är tillslagen sker inga förändringar i REALTIMEDATA, utan hamnar istället i GROUPREALTIMEDATA. Därför är det viktigt att avgöra om grupp- setupen använder sig av View Groupchannel eller inte.

SUBGROUPS
SUBGROUPS redogör hur subgrupper i gruppen skall se ut. För varje ny subgrupp skapas en definition av den specifika subgruppen. Subgrupperna är definierade genom GROUPID, SUBGROUPID, SUBGROUPLISTID, SUBGROUPNAME och SUBGROUPPARENTID. GROUPID kopplar subgruppen till rätt grupp. SUBGROUPID ger en numerisk identifikation till den berörda subgruppen, utgör även möjlighet för vidare koppling till ytterligare subgrupper. SUBGROUPLISTID anger vilket nummer i RC8s lista som subgruppen tillges. SUBGROUPNAME ger namn åt respektive subgrupp. SUBGROUPPARENTID kopplar subgruppen till dess överliggande grupp. Skrivs till -1 om subgruppens överliggande grupp är huvudgruppen, annars kopplas densamma till överliggande subgrupps SUBGROUPID.

TESTSETUPS
I TESTSETUPS behandlas testernas inställningar och hur testerna är kopplade till varandra till exempel genom subgrupper och kombinationstest. För varje nytt test skapas en ny test setup för respektive test.

Innehållet i TESTSETUPS förändras beroende på vilket test som körs. De variabler som ändrades för varje test var ID, BR_SRTESTID, BR_BRTESTID, BR_ARTESTID, MAPOPTIONSID, VEHICLEID, NAME, SUBGROUPS och TESTFILE. ID ger en numerisk identifikation till det berörda testet. BR_SRTESTID, BR_BRTESTID och BR_ARTESTID används för att knyta samman olika test i kombinationstesterna. MAPOPTIONSID länkar till DISPLAYID i POSITIONDISPLAYSETUP. VEHICLEID avgör vilken bil testet är kopplad till. NAME ger namn åt testet i RC8. SUBGROUPS anger vilken nivå i gruppsystemet testet är bundet till, tillskrivs -1 om nivå är samma som huvudgruppen annars kopplas rätt SUBGROUPID. TESTFILE utgör en pekare för vilken körfil som är kopplad till vilket test. Övriga inställningar ändras ej för nya test och kommer därför ej studeras vidare.

REALTIMEDATA, GROUPREALTIMEDATA och POSITIONDISPLAYSETUPS
Under rubrikerna REALTIMEDATA, GROUPREALTIMEDATA och POSITIONDISPLAYSETUPS sker förändringar vid upprepade test, hur de sker avgör variabeln VIEWGROUPLEVEL. VIEWGROUPLEVEL definieras i rubriken GROUPS vilken tillskrivs antingen 0 eller 1. När VIEWGROUPLEVEL är tillslagen sker övervakningen av testet på gruppnivå. När övervakningen sker på gruppnivå definieras testets övervakning endast en gång i GROUPREALTIMEDATA, annars definieras övervakning för varje specifikt test under rubriken för REALTIMEDATA och POSITIONDISPLAYSETUPS. Här återfinnes även variablerna TESTID och VEHICLEID som måste kopplas till rätt test.

TESTTRIGGERS
Under ovanstående rubrik definieras de specifika testets inställningar gällande test triggers, för varje test skapas en ny definition för det testet under denna rubrik. Samtliga inställningar här kommer att förbli samma som satts upp för den ursprungliga EXG filen Även här är det nödvändigt att sammankoppla TESTID och VEHICLEID för respektive test.

TIMETOLERENCE
Detta avsnitt tillskrivs en ny definition kopplat till varje nytt test som gruppkonfiguration skall innehålla.

Tidplan Robot Setup Generator — Volvo Cars — Thesis