VISUALISATION OF PHYSICAL SIMULATION USING JAVASCRIPT PHYSICS ENGINE WITH X3DOM

Olaf Berndt[1,2] and Linda Huber[1]

[1]Fraunhofer Institute for Computer Graphics Research IGD
Joachim-Jungius-Straße 11, 18059 Rostock, Germany
E-mail: {olaf.berndt, linda.huber}@igd-r.fraunhofer.de

[2]University of Rostock
Faculty of Computer Science and Electrical Engineering
Albert-Einstein-Str. 22, 18059 Rostock, Germany

KEYWORDS
X3DOM, JavaScript Physics Engine, Physical Simulation, Modelica

ABSTRACT

In order to visualise the results of a physical simulation and the effect on the related 3D geometry so far only the simulation tools themselves could be applied. A web-based application, though, could provide a powerful alternative for an efficient and realistic visualisation of data enabling users to edit the data and to share their results with customers.

In the course of the project physical modelling and simulation with OpenModelica was coupled with a rigid body simulation via the JavaScript physics engine Ammo.js and an X3DOM-based 3D graphical modelling and visualisation. Now it is possible to start and edit physical simulations as well as to view the effect of changes on the behaviour of the system in the web-based 3D model.

INTRODUCTION

In the scope of the efficient and multiple use of system simulation in the building process of special purpose vessels the connection of data from a physical simulation with a 3D graphical model for the realisation of a 3D Operator Training System is one of the key items (Berndt et al., 2015). Due to the large amount of data and the resulting size of the model it is essential to implement very efficient methods which basically work automatically and rarely need manual editing. In this article we present a method where execution of the physical behavioural simulation including the handover of the variables as well as visualisation of the results of the simulation will be done within a 3D model in the browser. Prior to their combining the graphical and the physical model were created separately. The work was completely realized by the efficient connection of open source tools without individual wrappers or additional frameworks.

As device under test the outrigger of a lifeboat davit system was used. To swing out the outrigger a cylinder is extended until the outrigger is in an upright position. From this position the outrigger continues to move outward due to his inertia and gravity. The functional principle of the davit system is displayed in Figure 1.

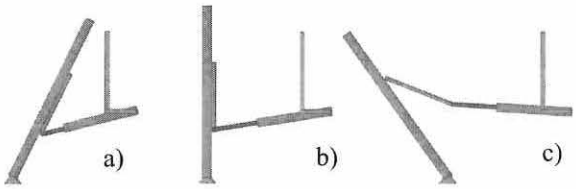

Figure 1: Functional principle of the davit: a) in standby position the davit lies on the cylinder, in case of operation the arm is erected by the cylinder; b) extension of the cylinder stops at the tipping point; c) davit continues moving outward by his own inertia.

RELATED WORK

The web-based visualisation of the results of a physical simulation within 3D models is actually evaluated for different applications. First, there is the possibility to share technical issues obtained from a physical simulation during the engineering process in a visually attractive and realistic environment with the customer (Chu et al., 2015). Furthermore, these methods can also be used for the large field of eLearning (Pang et al., 2013).

GRAPHICAL MODELLING WITH RIGID BODY SIMULATION

X3DOM is used for graphical modelling. It was coupled with a rigid body simulation via the JavaScript physics engine Ammo.js. A comparison with the other physics engines Cannon.js, Bullet.js and JigLib.js showed that Ammo.js fits the key requirements best (Huber, 2013). Beside performance those key requirements were the amount of predefined as well as user-defined shape types and constraints, the possibilities of data exchange and the developing activities. Cannon.js and Ammo.js were nearly equal in overall result, but with individual strengths and weaknesses. Cannon.js convinced with

better performance and ease of operation, but the amount of shapes and constraints was significantly lower. That is why Ammo.js finally was selected.

The original model of the davit corresponds to a real system (Figure 2a) and was provided by a supplier of davit systems. For further use the model was reduced to achieve a better performance, particularly with regard to large 3D objects (Figure 2b). The simplification of the structures, especially for curved and circular objects, resulted in a decreasing number of mesh triangles. In some cases the number of triangles could be reduced by 90%, which resulted in a significant improvement of performance. The reduced model was converted into X3DOM and connected with the physics engine. In this course all objects of the model were converted into rigid bodies. Afterwards these rigid bodies were connected via constraints. The resulting multi-body system out of rigid bodies and constraints can be enhanced by forces and velocities. Thereby drives can be added to the rigid body simulation as well.

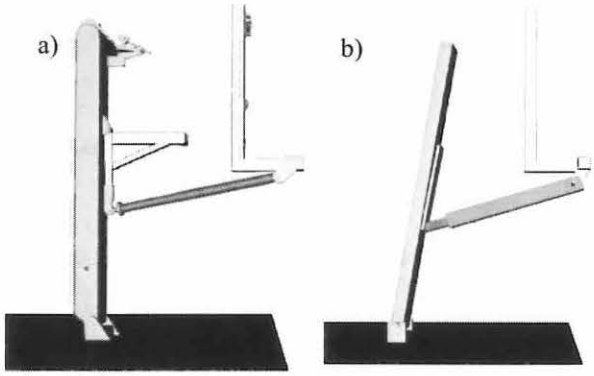

Figure 2: Model of the Davit; a) Original Model; b) Reduced Model;

PHYSICAL MODELLING

The physical modelling is done with Modelica and the open source tool OpenModelica 1.9 is used as tool for editing and simulating. Regarding the already described test scenario only the cylinder is relevant for the physical modelling as it is the only active component. The physical behaviour model of the cylinder component consists of three parts – the electronic control system, the electro-mechanical drive train and the calculation of the force exerted by the mass acting on the cylinder.

The drive train basically consists of an electric drive and a rotary linear transfer module (Figure 3). There is an electromagnetic force (EMF) which generates a rotation from DC voltage. The EMF is expanded by a rotational

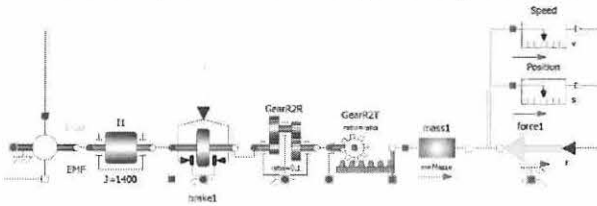

Figure 3: Physical Model of the Drive Train

inertia, a step-up gear and a brake. The transformation from a rotational to a translational movement is done by a converter. This translational motion moves a mass against a force.

The force, acting against the moved mass, is calculated from the mass of the davit and the lifeboat. The corresponding weight force F_M is distributed on two force vectors, the force component F_C, which is acting against the cylinder, and the component F_B, which is acting against the bearing at the fixed base point of the davit (Figure 3). The force F_C, acting against the cylinder, depends on the geometry of the davit and changes during the process of erecting the davit. When the cylinder is extended the length s is raised and the geometry is changed. F_C is given by:

$$F_C(s) = F_M \frac{sin(90°-\alpha)}{sin(180°-\beta)} \quad , \tag{1}$$

with

$$\alpha(s) = acos\left(\frac{a^2+b^2+c^2-s^2}{2a\sqrt{b^2+c^2}}\right) + acos\left(\frac{c}{\sqrt{b^2+c^2}}\right), \tag{2}$$

$$\beta(s) = acos\left(\frac{a^2+s^2-b^2-c^2}{2as}\right) \quad . \tag{3}$$

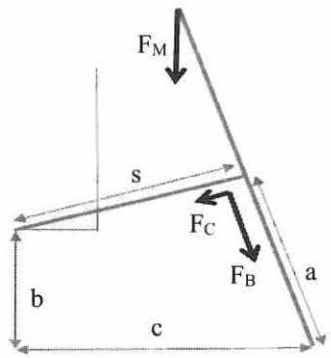

Figure 4: Geometry of the davit for the calculation of the acting forces;

The force F_C is one of the input values for the electronic control system. If the davit reaches the upright position F_C becomes zero. At this point the brake is activated with full power and the supply voltage for EMF is switched off. To reduce the mechanical stress for the drive train caused by the movement change there is a procedure to slow down the EMF. At a predefined position of the cylinder the polarity of the power supply for the EMF gets reversed and the EMF slows down the drive train.

INTERFACE BETWEEN PHYSICAL AND GRAPHICAL MODELLING

For the coupling of both components, physical and graphical modelling with rigid body simulation, an interface is required to provide the results of physical simulation for graphical modelling. At first it had to be determined, which data from the physical simulation could be used for the manipulation of graphical modelling. As mentioned before, graphical modelling consisted of a coupling of X3DOM and a JavaScript

physics engine. X3DOM presents the 3D model and can only handle changes in position and rotation as it consists of pure geometric data. The JavaScript physics engine Ammo.js is applied for rigid body simulation and is able to handle physical parameters out of physical modelling. Finally, three parameters are relevant for the coupling: time, velocity and position. Tus, depending on time, it will be determined how fast the inner part of the cylinder will be extended and how far the cylinder itself will be extended. These parameters can be processed by the physics engine in order to calculate the resulting changes in position and rotation of the movable parts of the davit. The results are transferred to the 3D model in X3DOM for visualising the movement of the davit. Graphical modelling is a purely web-based application.

After the coupling of physical and graphical modelling it should be possible to execute all user interactions, e.g. the start of the simulation or modifications of parameters, from the website. While graphical modelling, which is based on JavaScript, completely runs on the client, the application has to be enhanced for the coupling by a Java-based server. This is necessary for editing and executing files of physical modelling and simulation within the scope of a sandbox environment. After executing OpenModelica the simulation results can be generated as a .csv file. This file can be read by the server after starting the web application or after changing some simulation parameters and will be sent to the client in order to process the simulation results for graphical modelling. Another aim was to control the exported simulation results of OpenModelica via the web application and also to change parameters of the physical simulation, e.g. the combined mass of the davit and the lifeboat, via the web application in order to prevent media discontinuities. OpenModelica saves all parameters and settings for the simulation in an .xml file. Therefore, input fields were added to the GUI of the web application to enter different parameters for physical modelling and simulation, e.g. the mass (Figure 5). After entering the values these will be submitted to the server by pushing a button. The server is responsible for overwriting the corresponding values in the .xml file for the simulation settings of the physical modelling with the submitted values. After that the execution file of the

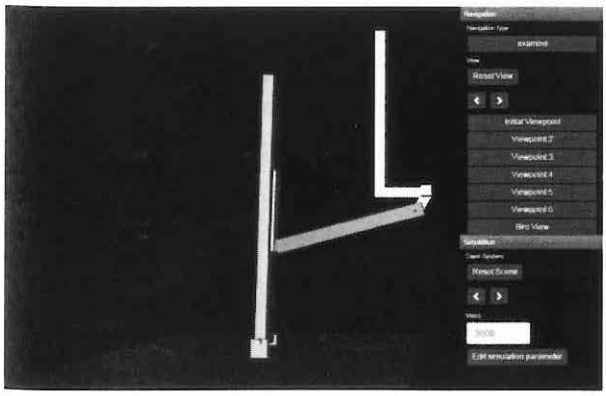

Figure 5: Web GUI with input field for changing parameters and a simulation control button.

OpenModelica simulation will be started by the server, which will generate a new .csv file with the updated simulation results. If the process of overwriting the .xml file and executing the OpenModelica simulation has been successful the server transmits a positive feedback to the client. Thereupon the client reads the required data for position, velocity and corresponding time stamp from the newly generated .csv file and submits the data to the physics engine for further processing. As a result of rigid body simulation changes in position and rotation of the davit's moveable objects are available and transferred to the 3D model in order to update the visual scene.

CONCLUSION AND OUTLOOK

In the course of the project we realised an efficient web-based coupling of physical modelling and simulation with 3D visualisation using the example of the outrigger of a lifeboat davit system. Changing parameters for the simulation settings and starting the simulation can be executed directly from the website based on a client server architecture. Hence the visualisation of the results of physical simulations within 3D models considering rigid body simulation is realised with open source technology and low demands on hardware and software equipment.

In the next step an interactive Modelica simulation via the web browser is going to be realised. Therefore, the Functional Mock-up Units according to the FMI 2.0 standard shall be connected with a JavaScript-based web application. The use of the Java library javaFMI with this application will be evaluated.

ACKNOWLEDGEMENT

This research has been supported by the German Federal State of Mecklenburg-Western Pomerania and the European Social Fund under grant ESF/IV-BM-B35-0006/12.

REFERENCES

Berndt, O., Lukas, U.F.v. and Kuijper, A. (2015) 'Functional Modelling and Simulation of Overall System Ship - Virtual Methods for Engineering and Commissioning in Shipbuilding' in *ECMS 2015*, pp.347–353.

Chu, Y., Hatledal, L.I., Sanfilippo, F., Schaathun, H.G., AEsoy, V. and Zhang, H. (2015) 'Virtual Prototyping System for Maritime Crane Design and Operation Based on Functional Mock-up Interface', MTS/IEEE.

Huber, L. (2013) 'Initial Steps for the Coupling of JavaScript Physics Engines with X3DOM', Eurographics Association, pp.81–90.

Pang, X., Dye, R., Nouidui, T.S., Wetter, M. and Deringer, J.J. (2013) 'Linking interactive Modelica simulations to HTML5 using the Functional Mockup Interface for the LearnHPB platform', pp.2823–2829.