Traffic Simulation based on the Robocar World Championship Initiative

Norbert Bátfai, *Member, IEEE,* Renátó Besenczi, *Student Member, IEEE,* András Mamenyák, Márton Ispány, *Member, IEEE*

***Abstract*—Robocar World Championship or briefly OOCWC is a new initiative to create a community of people who share their interest in investigating the relationship between smart cities and robot cars with particular attention to the spread of robot cars in the near future. At the heart of this initiative is the Robocar City Emulator. It is intended to offer a common research platform for the investigation of the smart city simulations. In this paper, we review the recent advances of the OOCWC.**

***Index Terms*—smart city, traffic simulation, robocar, programming competition**

I. Introduction

Based on some assumptions, by the year 2050, 70% of Earth's population will live in urban areas [1]. Urban infrastructures will face new challenges with this amount of population. In recent years, the domain called Smart City has become a vivid topic. Smart City applications contain solutions and research areas, such as intelligent city planning, crowd sourcing and crowd sensing, crisis and disaster management, etc. [2] The goal is to make the life of an urban population easier and provide sustainable development and an optimal distribution of available resources.

In recent years, many developments occurred in the automobile industry regarding the scope of autonomous cars, also known as driverless cars. Many companies are showing interest, such as the self-driving car pioneer, Google, or many German manufacturers, e.g. the Volkswagen Group. Also, the development of pure electric cars has shown major progress recently. These trends even seem to be accelerating. It is clear, one of the world's leading economic industries is facing a revolution; a paradigm shift is going to happen.

It can be an interesting question from the point of view of ICT and IT; how can these cars, which will be generally available in the near future, be assisted by city administration? What can a city controlled IT solution offer to these cars to allow them to be operated more practically, more economically and more efficiently?

Lets assume, in the near future, 50% of the cars in everyday life will be driverless. The car should have an optimal route if one would like to travel from one part of the city to the other. It can be a completely reasonable scenario that this route will be given by the city or, more specifically, by an IT solution coordinated by the city. This is reasonable, because the city has all the information necessary, such as traffic congestion, detours, accidents, road construction, rare events, etc, to produce an optimal route.

In this paper, we summarize the research and development results currently included in the RObOCar World Championship (OOCWC) platform. We will see in a working example, what we call "Police Edition" (described later), that the system can perform in an optimal way, to plan routes for individual entities and simulate traffic environments in almost every urban area of the world. We emphasize, this platform is currently a simulation platform, no actual traffic intervention occurs in the real world.

One useful feature of the system is adaptation to different uses. Different kinds of information is suitable for governmental travel, EMT actions, police events or civilian movements. As we will see in an example in Section II-B1, the system can be applied in each specific case.

The research and development process of the OOCWC is agile, with fast prototypes for every component. The platform, which is currently under development at the University of Debrecen, provides a competition where researchers and students can compare their knowledge in traffic analysis, route planning methods or even their coding skills. The currently available components of the system are open source and provide an opportunity for researchers and developers, who are interested in the relation of smart cities and autonomous cars, to try out their ideas and methods.

The OOCWC has many other possibilities. Besides providing routes to driverless cars and simulating traffic in cities, it will be able to collect data about cities with on-board traffic analysis. A database is being established for this data. Furthermore, the system can raise new questions in the ICT research domain: how can a car communicate with a city? The OOCWC is obviously a multidisciplinary platform (car industry, ICT and IT field, city administration, traffic analysis) which can bring us to the next step in smart cities: the automated city.

The rest of the paper is organized as follows. Sect. II sketches the overall architecture of the OOCWC platform to be developed and shows some main design requirements. In Sect. II-B we briefly introduce two rapid prototypes called "Justine" and "Justina". Finally, in Sect. III we give a brief summarization of the results from the first competitions and the results of our traffic measurement and analysis solutions.

All authors are with the Department of Information Technology, University of Debrecen, H-4010 Debrecen PO Box 12, Hungary, e-mail addresses: batfai.norbert@inf.unideb.hu, netid21@mailbox.unideb.hu, mamenyaka@gmail.com, ispany.marton@inf.unideb.hu. Corresponding author: Norbert Bátfai, tel: +3652512900/ext.75228.

This paper is an extended version of a conference article previously presented at the 13th International Conference on Telecommunications (ConTEL) 2015.

Manuscript received August 13, 2015, revised September 18, 2015.

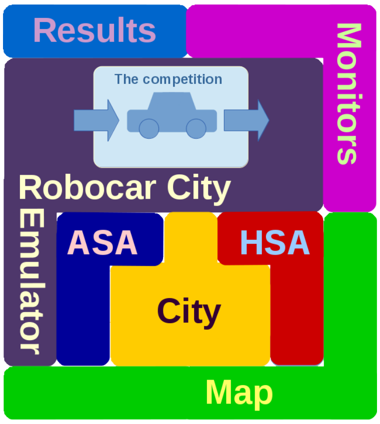


Fig. 1: The high-level "Tetris" plan of the OOCWC platform.

Related Works

It is not rare to use competitions in a given research and educational area, for example the RoboCup [3] focuses on artificial intelligence. We have suggested a similar initiative for smart city research.

The paper [4] presents a case study about investigation of replacing traditional cars with driverless cars in Singapore and [4, p. 10 Fig. 2] shows that the authors of the study have already used 10×10^5 number of vehicles but it may be noted that their simulation is not a realistic one in the sense that the simulated cars are not standalone agents.

Realistic traffic simulations have already been used in [5]. Here, it can be noted that our traffic simulation model can also be seen as a variant of the Nagel-Schreckenberg (NaSch) model [6] because we have used a cell-based approach as well.

Many traffic simulation models have been developed. In general, three classes exist: 1) The agent-based models (or microscopic models), which is the method presented in [7] and considered as a representative, modern approach of microscopic models. 2) The continuum models (or macroscopic models), there exists several solution, e.g. [8], [9]. 3) Hybrid models which work as a mixture of the two above, like [10]. Our rapid prototypes are agent-based systems. In addition, one of our prototypes called "Justine" can be considered as a standalone multi-agent system that has the ability to support multi-agent programming competitions.

II. THE OOCWC PLATFORM

Fig. 1 shows the high level architecture of the system to be developed [11]. 1) Map, City, The competition: each competition is played on a map and assigned to a given city. 2) ASA, HSA: Automated Sensor Annotations and Human controlled Sensor Annotations. The ASA subsystem automatically collects traffic flow data using video cameras. In contrast with this, HSA provides data collected by volunteers. The collected data will be used in the Robocar City Emulator. 3) Robocar City Emulator, Results, Monitors: The Monitor programs can visualize the working of the Robocar City Emulator. The results typically are competition reports but there might be simulation measurements in a given city, even industrial case studies.

A. Design requirements

Only some requirements are highlighted in this section. The full requirements can be found in the software repository of the project [11, doc/SRS] 1) Firstly, it must be specified that the Robocar Emulator can only be considered as an emulator for determining the traffic routing input and output of driverless cars, but in all other aspects it is a simulator. A distant goal is to provide a simulation in which the behavior of the driverless cars and human-driven cars are similar, as much as possible.

2) It is a very important requirement that the platform to be developed must be able to handle a very large number of cars. For example, [4] shows a simulation experiment in which there are 10^6 cars. 3) Another important requirement for the platform is, that it should be able to take into consideration the actual size of a car in such a way that a road can become closed because of the high density of traffic.

Crowd Sensing: The input of the Robocar City Emulator is stored in the City Cloud. This cloud system contains data about the city where the RCE operates. The collection of the data is a crowd sensing based solution. It can be performed in two ways: 1) A community of volunteers perform it manually with a cell phone software (HSA). 2) With specific hardware components developed for our purposes (ASA). By community based solutions we mean manually performed measurements. However, a cell phone application is being developed to assist the data recording. These smart phone applications are very simple and easy-to-use, they consist of only a few buttons. If a vehicle passes the point of measurement, a tap of the respective button is enough to record the event. After the annotation is finished, the application sends the time, position and recorded information to our cloud. Some similar, community based crowd sourcing solutions can be found in [12].

The specific hardware components, that are being developed for our purposes, can perform data collection without user interaction. These devices will be able to recognize vehicles via their video interface. The object detection will be performed with pre-trained cascade files, similar to those used in [13]. This type of object detection has good performance in face and eye recognition [14], [15]. Several solutions are being developed. The main difference between them is the type of processor assembled inside the hardware. The common part in our hardware design is the I/O parts and the usage of a Field Programmable Gate Array (FPGA). The FPGA component can perform tasks with very high speed. For the input, we use an image sensor (e.g. a high resolution video capture module) and a GPS module (to achieve accurate positioning). For the output, we use a standard GSM module to reach mobile data connectivity.

The first type of our hardware design is an ARM based solution [16], [17]. In this type, we use the advantages of the FPGA, mainly for I/O and memory management. The ARM gives us a standard option to perform calculations and image processing. This kind of hardware development can take place on development boards. The main advantage of this type is that we can use an Embedded Linux System, so the high level processing tasks can be developed in a standard Linux/UNIX environment. An example for this type of system is the widely applied and very popular Raspberry Pi single-board computer [18]. The selection of the suitable development board is currently in progress.

Another solution is based on a soft-processor. In this type, we do not have a physical processor, only an FPGA, so we should add a pre-defined one to our hardware design (e.g. a Xilinx MicroBlaze). In this case, we must specify the operation of the processor, however its instruction set is poorly defined.

The third type of solution has no processor at all, only a pure FPGA design. The whole process, including the I/O handling, image processing and some basic analytics is performed by an FPGA. Certainly, this solution gives us the fastest processing speed, but its development is more complicated.

We conclude that the ARM based solution is very flexible and it is easy to extend it with new features, but we are limited to its instruction set. Also, because OOCWC is an open source project, a hardware design developed in this environment can easily be opened for a community interested in such systems, because these components are cheap and widely used. On the other hand, the purely FPGA based solutions have an advantage in processing speed and we are not limited to the instruction set of ARM.

B. *Rapid Prototypes*

1) Justine: this is a rapid prototype for development of the OOCWC platform. It is entirely based on OpenStreetMap [19] (OSM) which is a worldwide, community-based open data project published under the Open Data Commons Open Database License. Justine is released under the GNU General Public License Version 3 and can be downloaded from the project's repository at https://github.com/nbatfai/robocar-emulator. It contains three main components: 1) the rcemu is the main software package and consists of Boost C++11 implementations of the smart city and the traffic simulator programs, 2) the rcwin and rclog are online and offline visualization programs for displaying and replaying rcemu simulation files 3) the DocBook 5.1 XML-based [20] documentation sub-project can also be found in the project's repository. The rcemu uses libosmium library [21] for the processing of OSM data. The result of the processing is the routing map graph that is placed in a shared memory segment by the smart city server. The other task of the smart city server is to simulate activities that may occur in the city. The traffic server can simulate the traffic flows. In addition, rcemu contains a sample program that has been written to demonstrate how a client can communicate with the traffic server through TCP and how the shared memory graph may be handled. From the viewpoint of functionality, clients may be routine cars, smart cars and guided cars. From these cars the guided cars are controlled by the competing teams. We are currently working on several editions of the prototype Justine. The main edition of Justine is called "Police Edition" in which cop multi-agents pursue gangster agents for ten minutes. Gangsters are smart cars and cops are guided cars.

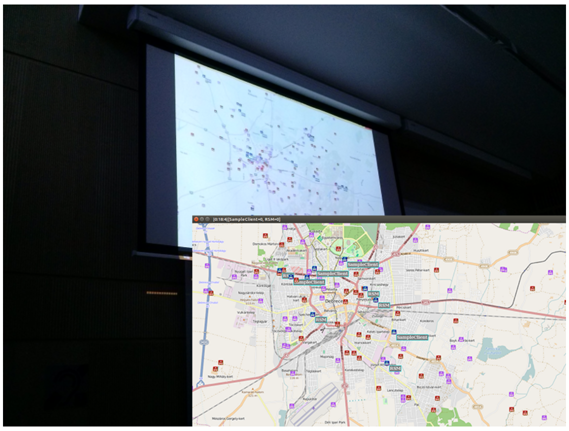


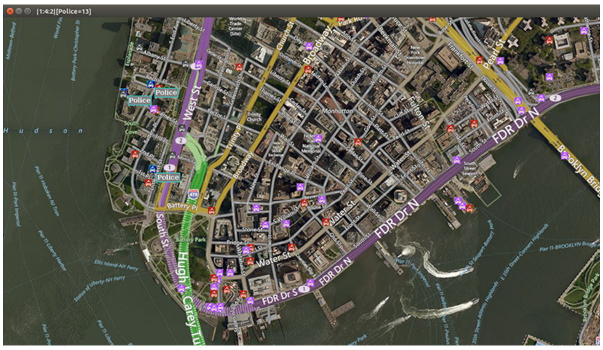
Fig. 2: The rcwin display program in action in the competition called Debrecen 2.

The rcwin and rclog have entirely similar structure, both of them are based on JXMapViewer2 [22]. The rcwin program can be seen in action in Fig. 2 and Fig. 3.

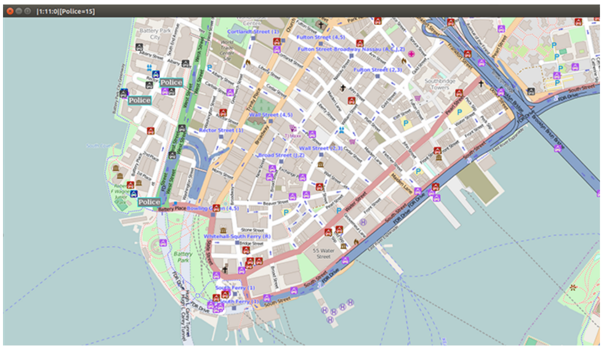
At the time of writing this paper, the official documentation is available only in the Hungarian language, but translations to many languages have already been prepared in the framework of the UDPROG project [23]. A YouTube video of the entire installation process and usage of the OOCWC platform can be seen at https://youtu.be/_FNoFqlygyE.

Operation of simulation model: The simulation has taken place in a rectangular part of the OSM map of the given city in which the competition is played. We call this part as "City Operating Area" or briefly COA. To be more precise, we have built two alternative data structures: one using a directed routing graph created from the OSM data and one using a Boost Graph Library (BGL) graph created from the previous directed graph. The simulation server is based on the directed routing graph but the custom clients may use the BGL graph as well. For example, in the case of Debrecen, the COA is bounded with GPS coordinates from 47.4095 to 47.652 and from 21.4268 to 21.8628. The corresponding routing graphs have 77591 edges and 37455 vertices. Using the terminology of the Nagel-Schreckenberg model, all edges are divided cells. If we interpret our simulation as an implementation of the NaSch model, then the cell length is equal to 3 meters that corresponds to 15 m/s = 54 km/h speed because our simulation cycles last 200 milliseconds. (It also means that all vehicles can move with the same velocity.) In contrast with the NaSch model, a cell may contain many cars, but each edge can only contain a given number of cars. This limit is calculated as the edge length divided by the length of the cell.

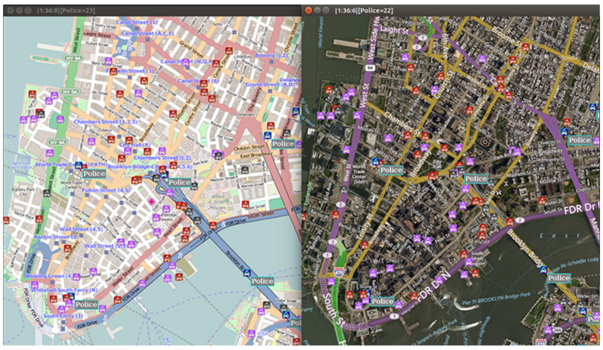
The routine cars move by random walk or ant simulations [24], [25]. In the latter case, for example, the next edge of a routine car will be chosen with a probability that increases with the number of past selections of the given edge. The sample client has a predefined implementation of Dijkstra's and Bellman-Ford algorithms to support the routing of guided cars. These implementations use the BGL's Dijkstra's and Bellman-Ford shortest path algorithms [26].



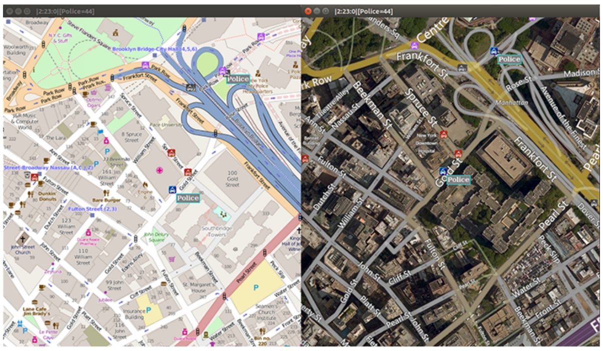
(a) Tile-set view



(b) Aerial view



(c) Hybrid view from distance



(d) Hybrid view closer

Fig. 3: The rcwin display program in action, hybrid view in Manhattan, New York City, USA. These views are provided by the JXMapViewer2[22].

2) Justina: this is another rapid prototype for development of the OOCWC platform and it is also based on OpenStreetMap. The main program contains both the emulator and the visualization components. The emulator uses the same libosmium library for the processing of OSM data. After the data is processed, the emulator cleans out the unnecessary parts, keeping only the roads, thus reducing the size of the map. From this data, a graph is created using the Boost Graph Library which holds the car objects and to which routing algorithms can be applied. The main routing algorithm for the non-routine car agents is the A* algorithm [26]. The algorithm takes into consideration the weight of the edges (roads). The weight of each edge is calculated from its length and the number of cars currently located on that edge, so the less congested roads are more likely to be part of the result. The routine cars move based on the ant simulation algorithm.

The visualization component uses the Qt development framework and can be seen in action in Fig. 4.

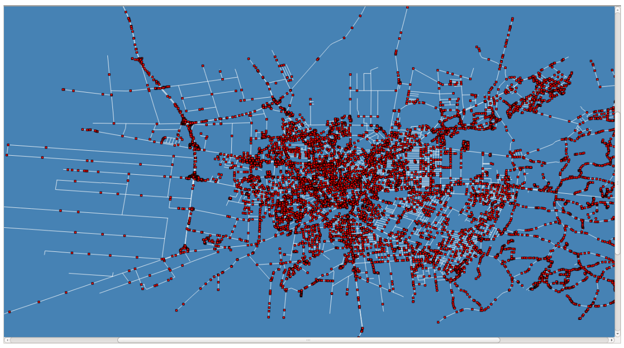


Fig. 4: The display program in action with 10122 cars

III. RESULTS

A. From Debrecen-1 to Debrecen-3

At the time of writing this paper, we have already organized three tournaments. The main purpose of these competitions was to gather first experience with the initial version of the OOCWC software and to find bugs in the source code. The Justine prototype has been used for tournaments. The first competition called Debrecen-1 was held in December 2014 with six teams. Three out of the six teams and three new teams participated in the second competition (Debrecen-2) in January 2015. The third competition called Debrecen-3 was held in March 2015, where 43 teams were competing. All three competitions were organized using the prototype "Justine". The detailed documentation (that contains Announcements, Team Qualification Papers, Traffic Logs and Competition Reports) of the last two tournaments is available in English and can be found at http://justine.inf.unideb.hu/2015/Europe/Hungary/Debrecen.

B. Data acquisition

An important part of the system is the data acquisition. To simulate real-world events (e.g. routing, real traffic simulations) we need real-world data. In order to deliver this data, we have implemented a crowd sensing device, which we mentioned in Sect. II-A. The device is an ARM based hardware and software solution and it will be suitable to install it into individual traffic entities (e.g. cars, buses). We should note that an initial testing phase has occurred. In this experiment, some streets of the city of Debrecen has been measured and served as an input for the Robocar City Emulator. The data acquisition project, called Real-Time Traffic Analyzer [16], was applied to proceed this experiment in the Crowd-sourced Traffic Simulator project [27]. The device and the experiment are discussed in [17].

In a brief summary, we give a picture about the operation of this subsystem. With a device, which can be assembled into vehicles, we are able to measure traffic environment around individual traffic entities (including private or public transport). It is important for us regarding these devices that they are in motion constantly, in contrast with the sensors installed on a fix position. So we should consider every factor during the development to aid our plan to install them into vehicles. The base element of the device is a development board, which is currently a Digilent Zybo. This board contains an FPGA and an ARM processor, working parallel (this solution is the first one in Sect. II-A). The FPGA part performs the I/O handling, the ARM processor performs image processing. Because the device is in motion all the time, we needed a GPS module to obtain position and connecting measured data to roads (e.g. streets). The measurement itself is a Haar-cascade based object recognition, the video stream is served by a camera module with the resolution of 640×480. The data is sent to our server through internet connection provided by a GSM module (GPRS connection).

The measurement itself is a car-counting function which gives us a car density value on each road. As many car is counted on a road as dense it is. On the server side, where we receive this data, an algorithm collects these density values by street, so we get a pair of street name and the corresponding intensity of the traffic flow.

In the simulation we use this pair. On the standard interface of the Robocar City Emulator we give this simple text file as an input which will be the initial state of the simulation. So the data collected by the Real-Time Traffic Analyzer, after some systematization, will be served as an initial simulation state. During the simulation we can observe the change of the distribution of the initial data, actually, how the traffic situation changes.

We can conclude that the data acquisition worked well during the testing phase. The object detection method recognized nearly every type of vehicle in the environment of the device, an accurate position obtained with the GPS and the data arrived to our server properly during the measurement. The Robocar City Emulator, after a minor development (and after the creation of the standard interface as well) was able to simulate the traffic outcomes from the initial data.

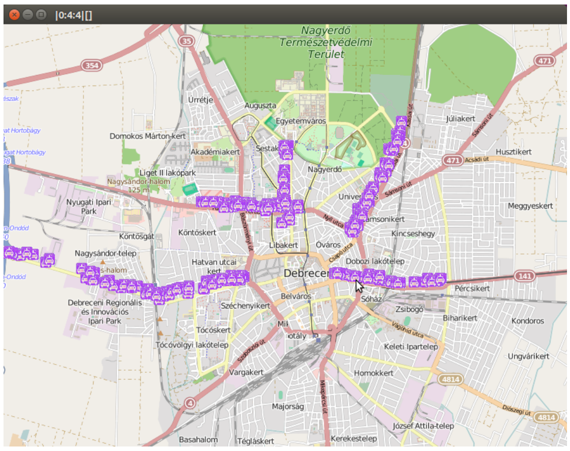


Fig. 5: The initial distribution of cars is based on the following roads Kassai út: 789, Egyetem sugárút: 317, Füredi út: 559, Kishegyesi út: 979, Faraktár utca: 271.

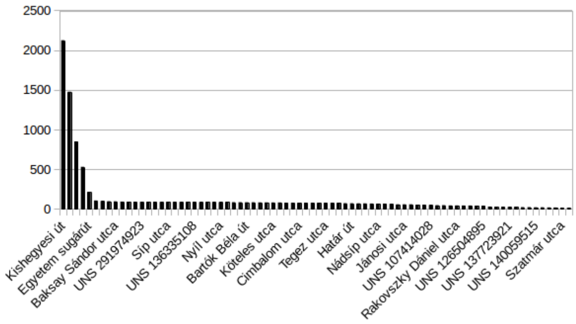
C. The Development of the Traffic Simulation Engine

As we mentioned in the introduction, the challenge of the *robocar-emulator* project is how to handle a large number of vehicles in the traffic simulation, for example, a million cars. It is clear that implementing cars in the simulation as individual software agents is a nearly impossible task. That is why we introduced the notion of routine cars. They are not individual entities like UNIX processes, Java threads or CUDA threads and they have no own individual properties. A routine car only has a probability distribution over the traffic graph nodes. It allows that routine cars can be built in the simulation software itself. The initial distribution of the cars is based on real measured or estimated data. In this work, the simulation starts from a modified version of the estimated distribution used in [17] as it can be seen in Fig. 5. Our main task in this paragraph is to investigate how the traffic simulation had changed this initial distribution of the cars. Fig. 6, 7 and 8 show the changing distribution of the cars over the traffic graph of Debrecen. These ordered histograms present the roads of Debrecen in decreasing order of the number of contained cars at different times of the simulation. It is an interesting observation that neither of these simulations change roughly the (Pareto) nature of the initial distribution but unfortunately it has not been enough. It is also very important that the ordering of the roads must stay the same during the 10 minutes of the simulation. It can be seen well in the presented ordered histograms that our simulation engine cannot fulfill this criteria at its current development phase.

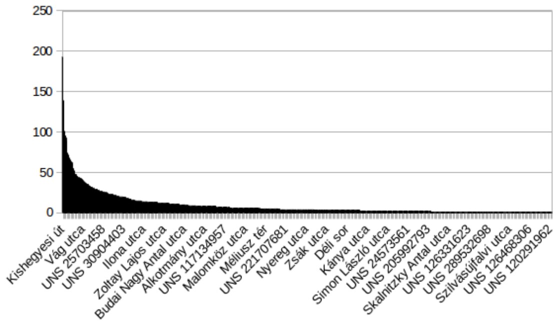
D. Future Work

In this paper, we have shown that random walk and ant simulations are not suitable to move routine cars for the real simulations. The main research goal of the OOCWC platform is to develop a new kind of traffic simulation that does not change the ordering of the roads in the ordered histograms generated during the simulation.

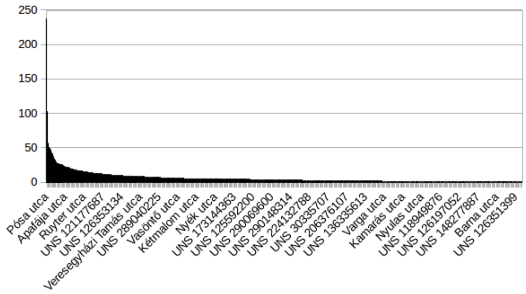
Regarding the crowd sensing subsystem the development and the testing phase has already been initiated. We should



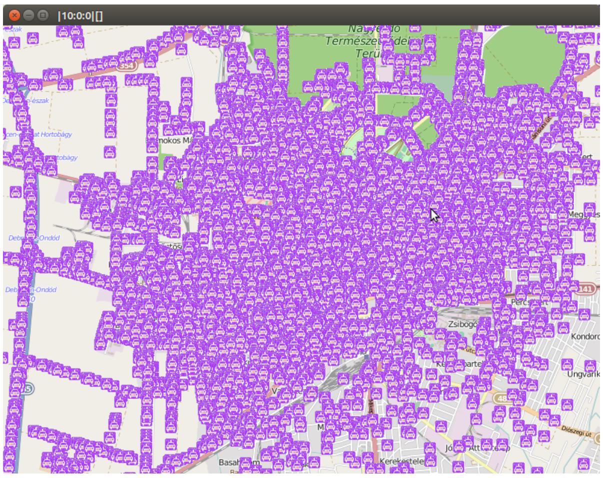
(a) At the beginning of the simulation the number of roads is 78.



(b) After a minute the number of roads is 1085.



(c) After the ten-minute simulation the number of roads is 1725.



(d) The previous time moment of the simulation is shown in the display program.

Fig. 6: Ordered histograms of cars per road. The x-axis shows the roads ordered by the number of contained cars. The y-axis shows the number of cars on the same road. These histograms only include roads that have at least one car. The simulation has started from a "measured" distribution shown in Fig. 5 and contains 10000 routine cars colored purple that move by random walk.

note that this prototype serve as a good foundation for further research, however to set up a system with 40-50 devices (in a city with the size of Debrecen) is necessary to obtain more experience in the analysis.

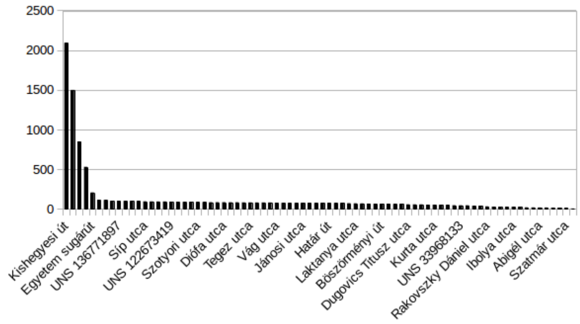
IV. Conclusion

In this paper, we have introduced the OOCWC initiative which is intended to offer a research platform for traffic simulations and attempts to organize a community through a coding competition. We have taken the first steps towards a successful implementation of the OOCWC. The "Justine" prototype has acquitted itself well in our local university environment. We found bugs, of course, but they were not too difficult to solve. Therefore, we believe that the time is ripe to build international partnerships around the OOCWC initiative. This paper is a step forward in this direction, because comprehensive use is the best catalyst for the development of a software system. Therefore, we invite researchers, university teachers and students to join the competition and organize their own championship in their own city.

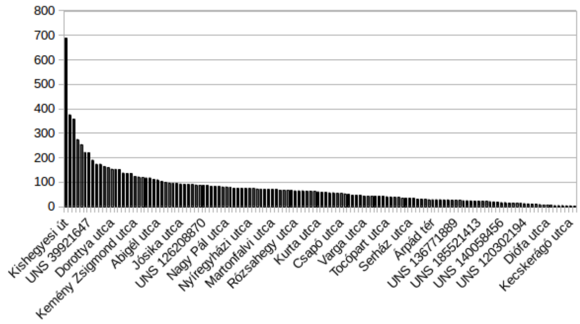
Furthermore, the OOCWC initiative has shown a great step forward in the research of the connection between smart cities and robotcars (mainly in the analysis of real traffic situations). After the proper management of the data collected by the Real-Time Traffic Analyzer and the fine tune of the Robocar City Emulator, we were able to simulate real traffic situations and observe their outcomes. Although, the build-in simulation algorithms need further development, the system has proven to be useful for such functions.

Acknowledgment

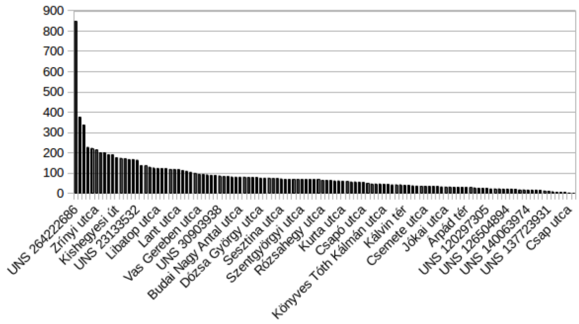
The authors would like to thank all actual and former members of the smart city group at the University of Debrecen. Special thanks to Prof. Joan Plubell Mattia, Tamás Katona and Fanny Monori for a close reading of the manuscript. We are especially grateful to all of the participants in the OOCWC competitions and the students of the BSc courses of "High Level Programming Languages" in the winter and spring semester of 2014/2015 at the University of Debrecen. Finally we would also like to give special thanks to Márton Vona and Balázs Kóti for logo and icons.



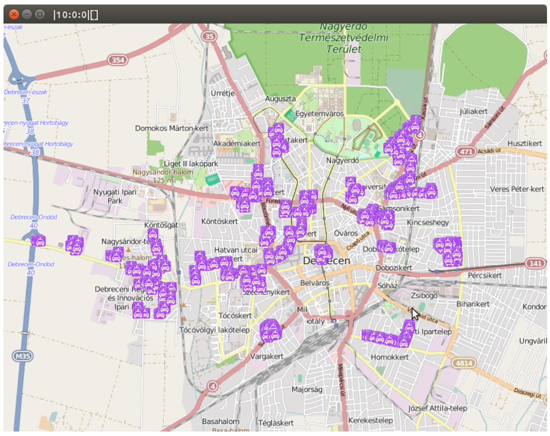
(a) At the beginning of the simulation the number of roads is 78.



(b) After a minute the number of roads is 134.



(c) After the ten-minute simulation the number of roads is 122.



(d) The final time moment of the simulation is shown in the display program.

Fig. 7: This simulation has been run with the parameter `traffict=ANTS`, namely the routine cars move by ant simulation. A comparison of different parameters (such as `ANTS`, `ANTS_RND`, `ANTS_RERND` or `ANTS_MRE`, see https://github.com/nbatfai/robocar-emulator/commit/5899ec98f5ce0acd796298929e53aa61037ba656 for details) of ant algorithms can be seen in a YouTube video at https://youtu.be/BwPGvoHls6c.

References

[1] "World Urbanization Prospects. The 2007 Revision," United Nations, 2007. [Online]. Available: http://www.un.org/esa/population/publications/wup2007/2007WUP_Highlights_web.pdf

[2] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban Computing: Concepts, Methodologies, and Applications," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 3, pp. 38:1–38:55, Sep. 2014.

[3] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "RoboCup: The Robot World Cup Initiative," in *Proceedings of the First International Conference on Autonomous Agents*, ser. AGENTS '97. ACM, 1997, pp. 340–347.

[4] K. Spieser, K. Treleaven, R. Zhang, E. Frazzoli, D. Morton, and M. Pavone, "Toward a systematic approach to the design and evaluation of automated mobility-on-demand systems: a case study in Singapore," in *Road Vehicle Automation*, ser. Lecture Notes in Mobility. Springer, 2014, pp. 229–245. [Online]. Available: http://dspace.mit.edu/handle/1721.1/82904

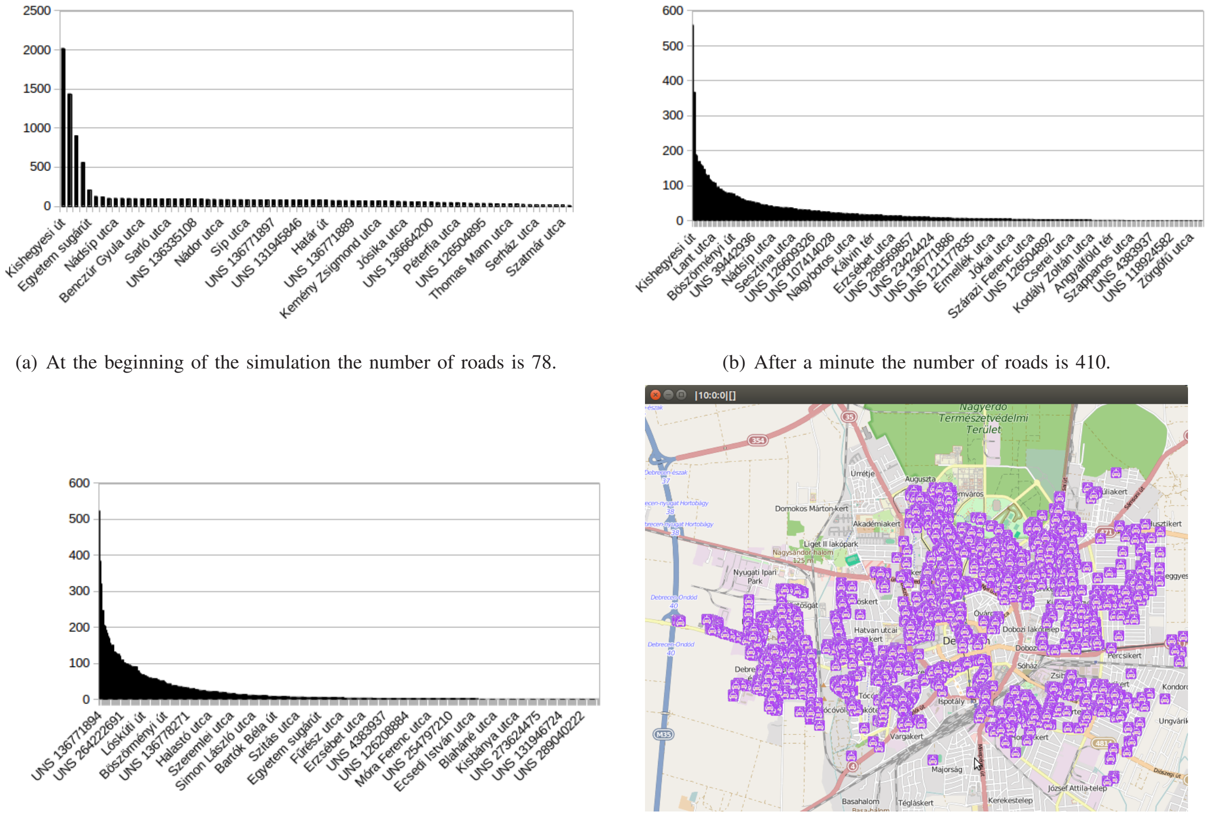
[5] J. Dallmeyer, A. D. Lattner, and I. J. Timm, "From GIS to Mixed Traffic Simulation in Urban Scenarios," in *Proceedings of the 4th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTools '11. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2011, pp. 134–143. [Online]. Available: http://eudl.eu/pdf/10.4108/icst.simutools.2011.245680

[6] K. Nagel and M. Schreckenberg, "A cellular automaton model for freeway traffic," *J. Phys. I France.*, vol. 2, no. 12, 1992.

[7] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E*, vol. 62, pp. 1805–1824, 2000.

[8] S. Fan, M. Herty, and B. Seibold, "Comparative model accuracy of a data-fitted generalized Aw-Rascle-Zhang model," 2014. [Online]. Available: http://arxiv.org/abs/1310.8219

[9] J. Sewall, D. Wilkie, P. Merrell, and M. C. Lin, "Continuum Traffic Simulation," *Computer Graphics Forum*, vol. 29, no. 2, pp. 439–448, 2010.



(a) At the beginning of the simulation the number of roads is 78.

(b) After a minute the number of roads is 410.

(c) After the ten-minute simulation the number of roads is 456.

(d) The final time moment of the simulation is shown in the display program.

Fig. 8: The simulation has been run with the parameter `traffict=ANTS_RND`.

[10] J. Sewall, D. Wilkie, and M. C. Lin, "Interactive Hybrid Simulation of Large-scale Traffic," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 135:1–135:12, Dec. 2011.

[11] N. Bátfai and et al., "robocar-emulator, Robocar World Championship - Robocar City Emulator," 2015. [Online]. Available: https://code.google.com/p/robocar-emulator/

[12] R. Szabó, K. Farkas, M. Ispány, A. Benczúr, N. Bátfai, P. Jeszenszky, S. Laki, A. Vágner, L. Kollár, C. Sidló, R. Besenczi, M. Smajda, G. Kövér, T. Szincsák, T. Kádek, M. Kósa, A. Adamkó, I. Lendák, B. Wiandt, T. Tomás, A. Z. Nagy, and G. Fehér, "Framework for smart city applications based on participatory sensing," in *Cognitive Infocommunications (CogInfoCom), 2013 IEEE 4th International Conference on*, Dec 2013, pp. 295–300.

[13] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[14] R. Lienhart, A. Kuranov, and V. Pisarevsky, "Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection," in *Pattern Recognition*. Springer Berlin Heidelberg, 2003, vol. 2781, pp. 297–304.

[15] A. Schmidt and A. Kasiski, "The Performance of the Haar Cascade Classifiers Applied to the Face and Eyes Detection," in *Computer Recognition Systems 2*. Springer Berlin Heidelberg, 2007, vol. 45, pp. 816–823.

[16] R. Besenczi, M. Szilágyi, and I. Oniga, "Real-time Traffic Analyzer project," 2015. [Online]. Available: https://github.com/rbesenczi/real-time-traffic-analyzer

[17] R. Besenczi, M. Szilágyi, N. Bátfai, A. Mamenyák, I. Oniga, and M. Ispány, "Using Crowdsensed Information for Traffic Simulation in the Robocar World Championship Framework," 2015, Accepted.

[18] "Raspberry Pi single-board computer," 2015. [Online]. Available: http://www.raspberrypi.org/

[19] "OpenStreetMap," 2015. [Online]. Available: https://www.openstreetmap.org/about

[20] N. Walsh, L. Muellner, and B. Stayton, *DocBook: The Definitive Guide*. O'Reilly, 2015.

[21] J. Topf, "Osmium Library," 2015. [Online]. Available: http://osmcode.org/libosmium/

[22] M. Steiger, "jxmapviewer2," 2015. [Online]. Available: https://github.com/msteiger/jxmapviewer2

[23] N. Bátfai and et al., "The Yearbook of the Programmers of University of Debrecen," 2015. [Online]. Available: https://sourceforge.net/projects/udprog/

[24] D. M. Gordon, "The development of organization in an ant colony," *American Scientist*, vol. 83, no. 1, pp. 50–57, 1995.
[25] V. Nugala, S. J. Allan, and J. W. Haefner, "Parallel implementations of individual-based models in biology: bulletin- and non-bulletin-board approaches," *Biosystems*, vol. 45, no. 2, pp. 87–97, 1998.
[26] "The Boost Graph Library (BGL)," 2015. [Online]. Available: http://www.boost.org/doc/libs/1_57_0/libs/graph/doc/index.html
[27] R. Besenczi, M. Szilagyi, and I. Oniga, "Crowd-sourced Traffic Simulator," 2015. [Online]. Available: https://github.com/rbesenczi/crowd-sourced-traffic-simulator

Norbert Bátfai is working as an assistant professor in Faculty of Informatics at the University of Debrecen, Hungary. He received his M.Sc. (summa cum laude) in Computer Science at 1998 from the Kossuth Lajos University (KLTE), Debrecen, Hungary. In 1999, he won the first prize in the Java Programming Contest organized by Hungarian Java Alliance: Sun, IBM, Oracle, Novell and IQSoft.
In 2004, his company won the first prize in the Hungarian Mobile Java Developer Contest organized by Sun Hungary and Nokia Hungary. In 2008, the Hungarian Chief Information Officers' Association has selected him as an IT trainer of the year. He received the Ph.D. degree in 2011. He won the Pollák-Virág award from Scientific Association for Infocommunications Hungary in 2012.

Renátó Besenczi is majoring in Engineering Information Technology MSc and working as a research team member at the University of Debrecen, Hungary. His main research areas are smart cities, crowdsensing, crowdsourcing and medical image processing.

András Mamenyák is majoring in Software Information Technology MSc at the University of Debrecen, Hungary. In 2013, he won the first prize in the XDA Tablet Z Development Competition organized by XDA Developers.

Márton Ispány received his Ph.D. degree in 1997 from the Kossuth Lajos University, Hungary. He is an associate professor at the Department of Information Technology at the University of Debrecen, Hungary. Dr. Ispány's research areas include integer valued time series analysis, branching processes, and data mining. He won the Alexits György award from the Hungarian Academy of Sciences and the Pollák–Virág award from the Scientific Association for Infocommunications, Hungary in 2012.