

PRESENTING THE RESULTS OF VEHICLE DYNAMICS SIMULATION IN VIRTUAL 3D ENVIRONMENT

Miha Ambrož, B. Sc.

Prof. Dr. Ivan Prebil

University of Ljubljana, Faculty of Mechanical Engineering, Centre for Element and Complete Design Modelling, Slovenia
142

ABSTRACT

Results of simulations done by mathematical models of driving dynamics have to be presented in a clear and appealing way. One possibility of this is presentation by means of animation in virtual 3D environment. For the mathematical model of control and riding dynamics of road vehicles, developed at CEMEK, University of Ljubljana, a library of low-polygon 3D geometrical vehicle models is being built. Animations are written in VRML2 and are composed of mathematical model output data, vehicle geometry and animation core. We have developed software to compose the animations, write them programmatically and display the results in virtual 3D environment. The required vehicle-specific input data is retrieved from an existing vehicle database. Currently, the mathematical model simulates riding control and riding dynamics, and deals with vehicles as systems of rigid bodies. The flexible structure of software for writing animations and of the animation core itself provides easy adaptation to possible future expansions of the mathematical model.

INTRODUCTION

To display the results of a driving dynamics simulation as an animation in virtual 3D environment, the vehicles, the driving surface and other involved objects have to be shown realistically. Therefore each vehicle has to be represented by a geometrical model, modelled upon a real vehicle. Since an integral library of low polygon-count 3D vehicle and road object models isn't yet commercially available, we have decided to start building our own.

One of the goals was to find a way to quickly and cheaply model geometrical models of vehicles, suitable for use in animations.

Two starting points were set:

- Vehicle models have to be modelled in enough detail to be recognisable from every point of view in virtual 3D space.
- The number of vertices (and thus the number of polygons) the vehicle model consists of, has to be kept as low as possible to enable display of animations on the widest possible range of hardware.

To achieve balance between these two contradicting requirements, special care was taken to model the distinguishing features of each vehicle with greater precision, while attempting to simplify the non-distinguishing ones. A few examples of such 3D vehicle models on a model of a driving surface are shown in Figure 1.



Figure 1

Examples of 3D vehicle models.

POSSIBLE WAYS OF APPLICATION

The primary use of a library of 3D geometrical models is in presentation of vehicle dynamics. This includes visualisation of traffic accidents and driving dynamics and preparing images for various reports and analyses.

The secondary use of 3D geometrical models of vehicles is in determining traces and damage on vehicles and other objects during minor collisions (Figure 2). This can find its use in investigating insurance frauds and similar, and requires geometrical models of sufficient accuracy.



Figure 2.

Presentation of a collision in virtual 3D environment.

PREPARING THE 3D VIRTUAL ENVIRONMENT

Generating 3D Geometrical Models of Vehicles

Several options of achieving the stated goals were considered before the actual start of development of the procedures of generating the 3D model library:

- Generating 3D geometrical models from projections of objects, by means of manually transferring points into 3D space,
- Generating 3D geometrical models by employing 3D scanning of real objects,
- Generating 3D geometrical models from series of photographs by means of commercial software for photomodelling.

The first option requires accurate drawings of at least three parallel projections of each vehicle and was thus soon rejected as too time consuming. The second option was rejected due to lack of suitable equipment for 3D scanning of live-sized vehicles and poor results obtained by experiments with scaled-down models of vehicles on a desktop 3D scanner. The third option has proven as a relatively quick and reliable way of producing large quantities of 3D geometrical models of vehicles.

To make the 3D geometrical models of vehicles suit the requirements of the animation engine, a process of generating 3D VRML2 geometrical model from series of photographs (Figure 3) has been devised.

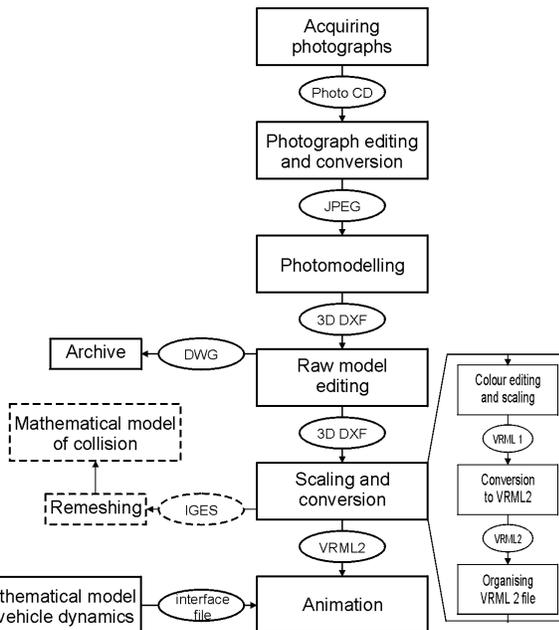


Figure 3.

Flow chart of generating geometrical 3D models.

The process consists of six main steps:

1. *Acquiring a set of digital photographs of the vehicle to be modelled.* This can either be done with a standard film camera or a digital camera. The benefit of using a digital camera is better and more consistent geometrical accuracy of photographs and the absence of having to digitise the photographs, while scanning negative films can yield higher image resolution and lower initial cost for equipment.

2. *Editing and converting the photographs into a format suitable for photomodelling.* In this step the digital photographs are edited (colour equalisation, sharpening

etc.) and converted to a format, suitable for input into the photomodelling software.

3. *Photomodelling.* By using photomodelling software (commercial products such as AEA Technology Geometra or EOS Systems PhotoModeler), in this step the data from the photographs is processed and used to generate 3D geometry. This step is the central part of the model generating process and represents around 70 % of the time used for creating a model. Since most ordinary road vehicles are longitudinally symmetrical, we usually model only one longitudinal half of a vehicle to conserve time and reduce the needed number of photographs. As a result of this step we get a raw model of one half of a vehicle (Figure 4). The term "raw model" in this case refers to a model that only contains basic geometry and to which the information (such as colour and scale) has to be added during the further phases of the model generating process.

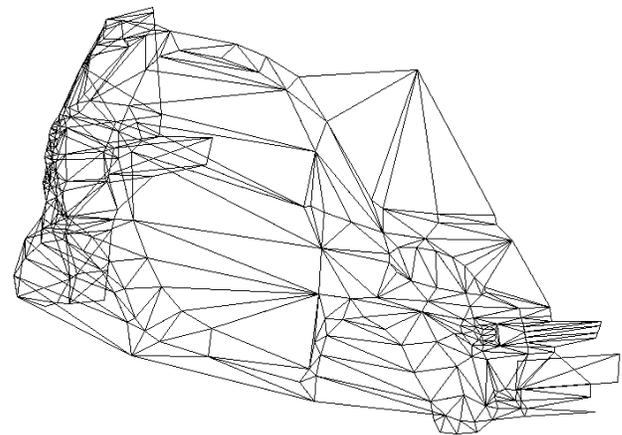


Figure 4.

Raw model of one half of a vehicle.

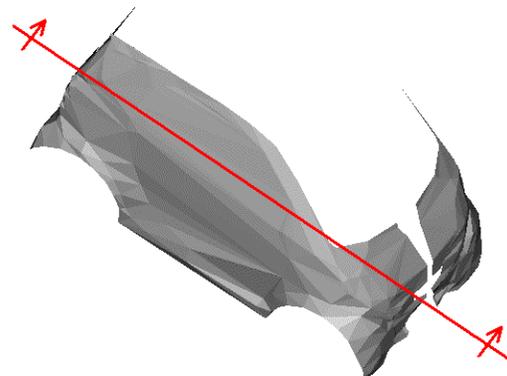


Figure 5.

Mirrored raw model of a vehicle.

4. *Raw model editing.* First, the raw model of a half of a vehicle has to be mirrored to get a raw model of the

entire vehicle (Figure 5). Additional features that cannot be photomodelled (vehicle bottom, hidden geometry of commercial vehicles etc.) are added to the raw model afterwards (Figure 6). This is also the step where we can correct any possible errors (such as missing or misplaced points) that may have been done during photomodelling.

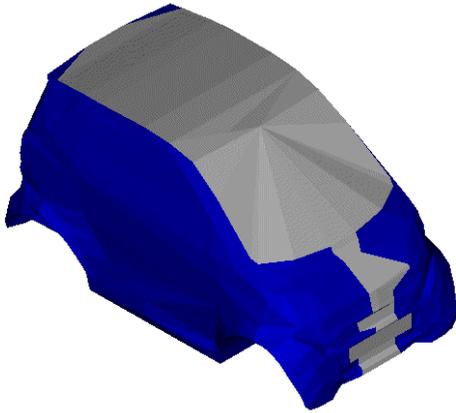


Figure 6.

Raw model of a vehicle with added surfaces that aren't photomodelled (grey).

5. *Scaling and conversion.* The raw model has to be properly scaled and coloured to reflect the size and the colour of the real vehicle (Figure 7). Special care is taken when assigning colours to groups of surfaces to make the vehicle model suit the colour changing mechanism of the animation core. Properly scaled and coloured model is then converted into VRML2 format and edited (this includes adding the "Driver's view" viewpoint and metacommands used by the VRMLPath programme interface). After this step the model is ready to be used in animation.

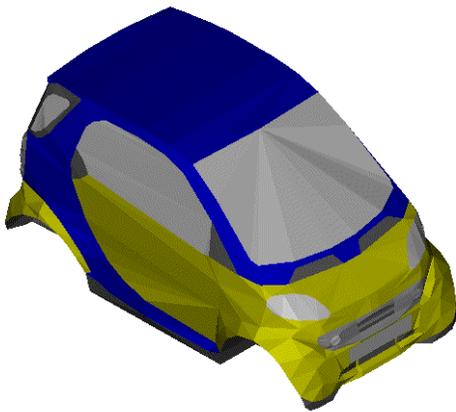


Figure 7.

Scaled and coloured model of a vehicle.

An example of a finished 3D geometrical vehicle model placed in a simple virtual 3D environment is shown in Figure 8.

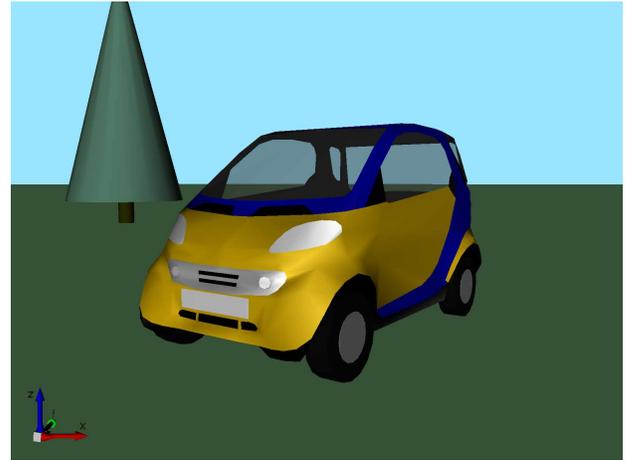


Figure 8.

3D vehicle model used in presentation.

THE ANIMATION ENGINE

There are several possible ways of creating animations from computed simulation data. These include:

- Using commercial software for dynamical analysis with its internal animation viewers.
- Preparing animations as non-interactive movies with generic animation software.
- Development of proprietary animation environment using standard 3D libraries (OpenGL, Direct3D).
- Development of an universal animation core in VRML and connecting it to a programme interface to automate the process of preparing animations.

After considering the benefits and downsides of each of the above options, VRML2 has been chosen as the tool for creating animations from the simulation results. The strongest reasons for this choice were:

- VRML2 is a public specification (Carey & Bell, 1997), which provides public availability of language references, ready-made solutions in form of protocols etc.
- Animations are stored in compact files. Objects and kinematics are stored as their geometrical descriptions, rather than sequences of large images.
- VRML2 is a structured language, which enables easy programmatic writing of animation files as well as easy manual debugging afterwards.
- Animations can be divided into objects and kinematics, which enables easy managing of the model library.
- VRML2 files are portable and can be used on any operating system as long as there is a VRML2 browser available for it.
- Animations are interactive (users can navigate the virtual world, change viewpoints etc.). Interactivity is

provided by the VRML2 browser itself and follows the same rules regardless of the browser used. The animation core, written in VRML2, has been made the base of the animation engine. The VRML2 specification includes language elements for creating keyframed animations. The animated objects can be anything from VRML intrinsic primitive shapes (boxes, cylinders, spheres, ...) to complex objects in "inlined" external files. A model of a conventional two-axle vehicle, for example, is composed of five objects: the bodywork and four wheels. These objects are visual representations of the five rigid bodies used in the mathematical model (Cigliarič, 1999). Each object can be animated independently. The animation core includes all the necessary means to describe the kinematics of the five bodies and provides placeholders for the geometrical models and the motion data computed by the mathematical model (Figure 9).

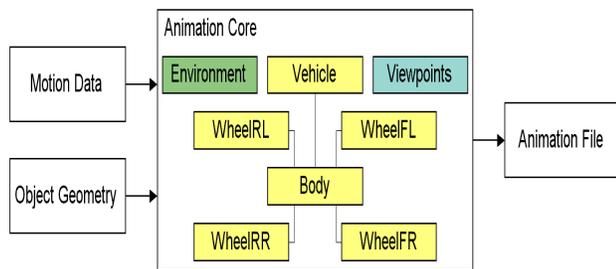


Figure 9.

Composition of the animation core.

The principle of animation in VRML2 is keyframing (Vince 1992), where known positions (keyframes) and orientations of objects on their respective motion curves are given in discrete time intervals and linear interpolation is done inbetween. In case of the presented animation core this is achieved by using interpolators of orientation and interpolators of position in conjunction with sensors that enable user interaction. An example sequence chart in Figure 10 shows an animation, where geometry is controlled with output data from two interpolators (orientationInterpolator and positionInterpolator), which generate output in the pace of timeSensor, which is controlled by touchSensor that activates on mouse click.

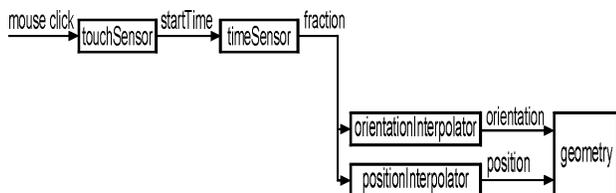


Figure 10.

Flow of animation.

For a perception of fairly smooth motion the animation has to run at at least 20 frames per second. This is

important, because the time interval at which the motion data is computed by the mathematical model has to be set according to this requirement. Smoothness of motion of an animation depends on the speed of the host system the animation is played on. If the animation is targeted to be used on the widest possible range of hardware then the length of the time interval between keyframes has to be set as to achieve balance between smoothness of motion, accuracy of display and capabilities of the hardware:

- too long a time interval may produce inaccurate animation because of the nature of linear interpolation (Figure 11),
- too short a time interval may overload the hardware and thus produce an animation that won't run in real time.

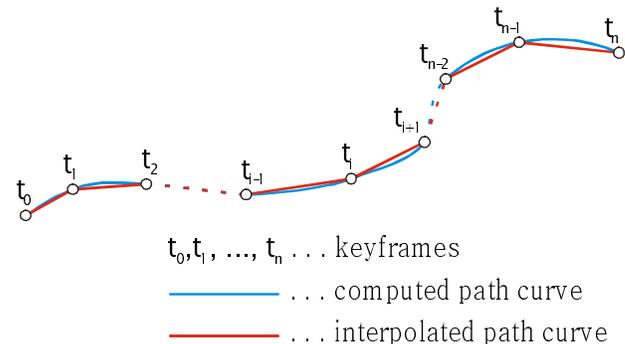


Figure 11.

Inaccuracy in motion curve because of interpolation.

Integration of mathematical model and animation

We have developed a Win32 application (called VRMLPath) that acts as the programme interface between the mathematical model of vehicle dynamics and the animation in virtual environment. It currently automates the procedure of creating animations to the extent where the user can:

- select a vehicle model from the list,
- display the vehicle model in the built-in preview window,
- manually edit the file with the vehicle model,
- select a pre-computed motion curve (or, if the required external modules are available, compute a new one using the current mathematical model),
- add a 3D model of a driving surface (as an external VRML2 file acquired from a GIS measurement),
- add custom objects (auxiliary objects, vertical signalisation etc.) to the animation and manage their position and appearance,
- prepare the animation and immediately display it in the animation window (Figure 12),
- write the animation into a file that can, in general, be transferred to any system and viewed with any VRML2 browser.

The user interface is designed to enable the user to accomplish these tasks quickly and is ready to provide

consistency throughout the application even with future expansions to the VRMLPath. Each step of the animation creation process is represented by a tab in the VRMLPath main window (Figure 13). This makes the user interface clear and any changes to input data easy.



Figure 12.

VRMLPath animation window.

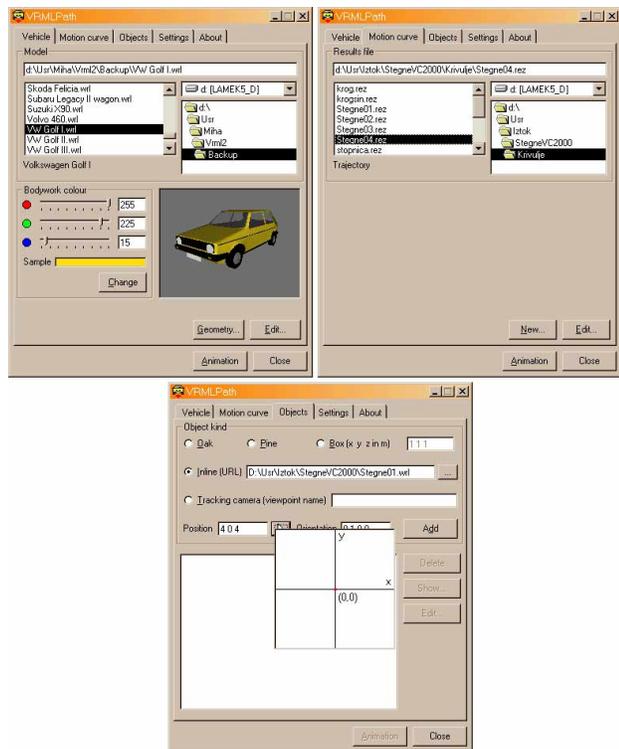


Figure 13.

VRMLPath main window - divided into tabs.

A flow chart of creating a typical animation with VRMLPath from the user's point of view is shown in Figure 14.

VRMLPath is built modularly, which makes it flexible in several ways:

- it is easy to adapt to frequent changes to the animation core during its test period,
- all the routines that interact with the mathematical model are in a separate module and can be changed without affecting the rest of code,
- it allows modifications and improvements to 3D object models without changes to the program code,
- it enables re-use of its modules in other applications,
- it enables integration of modules from other applications (e.g. the vehicle data window, Figure 15.),
- it enables implementation of new functions with minimal changes to the existing code.

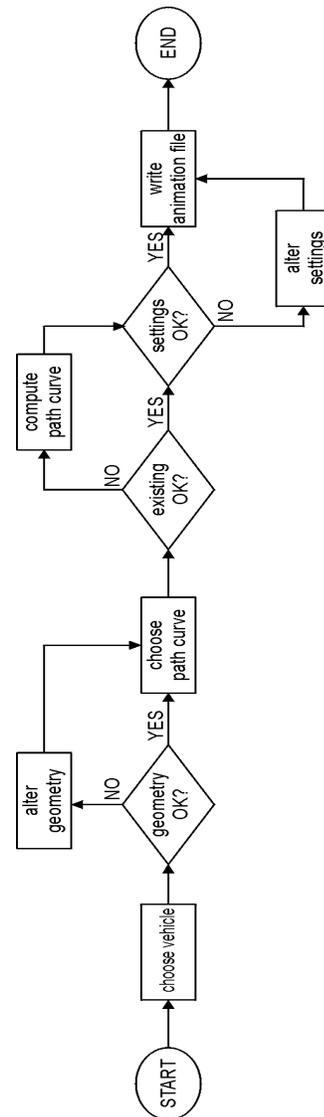


Figure 14.

Flow chart of creating an animation in VRMLPath.

VRMLPath is connected to an existing database of vehicle data. This connection is used to automatically retrieve the relevant data about the vehicle upon selection of its model. The geometrical data is presented

graphically in its own window that enables editing of data (Figure 15).

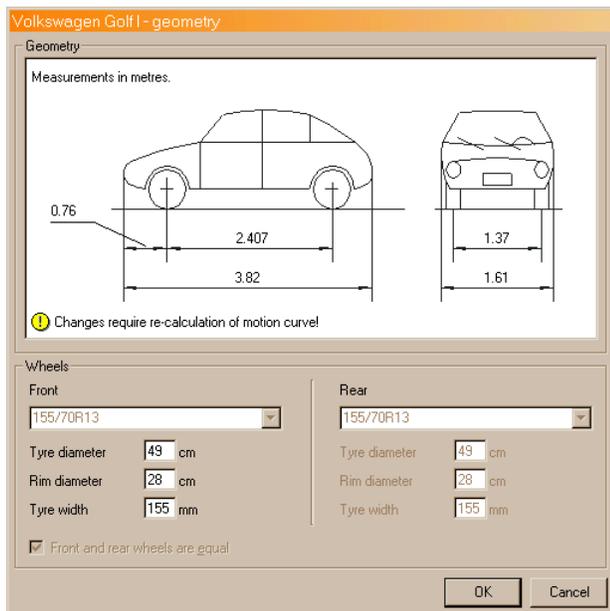


Figure 15.

VRMLPath: Vehicle data window.

CONCLUSION

The process of creating 3D geometrical models of vehicles has been developed to the extent where every land vehicle can be modelled and prepared to be used in animation within one working day using the routine procedures.

The current state of the animation engine and the programme interface enables quick and easy testing of mathematical algorithms and input data used by mathematical model of vehicle dynamics as well as the suitability of the animation engine itself. Future versions of mathematical model will include 3D model of

collision, which will require further development of vehicle models, the animation engine and the programme interface, to allow display and animation of deformable bodies.

The ultimate long-term goal of the work is to develop the current solution into a self-contained application, supported by a growing library of 3D models of vehicles and other traffic related objects.

Along with the development of the mathematical models, the future versions of VRMLPath will be developed to include modules for displaying animations of human body and its interaction with the vehicle and the driving surface.

REFERENCES

- Ciglarič, Iztok (1999). Approach to model development for the analysis of road vehicle dynamics, *Proceedings of IAT'99*, pp. 37-44, ISSN 1408-1679, Nova Gorica, April 1999, ZSITS, Ljubljana
- Carey, Rikk & Bell, Gavin (1997). *The Annotated VRML 2.0 Reference Manual*, Addison-Wesley Pub. Co., ISBN 0-12-249054-1, New York
- Vince, John (1992). *3-D Computer Animation*, Addison-Wesley, ISBN 0-201-62756-6, London

CONTACT DATA

Miha AMBROŽ, B. Sc., University of Ljubljana, Faculty of Mechanical Engineering, Aškerčeva 6, SI-1000 Ljubljana, Slovenia,
 E-mail: miha.ambroz@fs.uni-lj.si,
 Phone: +386 1 4771-127

Prof. Dr. Ivan PREBIL, University of Ljubljana, Faculty of Mechanical Engineering, Aškerčeva 6, SI-1000 Ljubljana, Slovenia,
 E-mail: ivan.prebil@fs.uni-lj.si,
 Phone: +386 1 4771-508

<http://www.fs.uni-lj.si/cemek>