Real-Time Control and Integration of a High-Fidelity Driver-in-the-Loop Motion Platform

Julian Hodgson, Giles Wood
Red Bull Technology Limited

ABSTRACT

Formula One racing is largely concerned with the issue of driving a vehicle at, or near, its performance limit. Attempts to extend the performance envelope through overall system development must be made with a view that any such improvements have to be exploited by a driver. The cost of track testing, regulatory limitations on the number of testing days, and the need for rapid turnaround mean that vehicle simulation is becoming increasingly important. To extract the maximum benefit from this, it is necessary to keep the driver in the loop since he forms a critical component in the overall system. For the driver’s control inputs to be valid, the simulation must be translated into accurate visual, vestibular, and auditory cues, and all of this must be done robustly and in real time. The high accelerations seen in Formula One, and the fact that a racing driver operates the vehicle in the region of its performance limit, make this a particularly challenging task.

Red Bull Technology is developing a new simulator with which it hopes to deliver the highest levels of simulation fidelity to both the driver and his engineers. The design of this simulator requires a real-time environment that is distributed, expandable, and tightly integrated with existing off-car systems and tools. These requirements have been met by new rapid prototyping tools developed by McLaren Electronic Systems Limited, built on MathWorks simulation and code generation tools, and deployed on real-time hardware supplied by Speedgoat GmbH.

INTRODUCTION

The development of Formula One racing cars has evolved considerably over the years. Where once it was sufficient to identify individual components that influence vehicle performance and optimize them as thoroughly as possible, we now see more of a systems approach being taken. This evolution was necessary to ensure that the vehicle is developed as a whole and that the performance gains made in individual subsystems (aerodynamics, suspension, drivetrain) can be translated into better lap times. It is therefore natural that this process should include the addition of perhaps the most important subsystem, the one that closes the feedback loop, the driver. Failure to include any consideration of the driver could easily result in the evolution of a car that is theoretically very fast but that is, in practice, almost impossible to drive. Clearly, this is not the desired outcome.

Figure 1. The closed loop vehicle system.
An analytical approach to this type of problem, where the open-loop response of the vehicle is used to predict closed-loop characteristics, is plausible but very difficult. The difficulty arises from the nonlinear nature of the vehicle response coupled with the fact that it is operated precisely where the nonlinearities are particularly severe. Figure 1 shows the primary subsystems and their coupling in the overall system, where:

- **P** Chassis dynamics – nonlinear, but approximately linear if formulated correctly
- **T** Tire dynamics – strongly nonlinear with saturation, operated about their peak, which makes sensible linearization difficult
- **A** Aerodynamics – weakly nonlinear, but high dimensionality
- **C** Controller – nonlinear, and also predictive
- **V** Vestibular – physiological detection and perception of motion, including vision

Clearly the only definitive approach to investigating these interactions is with a driver operating a real car during track tests. This is extremely costly and represents a difficult environment in which to undertake experiments but is ultimately the only objective way of assessing vehicle performance and drivability. An alternative approach is to make use of mathematical models of the vehicle together with a high-fidelity motion cueing system to emulate the feedback cues that the driver responds to (motion, visual, auditory, and somatic), as shown in Figure 2, where:

- **Z** Cueing filter
- **M** Motion platform
- **H** Motion control

![Figure 2. An alternative approach to vehicle development.](image)

This approach has a number of advantages but also presents challenges that require careful consideration. The cost, speed of turnaround, and flexibility of a virtual car mean that the performance of new concepts can be quickly assessed. However, this clearly requires that the vehicle models represent a sufficient level of fidelity to capture the physical phenomena of interest and that modeling tools are available that allow for easy modification of the models and automatic code generation for execution on a real-time system. Particularly true of a Formula One application is the requirement to have a high-fidelity thermo/mechanical tire model and accurate representation of the aerodynamics in the model. This typically requires extensive CPU capacity and a modular approach whereby additional capacity can be added as models evolve.
The other difficulty with this approach is providing the driver with accurate motion cues that are consistent with those that would be experienced on a real vehicle. Clearly the motion cues must be sufficiently accurate that the driver is able to sense, through whatever combination of feedback he uses, any vehicle modifications that are being studied; otherwise there is little point in having the driver in the loop. It is well known that the dynamics of the human vestibular system are such that it is not necessary to replicate the motion of the vehicle exactly; otherwise the only type of simulator that would suffice is one with a motion range equivalent to a track. One of the particularly challenging aspects of this is the fact that the vehicle can undergo very rapid transients, both longitudinally and laterally. This requires a very-high-performance motion system with a high closed-loop bandwidth. Red Bull has developed a proprietary motion system that for confidentiality reasons will not be discussed here. What matters is that there is a need for a high-performance motion system to cue the driver, and that adds an additional burden on the real-time system that is tasked with running the vehicle models, motion control, and safety logic.

SIMULATION AND CONTROL REQUIREMENTS

Like most Formula One teams, Red Bull uses model-based design tools, mainly Simulink, for its vehicle modeling and simulation, and the same approach will be used for the additional modeling and control design that is specific to the simulator. While we expect the control design and the hardware to remain relatively stable, once commissioned, the vehicle simulation will constantly evolve. Given this, it makes sense to look to implement the real-time simulation and control using rapid prototyping tools built on MathWorks Real-Time Workshop automatic code generation.

The architecture of the system and the appropriate choice of rapid prototyping tools and hardware depend on the operational requirements, performance targets, and physical design of the motion platform. Figure 3 shows the relationship of the main components; the requirements they put on the rapid prototyping tools are described below. The generation of graphics and sound is considered as a separate system and is not discussed here.

A scale model of the motion system has been constructed to allow us to experiment on and prove out both the concept and the rapid prototyping tools ahead of constructing the full-size system. Here, the I/O and control functions have been combined on a single node and the simulation node omitted since there is no driver-in-the-loop.

Figure 3. System architecture.
STANDARD ECU SOFTWARE AND TOOLS

The simulator’s role is to provide a realistic simulation not just for the driver but also for his engineers. In practice this means that race and performance engineers will need to interact with the simulator’s live and historic data using the same tools they would with the real car. So, the relevant parts of the standard ECU control software and its configuration and analysis tools, System Monitor and Atlas, must be supported. There are also Red Bull’s own in-house setup and analysis tools, not shown, that link to these components.

SCALABLE MULTINODE, MULTIPROCESSOR, AND MULTICORE

The simulator will be a large and electrically noisy environment, and processing the I/O in one place will create problems with cable management and signal integrity. The design of the motion platform and its electrical systems has an optimal solution that requires two I/O and control nodes.

The simulator will run a complex simulation of the vehicle, track, aerodynamics, ECU applications, and the driver’s vestibular system in order to generate the inputs for motion control. To run such a large body of code in real time will require multiprocessing support, both multicore and multiprocessor. It makes sense for this to be implemented in a third “simulation” node, so its performance is not compromised by the requirement to support I/O. A separate simulation node will also be easier to upgrade or replace in line with future demands.

Another reason for this three-node approach is that it separates out the core I/O and control responsible for the safe, though not necessarily correct, operation of the motion platform onto one node. This reduces the complexity of the failure mode effects analysis and any resulting containment actions required to ensure the safe operation of the simulator. The cost of having three nodes though is the requirement for a high-speed communications link for them to share data and synchronize execution.

GENERIC CODE

Not all our vehicle modeling is implemented in Simulink. While C code from these models can be wrapped up and included in Simulink with little overhead, this is quite an awkward solution in terms of providing the hooks for parameterizing and observing them. Ideally, we want the ability to support generic C code independently.

RAPID PROTOTYPING TOOLS

The requirement to support the standard ECU software and its tools is particularly problematic when considering existing rapid prototyping tools, such as xPC Target or dSpace. Proper integration of them is core to the way we want to operate the simulator, and this is a requirement we do not want to compromise on. Ideally they should be supported as seamlessly and transparently as possible, without introducing additional processes and procedures. The ECU code, and to a lesser extent its tools, is also subject to regular revisions, and the simulator will need to be kept up to date.

The obvious solution was for McLaren Electronic Systems to take its existing code generation tools that target the embedded environment of the ECU and port them over to a generic x86 hardware environment. The functional control components of the ECU software are already written in Simulink, and their code is generated through MathWorks Real-Time Workshop tools, so this is a relatively straightforward and logical evolution of these tools. This new product is called vTAG-RT, with “vTAG” standing for virtual TAG310 the product designation of the standard ECU and RT for Real-Time.

vTAG-RT uses the commercial real-time operating system RTOS-32. This has provided a neat solution to jump-starting the provision of I/O drivers by facilitating the use of any driver written for xPC Target, since this also uses RTOS-32. Currently single-core single-processor operation is supported, with multicore multiprocessor and multinode support in development and expected by the middle of the year.

Because vTAG-RT is so closely related to the ECU, it will be straightforward to drop a virtual ECU into the simulator and manage it as if it were a real ECU. When fully specified it will be an extremely powerful rapid prototyping environment in its own right, and its features should prove useful to the other components of the simulation. In particular, the tools System Monitor and Atlas, see Figure 4, are well proven in motorsport application and have a number of powerful features.

---

1McLaren Electronic Systems has the exclusive contract to supply all the Formula One teams with a standard electronic control unit (ECU) and software for the 2008–2010 seasons.
Some of the features of interest to us and differences with other rapid prototyping environments that are worth noting are:

- A common Simulink blockset with the standard ECU code generation tools. This includes blocks for specifying all configurable and displayable data, which also include additional information such as descriptions, groupings, units, conversions and enumerations, and blocks for generating highlighted events that are time-stamped and have text description.

- Support for up to nine user applications. Also, multiple System Monitor clients can connect to the same target, allowing different users to manage different applications. This will be particularly useful given the large and componentized nature of the simulation.

- System Monitor has a “data wizard” tool for merging and transferring parameter data between different versions of an application.

- Atlas supports a variety of configurable display types that can be combined on the same page and linked so they display the same cursor and time span and scroll together. It supports multiple layers of data that can each contain data from a different live or historic source. Layers can be overlaid for quick comparison.

- Multiple Atlas clients can connect and view data from the same target. Alternatively, the live data can be broadcast to a large number of clients via an Atlas data server, which is useful where a number of engineers will be observing different aspects of the simulation.

- Atlas supports post-processing of both live and historic data. This can reduce the computational burden on the target for signals that do not close the feedback loop.

Figure 4. System Monitor and Atlas.
REAL-TIME HARDWARE

Of course, all of this needs suitable hardware to run on and to interface to the motion platform and driver. In the choice of hardware, we wanted to keep to an industry-standard format, rather than proprietary, since this gives us greater flexibility, both now and in the future.

I/O AND CONTROL NODES

For the I/O and control nodes, we have chosen to use Speedgoat’s real-time modular hardware. Some of the key features Speedgoat has been able to supply us with are:

- Rugged modular compact PCI chassis that are suitable for an industrial environment and that have been tailored to meet our specific requirements
- A wide range of quality I/O modules with xPC Target driver support
- Support for FPGA-based I/O, which is the only practical way for us to interface to the number and variety of position servos and encoders used on the motion platform
- Support for SCRAMNet GT200 reflective memory, which provides fast (2.5Gbps) and robust (with an optional redundant channel) communications between the nodes

For the scale model we have combined the functions of the two I/O and control nodes into a single similarly specified system.

SIMULATION NODE

Separating the vehicle simulation from the requirement to support I/O allows the optimization of this node for multiprocessing. We will be using a multiprocessor server platform with up-to-date multicore Intel processors, with the only I/O it must support being a SCRAMNet link to the other nodes and Ethernet. This is a very cost-effective approach to providing a lot of processing power, and it would be quite realistic to upgrade this on a regular basis and recycle the old hardware.

CONCLUSION

We have presented the case for a driver-in-the-loop vehicle simulator in the development of a Formula One car. There are many benefits to be had from this approach, but the realization of a simulator of sufficient fidelity is a considerable challenge in itself. The design of Red Bull Technology’s new simulator places some demanding requirements on the real-time simulation and control environment and has driven the development of new rapid prototyping tools. The result is McLaren Electronic Systems’ new product, vTAG-RT. This combined with Speedgoat’s modular hardware is already up and running on the scale model and will be full deployed later this year on the full-size simulator.

ACKNOWLEDGMENTS

Red Bull Technology would like to thank McLaren Electronic Systems Limited and Speedgoat GmbH for their efforts in supplying these new tools.