Carnetsoft/WvW/1-01-22

# Course Research simulator

## Contents

# 1. Overview

In this course we will learn to use the simulator to perform an experiment.

Subjects to be treated in this course:

1) Where can I find which files ? Folders, locations of important files and programs….

2) A number of ways to measure, store and analyze data

3) Scenario script

4) Database references from script: RoadDesign

5) Perform experiment: measure workload while driving while the subject performs an additional cognitively loading secondary task: countback from 200 to 0 : script + database-> data specs ->testing ->measurement ->analyse data

# 2. Locations of files, folders and programs

## 2.1 Documentation

Documentation can be found in two folders. The documentation files are generally pdf's.

### 2.1.1 General documentation

C:\DriveSim3\Carnetsoft\documentation

- CarnetManual.pdf: driving simulator general manual. Contains information about starting and stopping the simulator, selecting a simulation, use of config program IOConfig, etc.
- Info about special modules, for example the clinical tests and the safety awareness modules.
- Info about recommended settings and button assignments

### 2.1.2 Research simulator documentation

C:\DriveSim3\Carnetsoft\ResearchSim\Documentation

- Scenario.pdf: manual of the scenario script language. Very important for doing research.
- RoadDesign.pdf: manual of the road designer. This is for creating virtual environments (databases). Most researchers use one or more of the 15 existing databases, but if you want to know the paths or make modifications, you still need the road designer.
- ExperimentSpecification.pdf: Description of how to use ExpPrep.exe, to make experiment specification files and data definition files for data storage.
- DataStorage.pdf: manual on how to store data and the different methods for doing that.
- A few other courses and recommendations.

## 2.2 Executables/programs

Here’s a list of all important executables

| **Program** | **Function** |
|---|---|
| StartSim.bat | This starts the simulator via a batch file that starts all relevant executables. Its located in c:\DriveSim3\Carnetsoft and has a desktop icon that can be clicked to start the simulator. It does the following:<br>- Set the proper environment path<br>- Start traffic.exe (scenario interpreter and traffic generator<br>- Start control.exe (user control interface<br>- Start Left and Right rendering channel (ppython_left.exe)<br>- Start Center rendering channel (ppython_center.exe) mainC.py, and mainLR.py are located in c:\DriveSim3\Carnetsoft and call all other python code in the *.pyc files. All source python files can be found in \ResearchSim\python-files\. Users are free to modify the rendering by changing these python source code files, but this requires python programming skills and is on your own risk. |
| ppython_center.exe | Panda3D renderer, that renders the center channel (forward view), accessed via the python file mainC.py. Located in c:\DriveSim3\python. It also does the voice message rendering and 3D sounds |
| ppython_left.exe | Panda3D renderer, that renders the left  and right channels (left and right views), accessed via the python file mainLR.py. Located in c:\DriveSim3\python |
| control.exe | User interface control program that does all datacomms via ethernet port 127.0.0.1 and direct memory access with the rendering engine and traffic.exe. It also does the static sound rendering (engine, wind and road contact sound) and the direct X interface to steering wheel, pedals and buttons. Located in c:\DriveSim3\Carnetsoft\SimCarnet. |
| traffic.exe | Program that contains the script interpreter and the traffic simulation. Sends out all traffic positions and speeds etc, traffic light data, animation data each frame to the rendering engines. Also contains the engine and vehicle model of the simulator car and communicates data with control.exe. Located in c:\DriveSim3\Carnetsoft\SimCarnet. |
| IOConfig.exe | Config program for the simulator, reads and stores most files in the ..\files folder. See documentation simulator. Located in c:\DriveSim3\Carnetsoft\SimCarnet. |
| ExpPrep.exe | Program to create .exp (experiment) files. *.exp files can also be created manually via a text editor. Located in c:\DriveSim3\Carnetsoft\ResearchSim\ExpPrep\. |

| | |
|---|---|
| RoadToolVC.exe | Program to create and modify virtual environments (graphics databases). Referred to as RoadDesign. Located in c:\DriveSim3\Carnetsoft\ResearchSim\RoadDesign\. |
| Dataproc.exe | Program to visualize and analyze binary datafiles (*.da0 files). The binary datafiles are created during a simulation and contain a stream of variables that are sampled with a fixed frequency (typically 10 Hz). Located in c:\DriveSim3\Carnetsoft\ResearchSim\DataProc\. |
| TextPad.exe | Text editor to write scenario script source files (*.sci and *.scn). Also recommended as a text editor for other text files and for python code files. TextPad is also used to run the script compiler that does a syntactical analysis of the script code and compiles it into a binary script file (*.scb). The simulator reads binary script *.scb files (and also *.exp files). Located in c:\TextPad 4\ |
| ModuleConfig.exe | For driver training software. Not relevant for research simulator. |
| NewStudent.exe | For driver training software. Not relevant for research simulator. |
| pview.exe | To view an *.egg or *.bam graphics database. Located in c:\DriveSim3\bin |
| egg2bam.exe | To convert an *.egg file to a *.bam file. Bam files load much faster than egg files and both are graphical databases. Located in c:\DriveSim3\bin |
| | There are a lot more programs in c:\DriveSim3\bin with a very specialized use. They are part of the Panda3D rendering engine. Only for users who want to develop databases. |

## 2.3 File extensions

Here’s a list of important file extensions.

| Extension | Type of file |
|---|---|
| .prc | Resource file for the rendering engine. Located in c:\DriveSim3\Carnetsoft. ConfigC.prc and configLR.prc contain the settings for the middle (centre and left/right rendering channels. The left channel represents the left out of the window view that is displayed on the left rendering monitor. The centre channel represents the forward view and the right channel the right view. .Prc files are loaded by mainC.py and mainLR.py. |
| .py | Python files that load the rendering engine for each channel separately. The rendering engine is Panda3D, and each channel is started from StartSim.bat. |
| .pyc | Compiled python files. These are stored in c:\DriveSim3\Carnetsoft. The .py source files are located in ..\ResearchSim\python-files\. If one of these python files is modified, make sure to copy it to in c:\DriveSim3\Carnetsoft and run the simulator again. A new .pyc file will be created automatically, if it does not contain any errors. |
| .sha | Shader files in c:\DriveSim3\Carnetsoft. Used by the rendering engine to create graphical effects, such as shadows. |
| .def | Resource files. Most of these are in the c:\DriveSim3\Carnetsoft\files\ folder. And most are modified and saved by the config program IOConfig.exe. All *.def files are text files that can be manually edited as well. |
| .exe | Important executables to run the simulator. Most files are located in c:\DriveSim3\Carnetsoft\SimCarnet |
| .bat | Mostly located in c:\DriveSim3\Carnetsoft\SimCarnet to perform special tasks. A number of these files are called from scenario script to load a spreadsheet with data and graphs (for example in the clinical applications). |
| .bam | Binary graphical database (virtual environment) made from an *.egg file with egg2bam.exe. All databases are located in the c:\DriveSim3\Carnetsoft\models folder. So, if you change or make a new database, make sure to copy it there. The *.bam database is read by the rendering software when a simulation (for example experiment) starts. It is referred to in the scenario script file. The original .egg file is created by the RoadDesign program, this *.egg file is manually edited if necessary and then converted into a .bam file via egg2bam.exe.<br>Animations are also *.bam files and are located in c:\DriveSim3\Carnetsoft\models\animations\. |
| .ref | A helper file with references to 3D models, that is read by the rendering engine together with the .bam file. Every graphical database consists of a .bam file and a .ref file, with the same filename. So, if there’s a file named rural.bam, then there also must be a file name named rural.ref. The .ref file is an editable text file that describes the position and rotation of 3D object files (these are also .bam files). Created by the RoadDesign program, together with the .egg file.<br>The .ref file must be located in the c:\DriveSim3\Carnetsoft\models folder. |
| .egg | The source file for a database .bam file. All database .egg files are located in |

| | |
|---|---|
| | c:\DriveSim3\Carnetsoft\ResearchSim\RoadDesign\databases\. *.egg files are output files of RoadDesign.exe. |
| .pft | These are particle animation files used for particle animations (snow and rain), by the rendering programs. Located in c:\DriveSim3\Carnetsoft\models\. |
| .net | The ‘logical’ counterpart of the graphical database. This consists of lists of intersections, road segments etc. that are used by traffic.exe by vehicles to traverse through the road network. This allows the system to accurately calculate distance to intersection, lateral position, TLC, number of lanes, traffic light status etc. Located in c:\DriveSim3\Carnetsoft\SimCarnet\scenegraphs\ |
| .rdb | The road database source file that is input to RoadDesign. Located in c:\DriveSim3\Carnetsoft\ResearchSim\RoadDesign\databases\ |
| .sdb | Also a road database source file that is input to RoadDesign. Contains all scenery information (trees, buildings etc.). Located in c:\DriveSim3\Carnetsoft\ResearchSim\RoadDesign\databases\ |
| .path | Output of RoadDesign: a list of paths, in text format. A path is a connection between 2 nodes (intersections or Cnodes, see documentation). A path always has a counterpath in the opposite direction. This file is only for the user to check the paths and intersections. It is not used by the system. Located in c:\DriveSim3\Carnetsoft\ResearchSim\RoadDesign\databases\ |
| .exp | This for an experiment file. An experiment file gives information about the subject ID, the scenario script file used and the datadefinition file (that defines which data variables are sampled and stored). The subjectID info is used to create unique filesnames, for each subject individually. *.exp files can be read into the simulator. Specifically used for experiments. *.exp files are usually put into a \subjects or \trials folder that contains a separate .exp file for each subject-condition combination. |
| .scn | Script source file that is compiled by the script compiled within TextPad.exe. When files are read by the *.scn file, they must be located in the \data folder with the folder where the *.scn file is located. |
| .sci | Script include file that is called from an *.scn file. Must be in the same folder as the *.scn file that calls the *.sci file. *.sci files typically contain scenario script that is reused in various other *.scn files. |
| .scb | Binary compiled *.scn file. The simulator reads *.scb files. |
| .da0 | Binary datafile created by traffic.exe if the script has indicated that data must be sampled and stored. Data are stored in the \data folder that must be located within the folder where the *.scb file is located. *.da0 files are read by Dataproc.exe. |

## 2.4 Useful resource files

Most resource files are located in the ..\files folder and have .def as an extension. The most important ones:

- **Screensettings.def**: to define the screen resolution and offset where control.exe is displayed. Is located in c:\DriveSim3\Carnetsoft.
- **Inputdevices.def**: to define the buttons and axes of the steering wheel, pedals and shifter. This is created by IOConfig.exe, but can also be manually edited. Located in c:\DriveSim3\Carnetsoft\SimCarnet.
- **Language.def**: language of the user interface of control.exe: English, German, French or Dutch. Located in c:\DriveSim3\Carnetsoft\SimCarnet.
- **Scriptfolder.def**: default start folder where simulations are read in control.exe. If you run an experiment you will set this to the folder where all subjects *.exp files are located. Located in c:\DriveSim3\Carnetsoft\SimCarnet.
- **Soundfolder.def**: name of the folder where the soundsamples (*.wav files) are located. Located in c:\DriveSim3\Carnetsoft\SimCarnet.
- **Sound.def**: settings for the sound. Located in c:\DriveSim3\Carnetsoft\SimCarnet.
- **Windowsettings.def**: resource file for the screen position of the subwindows of the userinterface of control.exe. Located in c:\DriveSim3\Carnetsoft\SimCarnet.
- **Automatic.def**: contains a flag to indicate if automatic gears are used. Set by IOConfig. Located in c:\DriveSim3\Carnetsoft\files\.
- **Leftseated**.def: contains a flag to indicate if the vehicle is leftseated (as in continental Europe) vs. Rightseated ( as in the UK). Set by IOConfig. Located in c:\DriveSim3\Carnetsoft\files\.
- **measuredata.def**: resource file for measuring data (into *.da0 and *.txt files) in the driver training modules (1-VehicleControl, 2-TrafficParticipation and 3-SpecialCircumstances), must be set to 1 if youwant to store data in addition to the driver errors.
- There are a lot more resource files in the \files folder which use is obvious.

## 2.5 Folders

Most files are stored in fixed folders. These are the most important ones:

| Folder name | Type of data |
|---|---|
| C:\DriveSim3\bin\ | All *.dll and *.exe files of the rendering engine |
| C:\DriveSim3\direct\ | ‘system’ python files of Panda3D |
| C:\DriveSim3\panda3d\ | Compiled python files of Panda3D |
| C:\DriveSim3\python\ | Executables of the rendering engine |
| C:\DriveSim3\Carnetsoft\ | Python files of the simulator software and StartSim.bat |
| C:\DriveSim3\Carnetsoft\documentation\ | General pdf documentation |
| C:\DriveSim3\Carnetsoft\files\ | Resource files used and set by IOConfig, control.exe, traffic.exe and the rendering software |
| C:\DriveSim3\Carnetsoft\models\ | All *.bam and *.ref graphical database files |
| C:\DriveSim3\Carnetsoft\models\ Animations\ | Animations of people and animals |
| C:\DriveSim3\Carnetsoft\models\ Cars\ | All vehicles\traffic participants used in the simulations |
| C:\DriveSim3\Carnetsoft\models\ Flare\ | Files for sun scattering effects |
| C:\DriveSim3\Carnetsoft\models\ Objects\ | All 3D objects that are positioned in the databases, referenced by the *.ref files. |
| C:\DriveSim3\Carnetsoft\models\ Popups\ | *.png files used by the StimPicture script procedure to put an image on the screen. |
| C:\DriveSim3\Carnetsoft\models\ Textures\ | Textures grouped by type of use: facades, fields, roadsigns etc. |
| C:\DriveSim3\Carnetsoft\SimCarnet\ | Simulator executables and resource files, dll’s etc. For example, control.exe and traffic.exe. |
| C:\DriveSim3\Carnetsoft\SimCarnet\ Lessons\ | Folder where all simulations, experiments are located. For example the test experiments BrakeRT, SteerRT and workload. The folder with the abbreviation of the country contains all driver training scripts. |
| C:\DriveSim3\Carnetsoft\SimCarnet\ Scenegraphs\ | All *.net logical road databases. |
| C:\DriveSim3\Carnetsoft\SimCarnet\ sounds\ | All *.wav sound files |
| C:\DriveSim3\Carnetsoft\SimCarnet\ Students\ | Only used by driver training system. Not relevant for research simulator. |
| C:\DriveSim3\Carnetsoft\ResearchSim\ | Contains all sub-folders with reserach simulator extentions |
| C:\DriveSim3\Carnetsoft\ResearchSim\ Dataproc | Data analysis and data inspection program Dataproc.exe. Used for veriables that have been sampled with a fixed sample frequency, reads binary *.da0 files. |
| C:\DriveSim3\Carnetsoft\ResearchSim\ | Research simulator software documentation |

| | |
|---|---|
| Documentation\ | |
| C:\DriveSim3\Carnetsoft\ResearchSim\ExpPrep\ | Utility for experiment preparation. Creates *.exp files. |
| C:\DriveSim3\Carnetsoft\ResearchSim\PsychoPy\ | PsychoPy tool for integrating simulator data with psychological standard experiments, such as the Sternberg or the Posner tasks. |
| C:\DriveSim3\Carnetsoft\ResearchSim\Python-files\ | All python source files for the graphical rendering. The user may modify these and copy the modified files to c:\DriveSim3\Carnetsoft\ to change rendering behaviour. Only recommended for programmers who have experience with the programming language python. |
| C:\DriveSim3\Carnetsoft\ResearchSim\RoadDesign\ | Program RoadDesign (RoadToolC.exe), that is used to create or modify graphical databases (virtual environments). The system comes with 15 databases (highway, motorway, rural roads, town, roundabouts, etc), so in most cases there is no need to create new databases. |
| C:\DriveSim3\Carnetsoft\ResearchSim\Scripts\ | All source script (*.scn and *.sci) files for the simulations in the ..\SimCarnet\Lessons folder. |

# 3. Measure and analyze data

For more details, read the document *DataStorage.pdf.*

There are three different types of data that can be measured and stored in this simulator:

1. **Raw data**, with a fixed sample frequency, typically 10 Hz. This is the typical type of data that is stored on most simulators. It is used for offline data analysis and inspection and consists of variables such as vehicle speed, yaw rate, time to line crossing, time headway etc.
2. **Processed data** that is measured and analyzed during the simulation and the results are written to an ascii file. This consists of data measured and stored in scripts and is typically of a higher level than data type 1, the raw data. Processed data may consist of a standard deviation of lateral position on a certain stretch of road, or the brake reaction time in a specific event.
3. **Driving errors** as stored in the student assessment system that comes with the driver training simulations. This contains information of the highest level, such as the number of times the speed limit was exceeded, or time headway was too small.

**Raw data**
Traffic.exe and the scenario scripting language generate functionality for binary datasampling and storing of variables in binary datafiles. The files can be read and processed by the Dataproc data visualization and analysis program. The scriptfile 'DataProc.sci' contains some predefined functionality for datasampling. DataProc.sci can be included in any script in which binary data sampling and –storage is needed. Although the data to be sampled and stored can be specified in script, it may be easier to use the application ExpPrep.exe for that purpose. See *ExperimentSpecification.pdf* for more details.

Scripts allows the user to:
- define the data to be sampled together with the sampling frequency
- create (open) and close a binary datafile
- set event-codes for inclusion in eventfiles.

Data are stored in a \data folder located in the same folder as where the script is located. So suppose there is a folder \myexperiment that contains the *.scb (binary scenario script) files, there MUST be a folder \data within \myexperiment. If there is no such folder, the data cannot be stored to file.

Script code to specifiy the sampling rate:
*Proc( SetSampleFrequency, 10 );*

Script code to define which data must be sampled:
*Proc( AddDataVariable, DataCode );*
*Proc( AddDataFunction, FuncName );*

Script code to start datasampling of binary data:
*Proc( OpenData, "thisdatafile", "Experiment 1" );*

After the experiment, all stored data can be visualized and analyzed with DataProc.exe.

### Processed data

Processed data are data that are measured, analyzed and stored DURING the simulation. After the experiment trial, you have these data available which can save a lot of time spent on data processing.

DataProc.sci contains some functionality for this, and it's easy to extend this in script. Examples are:

- compute SD of lateral position while driving on a specific stretch of road
- measure brake reaction time, minimum time headway or minimum time to collision during an experiment

In this method, you create an ascii file (using script) and fill it with the appropriate values.

### Driver errors in Student Assessment system

The highest level type of data that can be stored consists of driver errors that are measured as part of the student assessment system in the driver training software modules. This is explained in detail in *CarnetManual.pdf*.

The SAS is the central part of the simulator training. Every student has a spreadsheet (Excel spreadsheet) with the results stored. Each student has a separate spreadsheet to store the progress data. When a new student starts the training, a new spreadsheet has to be created for that student. These data can be used in the research simulator as well and can be useful when the 'quality of driving' has to be assessed. There are around 40 'lessons' available, ranging from simulated highway driving to driving in rural areas, on roundabouts and in villages and towns. Also night driving, driving in slippery road conditions or in fog is simulated.

These simulations can be modified by the user by changing and recompiling the scripts. Normally, a virtual instructor gives feedback on driver errors, but this can be disabled via IOConfig.exe. The scripts also give instructions via graphical popups, which can be disabled in the script.

To start an experiment using the SAS data, while measuring raw data plus processed data, do the following:

Lets say you want to use \2-TrafficParticipation\P-Traffic1 and you want to name the files p01.*, for example p01.da0, p01.txt and p01.xls (for the driver errors).

- go to the \data folder within 2-TrafficParticipation and edit (using TextPad 4) *subjectfile.txt*. This contains the name of the raw and processed data files. Change the contents into p01 and save the file
- start the simulator
- press <Student data > button.
- press <Select student> button, select the correct subject, in this case p01 (which is p01.xls)
- if the database has been loaded (takes a while), select the proper script, P-Traffic1 in this case, and press <Start Lesson>
- After the simulation (or lesson as it is called here) is finished, the data are stored in the spreadsheet, and the raw data plus processed data are stored in the \data folder within the folder where the selected 'lesson' is located.

# 4. Test experiments

There are 3 example experiments included with the research simulator installation:

- BrakeRT: located in c:\DriveSim3\Carnetsoft\SimCarnet\Lessons\BrakeRT\
- SteerRT: located in c:\DriveSim3\Carnetsoft\SimCarnet\Lessons\SteerRT\
- workload: located in c:\DriveSim3\Carnetsoft\SimCarnet\Lessons\workload\

Within these folders there are always 3 folders:

- data: contains the datadefs.dd file that lists all data to be sampled and stored with a fixed sample frequency. All data of the experiment are stored here. Also, if files must be read from the script file, they have to be located here.
- Source: contains the source script of the experiment. If needed, these can be changed in the TextPad editor and recompiled. The resulting *.scb file must be copied to the experiment main folder (for example \BrakeRT)
- Trials: contains all *.exp files. The .exp file is loaded into the simulator via <Start Simulation>

First, the BrakeRT and SteerRT experiments are discussed. After that the workload experiment is discussed in a bit more detail. Its recommended that you start with the workload experiment, check the scripts and see if you understand completely how the experiment works. After that, it is recommended to study the scripts used BrakeRT and SteerRT in more detail, if needed.

## 4.1 A bit of theory first

In it's essence, car driving consists of 2 basic tasks: **lateral control** and **longitudinal control**. Lateral control consists of steering, or in general controlling one's lateral position on the road to ensure the car stays within  the lane boundaries and on the road. Longitudinal control consists of speed control and maintaining a safe distance to vehicles in the same lane, in order to avoid collisions. These tasks are continuous control tasks that require a constant monitoring of the position of the vehicle in relation to its surroundings. In experienced drivers these tasks are more or less automatized, but in case of reduced driver fitness, fatique or less driving experience, they will require more effort of the driver.

Apart from lateral and longitudinal control performance, the mental workload will be measured via the Detection Response Tests, that is performed during the experiments. The DRT task consists of a red block presented on the middle screen (left of the center), to which the subject has to respond by pressing a button as fast as possible. This test is highly sensitive to mental workload: if the driving task requires more mental effort, or is experienced as more difficult, RT and missed signals increase on the DRT. (See for example: Van Winsum, W., Martens, M., Herland, L. (1999). The effects of speech versus tactile driver support messages on workload, driver behaviour and user acceptance. TNO report TM-99-C043. TNO Human Factors Research Institute, Soesterberg, The netherlands).

If the driver has to invest a lot of effort in the primary task to keep performance on an acceptable level, less attention will be given to other information and changing information in the environment of the car will be noticed less. This is considered dangerous because it increases the risk of missing a vehicle from left or right or pedestrians, etc. In other words, attention narrows to essential aspects of lateral and longitudinal task performance while other information is missed more easily. This is measured with the DRT. There must always be a baseline measurement of the DRT because of individual differences in reaction time. So the increase in DRT reaction time and missed signals relative to the baseline measures the workload induced by the driving task.

The tests have been designed to reduce the risk of simulator sickness as much as possible, by keeping the image complexity in the left and right screens low, and by keeping the accelerations and deceleration (both laterally and longitudinally) as low as possible. This is important because the risk of simulator sickness is largest with older (> 30 years of age) and more experienced drivers.

In all tests, the simulator car drives in AutoDrive mode: the simulator controls the speed to ensure that speed conditions are identical for all subjects. **In AutoDrive mode the gear shifter must be switched into the neutral position.**

## 4.2 BrakeRT experiment

This trial takes around 9 minutes. The subject drives on a winding continuous two-lane road. There's a constant stream of oncoming vehicles to make sure the driver stays in the left driving lane. There's a vehicle in front (lead car) that must be followed. The subject's vehicle drives automatically and maintains a time headway of around 0.7 second to the lead car. The lead car drives at a constant speed of 80 km/h, in AutoDrive mode (meaning that the simulator controls the simulator vehicle's speed) and at irreguar intervals it suddently brakes and decelerates to 50 km/h. In that case the following happens:

- The brake lights of the lead car switch on and it decelerates at 2.5 m/s$^2$ until it reaches a speed of 50 km/h
- As soon as the driver touches the brake pedal, the automatic speed control of the simulator vehicle is cancelled and it drives normally (human-controlled) and slows down.
- Brake reaction time is measured as the interval between the moment the lead car starts to decelerate and the moment the brake pedal is pressed >= 5 %.

During the 9 minutes tests there are approximately 16 brake events that result in a reliable measured average brake RT. This number of 16 measurements per trial is sufficient to give a reliable estimation of braking performance. While driving, the DRT (Detection Response Test) is performed and this results in around 140 measurements of RT's. At the end of the trail, all brake RT's are stored together with the number of events, average brake RT, standard deviation (SD) of brake RT's, and SD of lateral position. Also the number of collisions with the lead vehicle and with oncoming traffic is stored.

## 4.3 Steer Reaction experiment

This trial takes 10 minutes. The subject drives on a long straight two-lane road. The speed is controlled by the simulator and is maintained on a constant speed of around 100 km/h. This ensures that time pressure and task difficulty is the same for all subjects. If speed would be controlled by the driver, people generally tend to slow down if task difficulty is experienced as too high, in order to reduce mental effort. We want to ensure the task is forced-faced (instead of self-paced) in order to guarantee the task requirements are the same for all subjects. While driving in the left lane at a constant speed (AutoDrive mode, meaning that the simulator controls the simulator vehicle's speed) there are irregular (unpredictable) lateral disturbances (windgusts that act as forces on the side of the vehicle) from left or right, to which the driver is forced to respond quickly with a steering action (in the opposite direction), in order to stay within the lane boundaries. The steer reaction time is then measured as the interval between the start of the lateral disturbance and the moment the steering wheel is reversed at least 4 degrees in the opposite direction. During the 9 minutes tests there are approximately 20 events that result in a reliable measured average steer RT. This number of 20 measurements per trial is sufficient to give a reliable estimation of steering performance. While driving, the DRT (Detection Response Test) is performed and this results in around 150 measurements of RT's per trial. At the end of the trail, all steer RT's are stored together with the number of events, average steer RT, standard deviation (SD) of steer RT's, and SD of lateral position. SD lateral position is a generally used measure of steering performance in traffic psychology

## Instructions

All subjects need to receive the same instructions. For all tasks.

The instruction for the DRT (Detection Response Task) is:

"**The reaction task starts and ends with a dingdong signal. Look straight ahead to the horizon. As soon as you see a small red square a little bit to the left, press the response button as fast as possible. The response button is the middle red button on the right side of the steering wheel**".

The instruction for the Brake Test (BRT) is:

During the next test, the simulator vehicle starts to drive automatically. Make sure the gear shifter is in the NEUTRAL position. Always stay in the leftmost lane. There's a vehicle in front that pulls up to 80 km/h and at irregular intervals brakes until 50 km/h, after which it pulls up again. The simulator vehicle will follow and stay behind the lead vehicle, using a sort of cruise control. However, as soon as the lead vehicle brakes, cruise control stops working, and you have to brake as fast as possible in a controlled manner. This means you have to brake as soon as you see the lead vehicle braking, but never brake to a full stop: brake gently and make sure you keep a safe following distance, without colliding with the lead vehicle. As soon as the lead vehicle has reached a speed of 50 km/h, the cruise control takes over again, as soon as you release the brake pedal. The simulator vehicle will then pull up again and stay behind the lead vehicle. Never overtake the lead vehicle. The test takes around 9 minutes.

The instruction for the Steer Test (SRT) is:

During the next test, you will be driving on a long straight road and the simulator car will pull up until around 100 km/h. Make sure the gear shifter is in the NEUTRAL position. Always stay in the leftmost lane. At irregular moments, there will be windgusts from left or right. You have to make sure you don't drift to the right lane or off the road, when this occurs, but use the steering wheel to stay safely within the left lane. The test takes around 10 minutes.

## 4.4 Workload experiment

This experiment is discussed in a bit more detail.
The driving task consists of driving with a constant speed of 80 km/h on a curved road with oncoming traffic. The subject must stay within the lane boundaries and always stay on the road and avoid a collision. While driving, workload is measured with the PDT (peripheral detection task or DRT detection response task). There are 2 conditions: with or without a cognitive loading secondary task (counting back from 200). This secondary task only has a cognitive load. It can be performed while keeping the eyes on the road (no additional visual load).

The purpose of the experiment is:

- To see if workload can be measured of a simple driving task : is there a significant difference between the baseline PDT measures and the PDT measures during the _nosec condition (no secondary task) ?
- To see if the addition of a cognitive loading secondary affects workload: is there a difference between the PDT on the _sec and _nosec conditions ?
- To see if this cognitive loading secondary task affects performance on the primary task. The primary driving task has to subtasks: driving with a constant speed (measured by the SD of speed) and steering (measured by SD latpos). Also, behavioural compensation for increased workload can be measured as a reduction of mean speed during the _sec condition.

I have prepared a number of experiment files, four for each subject, plus a general one to practice driving. For example for subject 01:

- p01_baseline.exp
- p01_countback.exp
- p01_nosec.exp
- p01_sec.exp

plus practice.exp.

So, every subject has 5 trials:

- **Practice run** of approximately 4 minutes in which the subjects practice driving on the test circuit with the instruction to drive with a constant speed of 80 km/h (practice.exp).
- **Baseline PDT**: in which the baseline PDT, RT and missed signals is measured (p..baseline.exp),
- **PDT + countback task** while standing still (no driving), to measure the workload of the countback cognitive secondary task (p..countback.exp)
- **PDT + driving**: only driving while measuring the PDT, no countback task. This is to measure the workload demands of driving. (P..nosec.exp)
- **PDT + driving + countback task**: the subject drives while steering, maintaining a constant speed of 80 km/h, perform the PDT (to measure workload) and count back from 200 to 0 (to increase cognitive load): (p..sec.exp)

The *.exp files can be created with the ExpPrep.exe program. P01_sec.exp looks as follows:

```
"p01-sec"
"C:\DriveSim3\Carnetsoft\Lessons\workload_exp\trials\workload_exp.dd"
"C:\DriveSim3\Carnetsoft\Lessons\workload_exp\workload_exp.scb"
```

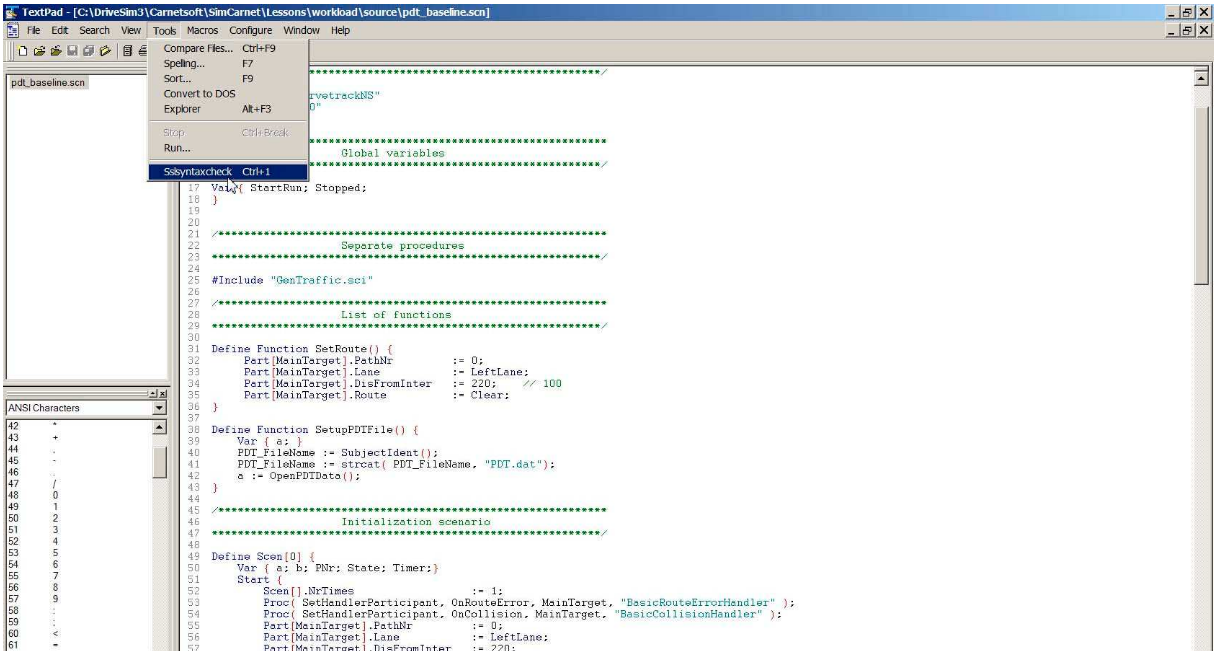
P01-sec, on the first line, is used as a filename for the output files.
The second line defines the location of the data definition file, so the system knows which raw data to store.

The third line gives the location and name of the compiled script file that defines the experiment.

P01_sec.exp is loaded into control.exe (Start Simulation). In the script (workload_exp.scn or one of the used *.sci files), this information is referred to:

The program c:\TextPad 4\TextPad.exe is used to make, change and compile scenario script. All *.scn and *.sci files can be double clicked and then Textpad opens. Syntax colouring and highlighting is used. To run the script, it must first be compiled into a binary script *.scb. Only *.scn scripts can be compiled: all include scripts (*.sci) are simply included in the binary files.
To compile the script:

- Menu Tools ->SslSyntaxCheck, see figure below.
- This runs the program SSLSyntaxCheck.exe with the current scriptfile as input
- When an error is found, TextPad points at the line with the error and an error message is provided.
- When there are no more errors, this is messaged to the user and a file with the same name and the *.scb extension is created.
- This file must be copied to the correct folder and can then be loaded and executed by the simulator.



```
*********************************************************/

rvetrackNS"
0"

**********************************************************
                    Global variables
*********************************************************/
Var { StartRun; Stopped;
}


/**********************************************************
                    Separate procedures
*********************************************************/

#Include "GenTraffic.sci"

/**********************************************************
                    List of functions
*********************************************************/

Define Function SetRoute() {
     Part[MainTarget].PathNr          := 0;
     Part[MainTarget].Lane            := LeftLane;
     Part[MainTarget].DisFromInter    := 220;    // 100
     Part[MainTarget].Route           := Clear;
}

Define Function SetupPDTFile() {
    Var { a; }
    PDT_FileName := SubjectIdent();
    PDT_FileName := strcat( PDT_FileName, "PDT.dat");
    a := OpenPDTData();
}

/**********************************************************
                    Initialization scenario
*********************************************************/

Define Scen[0] {
    Var { a; b; PNr; State; Timer;}
    Start {
        Scen[].NrTimes                   := 1;
        Proc( SetHandlerParticipant, OnRouteError, MainTarget, "BasicRouteErrorHandler" );
        Proc( SetHandlerParticipant, OnCollision, MainTarget, "BasicCollisionHandler" );
        Part[MainTarget].PathNr          := 0;
        Part[MainTarget].Lane            := LeftLane;
        Part[MainTarget].DisFromInter    := 220;
```

For example, in workload_exp.scn, in line 46:
**PDT_FileName := SubjectIdent();**
And in the next line, the string “PDT.dat” is appended to the filename.
SubjectIdent() is a system defined script function that requests the id (p01-sec, first line of the *.exp file) when the simulation has been started from an *.exp file.

In DataProc.sci, in the function StartDataSampling(), on line 118:
**EXT_DataDef := DataDef();**
This system defined function requests the name of a data definition file, (the second line of the *.exp file). In the following lines, this file is read and the data starage system is filled with variables via Proc( AddDataVariable, .. ) and Proc( AddDataFunction, .. );

To see which raw data will be stored, start ExpPrep.exe and Read File …\trials\wordload_exp.dd

It stores velocity, lateral position, steering Wheel angle, gas pedal position, brake pedal and clutch pedal plus gear. It also stored the segment id. After the experiment these data can be inspected and analyzed by DataProc.

Which data are processed online can be seen in DataProc.sci in the User Defined function WriteAsciiDataSpec() (line 241): mean vehicle speed, SD of speed, mean lateral position, SD of lateral position and number of collisions with oncoming traffic.

During all three trials, PDT is measured. In pdt.sci, the User Defined function WritePDTData() (line 45), it can be seen what data is stored: Nr of stimuli, mean reaction time, and fraction of missed signals (nr missed/nr stimuli).

The procedure is as follows:
Every subject first performs the baseline PDT measurement (for example p01_baseline.exp).
Then the cognitive load is counterbalanced: 50% of the subjects performs the task with the counting back secondary task (for example p01_sec.exp) as the second trial and the task without counting back (for example p01_nosec.exp) as the third trial, while the other 50% the order is reversed:

Order of tasks:

| 50% | 50% |
|---|---|
| practice | practice |
| _baseline | _baseline |
| _countback | _countback |
| _sec | _nosec |
| _nosec | _sec |

The instruction for the counting back task is:

“**Count back from 200 to 0, at a pace of 1 per second, for example, 200, 199, 198, etc. one number each second. Start with the task when you hear the first ‘dingdong’ signal (start of PDT test), and end when you hear the second ‘dingdong’ signal (end of PDT test). Counting must be spoken in a clear voice, like 200, 199, 198, etc. When you make a mistake, just continue. When you have reached zero, start again from 200**”.

After the task, write down the last number vocalized by the subject, because is a measure of the numbers counted down within the fixed amount of time.

The instruction for the driving task is:

“**Drive with a constant speed of 80 km/h. Stay in the left lane and make sure you don’t cross the lane boundary**”.

The instruction for the PDT is:

**“The task starts and ends with a dingdong signal”. Look straight ahead to the horizon. As soon as you see a small red square a little bit to the left, press the response button as fast as possible**”.

After the experiment, put the data in a table and analyze it (in SPSS, Excel or whatever you are using).

## 5. RoadDesign

For more detailed information, read *RoadDesign.pdf*.

The scripts refer to a database, and to objects and their properties with that database, such as:
- Set RoadNet “CurveTrackNS”
- Part[MainTarget].PathNr := 0;
- Part[PNr].DisToInter := Part[MainTarget].DisFromInter + 380;
- Etc.

So, the database that’s used in this experiment is CurveTrackNS.
RoadDesign is a designer for creating roadnetwork files for the Carnetsoft driving simulation software. With RoadDesign you design the roadnetwork geometry and this results in both a ‘logical’ database and a graphical database.
The graphical database consists of two ascii files:

- an *.egg file, which must be converted into a more efficient *.bam file
- a *.ref file

The logical database consists of an ascii file:

- a *.net file

Egg files are the database files used by Panda3d. Egg files are compiled into *.bam files because a bam file loads faster and requires less memory. The *.ref files contain references to 3D objects (buildings etc.) that are loaded in the graphics system together with the *.bam file. The rendering modules that read the *.bam and *.ref files are:

- c:\DriveSim3\python\ppython_center.exe
- c:\DriveSim3\python\ppython_left.exe

The *.net files contain definitions that are used by the traffic and scenario generation system:

- c:\DriveSim3\Carnetsoft\SimCarnet\traffic.exe
- in the script files a reference is made to a database and to paths and segments of the network of roads (as defined in the *.net file)

So we need to look for the following files:

- CurveTrackNS.bam (graphical database)
- CurveTrackNS.ref (references to 3d objects within the database)
- CurveTrackNS.net (logical road network accessed by script)