Vehicle Validation Visualization

Mikael Nybacka [1], Tomas Karlsson [2], Tobias Larsson [3]

(1) : Luleå University of technology
Division of Computer Aided Design
SE-97187 Luleå
+46920491698
E-mail : mikael.nybacka@ltu.se

(2) : Agency9
Aurorum Science Park 7
SE-97775 Luleå
+46706327799
E-mail : karlsson@agency9.se

(3) : Luleå University of technology
Division of Computer Aided Design
SE-97187 Luleå
+46920493043
E-mail : tobias@ltu.se

Abstract: The increasing complexity of embedded systems in the automotive industry calls for more efficient testing and verification processes early in the development phase. This paper presents a visualization application for distributed real-time vehicle validation. Real-time data from vehicles are transmitted via wireless network from the test track to a simulation framework. The processed data from the simulation are distributed to multiple web-based visualization clients. The Java based client presents data in a rich 3D environment, thus enabling non-experts to understand the dynamic behavior of complex vehicle systems. This application will contribute to an enhanced validation data presentation and the ability to collaborate in a distributed real-time virtual environment.

Key words: Virtual environment, distributed simulation, collaborative environment, real-time visualization, automotive validation.

1- Background

Due to the increasing complexity of embedded systems and software in vehicles, the automotive industry faces an increasing need to test and verify components and subsystems under realistic conditions. At the same time, vehicle manufacturer development cycles must be shortened to be competitive on the global market, and an increased amount of testing and verification must therefore be performed in less time. However, simply increasing the testing volume can be prohibitively costly, meaning that testing and verification processes must be made more efficient.

Since suppliers and subcontractors from all over the world frequently need to be involved in the testing and verification of vehicles, developing sophisticated methods and tools for distributed validation and simulation is necessary as well as incorporating these tools and methods into the overall framework for distributed product development become necessary. Vehicle testing, like winter testing, is often dislocated from the actual development centers, creating an even greater need for distributed technology support. The verification process of winter testing can be reduced if working in a real-time environment which will accelerate the product development process due to its real-time nature, and promote instant analysis of measurement and simulation data. To further improve the testing and validation process requires an application that can follow the validation and be capable of visualizing a multitude of data in real-time [NL1].

Winter testing facilities need to have the necessary wireless local area network (WLAN) infrastructure to work in a real-time environment. By using WLAN at the test tracks, live measurement data can be sent from vehicles at the test track back to the development site of the manufacturer for analysis and visualization, in real-time if desired. The possibility to share and view data in real-time from the test track will significantly reduce costs and lead-times for vehicle manufacturers. [NL1] [T1].

1.1 - Industry needs

Because engineers and decision makers are geographically dispersed during winter testing, a distributed environment that enables collaborative work in a virtual environment is needed. To increase the availability and decrease the Total Cost of Ownership (TCO) the system should preferably be web-based and available on a multitude of hardware, such as regular desktop PC's and Linux/Unix workstations.

The collected data at test sessions are often displayed on tabled data and graphs, making it difficult for both engineers and non-experts to understand the behavior of the whole system. Working in a rich 3D environment is needed to simplify the presentation, and thus give a better overview of

the system's behavior and performance.

2- Visualization concept

According to [G1], people perceive information primarily through vision. Therefore, if with the help of visualization aids, engineers can see the behavior of the vehicle, while seeing regular data presented in graphs or tables, more information from the test will be perceived. This is why real-time 3D rendering of test data is important to implement in the vehicle industry.

2.1 – Current test procedure

A typical test scenario consists of a numerous distributed groups of expertise. The groups are decision makers, system experts, test drivers and test entrepreneurs, all who have to work in a collaborative environment.

At a typical test one or more vehicles are driven at the test facility. During the test different types of sensors record and sample data from the test-drive. Logging of the data is done independently in each vehicle and the data are gathered from the vehicles after the session. A system expert will then analyze the data, which are often in tabled or graph form. Thereafter, the expert will write a report concerning the data and the findings of the test. The test process is quite sequential and the test scenarios are conducted in steps; test driving, analyzing the data, writing a report and decision-making.

Several problems can occur during the steps of a test day, such as during the test drive when the sensors are recording and logging data, the test drivers do not receive any direct feedback of the recording process. A problem during the test drive is only noticed during the gathering and analyzing of the test drive data. The problems can range from failing sensors, systems and human errors. Because the test drivers do not receive any feedback, they do not have the ability to directly change the test drive or the system to analyze the situation.

The system's expert has to wait until the test drive is finished before he can start analyzing the data. If the system's expert finds something that may need further testing, a new test drive with the same situation must be conducted the following day, which can be difficult. It is hard to reproduce the same situation when performing a retest the second time around the following day, since the conditions of the test track may have changed.

2.2 - The future test procedure

The approach taken in this work is to create a distributed real-time visualization application as a tool for vehicle tests. The idea is to create an optimum application for automotive testing, where engineers can work in a distributed environment.

In our test scenario the test drivers log-on via their laptop to a secured website where the visualization application is hosted. The sensors on the car log data locally as well as transmit data via WLAN to a data management server, see Figure 1. During the test, the driver follows the test virtually on his laptop in a 3D environment. The application gives feedback of the data transmitted from the sensors and the result of the actual test. The direct feedback not only verifies for the test driver that the actual test is being logged, but also gives the test driver the ability to see the result of the test in 3D. The test driver can directly find extreme system behavior and potential errors due to direct feedback of the results. Preferably the test driver is in a direct link with other test drivers who also have the same application to discuss and verify results.

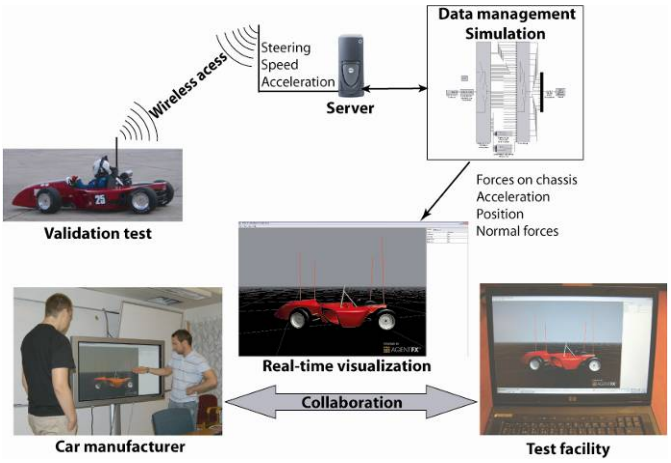

Figure 1: Future Vehicle Validation.

During the test the system expert monitors the test drive. Due to the easy-to-understand nature of real-time 3D representation [G1], the system expert can directly analyze the test data [NL1]. The system expert is also preferably connected via direct link to the test driver, so that they can discuss and analyze the data. Based on the analysis the system expert can conclude if a retest during the same conditions is required. The system expert can then directly instruct the test driver of the issue and order a retest from the test driver or one of the concurrent test drivers located in a similar location.

Due to the distributed nature of the application the location of the system expert is irrelevant. Another benefit of the distributed paradigm is that the system expert can invite other experts from other locations to directly verifying or discussing a problem without the need to stop the actual test.

For important decisions during the test the decision maker can be invited to render a decision about the findings during the test, such as increased budget for further research and testing.

2.2.1 – Distributed

With advanced 3D engines, such as AgentFX™ [A1], Java [J1] is a natural choice when creating web-based distributed environments. Due to the platform independence of Java an application can be executed on any computer regardless of operating system and CPU. Java also provides manage memory architecture which makes development phase less error prone.

An example of an optimum application is when engineers do not have to be on site, but instead can follow the test from their home office, reducing the total cost of testing (TCT). It is possible with the use of Java to web distribute a complete

application through the use of Webstart™ [J2]. With Webstart™, a complete Java application can be hosted on a secure web server. To access the application the user logs-on to a secure web server via a regular web browser. To start the application the user clicks on a link to the application, which is automatically downloaded and cached on the client. The process of web distributing removes the need to install a certain application on every client, and enables to ad-hoc invite external users. By removing the regular save, save as command for local storage, and replacing it with automatic persistence to a back end server, a user can access its current workbench anywhere in the world regardless of computer.

2.2.2 – Data acquisition

During the test sessions CAN-bus data are recorded from vehicle to data acquisition devices like Host Mobility MX-16 [H1], Ipetronik Data Logger [I1] or of a similar type with built in WLAN. In test facilities with WLAN infrastructure it is possible to send data directly from the test vehicle, and thus possibly view data from the car in real-time. This data can also be used to perform simulations in real-time, as further explained in 3.2.3.

A telematic test range was built in northern Sweden at Arctic Falls as a testing ground that facilitates distributed data acquisition from vehicles. This type of test facility makes it possible for automotive manufacturers and suppliers to perform distributed real-time validation of their products. [T1]

There are a number of different test scenarios concerning vehicle validation, all of which need to be accounted for in a vehicle validation application. The visualization application has to be able to present data from vehicle dynamics, climate control, vibration, etc., and therefore handle different types of inputs.

2.2.3 – Simulation

Simulation during product development and validation processes has to be used more and more efficiently. Simulation software enabling real-time performance is constantly evolving. This allows us the use of MBS software to simulate the dynamic responses of the vehicles behavior in real-time with input from the vehicle at the test site, and give engineers more information of the vehicle's performance. For example, the normal forces acting on the tires and the distribution of the forces on the four wheels can say a lot about the vehicle's stability and performance. And this is impossible to measure on a moving vehicle in real-time; hence, real-time dynamic simulation software is a necessity.

2.2.4 – Present graphics possibilities

The graphics market has seen an explosion in FLOPS the last decade. The market is driven by the ever-increasing demand of realistic environments in the entertainment industry, where visual quality meetings exceeding what was once only available in blockbuster movies are now common in modern 3D games. Standard PC has replaced the workstation as the best for real-time 3D content.

Today, triangles are the de-facto rendering real-time graphics primitive, as opposed to complex surfaces such as NURBS. Luckily, most modern Digital Content Creation (DCC) tools, such as Autodesk Maya [A3], support ways to take CAD models like STEP files and convert them to triangle models for more suitable real-time purposes.

With the introduction of programmable GPU hardware it is now possible to create lifelike realistic surfaces. The use of advanced rendering techniques can be used to introduce visual clues often needed to determine distances, material properties, etc.

2.2.5 – Rendering possibilities

We have the ability with 3D engines to display a complex object in a virtual environment, since we work in a 3D environment and are not limited to static positions, and can therefore offer a number of viewpoints such as free flight camera, drivers view, bird's eye, etc. We also control the projection and can offer the user different projections, e.g. orthographic or perspective depending on the situation. The complex objects can be further enhanced with textures to display surface attributes or to code measurement data on the vehicle, e.g. temperature distribution in the vehicle.

The image can be further enhanced by advanced lighting equations, such as per pixel phong shading that simulates a material's ability to reflect direct light [B1]. To enhance the visual appearance of materials such as chrome, environmental mapping techniques can be used to create realistic reflections [M2][WD1]. The depth perception can be enhanced with real time shadow techniques like shadow maps [R1] and shadow volumes [AW1], see Figure 3. The hierarchal structure can be used to visualize the chassis roll, pitch and squat with reference to the ground.

With the use of programmable shader, we can create surfaces that dynamically change on input. The input can be linked to sensors, simulations or be user controlled.

We also have the possibility to set up a clipping plane, thus allowing the end user to look at the interior of a car. By combining the complex 3D graphics with informative 2D overlays we can create an informative environment where we can follow test drives, see tabled data and still work in a 3D environment.

The environment can also be augmented with informative data, such as billboard and icons, for failure detection and display information of censors, or as in Figure 4 where the force on the tires is displayed as force axis.

3- Framework

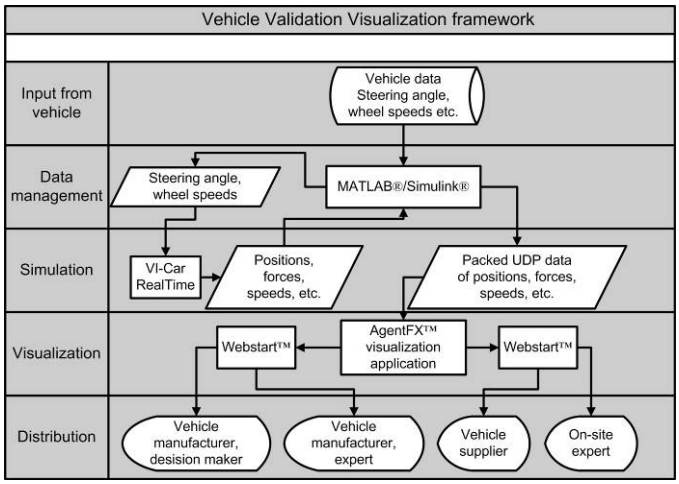

Figure 2: DRTSV framework.

The framework uses software that automotive companies already have licenses for, reducing the investment costs. They are also familiar with the software which makes it easy for them to implement within the organization. Figure 2 shows an overview of the Distributed Real-Time Simulation and Visualization (DRTSV) framework.

3.1 - AgentFX™

AgentFX™ is an advanced polygon based scene graph 3D engine dedicated to high quality real-time 3D graphics. AgentFX™ offers advanced 3D rendering capabilities such as picking, clipping planes, programmable shader and shadows. A unique advantage of AgentFX™ is that the user has complete control over the rendering process, allowing the creation of advanced user interface, with 2D overlays and augmented 3D views. With the use of COLLADA [K1], AgentFX™ can import models from a large selection of DCC tools.

Figure 3: Example of advanced 3D rendering, shadows and carbon fibre texture.

3.2 – MATLAB/Simulink

MATLAB/Simulink is used as the communication interface between VI-Car RealTime, the visualization tool AgentFX™ and the real-time data input from the car. The Simulink model collects input data to the real-time model from the car's CAN-bus (Steering, wheel speeds etc.). Simulink passes the data onto the MBS solver for dynamic calculations of the RealTime model. The MBS solver then passes the results back to the Simulink model, which packs the data needed for visualization into units suitable for network communication, and then sends it to the visualization application; See Figure 2.

3.3 – VI-Car RealTime

A formula SAE car was modeled in ADAMS/Car and exported to VI-Car RealTime. The ADAMS/Car model has 71 degrees of freedom (DOF) and the VI-Car RealTime model has 16 DOF. The RealTime model does not have linkages or bushings, and its steering system does not have parts for the steering wheel or rack. Instead, the model requires input parameters from, e.g. Kinematics and Compliance (K&C) test machine or data virtually obtained from the ADAMS/Car simulations. This makes it possible to run the dynamic simulation faster than real-time [M1].

3.4 – Architecture

The architecture of the DRTSV concept consists of MATLAB/Simulink, VI-Car RealTime and AgentFX™. MATLAB/Simulink receives data via UDP from the vehicle's CAN-bus, which are modified and sent to VI-Car RealTime solver via socket port. VI-Car RealTime calculates the dynamics of the vehicle and outputs various data to MATLAB/Simulink, e.g. vehicle and tire positions, forces, speeds, etc; see Figure 2. The data are processed in MATLAB/Simulink and passed to the AgentFX™ visualization application where the vehicle is rendered in a rich 3D environment. Additional data like contact forces on the wheels are also displayed with arrows that represent the X, Y and Z forces on the wheel (see Figure 4). The visualization application can be reached anywhere in the world through a web browser due to Java web start.

The application communicates over a custom UDP protocol, where the data are transmitted to the test client. The client streaming network is implemented on the Apache MINA network environment [A2]. MINA extends the Java's own network API and acts as a Rapid Application Development (RAD) tool for server/client development.

The test data can be recorded and stored. The system offers automatic persistence, where every clip is stored in a back end database and searchable on time, event or user. By removing the regular save, i.e. save as command for local storage, a user can access its current workbench anywhere in the world regardless of computer.

4- Visualization prototype

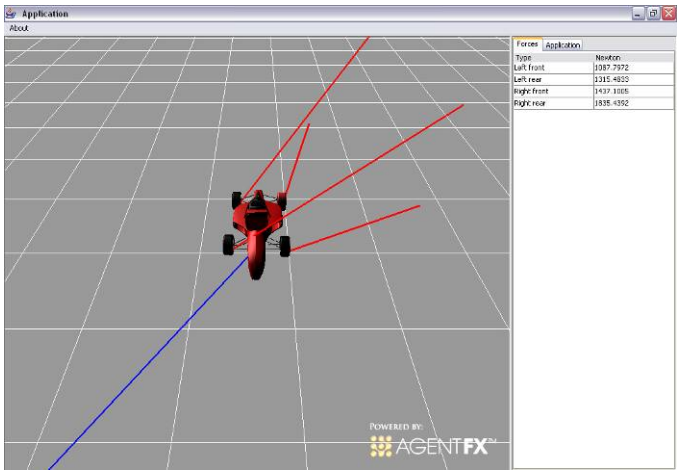

Figure 4: Prototype of the V³ application.

The prototype of the visualization application V³ consists of one dynamic handling overview that is visualized in a rich 3D environment. The application can be further enhanced with plug-ins for handling temperature, sound, vibration and other areas of interest. Figure 5, shows an example of how a user modified dynamics overview can look like, multiple views and data representation in graphs to the right.

In the dynamic handling overview the user can see the force distribution, displayed with red arrows, on each tire. The force distribution is also listed in a force table in the upper right corner of Figure 4. When reaching a certain amount of slippage, the vehicle's tires will turn red to indicate that the threshold has been reached. A blue arrow will indicate where the vehicle is heading and become elongated as the vehicle increases speed.

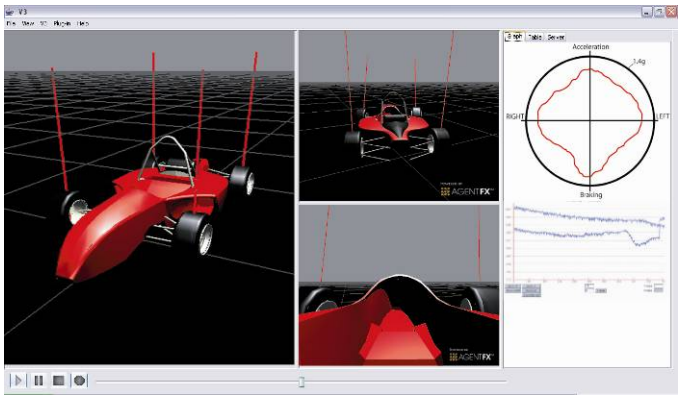

Figure 5: Example of future V³ application.

An example of possible plug-ins, the user can add temperature, sound and vibration overviews to the application. In the plug-in the user can control a clipping plane through the 3D model to see the distribution of temperature, sound or vibration. The user places control points in the model that are located in the same positions as in the real vehicle. Data will be sent from the real vehicle to the visualization application and the value of the specified point in the 3D model will be interpolated between each other to form a color scheme in the clipping plane. This will help the user view the distribution of temperature, sound or vibration in the clipping plane of the whole vehicle. Törlind [T1] explains this in a similar fashion.

5 – Future application

The visualization application will hopefully be tested in a real vehicle validation expedition in northern Sweden during the winter of 2006-2007, along with the need for additional functionalities in the application to be examined.

The possibility to present and visualize data from multiple test sessions in the same application will simplify the work process concerning distributed vehicle validation. The application must also be prepared for future test scenarios or customer specific scenarios, since it must be open for the customer to add their own layout of test scenarios.

Other areas besides the automotive industry that can make use of the visualization application will be examined, e.g. the mining industry to monitor their vehicles.

5.1 – Future rendering possibilities

The current generation of the GPU has been dedicated to rendering polygon data. The pipeline is highly optimized at handling triangles [G2]. However, triangle based models are not directly compatible with many of the CAD surface based models without the need of an intermediate step through a DCC tool. With more complex surface based models the triangle count can be become huge and result in heavy loading times and memory requirements. Large GPU manufactures have suggested that the graphics pipeline will be extended with a programmable geometry step with the release of OpenGL 3.0 [K2]. Whether this step will bring acceptance of more complex surfaces in real-time 3D graphics remains to be seen.

6 – Conclusions and discussion

This paper presents a Visualization application (V³) for Vehicle Validation. The application will contribute to enhanced validation data presentation and the ability to collaborate in a distributed real-time virtual environment. The visualization application will also help engineers see the behavior of the vehicle and in the same time see regular data presented in graphs or tables.

The V³ application can also be used during Hardware-In-the-Loop (HIL) simulations where the systems engineers want to se the behavior of the car running on the new hardware and software. With the V³ application the embedded systems engineer do not have to be located at the same place to see the results from the HIL simulation, they only need to log in via the web to see the simulation in real-time.

With the use of Java, it is possible to web distribute a complete application through the use of Webstart™. With Webstart™ a complete Java application can be hosted on a secure web server. The process of web distributing removes

the need to install a certain application on every client, along with enabling external users to be invited ad-hoc.

The use of Matlab/Simulink together with VI-Car RealTime allows for all data and data communications to be handled as well as simulate vehicle dynamics faster than real-time. The software's are also already used in industry, making it easy for the industry to start using Distributed Real-Time Simulation and Visualization. The combination of using Matlab/Simulink and VI-Car RealTime together with AgentFX™ has proven to be a successful combination. From the initial testing we see that the V³ application gives the user a better overview of the system's behavior.

We suggested an optimal application where the engineers and decision makers can work in a distributed real-time 3D environment. With the V³ application, we have made this possible. The remaining task is to validate whether the V³ application can enhance the future test procedure.

7 – Acknowledgements

Funding from Center for Automotive System Technologies and Testing through Norrbottens Forskningsråd and the support from Agency9 AB are greatly acknowledged.

8 – List of terminology

V³	Vehicle Validation Visualization
WLAN	Wireless Local Area Network
TCO	Total Cost of Ownership
CAN	Controller Area Network
MBS	Multibody System
DCC	Digital Content Creation
UDP	User Datagram Protocol
API	Application Programming Interface
RAD	Rapid Application Development
DRTSV	Distributed Real-Time Simulation and Visualization
TCT	Total Cost of Testing
HIL	Hardware-In-the-Loop

9 – References

[A1] AgentFX, accessed 20060901, http://www.agency9.com

[A2] MINA, accessed 20060901 http://directory.apache.org/subprojects/mina/index.html

[A3] Autodesk Maya, accessed 20060901 http://usa.autodesk.com/adsk/servlet/index?siteID=123112&id=7635018

[AW1] Aldridge G, Woods E. Robust, Geometry-Independent Shadow Volumes, 2004, p. 250-253, ISBN:1-58113-883-0.

[B1] J. F. Blinn. Models of light reflection for computer synthesized pictures. In Computer Graphics (SIGGRAPH '77 Proceedings), pages 192–198, July 1977.

[G1] Geisler G. (1998). Making Information More Accessible: A Survey of Information Visualization Applications and Techniques.

[G2] GPU Gems 2, 2005, p. 471-491, ISBN 0-321-33559-7.

[H1] Accessed 20060901, http://www.hostmobility.com

[I1] Accessed 20060901, http://www.ipetronik.com

[J1] Accessed 20060901, http://java.sun.com

[J2] Webstart™, accessed 20060901 http://java.sun.com/products/javawebstart/index.jsp

[K1] Accessed 20060901, http://www.khronos.org/collada/

[K2] OpenGL upcomming futures, accessed 20060901 http://www.khronos.org/developers/library/siggraph2006/OpenGL_BOF/

[M1] Accessed 20060901, http://www.mscsoftware.com,

[M2] Moller T., Haines E., Real-Time Rendering, 2002, p. 155-174, ISBN: 1-56881-182-9.

[NL1] Nybacka M., Larsson T., Johanson M., Törlind P. Distributed Real-Time Vehicle Validation. Proceedings of ASME IDETC/CIE 2006. DETC2006-99154.

[R1] Rege S. Shadow considerations. Accessed 20060901, http://download.nvidia.com/developer/presentations/2004/6800_Leagues/6800_Leagues_Shadows.pdf

[T1] Törlind, P., A Collaborative Framework for Distributed Winter Testing. In proceedings of eChallenges e-2004. October 2004: Vienna, Austria.

[WD1] Whyn C. Dietrich S., Cube Maps, accessed 20060901, http://developer.nvidia.com/attach/6666.

