ACTA IMEKO
ISSN: 2221-870X
December 2019, Volume 8, Number 4, 54 - 61

Simulator effectiveness test for e-training with the use of a DROMADER tele-operated vehicle

Igor Ostrowski[1], Andrzej Masłowski[1]

[1] Digital Mobile Robotics Division/Research and Academic Computer Network/ National Research Institute, Kolska 12, 01-045 Warsaw, Poland

ABSTRACT
This article presents the results of an effectiveness test of a simulator for the e-training operators of tele-operated vehicles with the use of a UGV DROMADER. The mobile platform DROMADER, its control panel, and the simulator system (hardware and software) are described. The results of the 3D modelling and construction of physical models are included. Graphical models using 3dsMax software from a CAD model and the mechanical part developed using Vortex Editor are discussed. Two tests conducted with the participants divided into two groups is undertaken. The first operated the DROMADER robot without previous training on the simulator, the second went through simulator training before operating the robot. The trial was composed to test driving and the manipulation operators' abilities. The results of effectiveness test are discussed.

Section: RESEARCH PAPER

Keywords: operator training; mobile robots; simulator test; humanitarian demining

Citation: Igor Ostrowski, Andrzej Maslowski, Simulator effectiveness test for e-training with the use DROMADER tele-operated vehicle, Acta IMEKO, vol. 8, no. 4, article 10, December 2019, identifier: IMEKO-ACTA-08 (2019)-04-10

Editor: Yvan Baudoin, International CBRNE Institute, Belgium

Received Dezember 5, 2018; **In final form** April 18, 2019; **Published** December 2019

Corresponding author: Igor Ostrowski, e-mail: iostrowski@wp.pl

1. INTRODUCTION

The detection and removal of landmines from infested fields has been a severe problem in recent decades, with vast political, social, and economic dimensions [1], [2]. In this context, the international scientific community has shown a broad interest in solving this problem from different points of view and using different procedures [3]-[7]. The anti-personnel mines, sub-munitions, and unexploded ordnance (UXO) could remain active for several years (in some cases, up to fifty years). They do not discriminate between military and civilian people, killing or injuring indiscriminately, including children and humanitarian aid workers [8], [9].

Many types of mobile vehicles that could take onboard sensors and other equipment to displace them over a mine-infested area (e.g. wheeled vehicles, tracked vehicles, and even legged robots) have been proposed to perform demining tasks efficiently and safely. Wheeled robots are the simplest, cheapest, and easiest to control [10], but tracked robots have an excellent ability to travel across almost any terrain. The need to develop mobile robots to operate in certain situations in which other humanitarian demining systems cannot operate properly or have low yield, have been investigated by several research centres, including the Royal Military Academy (RMA) and Free University of Brussels, Belgium; the Tokyo Institute of Technology, Japan; The Spanish National Research Council (CSIC), Spain; and others.

It is expected that diverse types of robots, differing in the kind of traction, load capacity, range, manipulation ability, and equipment with sensors are applied in the missions. A need to train a significant number of persons in the operation of these robots and to obtain a high proficiency in the operation thereof is particularly relevant in humanitarian demining for the sake of possible contact with explosive substances, creating a danger for operators, the population at large, and the environment. Training tasks require many hours of exercises with different types of robots. Conducting such training with the use of real robots would be unprofitable and probably unfeasible for technical and organisational reasons, including difficulties with the creation of all possible situations with which a robot operator must cope. The use of trainers, simulating robots' behaviour in different situations and circumstances, would be a necessity. Summarising the procedure for training an operator of demining platforms with a simulator can reduce the training time, as studied in [11]-[13]. Among others, the Unmanned Ground Vehicle (UGV) DROMADER used in humanitarian

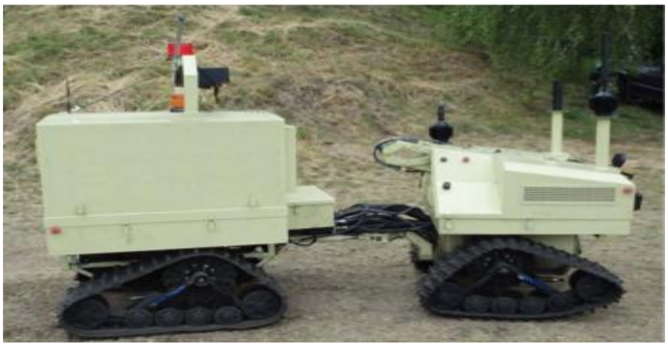

Figure 1. The demining UGV DROMADER.

demining is very hard to operate and a long time for training operators is required. For this reason, the application of a simulator system is advisable. This would reduce the training time and the time for which the robot is used for training purposes.

This article is organised as follows: We first discuss the results of the applications of the tools for 3D modelling and construction of physical models for a simulator dedicated to the e-training of demining robot DROMADER operators. Graphical models are developed using 3dsMax software from a CAD model. It is possible to perform straight conversion from the CAD model to a graphical game-ready model, but the results are poor. The most serious issues were performance and texturing. Graphical models had to be created from scratch. The mechanical part was developed using Vortex Editor. Vortex Editor is a tool that allows the creation of a mechanical system for the Vortex physics universe. It is a type of CAD software, which allows for creating projects, running test simulations, and tuning parameters without the need to export the model to a simulation framework.

Thereafter, the results of the effectiveness test of the simulator for e-training operators of tele-operated vehicles are shown. The test was conducted with the participants divided into two groups. The first group operated the DROMADER robot without previous training on the simulator. Members of the second group went through simulator training before operating the robot.

The trial involved two exercises. The first of these was devoted to testing the driving abilities of the operators. The task for the operator was to cover the distance in a short time and to not make any errors (such as deviations from the route or collisions with obstacles), which were scored negatively. The second exercise was devoted to testing the manoeuvre abilities, and the manipulation of heavy elements (concrete blocks) using the robot’s manipulator was the substance of the test.

Finally, some conclusions are offered in Section 6.

2. UGV DROMADER

The UGV DROMADER robot, shown in Figure 1, was designed and built by the Military University of Technology, Warsaw, Poland [14]. The robot's structure is characterised by a segmented chassis connected using an articulated joint. The turning angle of the front segment in relation to the rear can reach up to 90 °. This provides high manoeuvrability, allowing for easy turns in confined spaces. In conjunction with the track drive system and dual-axis drive, it allowed the DROMADER to achieve very good off-road mobility.

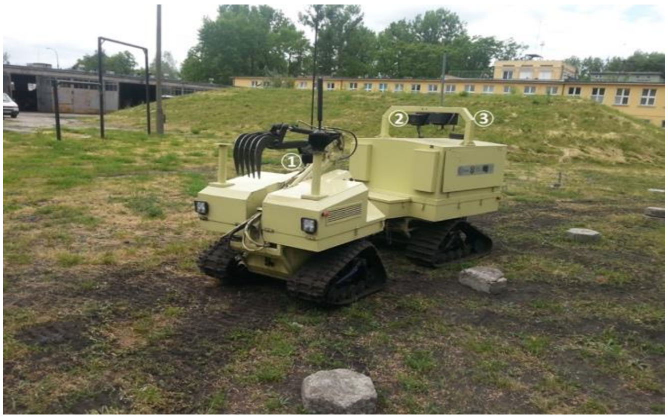

Figure 2. The DROMADER robot’s cameras marked as 1, 2, and 3.

The rear section houses a 15kW combustion drive unit, while the front section houses the control system and several components of the platform's drive system. Because of the desire to minimise the weight of the robot, the entire chassis of the robot is made of aluminium.

The robot may be equipped with an experimental protection system against remotely detonated Improvised Explosive Devices (IEDs), which may be mounted in the rear section above the engine inside the chassis. Using this configuration requires the detector and jammer antennas to be mounted in very specific locations to avoid interference and not to reduce their working area due to the chassis structure and other equipment mounted on the robot. The rear segment also contains a Wi-Fi transceiver antenna (2.4 GHz) for the remote-control system.

The manipulator was mounted on the front section of the robot. Part of its structure and location requirements was that, due to its transport position, it would not obscure the operator's field of view registered by the teleoperation system. The manipulator has three degrees of freedom allowing for the control of the effector's position in the vertical plane. Due to weight limitations, the horizontal plane control is achieved by turning the entire front section instead of an additional control mechanism.

The DROMADER is equipped with three cameras, as shown in Figure 2. A typical gamepad serves as the control console for robot operation.

The DROMADER is operated from an operation stand, with the use of camera views. The operation stand engaged during the validation trial is presented in Figure 3.

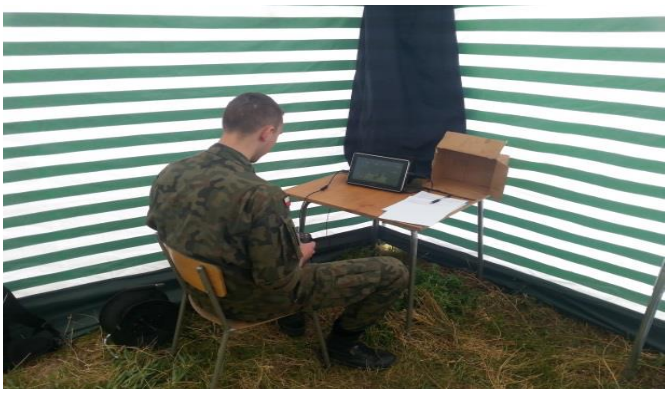

Figure 3. The DROMADER’s operation stand.

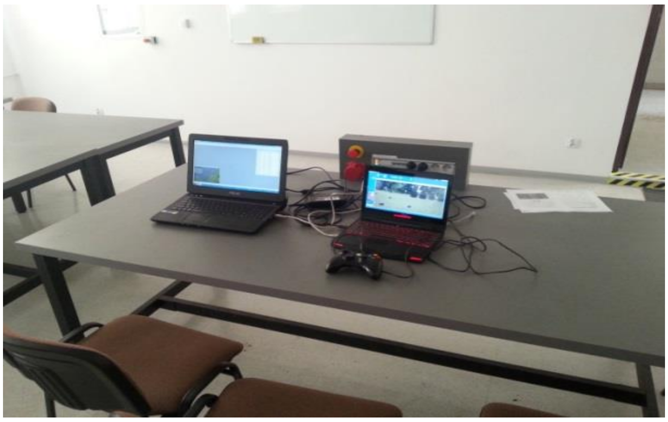

Figure 4. The DROMADER training stand.

The developed simulator tool is used for training DROMADER's operators. The training stand with the simulator, engaged during the validation trial, is presented in Figure 4.

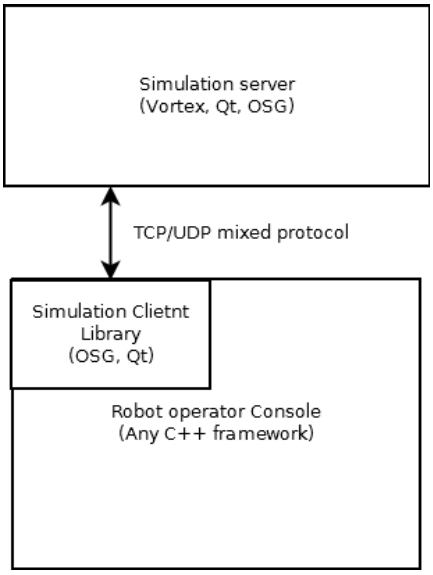

Figure 5. The simulation system concept.

3. UGV DROMADER SIMULATOR

3.1. Simulation framework description

The simulation system we developed is a universal tool for multi-robot operation. Figure 5 shows the simulation system concept. The created software solution can be divided into two sub-projects. The first is the simulation server, which is a standalone binary package. This program runs a simulation server, which is responsible for physics resolution, multiple-client scene synchronisation, and monitoring the mission progress. The second is a library that allows for communication with the simulation server. This software is a dynamic linked library. It allows for separating the simulation and trainees' consoles, so the simulation process can be run on multiple machines robustly. Another important advantage of the presented solution is the fact that the rendering scene with multiple cameras can be a heavy task for a computer, but it no longer affects physics simulation. The scene is synchronised by means of a mixed Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) protocol. It allows the connection of up to ten clients to the system.

For the physics simulation, the CM-Labs Vortex library is used. This library is a solution for serious games (games not mainly designed for pure entertainment), industry simulations (like cranes and excavators), and military training. It comes with a mature framework that allows for the development of a simulation of almost any type of vehicle. The framework also allows for the simulation of internal combustion engines, electric motors, a variety of transmission systems (including classical manual and hydrostatics transmissions), and nearly any kinematic layout. It is also possible to simulate complex structures, like robotics manipulators. Moreover, some simulation situations can be scripted using Python language. The library is proprietary and closed-source. It is well integrated with Open Scene Graph (OSG) [12], another library, which is an open-source, OpenGL-based graphics engine. It is a C++ library that is very often chosen in serious games and training systems. This library provides a number of useful tools, such as a hierarchical structure of the scene, a hierarchical structure of the graphical model with several types of relationships between model parts, graphical model loaders (including proprietary formats like like *.flt, *.cae, or *.fbx), and many levels of detail. It also supports OpenGL implementation from 1.1 up to 4.0, including OpenGL ES. The next part of the simulation framework is the Qt 5 library. This is a popular open-source C++ framework under the Lesser General Public License (LGPL). It is widely used in Graphical User Interface (GUI) applications, but it provides many solutions for non-GUI developers. In the project, QtXML and QtNetwork modules were heavily used. The last used library is Boost.Python. This library is part of the Boost open-source set of C++ libraries. Boost.Python allows for the exposure of some dynamically allocated C++ objects in the Python name-space. This allows for the creation of tools that can be used during the execution of simulated mission scenarios. For example, during mission scenario development, some triggers can be prepared. Those triggers can cause a number of actions, such as adding penalty points and time measurement. Because of the use of Python language, compilation is not needed. The code is loaded and interpreted by the integrated Python parser in runtime.

The simulation client used only OSG and Qt libraries. It makes this solution GUI-ready, multi-platform, and lightweight. It can be easily integrated with the operator or trainer console. This library deals with recreating the graphical scene in client software; rendering camera views; and sending and receiving data from and to robots. It can also receive some messages from the simulation server, which can be sent from the scenario's script, executed on the server's side.

3.2. The DROMADER robot simulation

The project for the robot in the simulation system can be divided into two main parts. The first is the graphical model, and the second is the physical model. It is important that the second part is based on the created graphical model. Graphical models are developed using 3dsMax software based on a CAD model. It is possible to perform a straight conversion from the CAD model to a graphical game-ready model, but the results are poor. The most serious issues were performance and texturing. The graphical models had to be created from scratch. The mechanical part was developed using Vortex Editor. Vortex Editor is a tool that allows the creation of a mechanical system for the Vortex physics universe. It is a CAD-like software, which allows for creating projects, running test simulations, and

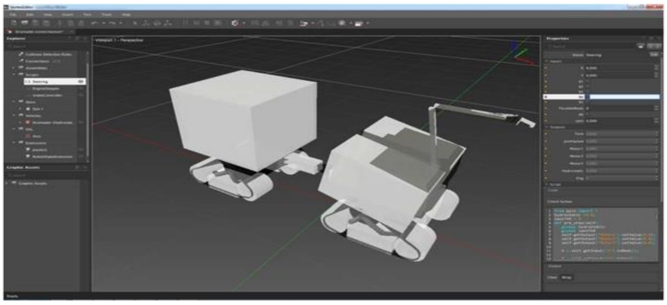

Figure 6. The physical model in Vortex Editor.

tuning parameters without the need to export the model to the simulation framework.

The graphical model is organised in a hierarchical structure, where every moveable part has its own node with its own coordinate system. During physics modelling, a number of properties were added. First, the collision geometries were created. The collision geometries are simple geometric primitives, like boxes and cylinders that interact with each other. Collisions are not resolved between graphical meshes. These shapes are organised into parts. Every collision shape has its own mass, inertia matrix, and materials. Thereafter, constraints were created. These are objects that create a connection between two or more parts. It could be, for example, prismatic or hinge-pair. Several parameters describe the constraints, like character (free, motorised, or locked), internal friction, damping, stiffness, motor, blockage, maximum force, and limits. In this fashion, hydraulics actuators were added to the simulation. The next robot-vehicle system was added to the physics model. There is a typical vehicle template, like cars (All Wheel Drive or Rear Wheel Drive or Front Wheel Drive with auto and manual transmissions), tracked vehicles with skid-steering (like excavators [hydrostatics torque division] or tanks [differential torque division]). The robot DROMADER is a unique vehicle solution, so a new template was created (Figure 7).

The template, after having been loaded to Vortex Editor, automatically creates complex structures like the AWD truck. Objects such as the engine, differentials, shafts, transmission system, driving constraints, and wheels with suspension are introduced and assigned proper relationships. The structure of the DROMADER robot is fairly simple. The engine is connected to the hydrostatics transmission and the ratio of every output motor can be set separately. The robot does not have differential steering (like an excavator), so the ratio for each track is the same. Driving is realised using hinge constraints with a velocity actuator. The final layer of the physics model is logic, which does not have to be hard-coded in the application code. Every robot's object can communicate with Python scripts. These scripts are an elegant way of implementing some logical dependencies. For example, the DROMADER robot has three driving modes. Switching driving modes affects motor velocity but also reconfigures the hydrostatics system, connecting the motors as parallel; thus, a higher torque is generated. This function cannot be implemented directly but can be realised as a script that would change the hydrostatics system ratio in a proper way.

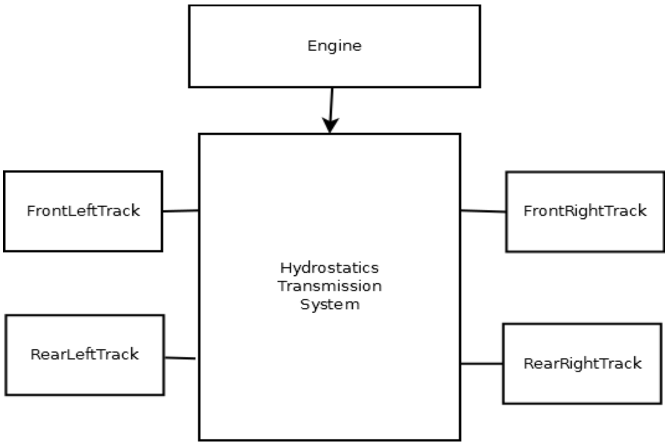

Figure 7. The robot DROMADER system template.

Finally, the Vortex High Level interface allows creating a universal interface for the vehicle model to communicate with the rest of the application. After loading the model into the final simulation application, it allows us to read and write the parameters from the C++ code.

3.3. Environment

The sample environment was created from point cloud. It is much easier to create a real location model using geodetic data

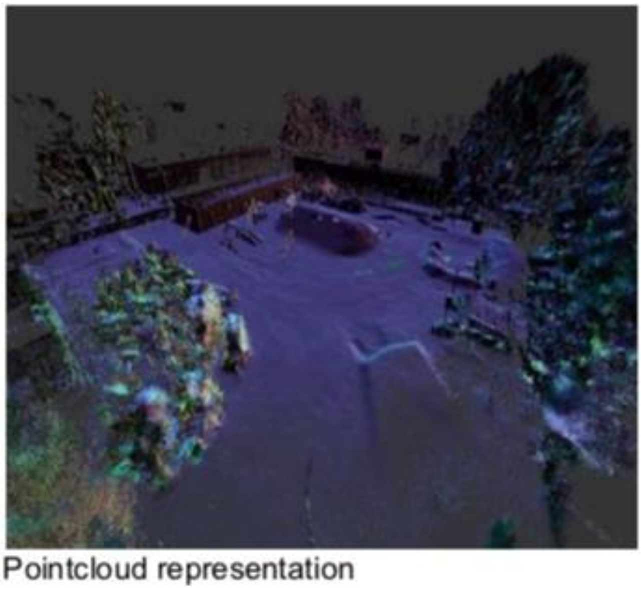

Figure 8(a). Environment representation.

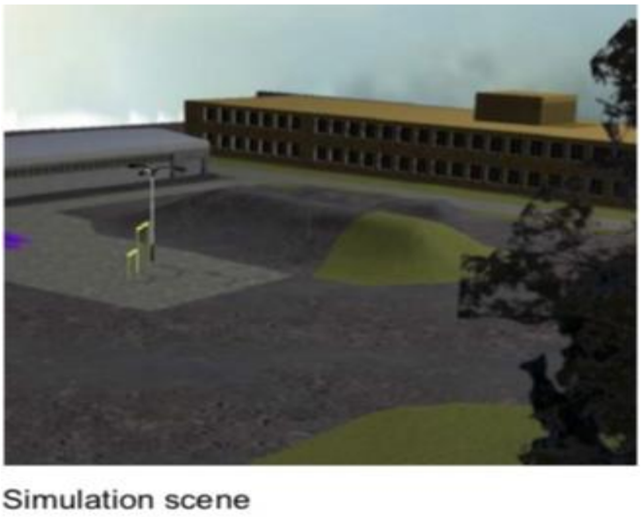

Figure 8(b). Environment representation

than only using photos [19], [20]. The environment was also created using 3dsMax. The building's floor plans were extracted from point cloud data, and an elevation map was created for the ground shape. Vegetation was replaced with a simple billboard representation. The billboard is a graphics method used for creating planar shapes that always face the viewer's camera. This method is widely used for simple vegetation rendering [22], [23].

The scene was textured and finally imported to Vortex Editor, where the parameters of the materials were applied to the surface [24], [25].

The configuration is an object-oriented XML file that the simulation server loads during initialisation. These files contain a great deal of information, like the spawn position of the robots, the scene type, and the position of third-person view cameras. It allows us to setup or modify the simulation scene in a highly convenient way. Furthermore, the XML file contains a script coded in Python language, which allows us to interact with the simulation, so some events can be dynamically created. It can create many opportunities to develop life-like, interactive scenarios.

3.4. Operator console

The operator console, as mentioned above, was created using the Simulation Client Library. The DROMADER operator's console is a handheld computer with a game-style controller. It communicates with the robot using a wireless connection. The operator can observe one camera and steer the robot and its manipulator using analogue sticks and buttons on the controller. Operations like connection; disconnection; and turning the robot's system on and off are realised by means of the touch-screen GUI. Due to a lack of access to the source code of the operator's console, the interface was recreated using our own framework. This program was successfully integrated with the Simulation Client Library, and finally, a realistic operator's console was made. The separation of the Simulation Client Library and the console's code is a great advantage because it makes a simulator independent from the input and output devices used and it allows for integration with real equipment. A view on the simulator screen is presented in Figure 9.

4. COURSE OF THE VALIDATION TRIAL

The validation trial was conducted with 20 participants (Military Academy students) divided into two groups. The first group, comprising 11 persons, operated DROMADER without previous training on the simulator. The members of the second nine-person group participated in simulator training before operating the robot.

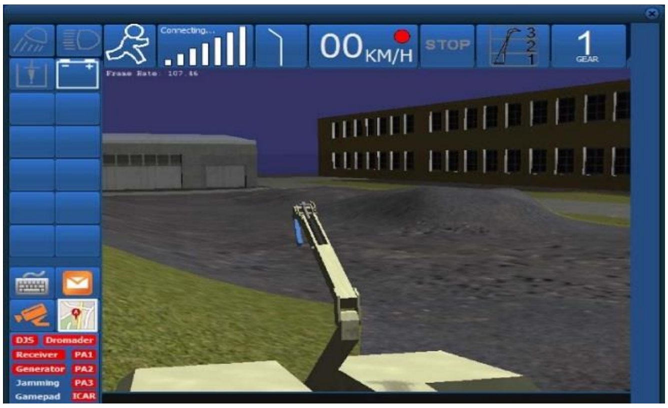

Figure 9. The operator console connected to the simulator server.

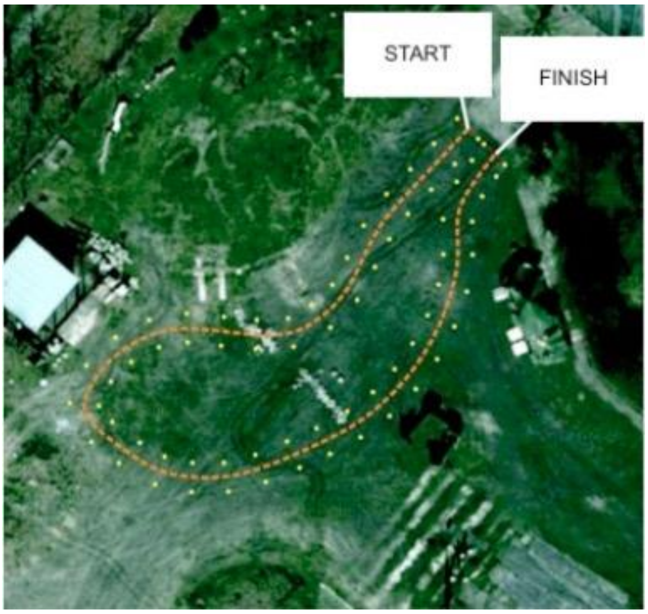

Figure 10(a). The route for testing the driving ability of DROMADER (aerial view).

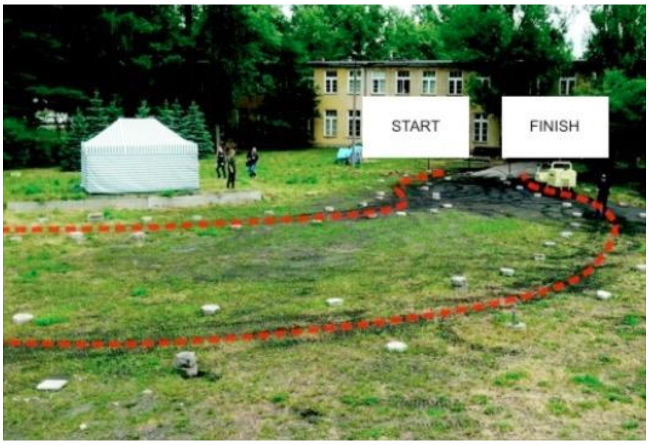

Figure 10(b). The route for testing the driving ability of DROMADER (ground view).

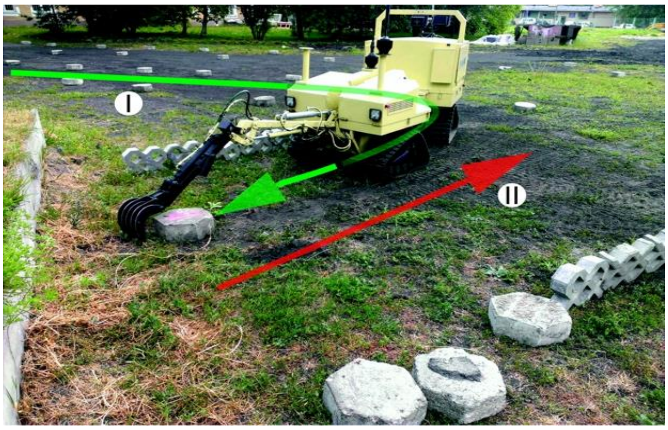

Figure 11. The exercise for testing the robot's manipulation abilities.

The trial involved two exercises. The first of these was devoted to test driving the abilities of the operators. The route used in this exercise is presented in Figure 10(a) and Figure 10(b). The task for the operator was to cover the distance in a short time and not make errors (such as deviations from the route or collisions with obstacles), which were scored negatively in accordance with [1].

The second exercise was devoted to test the robot's operator ability to manipulate heavy elements (concrete blocks). The exercise, presented in Figure 11, consisted of driving up to a certain block, localising it, grasping it (I), and then carrying it to a certain location (II). Errors were scored negatively. The block, being the manipulation subject, was located near a vertical wall. It represented an obstacle, like other concrete objects in the area.

5. RESULTS

In both the manipulation and driving exercises were the results obtained by comparison of the results for the participants of the two groups i.e. with and without previous training on the simulator. In both exercises, the time taken for fulfilling the task and the number of errors made were measured. The global outcome in relation to the time was calculated in two ways, as an arithmetic mean and as a median, according to the following formulas:

a) arithmetic mean (Equation 1):

$$x_{\mathrm{sr}} = \sum_{i=1}^{n} \frac{x_i}{n} \tag{1}$$

and median (Equation 2):

$$m_{\mathrm{e}} = \begin{cases} x_{\frac{n+1}{2}} & \text{for odd } n \\ \frac{1}{2}\left(x_{\frac{n}{2}} + x_{\frac{n}{2}+1}\right) & \text{for even } n \end{cases} \tag{2}$$

where:

n – number of participants;

x – time taken to fulfil the task.

The results obtained in the driving exercise are shown in Table 1 and Table 2.

After rejecting the extreme values, the arithmetic mean and median adequately amounted to 232 and 230.

After rejecting extreme values, the arithmetic mean and median adequately amounted to 306 and 294.

The differences in percentages between outcomes of two groups were calculated for the achieved times according to the following formulas:

$$\Delta x_{\mathrm{sr}} = \frac{x_{\mathrm{srnsz}} - x_{\mathrm{srsz}}}{x_{\mathrm{srnsz}}} \cdot 100\% \tag{3}$$

Table 2. Driving outcomes of participants not trained on the simulator.

Participant	Time taken to fulfil the task, in s	Duration of training in min	Number of errors	Arithmetic mean x_{sresz} in s
1	280	3	319	308
2	377	4		
3	324	4		
4	348	8		
5	260	4		
6	308	3		
7	262	2		
8	455	7		
9	254	0		
10	247	0		
11	398	8		

$$\Delta m_{\mathrm{e}} = \frac{m_{\mathrm{ensz}} - m_{\mathrm{esz}}}{m_{\mathrm{ensz}}} \cdot 100\% \tag{4}$$

The obtained values are as follows:

Δx_{sr} = 22 %, and for rejected extreme values 24 %,

Δm_{e} = 21 %, and for rejected extreme values 22 %.

The differences in percentages Δe between the outcomes of the two groups for errors made during the driving exercise, calculated in the same way, are as follows:

Δe = 29 % for the total number of errors,

Δe = 37 % for the maximum number of errors made by a single participant.

The results obtained in the manipulation exercise are shown in Table 3 and Table 4.

The differences in percentages between the outcomes of the two groups for the time taken to fulfil the task, calculated in the same way as for the driving exercise, are as follows:

Δx_{sr} = 44 %,

Δm_{e} = 42 %.

6. CONCLUSIONS

The test described in this paper confirms that using simulators in training robot operators is an effective way of reducing the cost of training and of allowing trainees to make errors with lower levels of stress. Comparing our method with

Table 1. Driving outcomes of participants trained on the simulator.

Participant	Time taken to fulfil the task, in s	Duration of training in min	Number of errors	Arithmetic mean x_{sresz} in s	Median m_{esz} in s
1	217	40	0	249	243
2	302	20	1		
3	243	30	3		
4	245	30	3		
5	385	30	4		
6	193	60	4		
7	190	60	5		
8	246	30	5		
9	216	30	0		

Table 3. The manipulation outcomes of the participants trained on the simulator.

Participant	Time taken to fulfil the task, in s	Duration of training in min	Number of errors	Arithmetic mean x_{sresz} in s	Median m_{esz} in s
1	207	40	1	190	192
2	175	20	3		
3	231	30	0		
4	143	30	0		
5	233	30	3		
6	174	60	1		
7	192	60	3		
8	165	30	0		
9	192	30	1		

Table 4. Manipulation outcomes of the participants not trained on the simulator.

Participant	Time taken to fulfil the task, in s	Duration of training in min	Number of errors	Arithmetic mean x_{sresz} in s
1	305	4	341	334
2	414	2		
3	283	4		
4	334	1		
5	383	1		
6	294	1		
7	316	2		
8	415	5		
9	249	1		
10	403	2		
11	354	0		

the use of real machines during the first lessons of robot use, our method is a much faster way of getting satisfactory results.

Our results were validated by determining how the simulator under consideration meets the requirements of its performance threshold. For the simulator, a triple-value threshold was set with three performance levels:

- Level U – unsuitable tool – less than 10 % difference between the groups' outcomes;
- Level I – the tool needs improvement – difference between 10 % and 30 %;
- Level S – suitable tool – difference larger than 30 %.

Differences of outcomes in driving exercises are in the 21 % - 24 % bracket for the time taken to fulfil the task and in the 29 % - 37 % bracket for the number of errors made.

Differences of outcomes in manipulation exercises are in the 42 % - 44 % bracket for the time taken to fulfil the task and reach 40 % for the number of errors made.

Considering that the time taken to fulfil the task of approach in demining missions is of secondary importance [4], it is justified to decide the level S obtained by the trainer-simulator as a result of the validation trial.

REFERENCES

[1] Y. Baudoin, M. K. Habib, I. Doroftei Mobile robotics systems for humanitarian demining and risky interventions, in: Using Robots in Hazardous Environments Y. Baudoin, M. K. Habib (editors). Woodhead Publishing Ltd, Elsevier, UK, 2011, pp. 3-31.

[2] Y. Baudoin, I. Doroftei, TriDem – a wheeled mobile robot for humanitarian mine clearance, Proc. of the 6th IARP Workshop on Humanitarian Demining (HUDEM), 24-26 April 2012, Sibenik, Croatia, pp. 93-98.

[3] S. Hirose, K. Kato, Quadruped walking robot to perform mine detection and removal task, Proc. of the 1st International Conference on Climbing and Walking Robots, 1998, Brussels, Belgium, pp. 261-266.

[4] P. Gonzalez de Santos, J. Cobano, E. Garcia, J. Estremera, M. Armada (2007) A six-legged robot-based system for humanitarian demining missions, Mechatronics 17 (2007) pp. 417-430.

[5] E. Fukushima, M. Freese, T. Matsuzawa, T. Aibara, S. Hirose, Humanitarian demining robot gryphon. Current status and an objective evaluation, Intl. Journal on Smart Sensing and Intelligent Systems 1(3) (2008) pp.735-753.

[6] R. Fernández, H. Montes, C. Salinas, P. González de Santos, M. Armada, Design of a training tool for improving the use of hand-held detectors in humanitarian demining, Industrial Robot: An International Journal 39(5) (2012) pp. 450-463.

[7] H. Montes, L. Mena, R. Fernandez, J. Sarria, M. Armada, Inspection platform for applications in humanitarian demining, Proc. of the 18th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, 6-9 September 2015, HangZhou, China, pp. 446-453.

[8] ICBL, International campaign to ban landmines: 12th annual landmine monitor report, 2010. http://www.the-monitor.org/media/1641811/Landmine_Monitor_2010_lowres.pdf [Online, accessed: 14 June 2016]

[9] ICBL, International campaign to ban landmines: 17th annual landmine monitor report, 2015. http://www.the-monitor.org/media/2152583/Landmine-Monitor-2015_finalpdf.pdf [Online, accessed 19 September 2016]

[10] P. Kopacek, L. Silberbauer, A new locomotion concept for humanitarian demining robots, Proc. of the 7th IARP International Workshop on Humanitarian Demining, 28-30 March 2008, El Cairo, Egypt, pp. 32-36.

[11] A. Maslowski, Training in military robotics and EOD unmanned systems, in: Conference and Seminars, NATO EOD Center of Excellence Publishing, Trenčin, Slovakia, 2014.

[12] T. Łuczynski, I. Ostrowski, M. Zoppi, J. Bedkowski, A. Maslowski, M. Przybylko, P. Wojcieszynska, M. Szczepaniak, Mobile demining devices modeling for the training purpose using VORTEX framework, Proc. of the International Conference on Military Engineering: Problems and Prospective, 23-25 April 2013, Institute of Military Engineering Publishing, Wroclaw, Poland, pp. 233-240.

[13] A. Kaczmarczyk, M. Kacprzak, A. Maslowski, Muti-level simulation training of devices operators: an application to mobile robots operators, Elektronika 11 (2009) [in Polish].

[14] J. Łopatka et al., Intelligent Communication Supervising on the Battlefield with the use UGV DROMADER, Warsaw Military University Bulletin 62(3) (2013) pp. 13-25 [in Polish].

[15] J. Bedkowski, K. Majek, I. Ostrowski, P. Musialik, A. Maslowski, A. Adamek, A. Coelho, G. de Cubber, Methodology of training and support for urban search and rescue with robots, Proc. of the 9th International Conference on Autonomic and Autonomous Systems, 2013, Lisbon, Portugal.

[16] TIRAMISU consortium, Training tools for the TIRAMISU project: Annual Report [Online] http://www.fp7-tiramisu.eu/.

[17] P. Musialik, J. Bedkowski, I. Ostrowski, T. Łuczyński, K. Majek, A. Maslowski, A. Adamek, Virtual Environment Modeling for the Training of Unmanned Vehicle Operators. Air Force Academy Publishing, Deblin, Poland, 2015.

[18] J. Bedkowski, A. Maslowski, G. de Cubber, Real-time 3D localization and mapping for USAR robotic application, Industrial Robot: An International Journal 39(5) (2012).

[19] P. B. Silva, A. Coelho, R. Rodrigues, A. A. de Sousa, A procedural geometry modeling API, Proc. of the International Conference on Computer Graphics Theory and Applications, GRAPP & IVAPP 2012, 24-26 February 2012, Rome, Italy, SciTePress.

[20] P. B. Silva, A. Coelho, Procedural modeling for realistic virtual worlds development, Journal of Virtual Worlds Research, Vol. 4, No. 1, 2011.

[21] D. Jesus, A. Coelho, C. Rebelo, A. Cardoso, A. A. Sousa, A pipeline for procedural modelling from geospatial data. Proc. Eurographics 2012, pp. 9-10, 2012.

[22] T. Martins, P. B. Silva, A. Coelho, A. A. Sousa, An urban ontology to generate collaborative virtual environments for municipal planning and management, Proc. GRAPP 2012 -7th International Conference in Computer Graphics Theory and Applications, pp. 507-510, 2012.

[23] J. Bedkowski, K. Majek, A. Nüchter, General Purpose Computing on Graphics Processing Units for Robotic

Applications. Journal of Software Engineering for Robotics, vol 4, no 1(2013), pp. 22-33.

[24] M. A. Fischler, R. C. Bolles, Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated aartography, Comm. of the ACM 24 (6), June 1981, pp: 381–395, doi :10.1145/358669.358692.

[25] S. Schumacher, J. Bohm, Georeferencing of terrestrial laser scanner data for applications in architectural modelling, 3D-ARCH 2005: Virtual reconstruction and visualization of complex architectures, Mestre-Venice, Italy, 22-24 August 2005, Universität Stuttgart, 2005.

[26] J. Elseberg, D. Borrmann, A. Nüchter, Efficient processing of large 3d point clouds. Information, Communication and Automation Technologies (ICAT), 2011 XXIII International Symposium on. IEEE, 2011.