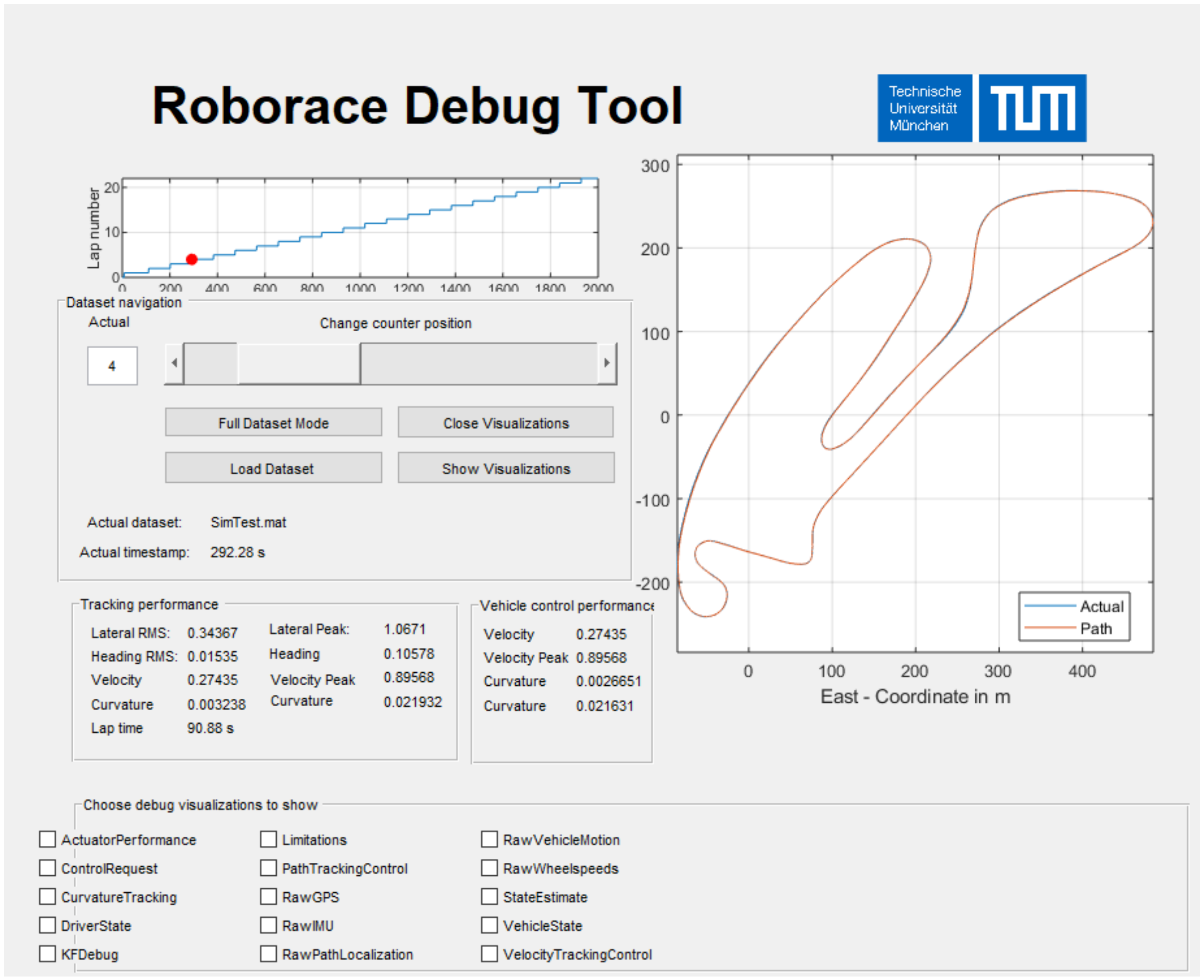
Roborace Control Debug GUI

General overview:



Actual: 4

Actual dataset: SimTest.mat

Actual timestamp: 292.28 s

Tracking performance

Lateral RMS:	0.34367	Lateral Peak:	1.0671
Heading RMS:	0.01535	Heading	0.10578
Velocity	0.27435	Velocity Peak	0.89568
Curvature	0.003238	Curvature	0.021932
Lap time	90.88 s		

Vehicle control performance

Velocity	0.27435
Velocity Peak	0.89568
Curvature	0.0026651
Curvature	0.021631

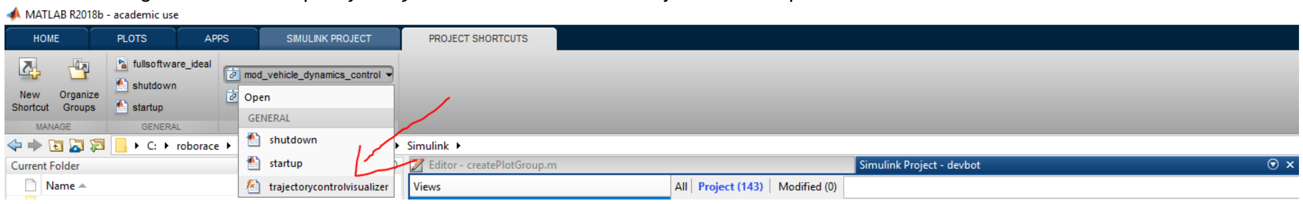
To faciliate an easy and convenient debugging workflow, a GUI to visualize the relevant data was developed within the Roborace team. It is used for all data concerning algorithms running on the motion control unit. This includes low level startup and safety logic, low level vehicle interfaces, the final sensor fusion and control algorithms. The tool is based on a common data format for all measurement data, independently if they are created by the simulation or the real vehicles. To enable this, there are scripts to convert the data from the vehicle or the simulation to the necessary format and store it as .mat files.

The tool allows to display either the full file data or separate it based on laps (indicated on the left top corner). The navigation can be done via the slider below. On the right side the global track is shown (target path and actual path). Below some key performance indicators are displayed. The checkboxes at the bottom can be used to select which detail plots are displayed when clicking on "Show visualization".

Workflow for data analysis:

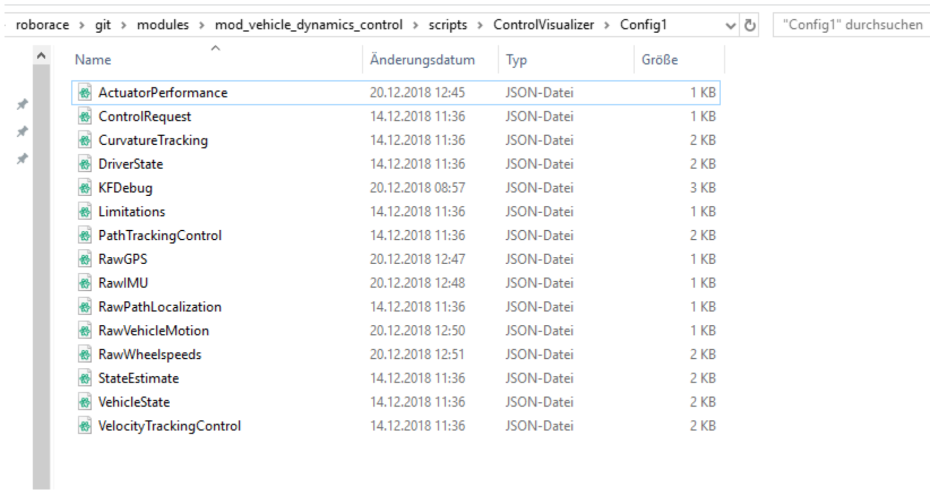
- Get the logs and convert them according to the steps given in the vehicle specific documentation
- Open the Simulink project of your vehicle

- Start the Debug GUI via the script *trajectorycontrolvisualizer* in the Project Shortcut pane



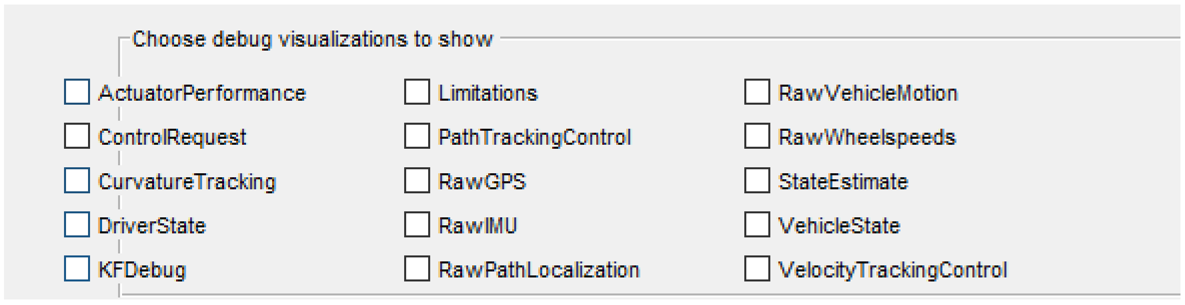
- The tool is asking you to load a data analysis configuration during startup. A configuration is represented by a folder containing multiple .json files. Each .json files describes one detail analysis plot which can be opened in a separate figure window. Choose the folder where your config is located to finish the startup. You can find an example configuration under *modules/mod_vehicle_dynamics_control/scripts/ControlVisualizer/Config1*.

:nt

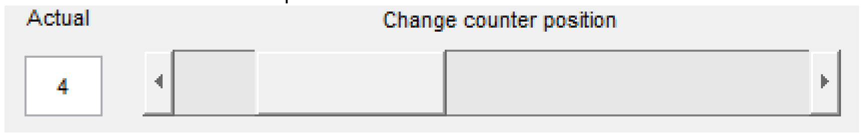

roborace › git › modules › mod_vehicle_dynamics_control › scripts › ControlVisualizer › Config1
"Config1" durchsuchen

Name	Änderungsdatum	Typ	Größe
ActuatorPerformance	20.12.2018 12:45	JSON-Datei	1 KB
ControlRequest	14.12.2018 11:36	JSON-Datei	1 KB
CurvatureTracking	14.12.2018 11:36	JSON-Datei	2 KB
DriverState	14.12.2018 11:36	JSON-Datei	2 KB
KFDebug	20.12.2018 08:57	JSON-Datei	3 KB
Limitations	14.12.2018 11:36	JSON-Datei	1 KB
PathTrackingControl	14.12.2018 11:36	JSON-Datei	2 KB
RawGPS	20.12.2018 12:47	JSON-Datei	1 KB
RawIMU	20.12.2018 12:48	JSON-Datei	1 KB
RawPathLocalization	14.12.2018 11:36	JSON-Datei	1 KB
RawVehicleMotion	20.12.2018 12:50	JSON-Datei	1 KB
RawWheelspeeds	20.12.2018 12:51	JSON-Datei	2 KB
StateEstimate	14.12.2018 11:36	JSON-Datei	2 KB
VehicleState	14.12.2018 11:36	JSON-Datei	2 KB
VelocityTrackingControl	14.12.2018 11:36	JSON-Datei	2 KB

- You should now see the GUI without data in front of you. Your config is parsed and for each .json file a checkbox is created at the bottom of the GUI. This can be used to access the plots later on.



- Press *Load Dataset* and choose the dataset you want to analyze. Depending on the size of the log file, this might take some time.
- You have to options to analyze the data. Either lap based or the complete dataset. After loading the dataset, the config is in Single Lap Mode. This means that you can use the slider to navigate through the dataset. Note that the overview plot on the right always displays the right time frame. If you want to display the complete dataset, click on *Full Dataset Mode*. This disables the slider and sets the relevant timeframe to the complete dataset.



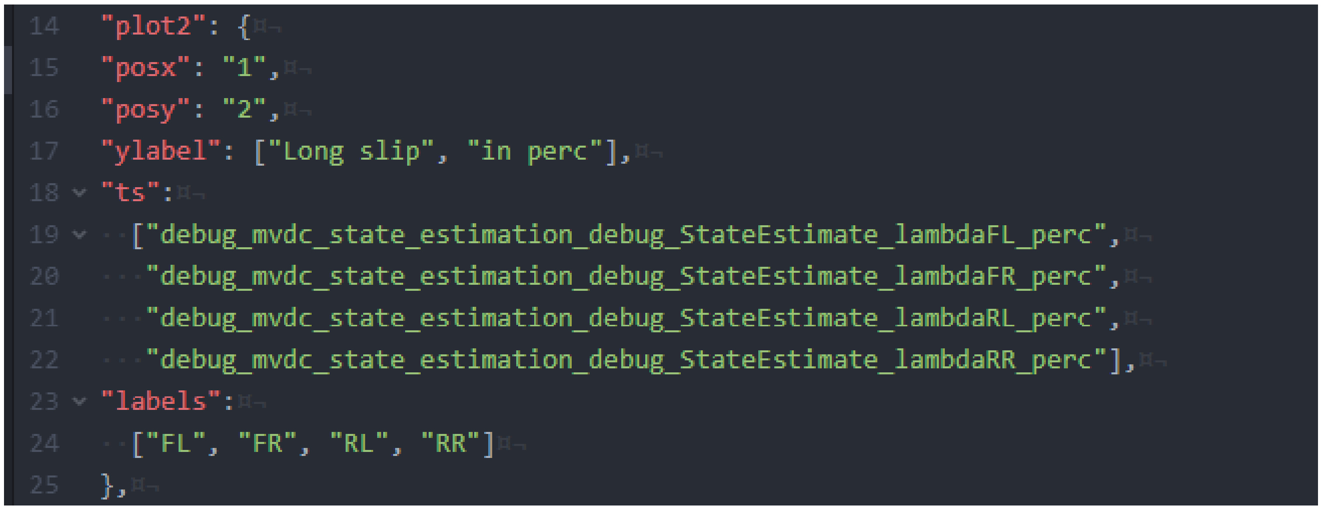
- Tick the checkboxes at the bottom for the detail plots you want to see (multiple are possible).
- Press *Show Visualizations* and wait until all the plots are displayed
- If you want to close all plots at the same time, press *Close Visualizations*

Create your own config:

While the standard config might cover many cases, you might want to create your own figures and put them together in your own way. To do this, follow this workflow:

- Create a new folder with a meaningful name
- Place all .json files for your plots in here
- Open it after starting the GUI
- Done!

The design of new figures is a little bit more involved. In general, each .json file represents one figure. The Filename is used as figure title and checkbox name. The file itself, is a collection of multiple plot definitions. A single plot definition looks as follows:



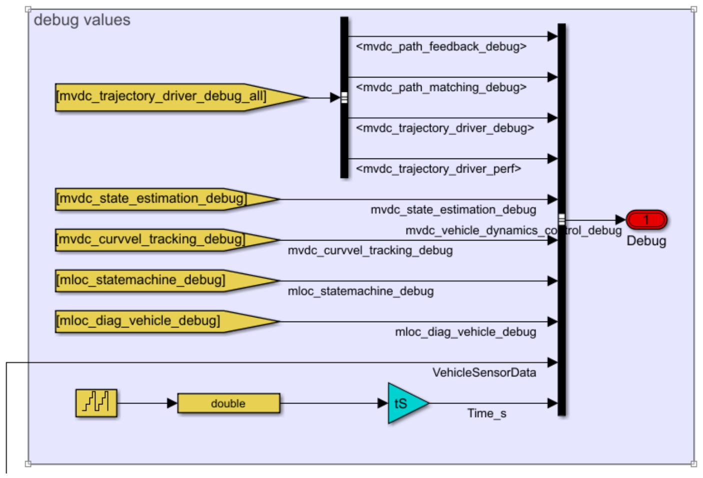
```
"plot2": {
"posx": "1",
"posy": "2",
"ylabel": ["Long slip", "in perc"],
"ts":
  ["debug_mvdc_state_estimation_debug_StateEstimate_lambdaFL_perc",
   "debug_mvdc_state_estimation_debug_StateEstimate_lambdaFR_perc",
   "debug_mvdc_state_estimation_debug_StateEstimate_lambdaRL_perc",
   "debug_mvdc_state_estimation_debug_StateEstimate_lambdaRR_perc"],
"labels":
  ["FL", "FR", "RL", "RR"]
},
```

The name ***plot2*** is not used in the programm. ***posx*** and ***posy*** specify the subplot position in the figure. **posx** represents the column and ***posy*** the row in a grid. They start with 1 and can be arbitrary high. Experience has shown that single column figures are handy during analysis, furthermore it is advised to not go for more than three rows. They become hard to read on a laptop screen during track operation. ***ylabel*** represents the label of the y-axis. The multiple strings allow to put a multi-line label. The ***ts*** structure holds the variable names of the timeseries stored in the *debug* structure located in the log file. These must match exactly, otherwise the plot cannot be drawn. Finally, the ***labels*** struct ure holds the legend entries for each of the timeseries. They are matched in order of their appearance. Each timeseries must have a label.

In general, it is best to start from an existing .json file and to modify it according to your needs. This will help you to gain the necessary insight and create your own debug plots fastly.

How to add new signals:

In general, all signals are available which can be found in the *debug* structure in the log files. It is created from the debug bus in the software. You find it in the main model of the shared software parts mvdc_vehicle_dynamics_control. slx.


debug values
[mvdc_trajectory_driver_debug_all]
<mvdc_path_feedback_debug>
<mvdc_path_matching_debug>
<mvdc_trajectory_driver_debug>
<mvdc_trajectory_driver_perf>
[mvdc_state_estimation_debug]
mvdc_state_estimation_debug
mvdc_vehicle_dynamics_control_debug
[mvdc_curvvel_tracking_debug]
mvdc_curvvel_tracking_debug
[mloc_statemachine_debug]
mloc_statemachine_debug
[mloc_diag_vehicle_debug]
mloc_diag_vehicle_debug
VehicleSensorData
double
tS
Time_s
1
Debug

If you need additional signals, please add them to the available busses or add a new debug bus in case you have a new software module. Adding other signals than the *debug* structure is in general possible, but the workflow for this bus is automated (including generation of logging models etc.).