OpenSD2S: New features and perspectives for an Open-Source Driving Simulation software

Paul George, Andras Kemeny, Chunshi Guo
Renault
Technical Center for Simulation
Guyancourt, France
paul.george@renault.com
andras.kemeny@renault.com
chunshi.guo@ensam.eu

Frederic Merienne, Feng Wang
Arts et Metiers ParisTech
Institut Image
Cluny, France
frederic.merienne@ensam.eu
feng.wang@ensam.eu

Thierry Joubert
Theoris
Paris, France
thierry.joubert@theoris.fr

ABSTRACT

Driving simulation is widely used by automotive companies as well as road traffic, training and safety research laboratories for efficient vehicle engineering design, driver behavior studies, better road safety and driver training. Though a number of industrial software solutions are available on the market such as SimVehicleLT™, SCANeR™, STISIM Drive® or Virtual Test Drive, their deployment may be heavy, often requires specific training and adaptation for specific need can be lengthy and costly.

As an alternative, open source codes are free of charge and typically easy to adapt and/or modify, among them OpenSD2S. This recently introduced Open Scalable Driving Simulation Software has a modular and scalable architecture based on MPI (Message Passing Interface) which allows for distributing the computational load over a computer cluster. The simulator is split into as many modules as features: a car dynamics module based on ODE (Open Dynamics Engine), a visual module based on OGRE (Open source 3D GRaphics Engine) or OSG (Open Scene Graph) which allows for multi-screen support, a sound module based on OpenAL (Open Audio Library), among others.

To meet the requirements for driver behavior studies, a traffic module, based on the OpenDRIVE road format and a simulation data recorder have been implemented. Thanks to its flexibility and these new features, OpenSD2S is now used for road driver and traffic research experimentation, such as at University of Lyon, where driver cognitive load study is carried out during road exit maneuvers. The used hardware configuration is an affordable solution for such an experiment, using a mainstream gaming steering wheel, pedals and a standard driver seat.

Future work will be carried out to improve the vehicle dynamics realism (collisions and suspension support), tools for visual and road database generation as well as advanced traffic model. The currently existing embedded version of the vehicle dynamics will also be extended to an embedded OpenSD2S version.

ABOUT THE AUTHORS

Paul George is a PhD student in computer science at Arts et Métiers ParisTech with Renault.

Andras Kemeny, ScD is Professor and Head of the Technical Center for Simulation of Renault. He has directed the development of a large driving simulation software package – SCANeR©. Currently he is also Director of the Laboratory of Immersive Visualization, a joint Renault - Arts et Métiers ParisTech research laboratory.

Chunshi Guo, MS in Automotive Engineering from Tongji University, China & Arts et Métiers ParisTech, France.

Frederic Merienne, PhD is Professor and Director at Institut Image, Arts et Metiers ParisTech.

Feng Wang is an MS student in Mechanical Engineering and Automation from Xi'an Jiaotong University, China & Arts et Métiers ParisTech, France.

Thierry Joubert is the CTO and co-founder of THEORIS. Thierry has been actively engaged in Embedded-system design and real time Simulation application development for over twenty five years.

OpenSD2S: New features and perspectives for an Open-Source Driving Simulation software

Paul George, Andras Kemeny, Chunshi Guo
Renault
Technical Center for Simulation
Guyancourt, France
paul.george@renault.com
andras.kemeny@renault.com
chunshi.guo@ensam.eu

Frederic Merienne, Feng Wang
Arts et Metiers ParisTech
Institut Image
Cluny, France
frederic.merienne@ensam.eu
feng.wang@ensam.eu

Thierry Joubert
Theoris
Paris, France
thierry.joubert@theoris.fr

INTRODUCTION

Motivations

Driving simulation is widely used by automotive companies as well as road traffic, training and safety research laboratories for efficient vehicle engineering design, driver behavior studies, better road safety and driver training. Though a number of industrial software solutions are available on the market such as SimVehicleLT™, SCANeR™, STISIM Drive® or Virtual Test Drive (Romano, R.A., 2000; Kemeny, A., 1993; Kemeny, A., et al., 1994; Lacroix, B., et al., 2013; Allen, R.W., et al., 2004; von Neumann-Cosel, K., et al., 2009), their deployment may be heavy, often requires specific training and adaptation for specific needs can be lengthy and costly.

SCANeR™ for example, initiated by Renault and now marketed by Oktal, is not only used by Renault, but also Peugeot, Nissan and many others with a good level of satisfaction. Nevertheless, even after a short training session, researchers may find long and fastidious the preparation of very specific driving simulation experiments as the SCANeR™ human machine interface has to take into account all the possibilities of such sophisticated driving simulation scenarios. Another issue is the existence of a large number of software tools on the market one would like to use to optimize database, scenario control or rendering capabilities. Also, acquisition of driving simulation software can be too costly for academics as high level rendering of image or motion is not always needed for experiments which do not focus on realism (Kemeny, A. and Panerai F., 2003).

One way to overcome these problems is to build one's own driving simulation software which would allow a high level of customizability to fit every need. This kind of software involves complex relationships between heterogeneous technologies and thus requires a wide variety of competencies such as image rendering, physics simulation, vehicle dynamics modeling, sound generation, traffic simulation, etc. However, the open source community already provides state-of-the-art implementations of some of these functionalities and the assembly of them could lead to fully open-source driving simulation software.

Based on this idea the Technical Center for Simulation of Renault in cooperation with Arts et Métiers ParisTech started the Open Scalable Driving Simulation Software (OpenSD2S) project: an open-source driving simulation software designed to be flexible and modular and which allows easy and quick modification or interfacing with existing or easily modifiable data or software. This could be a significant advantage for efficient implementation of use-cases not yet proposed as standard solutions on the market such as for Advanced Driver Aid System (ADAS) and Automatic Vehicle applications. In addition, one can hope that capturing a small part of those developments, may lead to a real alternative to commercial products with the integration of new trends, tools and capabilities. Usable by academic research lab as well as companies, it remains affordable since mainstream gaming wheel, pedals and seat can be used as well as high end professional simulator. It is released under the LGPL (GNU Lesser General Public License) which means it can be integrated in a commercial product (Free Software Foundation, 2007).

Project history

A first version was developed in 2010 at Arts et Métiers ParisTech in cooperation with Renault, Technical Center for Simulation (Filliard, N., et al., 2010). The same year, a first running version was published on sourceforge. In 2012, Theoris SAS, a software development company specialized in embedded systems and augmented reality,

joined the project. Since its introduction, OpenSD2S has drawn the attention from the academic community and several improvements and new features came from their feedback.

ARCHITECTURE

OpenSD2S main principles are modularity and scalability.

Modularity

Modularity is achieved through a component oriented design which helps make the software more easily maintainable by isolating independent features. This approach is well suited for integrating heterogeneous modules handling parallel computations. Every specific library is encapsulated in a module providing a standard interface which abstracts its functioning. This loosely coupled architecture is more flexible and favors testability of the system as evolution of the code is localized, which is a critical requirement in the conception phase of such composite applications. Therefore, it enables future evolutions as the addition of new functionalities is simply achieved through the addition of new modules, and it makes it possible to propose different alternatives for each module (e.g. different 3D Engines, vehicle dynamics models). Notably, modules based on open source libraries can be transparently replaced by in-house development or proprietary libraries, provided that a SDK (Software Development Kit) is available.

In the current version of OpenSD2S, modules consist of standalone operating system processes, coordinated using MPICH (Gropp W.D., 1995), a MPI compliant message passing interface (Clark L. et al., 1994). The use of an MPI-based inter-module communication layer enables the optimized transparent execution of the simulation software on a variety of hardware architectures ranging from Ethernet-based computer clusters to multi-CPU computers.

Scalability

Scalability is necessary so that the software is ready to adapt to a wide range of simulators, starting from low-cost gaming configurations to professional high performance simulators supporting the increased demand for computational power. For this purpose, the different modules are organized according to a star-shaped design pattern (Figure1): each module is connected to a central kernel, which manages a database containing all simulation data (e.g. car position, viewpoint, steering wheel angle). Modules can update these data or receive their last value upon request. This loose synchronization is intended to reduce the inter-module execution coupling and allow for heterogeneous execution frequencies to coexist (e.g. the visual module will typically run at 60Hz while the vehicle dynamics module will run at 1000Hz). Such a centralized data management would simplify the implementation of a monitoring tool for messages exchanges and data accesses on the server which is useful for debugging.

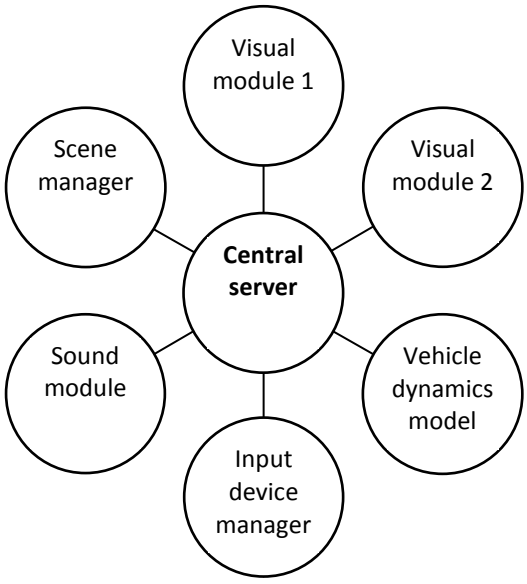

Figure 1. General module organization

FEATURES

Visual rendering

The visual module is used to display a view of the virtual environment from the driver's current viewpoint. Before the rendering of each frame, the visual module sends a message to the central server to fetch the last position and orientation of the viewpoint. The visual scene is rendered using OGRE (Streeting, S et al., 2000), an object-oriented open-source 3D render engine written in C++, customizable and extensible through the freely available plugins, enabling for instance OpenGL and DirectX support or bindings to multiple physics engines. Moreover, OGRE benefits from an active community.

In the simplest case, the module runs on a single computer and displays the road on a screen in front of the driver's seat. The visual renderer can also handle several more complex screen configurations as it can be run multiple times on each computer. It allows some basic, yet powerful cluster rendering.

To improve immersion and driving sensations, some mechanisms have been implemented at the visual level. Configurations where screens are surrounding the driver are supported. An asymmetric viewing frustum is computed for each screen to ensure having the correct simulated view according to the head position. Some configurations files are used to describe the screens spatial configuration as well as the head position, making it flexible enough to adapt to every possible simulator.

Most of the time, an approximation of the head's position derived from the seat's position is enough to provide good driving sensations. However, more precise data might be required to enable motion parallax thus improving depth perception. The renderer has been designed to allow further evolution, thereby paving the way to a future head-tracking module. The implementation of such a module would be painless as there are already open source tools to do such things such as VRPN. It would just be a matter of sending a MPI message carrying the head position.

Active stereoscopic 3D is supported for NVIDIA Quadro graphics card which features Quad Buffer and provided that the displays are 3D active ready which means that they can run at ~120 Hz or more. Also, passive stereoscopic is already natively handled by OGRE.

One main issue when implementing clustering in the case of 3D graphics generation is the synchronization of all the nodes of the cluster. Without it, images may seem discontinuous at screen junctions, or between the two eyes when doing 3D stereoscopic, guaranteeing headaches. Several layers of synchronization are supported by the visual module. Framelock is the synchronization of the 3D scene on each node of the cluster, in our case the critical node is the virtual car. Swaplock ensures that each frame is displayed on each screen at the same time. All these synchronizations are software implemented using MPI synchronization mechanisms.

Figure 2 : Virtual view of Guyancourt, rendered in OpenSD2S

Selecting a particular rendering library also implies constraints on the possible readable database description file formats. OGRE only supports its own native binary format, but a number of conversion plugins and exporters exist for main 3D authoring tools, including open source software like Blender. For the purpose of our tests, a visual database modeled in Autodesk® Maya® used in typical simulation scenarios at Renault has been exported using one of these plugins.

Vehicle dynamics

The vehicle dynamics model plays a key role in the driving simulation software, as it takes driver's commands as input and calculates vehicle's position, orientation, as well as the other status information like vehicle speed and engine speed. Currently there are no open source realistic vehicle dynamics models available as a single library. Most of the open source vehicle simulators are related to videogames and focus on the game enjoyment and playability instead of the correctness of the vehicle behavior. That's why the dynamics module has been created from scratch using ODE (Open Dynamics Engine) (Smith R., 2001) and MATLAB Simulink, a powerful modeling and simulation tool for designing complex dynamic systems which can automatically generate C code for real-time implementation.

The vehicle model has 10 degrees of freedom, which include longitudinal and lateral motions, yaw motion and body roll motion of the vehicle, the rotational dynamics of the four wheels as well as one degree of the powertrain system. As for the powertrain system, the engine is represented by the Engine Universal Performance Characteristics Map (Zhang Z.P. and Zhang J.Z., 2012), while a friction model coming from the literature (Bățăuș, M. et al., 2011) has been employed to model the clutch connected with a manual gearbox. Moreover, a braking system with ABS was integrated as are 1-D aerodynamics in the longitudinal direction. In this version, the vertical dynamics are not considered, for this reason the suspension was simplified as front and rear anti-roll bars. The latest version of the well-known Pacejka's Magic Formula (Pacejka, H.B., 2002) has been used to model the tire. In particular, the transient state is taken into account. The integration of the relaxation lengths both in the longitudinal and the lateral directions allows stabilizing the vehicle at the low speed and driving in reverse (Bernard, J.E. and Clover, C.L., 1995). The overall simulation runs at 1000Hz independently of the frame rate which is typically about 60Hz thanks to the kernel based architecture. A real-time version of the vehicle dynamics engine running on a Raspberry pi has been developed for the purpose of performances enhancements. The Raspberry pi is a credit-card-sized single-board computer which is cheap (US$25) and popular.

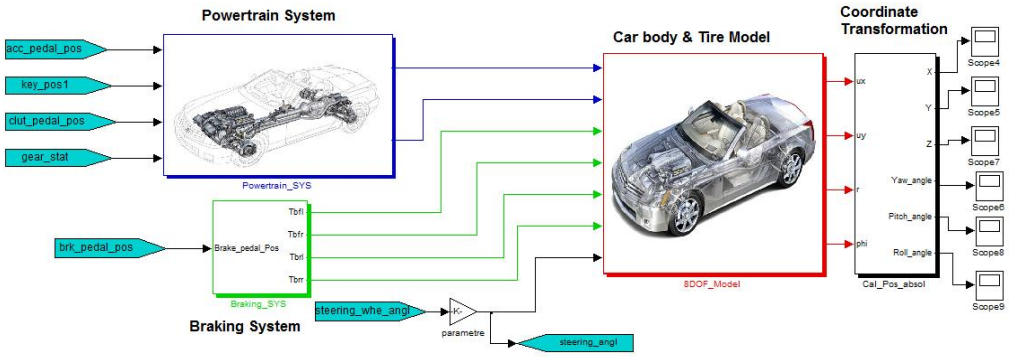

Figure 3. Vehicle dynamics model in Simulink

Sound

The sound module is based on OpenAL (Open Audio Library) (Loki Software, 2000) and is in charge of generating the sound environment. Currently, only the ambient sound of the car is generated (contact, engine, gear shift etc.).

Road Network

When building a driving simulator, a 3D model of the roads is generally sufficient to be able to drive. However, driving in an empty world is not really immersive and representative of a real world driving. That's why traffic is a major feature. While it's not currently operational, we paved the way to its implementation by adopting the OpenDRIVE standard (VIRES, 2006), for describing the road network.

Traditionally, most driving simulators have used their own proprietary formats for the logic description of the road system, which has made it impossible to share road data between simulators. In 2006, an initiative to create an open format for the road description was launched by VIRES GmbH. Based on XML format, OpenDRIVE contains almost all key features of real road networks, including road geometry, lanes, junction and objects. This data is essential for an eventual traffic module as it contains the required information to maneuvers all the autonomous virtual vehicles that would populate the scene. Also, OpenCRG (VIRES, 2008), a road surface description has been integrated in OpenDRIVE to support vibration simulation and the evaluation of the passenger comfort which makes OpenDRIVE a feature rich solution.

In the current state of OpenSD2S, a basic programming interface with OpenDRIVE has been integrated. It allows parsing an XODR file (the extension of OpenDRIVE) to access.

Road generation

The core of OpenSD2S aims at rendering the driving sensations by means of realistic sound, graphics and dynamics simulations. One of the final purposes is to speed up the design of a virtual driving experience. However, OpenSD2S is just the last link in this process and content creation is still required in the early phases of design. Different contents are required for a full simulation: car dynamics characteristics, according engine sound, 3D model of the cockpit and virtual driving circuit.

The virtual driving circuit creation is the major issue as it requires some artistic competencies and a long creation process which can take months for a single artist. Besides, some additional data are needed to feed the road database such as road logical network or even Bezier patches for future use by the dynamics engine, all of this can be quite constraining for a tiny structure. One might be tempted to reuse the resources of a video game but this content is copyrighted. Another solution would be to use open database that are available on the Internet and which propose free 3D contents. However these contents are very few, contain no circuits or very basic ones of poor quality, are dependent on a specific file format and lack realism. The main goal of simulation is to reproduce reality. Driver behavior studies are done on real situations. Based on this observation, it appears that the best way to reproduce the reality would be to digitize it.

A tool for visual database generation based on OpenRoadEd – an OpenDRIVE file generator – and drawing its data from various open cartographic sources such as OpenStreetMap (OpenStreetMap contributors, 2004) and SRTM (Hennig, T., et al., 2001) is under development at Theoris. OpenStreetMap is a collaborative project to create a free editable map of the world while SRTM (Shuttle Radar Topography Mission) is a NASA mission conducted to obtain elevation data for most of the world. These data are merged to generate a 3D model of the road as well as the associated description of the road network. The main advantage is the speed of the process as it allows selecting a geographical area and exporting it directly in an OpenSD2S suitable file format, ready to go. Also the road network is coherent and its geometry respects the norms of curvature while an artist design of a road can be unrealistic. At this stage, roads can be exported and buildings are roughly modeled as generically texturized blocks.

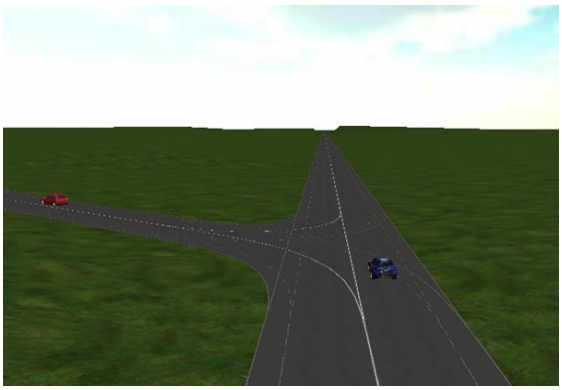

Figure 3: A road generated from OSM data with cars along the road

APPLICATIONS

OpenSD2S has already been used at Renault for Eco-driving performance assessment with in-car visual and haptic feedback assistance (Parodi-Keravec, A., et al., 2011). Thanks to its flexibility and these new features, it is now used for road driver and traffic research experimentation, such as at University of Lyon, where a driver cognitive load study is carried out during road exit maneuvers. In each case, the hardware configuration was an affordable solution for such experiments, using a mainstream gaming steering wheel, pedals and a standard driver seat.

FUTURE WORK

In the current version, autonomous vehicles follow the road network while respecting speed limits. Future work will be focused on providing comprehensive interactive traffic where all vehicles will communicate with each other and take into account the rules of the road. Another major feature is a scenario building GUI. Currently if one wants to add some special events according to a scenario, he must program it directly into the code which lacks flexibility. The aim is that OpenSD2S can be used without any technical knowledge. Finally a data logging module for further analysis is required as it is useful while conducting experiments.

CONCLUSION

The article presents an overview of the current state of OpenSD2S, a highly customizable, modular and scalable open source driving simulation software as well as its future features. Based on open standards, it's meant to be an affordable solution to meet the needs of both industry and researchers. A running version featuring a vehicle dynamics engine, a cluster-ready visual module with sounds has been integrated while some new features are on their way such as road network generation and traffic. OpenSD2S is already used by some academics, benefits a dynamic user community and is under sustainable development. The simulator is licensed under the LGPL and thus can be used in a commercial product. The latest version can be found on http://opensd2s.sourceforge.net.

ACKNOWLEDGEMENTS

The authors would like to thank Nicolas Fillard who contributed strongly in the early stages of development, and Sebastien Gerin, Jean-Luc Martinez, Joel Vedrenne, Julien Ryard and Vincent Benedet of Arts & Métiers ParisTech for helpful comments.

REFERENCES

Romano, R.A., (2000), Realtime Driving Simulation Using A Modular Modeling Methodology, *SAE Technical paper*, Series No. 2000-01-1297

Kemeny, A., (1993). A Cooperative Driving Simulator, *Proceedings of the International Training and Equipment Conference*, London, May 1993, pp.67-71

Kemeny, A., & Reymond, G., (1994). Automotive training simulator for driver safety enhancement, *Proceedings of the Image VII Conference*, Tucson, June 1994, pp.113-118

Lacroix, B., Mathieu, P., Kemeny, A., (2013). Formalizing the Construction of Populations in Multi-Agent Simulations, *Journal of Engineering Applications of Artificial Intelligence*, Vol. 26, no. 1, pages 211-226, 2013

Allen, R.W., Park, G., Cook, M.L., Rosenthal, T.J., & Aponso, B.L., (2004). Results and Experience from a Large Novice Driver Training Study, *Proceedings of the DSC Europe Conference 2004*, Paris, September 2004

von Neumann-Cosel, K., München, T.U., Dupuis, M. (2009), Virtual test drive - Provision of a consistent tool-set for [D,H,S,V]-in-the-Loop, *Proceedings of the DSC Europe Conference 2009*, Monte-Carlo, Monaco

Kemeny, A., & Panerai, F., (2003) Evaluating perception in driving simulation experiments, Trends in Cognitive Sciences, 7(1):31-376

Streeting, S., Walsh, J., & Johnstone, B. (2000). OGRE: Open source 3D GRaphics Engine (Version 1.8) [Software]. Available from http://www.ogre3d.org

Gropp, W.D., (1995). MPICH: High-performance and Widely Portable Message-Passing (Version 2) [Software]. Available from http://www.mpich.org

Free Software Foundation (2007), LGPL: GNU Lesser General Public License (Version 3). Available from http://www.gnu.org/licenses/lgpl.html

Filliard, N., Icart, E., Martinez, J.-L., Gérin, S., Mérienne, F., & Kemeny, A., (2010). Software Assembly and Open Standards for Driving Simulation, *Proceedings of the DSC Europe Conference 2010*, France, Paris

Clark, L., Glendinning, I., & Hempel, R. (1994). The MPI Message Passing Interface Standard. Technical report, Edinburgh Parallel Computing Centre, The University of Edinburgh.

Loki Software (2000), OpenAL: cross-platform 3D audio API [Software]. Available from http://connect.creativelabs.com/openal

The MathWorks Inc. (2000), MATLAB (Version 6.1) [Software]. Available from http://www.mathworks.fr/products/matlab

Smith R. (2001), ODE: Open Dynamics Engine [Software]. Available from http://www.ode.org

VIRES (2006), OpenDRIVE: managing the road ahead. Available from http://www.opendrive.org

VIRES (2008), OpenCRG: Open Source Road Surface Description Format, Available from http://www.opencrg.org

OpenStreetMap contributors (2004), OpenStreetMap: Free Editable Map of the World, http://www.openstreetmap.org

Hennig, T., Kretsch, J., Salamonowicz, P., Pessagno, C., & Stein, W., (2001). The Shuttle Radar Topography Mission, *Proceedings of the First International Symposium on Digital Earth Moving 2001, Springer Verlag*, London, UK.

Bernard, J.E., & Clover, C.L., (1995). Tire Modeling for Low-Speed and High-Speed Calculations, *SAE Technical Paper* 95031

Zhang, Z.P., & Zhang, J.Z., (2012). Drawing Engine Universal Performance Characteristics Map Method Based on MATLAB, *Advanced Materials Research*, 614-615, 361

Pacejka, H.B., (2002). Tire and Vehicle Dynamics, *Butterworth Heinemann*, Oxford

Bățăuș, M., Maciac, A., Oprean, & M., Vasiliu, N., (2011). Automotive Clutch Models for Real Time Simulation, *Proceeding of the Romanian Academy*, 12, pp. 109-116

Parodi-Keravec, A., Azzi, S., Filliard, N., Vailleau, B., Icart, E., Kemeny, A., Mérienne, & F., Martinez, J.-L., (2011). Eco-driving performance assessment with in-car visual and haptic feedback assistance, AFRV 2011