Towards a Simple City Driving Simulator Based on Speed Dreams and OSM

Tigran Avanesov[†], Nicolas Louveton[†], Roderick McCall[†], Vincent Koenig[‡], Martin Kracheel[†],

[†]SnT, Université du Luxembourg
6, rue Richard Coudenhove-Kalergi
L-1359 Luxembourg, Luxembourg

[‡] SnT and EMACS, Université du Luxembourg
Route deDiekirch
L-7220 Walferdange, Luxembourg

<firstname>.<lastname>@uni.lu

ABSTRACT

This paper presents an architecture and partially built simulation platform which is designed to offer a flexible open ended approach for conducting laboratory experiments. The emphasis is on supporting multiple drivers and the ability to swap in and out different software components and devices.

Categories and Subject Descriptors

J.4 [**Social and Behavioral Sciences**]: Sociology, Psychology; D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User interfaces*

Keywords

City driving simulator, OSM, Speed Dreams, Usability

1. INTRODUCTION

The City of Luxembourg is the 10th most congested city in Europe [9]. In order to overcome this problem, the I-GEAR project was created and it specifically explores how we can encourage commuters to subtly change their mobility behaviour through the use of game-like mobile applications. The project has two key testing stages that explore these issues which need a specific simulation environment.

A key requirement is that the simulator must be highly modular and customisable, for example, supporting new graphical engines as they become available, large city-like environments with complex layouts, having more than one driver at any given time and finally also allowing us to change/modify and remove hardware components such as tablet PCs at short notice. We chose not to buy a commercial platform as these are often heavily locked down and are too expensive when multiple cockpits are required. As a result we chose to base our simulator on an architecture of open source components coupled with a robust underlying highly flexible server application. This approach allows us for example to completely change the 3D environment with minimal impact on other aspects and to use the programming languages of our choice for controlling the simulation environment.

2. SIMULATOR

For the 3D environment we are using Speed Dreams (SD) [7] which is an open source motor sport simulator which is forked from the Torcs platform [8]. The platform supports a range of cars, standard driving controls (e.g. pedals) and is edging slowly towards providing networked game play with multiple drivers. It is written in C++, runs on Linux, Windows and soon Mac OS X and is easily customisable.

2.1 Speed Dreams track model

SD is focused on supporting motor racing and each track consists of a sequence of segments such that the end of the last segment coincides with the beginning of the first to form a lap. A segment may be given a curvature radius, a tilt and also may have different starting and ending widths and altitudes. This track information is stored in an XML file. A car can move only inside the track limited with the barriers. Any other objects, like trees or buildings, have no impact with the car, and they are usually placed outside the track.

2.2 Problems Encountered

In order to use SD for our purposes, we had to solve a couple of problems:

- Road intersections are not foreseen, while these are a common case for city road-maps (a workaround is described in Section 2.3).
- There is no 3D scene of Luxembourg city integrated in the simulator (see Section 3).
- There is no traffic. Moreover, SD robots rely on xml description of the track (not yet addressed).
- There are no traffic lights (not yet addressed).

2.3 Source code adaptation

As mentioned above, SD could *not* be used *as it is* for city driving due to impossibility of defining road intersections. Thus, we introduced some changes in the source code based on the following idea. If we ensure that the track describes a flat world (i.e. the track has a constant altitude) and disable the collision test with the barriers, then the car can drive to any point of the scene. Then, if a driver will not go off the road[1] which he/she sees as a part of 3D scene, we obtain a solution: one can drive on an arbitrary road-map ignoring the track information.

Another change we applied is an integration of telemetry data transmission via TCP (see Section 4).

[1]Since we disabled the collision detection with barriers, we also have to implement it with using objects in the 3D model

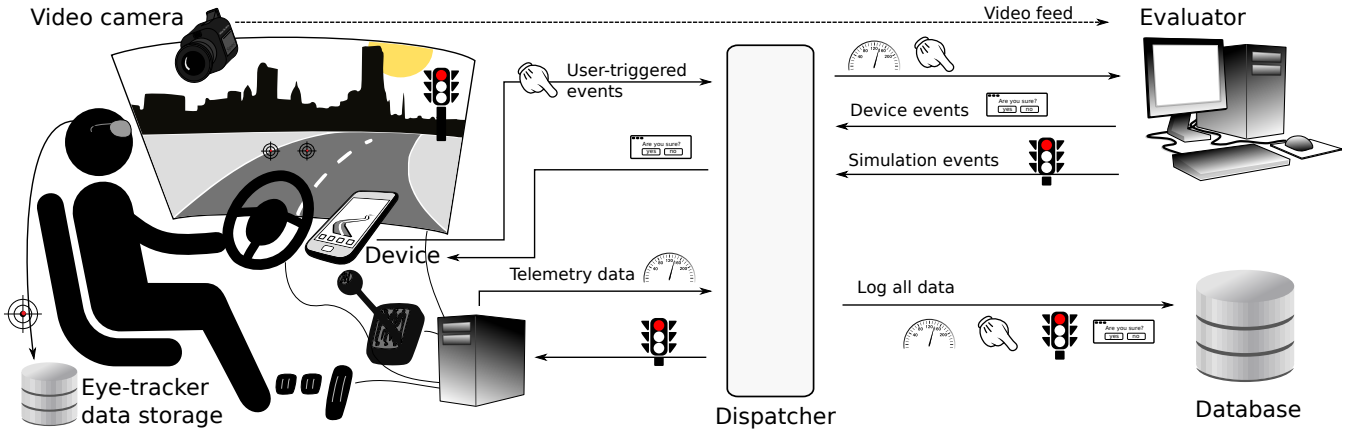

Figure 1: A scheme of the simulator usage in I-GEAR

3. 3D CITY MODEL

SD uses AC3D format [1] to describe 3D scene. While this is flexible, building a 3D model of Luxembourg City is a complex task and in order to speed up the process we used data from OpenStreetMaps [10] (OSM) and it's satellite projects.

First, we had to crop the OSM country description to reflect our area of interest (Luxembourg City). The tools [6] and [5] provide such functionality by taking as input an OSM data file and a polygon description of the desired area. The latter can be created in [3] by selecting the desired area on a the map. *Second*, we used OSM2World [4] to create an initial 3D scene of the area which then exported to a Wavefront `.obj` file. *Third*, we imported the .obj file into Blender [2], a 3D open content creation suite, where the scene can be edited (e.g. by adding building facade textures) and using 3rd party scripts, the result was then exported to AC3D formatted file (`.ac`). *Finally*, the `.ac` file can be used by SD.

We note that some objects generated by OSM2World have a relatively high number of polygons which in turn causes performance issues (if not SD crashes) during the simulation. That is why we had to replace some objects with lower polygon counts. This was carried out by editing the scene in Blender, however we plan to use OSM2World.

4. EVALUATION ENVIRONMENT

Our primary focus within the simulator is to observe driver behaviour and to log their actions. We plan to log various data relevant to driver's behaviour, vehicle and in-car devices. In order to achieve this objective we have outlined the core components in the following section and in Figure 1.

Evaluator PC acts as control centre for the evaluator and displays the current telemetry data, status of the device and provides live video feeds of the driver and 3D model. Additionally it supports setting up and controlling each experiment as well as triggering events both in the 3D model (e.g. switching traffic lights) and on devices like tablet PCs.

Video camera A video feed of the drivers cockpit including the 3D model and controls.

Eye tracker An SMI head-mounted mobile eye tracker is used to track the drivers gaze position, dwell time and eye movements.

Device Any in-vehicle device that is used for capturing and displaying data (smartphone, tablet PC, etc.).

Database Data from the simulation is logged in SQLite database. Can be easily switched to any other DBMS.

Dispatcher Dispatches the data flow as shown in the Figure 1. We implemented it in Python.

As for communication protocol we use JSON. Telemetry data contains current speed, acceleration, car position, controllers status etc. Every user action on a device is logged and sent to the dispatcher. All this data is stored in the database and can be analysed later on.

5. CONCLUSIONS

We have presented an overview of the I-GEAR simulator environment which will be used for the analysis of driving patterns and human-factors issues. Among the key benefits of our approach are: the relative low cost, support for multiple drivers, high modularity and the ability to support any part of the world through the use of Open Street Map.

6. ACKNOWLEDGMENTS

Supported by the National Research Fund, Luxembourg (Project I-GEAR).

7. REFERENCES

[1] The AC3D file format. `www.inivis.com/ac3d/man/ac3dfileformat.html`.
[2] Blender project. `www.blender.org`.
[3] Digitizer tool. `www.birdtheme.org/useful/googletool.html`.
[4] Osm2world tool. `osm2world.org`.
[5] Osmconvert tool. `wiki.osm.org/wiki/Osmconvert`.
[6] Osmosis tool. `wiki.osm.org/wiki/Osmosis`.
[7] Speed dreams project. `www.speed-dreams.org`.
[8] Torcs project. `torcs.org`.
[9] TomTom 2010 congestion index. `http://www.whatcar.com/car-news/tomtom/249079`, 2010.
[10] M. M. Haklay and P. Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7:12–18, 2008. See `www.osm.org`.