RIMAC AUTOMOTIVE CHALLENGE

INSTRUCTIONS FOR STAGE B

1. VEHICLE MODEL

The vehicle model provided in the package has the same parameters as that provided for Stage A. However, the simulation speed has been improved significantly. The FMU now runs at half the speed of real time with a step size of 10ms. The model has been optimized to run at this step size and it corresponds to the communication interval between the platforms on which the model and controller will be implemented. Also, the outputs are cleaner- related quantities (like velocity in the orthogonal directions) have been collapsed into vectors.

1.1 Tire Model

Given the significant complexity of the tire model and inexperience of contestants with the Pacejka model, a new Tire Test FMU has been provided which accepts all relevant tire states (longitudinal speed, angular speed, slip angle and normal load) as inputs and produces tire rolling radius, forces and moments as outputs, which allows for more direct use of the tire model and obviates the need for contestants to independently reproduce it.

Note: The Wheel Radius parameter mentioned in the previous document corresponds to the radius of the wheel and not the tire. The unloaded radius of the tire can be found in the .tir file and corresponds to approximately 0.34m. The effective rolling radius is now available as an output from the Tire Test FMU.

1.2 Simplified Vehicle Model

In order to assist contestants, a simplified vehicle model implemented in Simulink has been included in the package. The chassis, implemented as a MATLAB Function, has three degrees of freedom (longitudinal and lateral translation and rotation about the vertical axis) and simplified load transfer effects. The new Tire Test FMU provided with the package is used along with the wheel dynamics implemented using Simulink blocks. The purpose of this model is solely to demonstrate how a simplified model of the vehicle could be constructed to replicate the detailed model and has not been parameterized too accurately. Contestants should use their discretion to modify and tune the simple model to fit their needs.

2. REWARD FUNCTION

The reward function used to evaluate control algorithms in Stage A was defined to assign maximum weight to vehicle side-slip angle but also included contributions from longitudinal speed and lateral offset. However, we noticed that contestants optimized for the reward function structure by simply driving slowly along the edge of the track. To encourage contestants to target the primary objective of the challenge- to sustain high vehicle side-slip angles through the course of a given circuit- the structure of the reward function has been modified significantly and is now as follows:

$$f(t) = \frac{1}{t}\int_0^t (\beta)^2 dt \qquad \boldsymbol{\beta} - vehicle\ sideslip\ angle\ (deg)$$

The rationale behind this structure is that it is more important to maintain higher sideslip angles- it is a drift competition after all. Averaging the reward function over time incentivizes maintaining a higher longitudinal speed as the time taken to complete the circuit would then be lower and conversely avoids a higher reward function being generated from driving slowly with a low sideslip angle (which is obviously easier to achieve).

3. HARDWARE PLATFORMS

As mentioned in the Challenge Description document, the vehicle model will be simulated on a dSpace SCALEXIO Hardware-In-the-Loop (HIL) test rig and the control algorithm will be implemented on a dSpace MicroAutoBox II Rapid Development Platform (RDP). More information regarding the MicroAutoBox II can be found here. The two platforms will communicate via a CAN bus as defined below.

3.1 CAN BUS

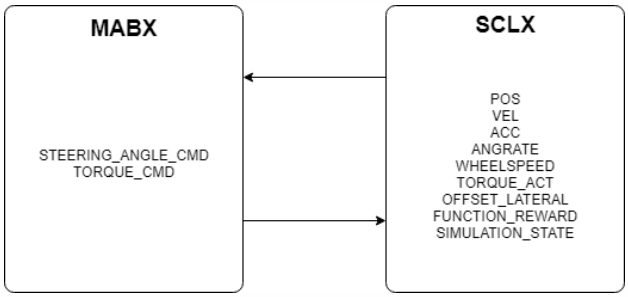

Fig: Transmitted signals from MABX (MicroAutoBox) and SCLX (SCALEXIO)

The *BUS.dbc* file provided contains the definition of the CAN BUS for communication between the MicroAutoBox and the SCALEXIO. The vehicle model, running on SCALEXIO, transmits the model states to the MicroAutoBox and receives the actuation commands from it, as depicted in the figure above.

To transmit commands to the vehicle model and receive states from it, you must use the RTICAN Transmit and RTICAN Receive blocks from the dSPACE RTI CAN blockset in your Simulink model of the control algorithm. The RTI CAN library would be installed during installation of the dSpace software suite. Make sure to select the RTI1401 platform in Matlab during startup after installation of the dSpace library. An example model demonstrating CAN communication has been included in the package.

3.2 Exporting Your Control Algorithm

Control algorithms should be exported to C code which is executable on the RDP using Simulink Coder. This process generates a number of files, all of which are necessary to program and monitor the MicroAutoBox II platform. The files (when using the default "solution.slx" model name) are as follows:

- solution.map
- solution.ppc
- solution.ppc.hex
- solution.ppc.srec
- solution.sdf
- solution.trc
- solution.trz

Contestants can send their compiled control algorithms (a .zip file containing all the files above) to the Rimac Automotive Challenge Team through WeTransfer at any time during the week. The Rimac Automotive Challenge team will test the submitted code on a MicroAutoBox II platform every Friday, and report the results (execution time, faults generated) to the submitting team.

3.3 dSpace Software Suite

To install the dSpace software suite, follow the instructions available here.
The Ticket Group ID is: **ACT-8E933-9229D-084F3-7A654-0303D**.