Transport Delay Characterization of SCANeR Driving Simulator

François Saidi, Guillaume Millet, Gilles Gallée
Oktal SA,
19, Bd des Nations Unies, 92190 Meudon - France
francois.saidi@oktal.fr, guillaume.millet@oktal.fr, gille.gallee@oktal.fr

Abstract – *Considering the high impact of system latency on simulator sickness and fidelity, the overall driving simulator as well as each subsystem must be carefully designed to limit system latency (e.g. system delay or transport delay) and the variation of this latency. This paper first describes improvement done in SCANeR software architecture to reduce and stabilize transport delays values. The most important source of transport delay is identified as being related to the asynchronous architecture of the software. To overcome that problem a scheduler was developed to precisely control the synchronization of the whole software component. In a second part the paper presents the approach used to measure hardware related transport delay in order to specify a suitable configuration for a high performance full scale dynamic driving simulator. This study includes hardware and software instrumentation in which the latency of each subsystem is measured.*

Résumé *- Etant donné l'importance des effets de la latence sur la mal du simulateur ainsi que sur la fidélité du rendu, aussi bien le simulateur de conduite au complet que chaque sous-système doivent être soigneusement conçus pour limiter la latence du système et les fluctuations de cette latence. Cet article décrit dans un premier temps l'étude de l'architecture du logiciel de simulation de conduite SCANeR ainsi que les améliorations faites pour réduire et stabiliser la latence du système. L'étude à montré que la source principale de latence est due à l'architecture asynchrone du logiciel. Afin de remédier à problème, un ordonnanceur à été mis au point afin de contrôler finement la synchronisation des différents composants du logiciel. Dans une deuxième partie l'article présente le protocole expérimental ainsi que les résultats des mesures de latences dues aux périphériques matériels du simulateur, dans le but de fournir des spécifications pour un simulateur dynamique de conduite. Cette étude se base sur une instrumentation aussi bien logicielle que matérielle pour fournir des mesures de latence de chaque sous-système.*

Introduction

Considering the high impact of system latency on simulator sickness and fidelity [5], the overall driving simulator as well as each subsystem must be carefully designed to limit system latency (e.g. system delay or transport delay) and the variation of this latency.

This paper first describes improvement done in SCANeR software architecture to reduce and stabilize transport delays values. In a second part the paper presents the approach used to measure hardware related transport delay in order to specify a suitable configuration for a high performance full scale dynamic driving simulator.

Background

Simulator transport delay characterization received a lot of attention from simulator operator and manufacturers [1] [2] [3]. Different frameworks have been proposed, most of them relying on real time OS and expensive hardware for inter-host communication.

With the ever-decreasing price of multi-core CPU and high bandwidth Ethernet networks these choices should be reconsidered. Using a different approach, SCANeR™ can run on non real time os and conventional hardware reducing development cost without sacrificing high performance and determinism.

Different sources of delay have been classified [4]:

1. **Off-host delay:** Duration between the occurrence of a physical event and its arrival on the host.
2. **Computational delay:** Time elapsed while the data is in the host system and while the system is doing computations.
3. **Rendering delay:** Time elapsed while the graphics engine is generating the resulting picture.
4. **Display delay:** Time elapsed between sending images to the display and the display actually showing them.
5. **Synchronization delay:** The time in which data is waiting between stages without being processed.
6. **Frame-rate-induced delay:** Between two frames the display is not updated, causing the user to see an outdated image stream.

Each type of latency should be kept as low as possible. This cannot only be achieved by efficient software architecture but a high performance hardware selection and tuning is also necessary.

The system latency is measured between the activation of the steering wheel and the system response (through the visual system or the motion system).

Four latencies define the performance level (Figure 1): The Motion latency, The Visual latency, The Steering wheel latency, and The latency gap between Motion and Visual.

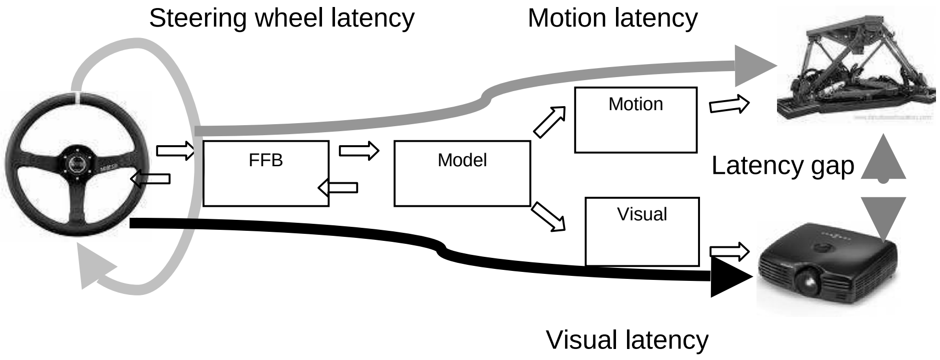

Figure 1. System latencies

The latency gap measures the delay between the effect of the steering wheel input on the platform and the effect on the display. In other words this latency gap expresses the asynchronism between vision and motion. This is considered to be the most important latency involved in the simulator sickness and should be kept as small as possible.

Software architecture measurement and improvement

We undertook extensive studies on transport delay of SCANeR™ software. These studies include source code instrumentation and data analysis; they revealed two main areas for improvement:

1. Synchronization between modules
2. Precision of the system timer

The following sections present a brief summary of these results and the solutions and improvement made in each area.

Synchronization issues

The simulation software relies on a distributed architecture taking advantage of modern multi-cpu hardware. This distributed architecture has nevertheless some complexity drawback and special care must be taken for the synchronization of the various modules. In previous versions of the software, synchronization of the different module was random and a high jittering configuration could happen frequently (Figure 2). In the figures below, M1, M2 and M3 are the critical components involved in the motion latency (we find a similar scheme for the visual latency).

M1 is the Force Feed Back (FFB) module, M2 the Dynamic Model and M3 the Motion module. All modules are running at 500 Hz in these tests.

M1 and M3 modules have a small execution time compared to their time slice, whereas M2 (the dynamic model) can use as much as 80% of its time slice. In such a configuration, the starting time of the module M3 was near the ending time of M2 and the transport delay jittered between 3 and 5 ms (Figure 3).

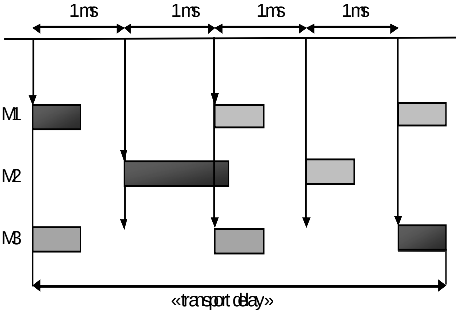

Figure 2. Random synchronization

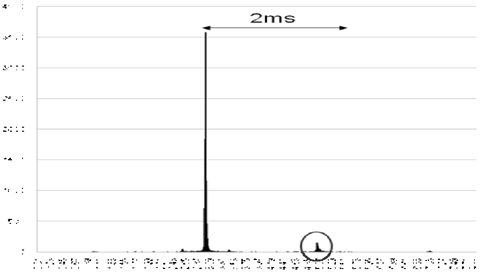

Figure 3. Unscheduled transport delay histogram (horizontal axis in ms, vertical is the number of occurrences)

However, a better synchronization configuration, as shown on Figure 4, could happen but this was also random. Such a configuration provided a transport delay stabilized around 3 ms (Figure 6).

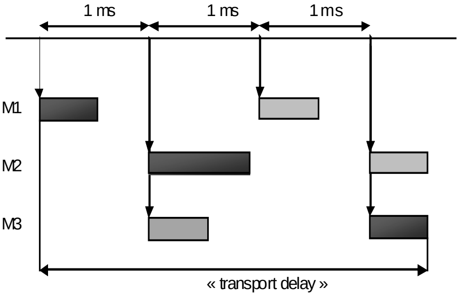

Figure 4. Scheduled synchronization

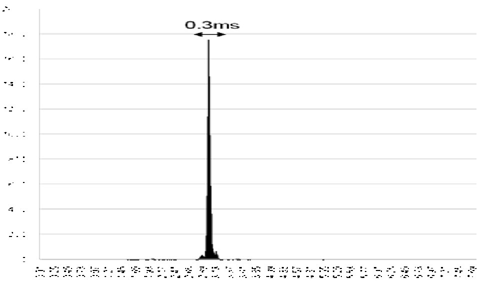

Figure 5. Scheduled transport delay histogram (horizontal axis in ms, vertical is the number of occurrences). The peak is at 3 ms

In order to precisely control the synchronization between the critical modules, a scheduler is needed. This scheduler should guarantees that the user defined synchronization configuration is maintained all aver the simulation providing a stabilized transport delay.

Precision of the system timer

SCANeR™ software and in particular the high frequency scheduler relies on the operating system timing performance.

The histograms below show the frequency stability of the system timer on both Microsoft Windows XP and VISTA/SEVEN.

We notice that the frequency regulation of a 500 Hz timer on windows XP (Figure 7) is less precise (most values are around 1.8 ms and few of them around 2.8 ms). On windows VISTA 99% of the values are around 2 ms. This study reveals a more precise timer regulation under VISTA OS, similar results are found under windows SEVEN which additionally embed a faster user interface. The use of windows SEVEN is mandatory when running a scheduling component at high frequencies (1000 Hz).

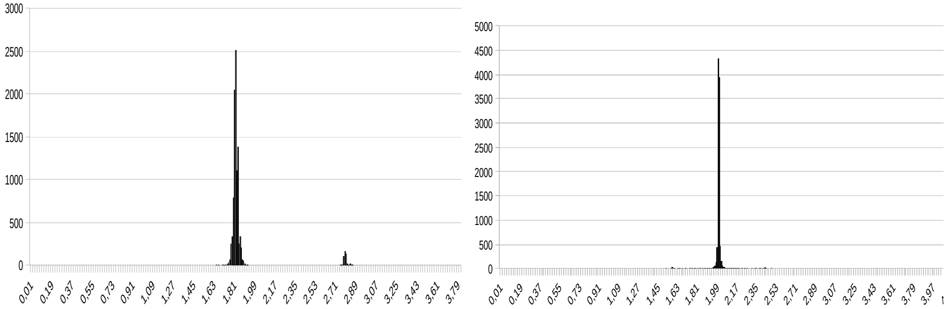

Figure 7. Frequency histogram of Windows XP timer (left) and VISTA (right) (horizontal axis in ms, vertical is the number of occurrences)

Scheduler conception

The scheduler is a software module which should run on the same computer as the FFB, Dynamic Model and Motion module (the critical modules). It controls the starting sequence of each step of the critical modules. The scheduler module runs at 1000 Hz and a configuration file defines the timings between the various modules that must be scheduled.

The scheduler is divided in several threads (Figure 8). A starter thread which sends the start event for each scheduled module and a set of listener threads (one for each scheduled module) to receive the execution end notification of a given module.

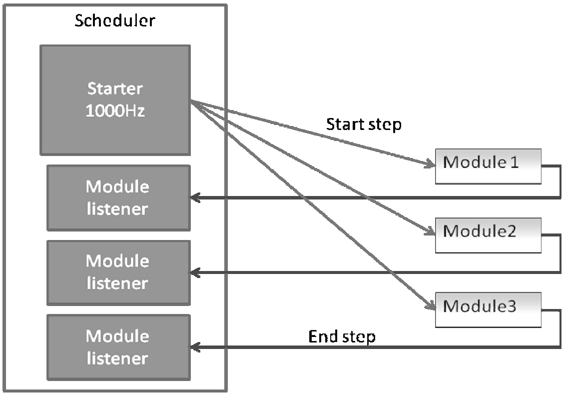

Figure 8. Scheduler architecture

The evolutions described above led to significant performance improvements of the simulation software which reaches a 3 ms software transport delay with 0.3 ms jittering for the path from FFB through Model and to the Motion module.

Hardware interfaces subsystem performance evaluation

This section studies the performance of each subsystem of the simulator independently. These tests were done under a Quad-Core Q6600 CPU under Windows VISTA OS. Test were performed using an oscilloscope together with an accelerometer for moving devices (motion platform and steering wheel), photodiodes for projectors, and I/O cards for signaling internal events in the software.

Steering wheel latency

The first experiment was to measure the steering wheel latency. The force feedback loop architecture is designed to optimize the transport of information from the steering wheel to the vehicle model (angle acquisition) and from the vehicle model to the steering wheel motor (steering torque). The advanced vehicle dynamics model is in charge of the computation of the steering torque according to the driver input and the driving conditions. Experiments were done on a Sensodrive SD-LC with two different kind of CAN interface a USB and PCI with no noticeable performance difference.

A specific version of the FFB module has been used to perform this test. This FFB module periodically sends a high torque value command to the steering wheel controller and simultaneously signaling this event by a 1 value on the pin 0 of the parallel port. This module also reads the speed value of the steering wheel and sets the pin 1 of the parallel port to 1 when a non-zero value is detected. An accelerometer was fixed on the steering wheel and connected to the oscilloscope. Parallel port pin 0 and 1 were connected to the oscilloscope to as described on the Figure 9. The result of this test is shown on Figure 10.

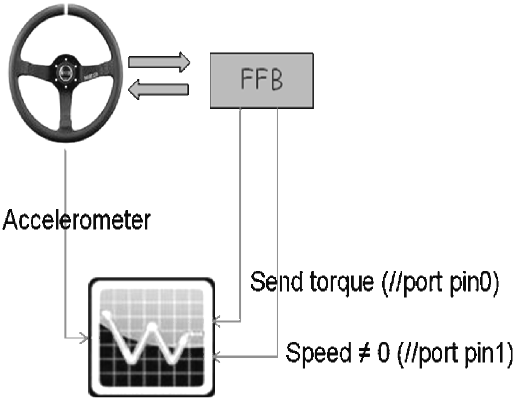

Figure 9. Steering wheel measurement

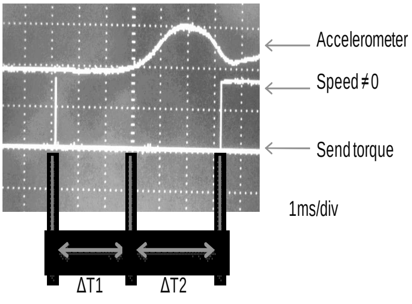

Figure 10. Results with FFB at 500 Hz

The minimum (6 ms) value has been reached for a FFB frequency of 500 Hz. Over 500 Hz, we did not notice any improvement, the bottleneck being shifted to the steering wheel internal control or the CAN bus communication.

Motion platform latency

The aim of this second test was to measure the motion platform delay. The tests were done on a Rexroth 6Dof motion platform with a 2500 kg payload.

A specific version of the Motion module has been used to perform this test. This Motion module periodically sends an acceleration value command to the Rexroth control computer and simultaneously signaling this event by a 1 value on the pin 0 of the parallel port. Additionally the Rexroth 6Dof motion control signals the reception of acceleration through an IO card. An accelerometer was fixed on the motion platform and connected to the oscilloscope. Pin 0 and the output of the 6Dof Motion controller IO card were also connected to the oscilloscope as described on Figure 11.

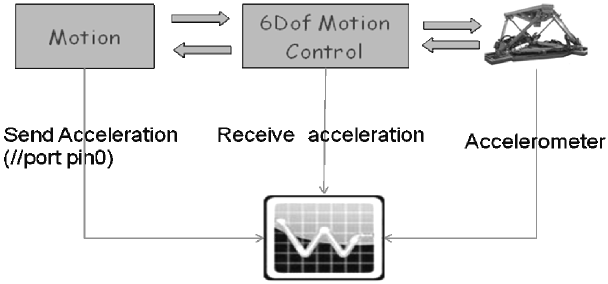

Figure 11. Motion platform measurement

The measured platform delay changes depending on the cueing cut-off filtering value (Figure 12). For a low-pass filter of 5 Hz the transport delay is around 25 ms and for 75 Hz it drops to 12 ms. A series of other experiments were conducted which showed that the platform delay can reach 35 ms. This variation of the platform transport delay does not occur during a simulation but changes only when the cueing settings are modified.

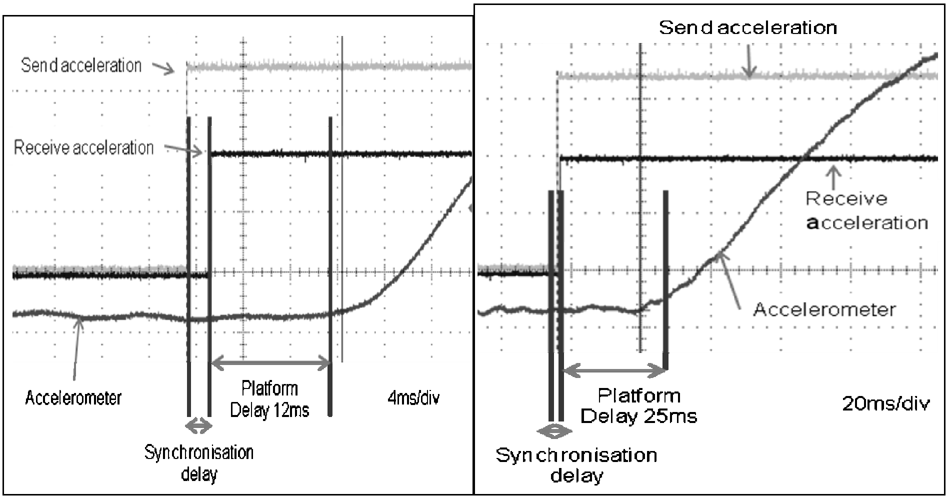

Figure 12. Motion platform results with a cueing filtering cut-off value at 75 Hz on the left and 5 Hz on the right

Visual system

The last test was about measuring the display delay of the video projector. The tests were done with two different DLP projectors. The dynamic model and the visual module were running on two computers. A specific version of the Dynamic Model and the Visual module were used to perform this test. The Model module periodically toggles between a vehicle position corresponding to a bright area of the database and a dark area; this causes a sudden change in the projector intensity easily measurable by a photodiode. Simultaneously the Model module signals the position change by 1 value on the pin 0 of the parallel port. The Visual module is modified to signal a position change by 1 value on the pin 0 of his parallel port. The Visual module frequency (display rate) is 60 Hz. Figure 13 shows the experimental setting, with a photodiode in front of the projector linked to the oscilloscope along with parallel port pin 0 of both computers running Dynamic Model and the Visual module.

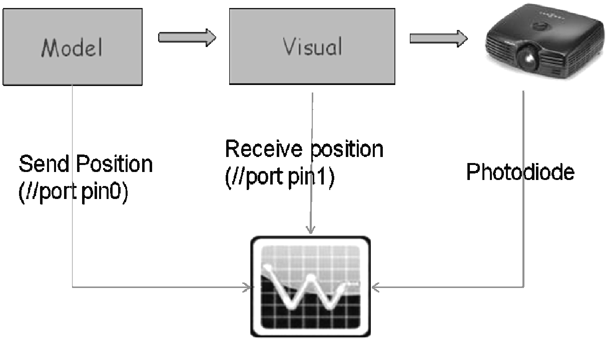

Figure 13. Visual system measurement

Results showed an important disparity between low cost projectors and high cost, simulation specific, projectors. The display delay vary from one to two frames from the time the frame start to flow out of the VGA connector to the moment where it is fully displayed on the screen. This confirms that the choice of low latency projector is an important issue for high performance simulator.

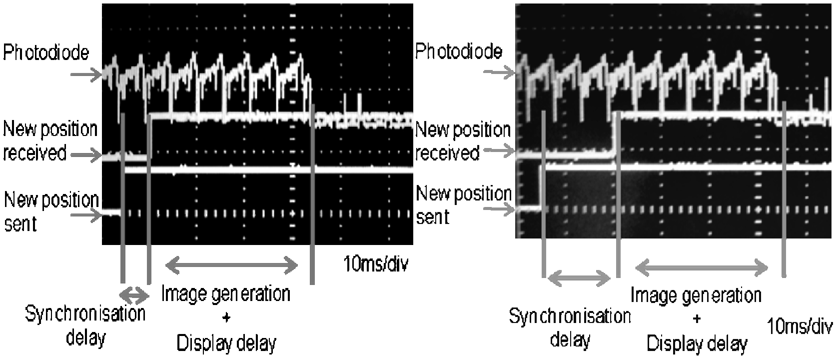

Figure 14. Visual transport delay of 2 consecutive measured with a low latency projector

Another important issue is the synchronization delay (Figure 14) between the model and visual. Because both process run on separated computers, this delay will vary from 0 to 16 ms (the visual module frequency is 60 Hz). A way of generating a synchronization signal from the scheduler to lock the image generation process is under study.

Conclusion

According to previous results we could estimate the global transport delay of a simulator based on SCANeR™ software and the selected hardware:

1. The average value of the Visual latency is lower or equal to 48.5 ms (± 8.5 ms)
2. The average value of the Motion latency can be between 19 ms and 37 ms (± 1 ms) depending on the cueing settings.
3. The average value of the latency gap can be between 29.5 and 11.5 ms without delay compensation.
4. The value of the steering wheel latency is around 9 ms.

These values are rather low compared to other simulator, and can easily stand the comparison with simulators based on real time OS. These values show that the bottleneck is the image display and generation frequency leaving some room for improvement.

Ongoing work aims at measuring the end-to-end transport delay of a complete simulator. Moreover an online monitoring of the transport delay is also planned.

The high performance architecture of this simulation software is flexible and optimized, allowing to run the dynamic model and the steering wheel force feedback loop at high frequencies pushing back the frontiers of non real time OS.

Keyword: Transport delay, optimization, synchronization

Bibliography

Papelis, Y., Qidwai, M., Schwarz, C., Dolan, J., (2005) Characterization of NADS Subsystem Transport Delays. *Proceedings of the Driving Simulation Conference*, DSC USA 2005.

Worl, R., (2006) Dynamic Measurement of Simulator Transport Delays. *Defense Science and Technology Organization,* Australia.

Shwarz, C., Gates, T., Papelis, Y., (2003) Motion Characteristics of the National Advanced Driving Simulator *Proceedings of the Driving Simulation Conference*, DSC USA 2003.

Jacobs, M. C., Livingston, M. A., State, A., Managing Latency in Complex Augmented Reality Systems, *Proceedings of 1997 Symposium on Interactive 3D Graphics*, ACM SIGGRAPH 1997.

Kemeny, A., Simulation and Perception, *Proceedings of the Driving Simulation Conference*, DSC Europe 2000.