Generic Model of Vehicle Dynamics for Low-Cost Simulators and Serious Games

Radoslav Stojic, Ivan Vujic, Milan Stojic

1 Faculty of Information Technologies, Metropolitan University, Tadeusa Koscuska 63, 11000 Belgrade, Serbia, 2 Microsoft Development Center Serbia, Spanskih boraca 3/3, 11070 Belgrade, Serbia, 3 Microsoft Development Center Serbia, Spanskih boraca 3/3, 11070 Belgrade, Serbia

ABSTRACT

The paper presents an approach for simulation model development adapted for low-cost simulators and serious games, based on mathematical modeling of vehicle dynamics which sufficiently well represents functional motions of class of simulated ground vehicles. The model is simple, easy to use and suitable for efficient model development and validation. By tuning the parameters of the basic generic model it may be obtained particular behavior of various mobile objects (tanks, tracks, cars, mobile robots) implemented in games and simulating devices.

An example of passenger car dynamic model, implemented in serious game-driving simulator, is described to illustrate feasibility and features of proposed modeling procedure. Initial modeling is done using parameters extracted from technical specifications of the vehicle and Excel/Matlab/Simulink model validation. The Excel/Matlab model is used to check and validate real-time simulation model. The final step of model tuning and validation is done using comparison of operative simulating device behavior and the data recorded on real vehicle.

Keywords: Simulation technology, Serious games, Training simulators, Vehicle dynamics, Vehicle modeling, Model validation

1. INTRODUCTION

Reducing cost and increasing safety of education and training, simulator technology exhibits tendency of rapid development in almost any area the military and civilian practice. Fidelity of simulation (i.e. closeness of behavior of simulated and real system) is vital for successful simulator based training [1].

That is why often a sophisticated mathematical models are implemented in simulators (six degrees of freedom rigid body model is almost standard approach in flight simulation [1-4], and even multi-body model is used in car dynamics simulation [5]). On the other hand, this usually implies comparison of measurement and simulated results and use of sophisticated and often expensive validation and testing techniques which increases cost of model development and overall cost.

Similarly 3D video game technology (except the price), called serious games, offers another possibility for training and education [7-10]. In fact, these systems (often called lightweight simulators) are not games, although their core technologies come from games, as well as from artificial intelligence, robotics and learning theory. Serious games have become increasingly popular in military and civilian training and education. Fields of application include also medicine, business and marketing, cinema industry, politics, religion and others.

Serious games are based on the idea to use the games for a purpose other than pure entertainment, and obtain additional benefit, still preserving attractiveness and low-cost characteristics.

However, if applied as it is, the video games paradigms will not necessary give good results without adequate adaptation to the learning and training objectives. This primarily affects development of the mathematical model which should be sufficiently close to simulated object, but unlike expensive simulators and special-purpose training systems, free of all expensive methodologies.

It is postulated here that a serious game needs more adapted model development methodology as well as more adapted validation methodology [4]. This aspect, to the knowledge of authors is still not well covered in the literature.

This paper attempts to borrow experience from the model development technique well established for low-cost simulation devices [11, 16] and attempts to adapt it to the serious games.

The plan of paper is as follows:

We restrict here to a class of ground vehicles as typical mobile objects represented in simulation devices. In section 2. We present mathematical model of vehicle dynamics representing functional motion over the ground surface (not necessary flat but sufficiently smooth).

An implementation of this model in the serious game, developed in the scope of student competition, is presented in section 4.

Model validation and parameter tuning to attain desired fidelity of simulated vehicle behavior is described in section 5.
Finally, section 5 also discusses aspects of application presented modeling methodology to simulation of wider class of mobile objects.

2. APPROACHES TO MODELING OF GROUND VEHICLE DYNAMICS

The simplest model of mobile object is so called point mass approximation, very popular in 2D as well in 3D videogames [12]. For motion in a horizontal plane (important for a large class of ground vehicles) the equations takes very simple form:

\[ \dot{x}_0 = V \cos \psi, \quad \dot{y}_0 = V \sin \psi \]

where \((x_0,y_0)\) denote object position (in inertial reference axes), and \(\dot{x}_0, \dot{y}_0\) its derivatives, \(V\) is speed and \(\psi\) is heading angle, \(c \equiv \cos(\psi), \quad s \equiv \sin(\psi)\). If effect of inertia is important, than equation is added:

\[ \ddot{V} = \frac{F}{m} \]

where \(m\) is vehicle mass and \(F\) is propulsive force.

This model may be used to describe motion over the surface (not necessary flat) in 3D. A few variables, few parameters give a model simple to test and simple to tune, fitting well in a game development context.

On the other hand, training simulators usually use well known rigid body model with six DOF (degrees of freedom). The differential equations describing three translations (east, north, height) and three rotations (pitch, roll, and yaw) may be found in numerous textbooks [2-5].

Increased realism of 6 DOF model is followed by high model complexity, (incomparable to 2D point mass approximation (1,2)), which requires more effort and time to develop equations, tune its parameters and validate software. Model development usually includes measurements on real vehicle followed by sophisticated filtering and parameter identification techniques [1,2].

Obviously, this increases overall cost of product which may easily leave “low-cost club”.

It is clear that low cost hardware configuration as a platform for implementing serious games and training simulators (modest when compared with top quality products e.g. full flight simulators with collimated visual system) offers also modest device performances. But, low cost devices may have higher performance/cost ratio, and this should be primary objective of their design.

It is postulated here that it not justifiable to implement full scale 6DOF vehicle model since it offers more than necessary. The lower complexity model may give performances compatible with the chosen (low cost) hardware configuration, but with reduced cost.

For a class of ground vehicles whose dominant motion belongs to surface is possible to try with model slightly more complex than point mass model to obtain necessary level of fidelity specified for a serious game (or low cost simulator).

3. SIMULATION MODEL FOR A GROUND VEHICLE

We need a mathematical model of wheeled ground vehicle (car or airplane on ground) which is representative of his basic handling and performances, and still simple to use in analysis of motion and simulation. Thus, resulting model will represent planar motion with acceleration effects, steering and braking effects.

The coordinate systems and motion variables are chosen according to SAE vehicle dynamic terminology [5].

It is assumed:

a) Vertical motion, pitch and roll motion are small (body has yaw, longitudinal and lateral motion).

b) Vehicle is symmetric (left and right wheels have identical state, i.e. forces, velocities …)

Control variables are chosen as follows:

\( P_1 \) - Power lever (pedal)
X_B - Brake pedal displacement
X_n - Steering wheel deflection

State variables describing position, orientation and velocity are (see fig 2),

\[ \begin{align*}
( x_o, y_o ) & - \text{Position of vehicle center of mass in the road (inertial) axes} \\
\psi & - \text{Yaw angle (between longitudinal axis and north)} \\
(u,v) & - \text{Velocity in the body axes} \\
r & - \text{Yaw rate}
\end{align*} \]

Kinematic relations (derived from 6 DOF model [2]) are

\[ \begin{align*}
\dot{x}_o &= u \cos \psi + v \sin \psi \\
\dot{y}_o &= u \sin \psi - v \cos \psi \\
\dot{\psi} &= r \\
a_x &= \dot{u} - r v \\
a_y &= \dot{v} + r u
\end{align*} \] (3)

where \((a_x, a_y)\) are longitudinal and lateral accelerations.

The forces acting on vehicle are:

\[ \begin{align*}
P_w & - \text{traction force (on front or rear wheels)} \\
F & - \text{friction force (on each of wheels)} \\
F_a & - \text{aerodynamic drag} \\
W & - \text{weight of vehicle (} W = m g \text{)}
\end{align*} \]

Longitudinal friction force \(F_{xw}\), and lateral friction force \(F_{yw}\) in wheel coordinates are given by

\[ \begin{align*}
F_{xw} &= \mu_x F_s \sin \alpha u_w \\
F_{yw} &= \left| \mu_y \right| F_s \sin \beta w
\end{align*} \] (4) (5)

Where

\[ \begin{align*}
(u_w, v_w) & - \text{wheel velocity in the wheel axes} \\
F_s & - \text{Vertical force acting on wheel} \\
(\mu_x, \mu_y) & - \text{Longitudinal and lateral friction coefficients}
\end{align*} \]

Let us note that \((x_w,y_w)\) denote wheel fixed axes and index \(w\) denotes variables defined with respect to these axes.

For wheel \(i = f, r\) (f - front, r - rear), at position \((x_i, y_i)\) in the body axes (fig 3), one has

\[ \begin{align*}
u_i &= u - y_i r, & u_{wi} &= u_i c \psi + v_i s \psi \\
v_i &= v + x_i r, & v_{wi} &= - u_i s \psi + v_i c \psi \\
\tan \beta_{wi} &= v_{wi} / u_{wi}
\end{align*} \]

Wheel steering angle (between wheel and vehicle axes) is given by

In our case \(y_f = y_r = 0, \ x_f = a, \ x_r = -b\), since the assumption is to observe identical velocities and forces on left and right wheels (see fig4)
\[ \psi_w = X_w \psi_{wMax} \]

where \( \psi_{wMax} \) is maximal wheel steering angle.

Vertical force on wheel may be calculated as

\[ F_d = m g \frac{b}{(a + b)} \]

\[ F_s = m g \frac{a}{(a + b)} \]

Aerodynamic force is collinear with the vehicle velocity and equal to

\[ F_a = C_D S \rho V^2 / 2 \]

where \( C_D \) – coefficient of aerodynamic drag, \( S \) – Frontal cross-section area, \( \rho \) – air density and \( V \) – vehicle velocity

\[ V = (u^2 + v^2)^{1/2} = (\dot{x}_0^2 + \dot{y}_0^2)^{1/2} \]

In the body axes:

\[ F_{ax} = -F_a \frac{u}{V} \quad F_{ay} = -F_a \frac{v}{V} \]

The force delivered to the wheel is depends on wheel radius (or "rolling radius") \( R_w \)

\[ P_w = M_w / R_w \]

Torque delivered by engine to wheels depends on gearbox and final drive ratios

\[ M_w = k_g (i_{Br})^* k_d^* M_m - \]

where \( k_g (i_{Br})^* \) – gearbox ratio, \( i_{Br} \) – gear, \( k_d^* \) – final drive ratio.

The engine torque \( M_m \) depends on power lever position \( P_l \) and engine RPM \( \omega_m \) in [rad/s] :

\[ M_m = M_m(\omega_m, P_l) \quad (8) \]

Where \( \omega_m \) – engine angular velocity (RPM expressed in [rad/s]), \( P_l \) – power lever position.

Force and moment equations are

\[ m a_x = P_{rw} + F_{xrw} + (P_{tw} + F_{xrw}) \psi_w = F_{xrw} \psi_w + F_{ax} \]

\[ m a_y = F_{yrw} + (P_{tw} + F_{xrw}) \psi_w + F_{yrw} + F_{ay} \]

\[ J \ddot{r} = a (P_{tw} + F_{xrw}) \psi_w + F_{yrw} \psi_w - b F_{yrw} \quad (9) \]

where \( m \) is vehicle mass and \( J \) is vehicle moment of inertia with respect to vertical (z) axis.

This completes system equations.

Obtaining accelerations \( (a_x, a_y) \), one may calculate \( \dot{u}, \dot{v} \) from (3) and having all state derivatives integrate system of differential equations.

4. DRIVE-ON MODEL IMPLEMENTATION AND VALIDATION

The model just described is implemented in serious game called DriveON, finalist of Imagine Cup student competition [13].

DriveON is a low-cost virtual simulation environment for learning of car driving. It is based on DIS (Distributed Interactive Simulation) concept i.e. the software implemented on the several personal computers communicating over network (LAN).

Student and instructor interface is represented by physical driver controls and visual system and virtual instruments (fig 4, 5).

Fig 4: Panoramic visual system of DriveON

The main part is car dynamic model, designed to sufficiently well represent all relevant static and dynamic characteristics of real vehicle. In the DriveON the model represents OPEL Corsa 1.0, which is used for test and model tuning and validation.

Fig 5: Driver controls of DriveON
The traffic is represented by a number of computer-driven vehicles (i.e. by virtual intelligent drivers), which respect the traffic rules.

DriveON has implemented different scenarios where students may be trained in different situations like city traffic, open road or country side, high way, dry, wet, snow or iced surface.

By using Instructor Station, the traffic condition may be changed, like control of every single car, (tell them to go on next cross-road left or right turn), also weather condition, traffic light signalization may be changed. Instructor station has its own driving tests, for efficient education of students.

The data about teaching are easily collected and may be used for further analysis to improve teaching or to adapt to particular students.

In total up to 5 computers with specific input/output devices, connected over LAN, are used to implement DriveON simulation environment. Three computers are used for visual system with attached projectors to create wide angle field of view (180 degrees horizontal). One computer is used for both traffic simulation and car dynamic computation and one is used for Instructor Station module.

Next figure shows simulator architecture.

**Fig 6: Architecture of DriveON simulator**

Unlike to the traditional training DriveON offers safe training in risky situations as well as in adverse weather conditions. It may be easily adapted to other types of vehicles, extended by driving-learning assistance system or virtual (artificially intelligent) instructor.

5. VEHICLE MODEL DEVELOPMENT

It is obvious that DriveON project, conducted in the scope of student competition was (really) low budget project. On the other hand, goal to build realistic training environment has been attained.

How the data is collected and model validated to represent behavior of a concrete vehicle?

The necessary methodology is borrowed from low-cost flight simulator development described in [15,16] and adapted to serious games. It is characterized by few principles: use public data, primarily available on internet, and sophisticated data extraction and parameter tuning techniques. Before model implementation in real-time simulator environment, the parameters and functions in the model are validated comparing predicted performances with those observed on real vehicle.

Description how mathematical model of small car Opel Corsa 1 (44kW) has been developed and implemented in DriveON, follows.

If we put some set of parameters \( p \) (i.e. values of constants \( m, J, a, b, \) etc.) into equations (1-9) and integrate, we can obtain a set of characteristics \( C_h \) of interest (like vehicle trajectory versus time, maximal speed and acceleration, cornering speed,…). This may be symbolically noted as

\[
f (p, C_h) = 0 \quad (10)
\]

where \( f \) represents appropriate function. Note that \( f \) is not a function in classical sense, given by analytic expressions, but mapping which puts into correspondence simulation results \( C_h \) with input parameters \( p \).

If we get some characteristics observed on real vehicle, let's denote them as \( \hat{C}_h \), then it is likely that will not be identical to those predicted by simulation model. In fact, instead of (10) we may write

\[
f (p, \hat{C}_h) = \bar{\epsilon} \quad (11)
\]

where \( \bar{\epsilon} \) is an additive term necessary to obtain equality in previous equation. Eq. (11) will hold also if we put "exact" values \( \hat{p} \) of parameters corresponding to real vehicle (there will be always measurement, round of and other errors).

The term \( \bar{\epsilon} \) may be interpreted as an error in data or a degree of non-consistency in the set \( \{ \hat{C}_h, \hat{p} \} \).

It also may be interpreted as a measure of model fidelity (i.e. closeness of simulated behavior with that observed on real vehicle).
Finally, if we have some trusted set of data (e.g. \( \hat{\mathbf{C}}_h \)) then we may exploit relation (11) to determine the best choice of parameters \( \mathbf{p} \), giving minimal error \( \hat{\varepsilon} \) (actually, minimizing sum of squares of errors).

### 5.1 Data collection

Primary sources for the vehicle model definition are: certification or technical data sheet and vehicle user manual as officially approved documents. Additional data may often be found in some of textbooks and in various technical reports.

The next level of data collection concerns the similar vehicles (i.e. those having similar or same purpose, certification category, engine power, weight and size). Using principle of system similarity and model equations (9), this data may be used to predict parameters of the target vehicle.

Vehicle model parameters may be predicted using standard techniques for vehicle design, available in the textbooks [1-3, 5] and technical reports. Often the books are accompanied by software ready to use.

Finally, some data may be gathered on vehicle using simple "paper and pencil" methods and video records. Illustration of this by example on our target vehicle Opel Corsa1.0, follows.

Vehicle data sheet downloaded from manufacturer’s site (file new_corsa.pdf, available from authors), contained the following (relevant) data:

<table>
<thead>
<tr>
<th>Table 1: Parameters in manufacturer’s data sheet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Model year:</strong></td>
</tr>
<tr>
<td><strong>Body</strong></td>
</tr>
<tr>
<td>Drag coefficient (cD):</td>
</tr>
<tr>
<td>Frontal area (A in m²):</td>
</tr>
<tr>
<td>Index (c WxA):</td>
</tr>
<tr>
<td><strong>Weights and Dimensions</strong></td>
</tr>
<tr>
<td>Length (mm):</td>
</tr>
<tr>
<td>Width (mm):</td>
</tr>
<tr>
<td>Rim width (inch/mm)/tire size:</td>
</tr>
<tr>
<td>Turning clearance circle/turning circle (m):</td>
</tr>
<tr>
<td>Kerb weight/max. allowable weight/additional load (kg):</td>
</tr>
<tr>
<td>Power to weight ratio (kg/kW; kg/hp)(empty):</td>
</tr>
<tr>
<td><strong>Performance</strong></td>
</tr>
<tr>
<td>Top speed (km/h):</td>
</tr>
<tr>
<td>Acceleration 0-100 km/h (sec):</td>
</tr>
<tr>
<td><strong>Engine Data</strong></td>
</tr>
</tbody>
</table>

### Output (kW/hp CEE at 1/min): 44/60 at 5600
Max. torque (Nm at 1/min): 88 at 3800

### Transmission
- Drive axle: front wheel drive
- Transmission, type: manual

### Gear Ratios

<table>
<thead>
<tr>
<th></th>
<th>1st</th>
<th>2nd</th>
<th>3rd</th>
<th>4th</th>
<th>5th</th>
<th>Reverse</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gear Ratios ( k_g )</td>
<td>3.73:1</td>
<td>2.14:1</td>
<td>1.41:1</td>
<td>1.12:1</td>
<td>0.89:1</td>
<td>2.90:1</td>
</tr>
</tbody>
</table>

Thus, we may immediately extract the parameter values given in following table:

### Table 2: Parameter values for vehicle model

<table>
<thead>
<tr>
<th>m</th>
<th>Mass (empty + 2 passengers)</th>
<th>1150 kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_D )</td>
<td>Drag coefficient</td>
<td>0.36 ---</td>
</tr>
<tr>
<td>( A_x ) (S)</td>
<td>Frontal area</td>
<td>2.06 m²</td>
</tr>
<tr>
<td>( \rho )</td>
<td>Air density</td>
<td>1.225 kg/m³</td>
</tr>
<tr>
<td>g</td>
<td>gravity acceleration</td>
<td>9.81 m/s²</td>
</tr>
<tr>
<td>( R_w )</td>
<td>Rolling radius of wheels</td>
<td>0.2 m</td>
</tr>
<tr>
<td>( k_d )</td>
<td>Final drive ratio</td>
<td>3.94</td>
</tr>
</tbody>
</table>

Next for simulation model is engine torque \( M_m \) (\( \omega_m, P \)) for every \( \omega_m \) (engine RPM) and \( P \) (throttle position), but in the Table 1; we have specified only engine maximal power and torque and corresponding RPM. We can ask engine manufacturer for these data, but what we eventually get is open throttle characteristics (\( P_l=1 \)) like those in fig 7.

We may interpolate data for other throttle positions (\( P_l \neq 1 \)), but we still need to compare resulting characteristics with real engine behavior and tune the data, especially for low throttle opening.

Fig 7: Full throttle engine characteristics [17]
As an alternative, it is possible to start with some characteristics (like those in fig 7, scaled to satisfy known maximal torque-power-RPM values) and compared to real vehicle behavior in steady state. This option is chosen in DriveON project, and it will be explained how it is done. For a given fixed $P_l$ vehicle is steered in straight line until speed reaches study value, the speed and RPM are read from instruments and noted in the table for a given gear and $P_l$ value. The throttle position $P_l$ may be controlled (measured) by mechanical limiter of pedal travel. The procedure is repeated for every gear and several $P_l$ values.

Now, we can use model (1-9) to calculate values of engine torque for specific throttle/RPM conditions.

5.2 Straight Line Vehicle Movement

For practical reasons it is convenient to adapt the equations (1-9) for particular purpose – straight line movement, keeping only variables associated with x-axis an putting all other to zero (i.e. $v = 0$, $r = 0$, ...). As a result, we obtain the equations

$$m \ddot{u} = P_w + F_x + F_{ax}$$

where $P_w$ is total tractive force (on all powered wheels), is total $F_x$ friction force (on all wheels).

**Fig 8: 1 DOF car model**

Given vehicle speed (state variable) $u$ and driver input $P_l$, one obtains wheel and engine angular velocities, and engine and wheel torques respectively

$$\omega_m = \frac{u}{R_w}$$
$$\omega_m = k_g (i_{th})^* k_d^* \omega_w$$
$$M_m = P_l M_1 (\omega_m) + (1 - P_l) M_0 (\omega_m)$$

where $M_1 (\omega_m)$ is engine torque for $P_l = 1$ and is $M_0 (\omega_m)$ engine torque for $P_l = 0$ given by

$$M_0 (\omega_m) = (h_0 + h_1) \frac{\omega_m}{\omega_{00}} M_{max}, \quad h_0, h_1 = \text{const}$$

$$M_w = k_g (i_{th})^* k_d^* M_m$$
$$P_w = M_w / R_w$$
$$F_{ax} = C_D S \rho V^2 / 2$$
$$\mu_s = \mu + (h_0 - 1) \mu_t$$
$$F_{ax} = -\mu_s mg$$

Adopting friction coefficients for dry asphalt (details are omitted)

$$\mu_s = 0.024, \quad \mu_M = 0.65$$

one may calculate acceleration $\ddot{u}$ and integrate to obtain velocity, position and other related variables versus time.

If we put $\ddot{u} = 0$, we can solve this set of relations as algebraic equation with respect to one unknown variable, and that is what is necessary to tune unknown (or not well known) engine parameters from described steady state observations on vehicle.

Above set of relations is implemented in Excel and used built-in function Goal Seek to solve equation $\ddot{u} = 0$ with respect to any chosen unknown parameter.

Thus, conducting simple tests on vehicle and using pencil and paper "as recording device", one can adjust the unknown engine (or other) characteristics.

The final results (corresponding to $h_0 = 0.4, h_1 = -0.2$), implemented in DriveON simulator are shown in fig 9.

**Fig 9: Predicted Corsa 1.0 Engine torque**

$(P_l=0, 0.25, 0.5, 0.75, 1)$

Note.

If it is not necessary to have objective validation of simulated vehicle performances, one can apply this procedure using opinion of experienced driver, avoiding observations on vehicle (which should be done with all safety precautions).

5.3 Model Validation

The procedure described in previous subsection includes model validation (although for straight line movement only) since the parameters are tuned to represent given vehicle behavior. However, when dealing with experimental data, we usually obtain more data than unknown parameters. Then, we get situation represented by eq. (11) where every instrument reading correspond to one component of vector $e$ - error in data consistency. In this case the best choice of parameters $p$, giving minimal sum of squares of errors – components of $e$ may be obtained by built-in functionality in Excel (Solver finding minimum of function of several variables).
The next step is validation of 3DOF model which is done implementing the equations in matlab/simulink and comparison of simulation results with real vehicle behavior. Note again that this comparison may be objective and subjective (i.e. "expert based" and "measurement based"). We need firsts to determine rest of unknown parameters. Using already described methods of data extraction from public sources (i.e. using data for similar vehicles and applying engineering judgment), the following values of parameters are chosen.

Table 3: Parameter values for lateral dynamics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>J_\text{yaw inertia}</td>
<td>1850 kgm$^2$</td>
</tr>
<tr>
<td>aDistance of CG to front axle</td>
<td>1.064 m</td>
</tr>
<tr>
<td>bDistance of CG to rear axle</td>
<td>1.596 m</td>
</tr>
<tr>
<td>C_{yf} Lateral tyre stiffness (front)</td>
<td>140 kN/ rad.</td>
</tr>
<tr>
<td>C_{yr} Lateral tyre stiffness (rear)</td>
<td>120 kN/ rad.</td>
</tr>
</tbody>
</table>

The series of model validation tests is conducted.

The results of one of them, slowly increasing steer test is represented in the following.

Steering command $X_n$ is increased from zero with 15 deg/s, while power command is held constant ($P_l=1$), without breaking.

![Fig 10: Driver input for slowly increasing steer test](image1)

The test is performed on two road surfaces (dry and wet asphalt) with maximal friction coefficients $\mu_M = 0.95$ and $\mu_M = 0.65$ respectively.

Resulting heading angles and vehicle trajectory are shown in following figures.

![Fig 11: Heading angle in slowly increasing steer test](image2)

![Fig 12: Vehicle trajectory and heading in slowly increasing steer test](image3)

Obviously, final vehicle model validation is done on simulating device with a driver performing realistic maneuvers.

In this way we have described equations, corresponding parameters and validation technique applied to model dynamic behavior of a small car to be implemented in DriveON simulator. More details are given in ref [13,14].

5. CONCLUSION

The lack of proper modelling approach for serious games which guarantees the necessary educational content, as well as keeping the cost at low level is addressed in this paper.

An approach for simulation model development adapted for low-cost simulators is presented. It combines well established principles and methodology from the flight training devices with an efficiency of game development procedure.

An example of application to contemporary low cost car driving simulator, finalist of the student world competition in software design is also presented.

It is shown in this paper that simplified model with respect to a model implemented in high fidelity simulation devices, when implemented into game development...
framework, may give good results in education and training, still keeping the cost at the acceptable level.

The methodology is already applied to the representative low-cost simulator based on the distributed simulation concept and adapted to the serious games.

The model development methodology may be adapted to other simulated objects in serious games as a tradeoff between educational quality and cost, still preserving entertainment challenges.

REFERENCES
[10] Iuppa N. Borst T., End-to-end game development: creating independent serious games and simulations from start to finish, Elsevier, Oxford UK, 2010
[12] Mat Buckland, Programming Game AI by Example, Wordware, 2005
[14] Stojić M., "Intelligent Virtual Environment for Driving Simulator", bsthesis, Faculty of Organizational Sciences, University of Belgrade, 2007