Simple Mobile Robots and Self-Adaptive Wireless Networks

Hartmut Surmann, Fraunhofer IAIS, hartmut.surmann@iais.fraunhofer.de, Germany
Rainer Worst, Fraunhofer IAIS, rainer.worst@iais.fraunhofer.de, Germany
Erik Zimmermann, Fraunhofer IAIS, erik.zimmermann@iais.fraunhofer.de, Germany
Stefan Wilkes, University of Applied Science Gelsenkirchen, stefan.wilkes@gmail.com, Germany
Tom-Marvin Liebelt, University of Applied Science Gelsenkirchen, fh@tom-liebelt.de, Germany
Christopher Eulering, University of Applied Science Gelsenkirchen, christopher.eulering@gmx.de, Germany

Abstract

Disaster areas require mobile robots with extreme capabilities. This paper presents an approach for setting up a network infrastructure to operate such mobile robots. We present a waterproof netserver box as a main component and mobile router robots (RC cars) to extend the network capabilities. The simple mobile robot consists of cheap standard RC components. It provides a view into the disaster area with its sensors, e.g., cameras, and in addition and very important, it sets up a persistent communication between all mobile robots and rescuers. All components use ROS as a middleware and can be integrated in the overall system. In addition to infrastructure networks, mesh networks are also supported to replacce the destroyed network infrastructure. Furthermore, the network and robots are prepared for cloud computing.

1 Introduction

1.1 Motivation

Urban search and rescue (USAR) missions need support from modern computer technologies to fulfill their task to help people. In the past decade robotic researchers and computer scientists enhanced robots especially to help in rescue missions, i.e., funded by the EU and DARPA [1, 3, 17]. These projects lead to complex and expensive robots, e.g., tracked robots, humanoids or drones, but some problems are still open, e.g., how to ensure communication ([18]) and what happens with robot suicide missions. Cheap, disposable robots would make the decision for suicide robot missions much easier for the mission commander. Furthermore, simple robots could help to gather information as sensor nodes to clear up rescue scenarios and to help the commander to make good decisions. For example, the collapse of the historical archive of the city of Cologne in march 2009 lead to hundreds of cracks, which had to be observed manually during the first five days of the mission[20]. Changes of the cracks were not well documented since, e.g., short time interval photos were not available. Beside the use of tracked robots (UGV) and drones (UAV) the NIFTi and TRADR projects [2, 3] have also to cover the communication problem and therefore to build robust network nodes to set up a network for the robots. Simple communication robots extend the adaptaibility of the network structure and will also be presented in this paper. The paper is structured as follows;

After a short state of the art in the next section we present the network components in chapter 2 followed by chapter 3, which presents use cases and performance measurements.

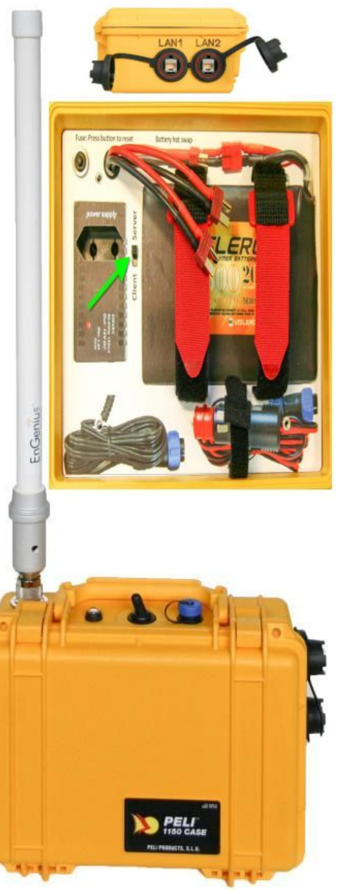

Figure 1: Outdoor resistant WLAN box. Hardware components: RaspberryPI, battery pack, Bullet M2.

1.2 State of the art

Nowadays computer networks consist mainly of ethernet and WLAN networks. WLAN networks can be used in infrastructure or in ad-hoc mode. This allows several network architectures [19, 4]. For each of the network topologies thousands of network adapters and routers exist e.g. [8, 9]. Furthermore ,cheap and small PC boards are available, e.g., the RaspberryPI with a huge amount of commercial and open source software. Also RC cars are very common, cheap, and widely used e.g controlled by special hardware and a router [12]. In general sensor networks and nodes is a wide area [5]. The integration of these components into network nodes suitable for rescue missions is a new approach and looks very promising.

2 Network design

2.1 The Netserver Box

The waterproof netserver box is designed to provide LAN and WLAN capabilities in field operations and in a lab environment (**Figure 1**). It can be configured by a hardware switch to act as a server or as a client in the network and is protected against overvoltage and overcurrent. Usally one box works as a server in the network. The client mode can be set quickly with the switch, when it is required to connect several boxes together. The box can be powered by a LiPo battery (Vislero LiPo 6600, 14,8V, 4S1P flat pack), wall socket (230VAC, AC/DC converter) or automobile plug (12VDC); it contains a battery low indicator and a battery hot swap over a second T-plug. Usually, in field missions, the LAN1 waterproof socket (type PX0834) of the server is connected to a Hub in the command center to connect all stationary computers to the field, i.e., mobile robots or in-field rescuers. It offers DNS, NTP and DHCP services, which are necessary for all robots and drones that run Ubuntu and ROS. This setup establishes a complete independent network in the field. The additional LAN2 socket can be used to connect the box to an external network ,e.g., internet if avaliable in the field of operation. This network adapter is automatically configured via external DHCP. If both networks are connected, computers from the internal network can also access the external network (maybe internet), which is very helpful for the maintenance, administration and setup of the complete system. WLAN can be provided by different kinds of bullets, e.g., at 2.4 GHz and 5 GHz, which are connected to a RaspberryPI. The whole network consists of the following components:

- RaspberryPI (running Raspbian image of the dedicated netserver for DHCP, DNS, NTP, ...)
- Bridge node at ground station, e.g., Ubiquiti 2,4 GHz Bullet M2 802.11N or Ubiquiti 5 GHz Bullet M5 802.11 and Omni-directional antenna
- Cables and power supply for each Ubiquiti Bullet
- Switch at ground station,extension cables, and power bars

The use of bridge nodes for the WLAN communication has the advantage that no WLAN device drivers for Linux are needed, which in general could have reliabilty problems over a longer period of time. A crucial issue for ROS-based ([16, 6]) communication is exact synchronization of the nodes involved. To provide a reliable time base for all PCs in the network to synchronize with a NTP (Chrony) server is running on a RaspberryPI ([7]) inside the netserver box, which also provides DHCP and DNS services. Tests of the network configuration are given in section 3.2.

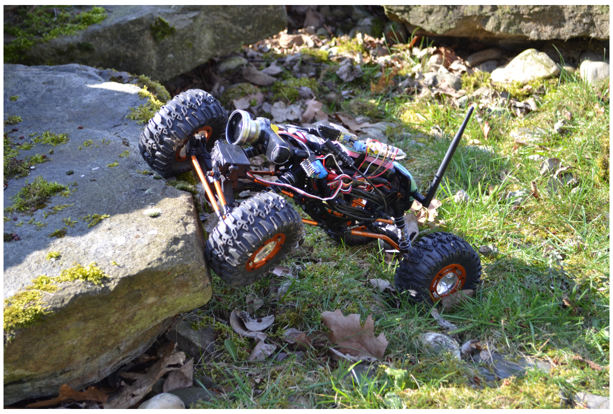

Figure 2: The Rock Crawler in action. Attached components: RaspberryPI, battery pack, ALFA WLAN Adapter and a wide angle Logitech webcam with a microphone. A video can be found at: http://www.youtube.com/watch?v=St8QtC1kut0.

2.2 Simple Mobile Robots act as mobile access points

To adapt the network to different configurations, we set up two mobile robots to operate as movable WLAN bridges as seen in **Figure 2** and **Figure 3**. The robots should be as simple and cheap as possible to interact in so called suicide robot missions. In some cases the robot has to be fast to travel long distances and in other cases the robot has to be powerful to crawl through difficult terrain like small rocks or sidewalks.
For that reason, we have designed a computational unit that is 100% compatible with the network box introduced in the previous section and which is able to control several PWM channels. This computational unit can be attached to almost every commercial RC model. The result is a disposable robot which can be configured for specific scenarios like ground, aerial or water operations.

2.2.1 Hardware Design

We have build two variations of a mobile robot using RC model cars. The first robot is a fast mover based on a Reely Truck, e.g., [10] for about 170,- Euro. The car can reach a velocity up to 70 km/h. The second robot is a rock crawler based on an Axial AX10 Scorpion, e.g., [11] for about 200,- Euro. This robot (5 km/h) is not that fast as the the first one, but it is able to drive through difficult terrain and climb up obstacles with a slope up to 70Â°. The only modification of these RC models is the detachment of the RC receiver unit, so the computational unit can be connected to the PWM signals directly. Usually a RC model has two PWM channels for full operation. One channel is to control the steering servo. This channel needs PWM pulses with a period of 20 Hz. The duty cycle of the signal sets the position of the servo motor, e.g., 1,5 ms interval is the central setting while 1 ms and 2 ms duty cycles are standing for the maximal deflections. The other channel is to control the linear velocity. There can be two operation modes. One is exactly like the servo steering method, which means that the model can be in a neutral position or in full forward / reverse mode. The second operation mode is trivial. In this case the duty cyle controls the power which flows through the motor immediately.

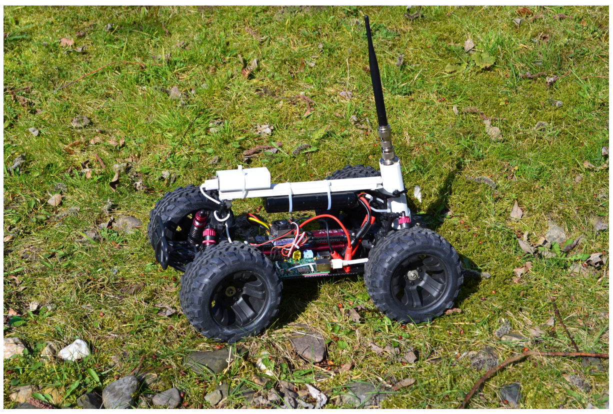

Figure 3: The Fast Mover. Attached components: RaspberryPI, battery pack, Bullet2M Router. A video with an attached network camera can be found at: http://www.youtube.com/watch?v=qVALTltiF6o

The computational unit is based on a RaspberryPI ([7]), like in the netserver box above. A big advantage is that the Debian operating system, installed on the PI, allows the usage of the robot operating system ROS Hydro [16, 6]. A USB power box, attached to the RC models, provides the necessary operation power of 5V to the PI. The main reason, why we are using a separate power pack, is that the power supply for the computer system is independent from the power supply of the mobile base. Therefore, we can reach an online time up to 10 hours, depending on connected hardware. For example, the mobile network is still accessible and provides sensor information even if the robot's base has run out of power. Furthermore, several ROS compatible sensors like USB cameras or infrared cameras can be attached individually to the PI. In our setup the Rock Crawler has a wide angle Logitech webcam attached, which provides visual information over the network. To teleoperate the Crawler in real time we provide an image stream with a resolution of 160x120 pixels at 20 frames per second and transmit the stream over the network. The image stream generates a CPU load of about 50% at the RaspberryPI. Both robots can be adapted with further sensors, e.g., an action cam like the GoPro Hero, to retrieve high resolution wide angle pictures after a mission. Also the network interface can be changed. While the Fast Mover has still the Bullet2M Router attached, we are using an ALFA AWU036NH, well known in the Wardriving community, for the Rock Crawler. This allows more compact hardware setups and thus smaller and cheaper robots.

2.2.2 Software Design

On the base of ROS and the GPIO unit of the RaspberryPI, the PWM signals can be created straight in software. No additional hardware is needed to communicate with the RC models. To achieve this functionality, a module called PIBlaster is installed on the PI [14]. This module provides 8 PWM channels that can be directly attached to the corresponding GPIO pins. We have modified the PIBlaster module so that we obtain a stable 20 Hz PWM signal to control the servo motors with a processor load on the PI of less than 5%.

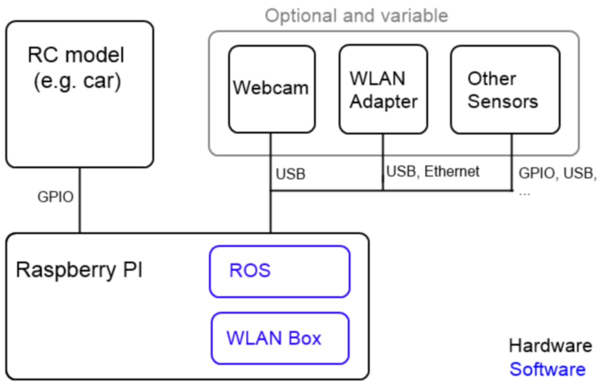

Figure 4: Communication between hard- and software components for individual RC robots

The core of the software is a ROS node called rc_bot. In this node, we have implemented a conversion of standardized ROS messages for velocity to the corresponding PWM duty cyles. The type of operation mode for the velocity PWM and the connected channels to the mobile base can be set in the node. The minimum and maximum duty cycles for the servo deflections can be adapted, because some servo motors interact more accurate than other ones. In our case the Rock Crawler steering servo has a deflection space between 0,9 ms and 2,1 ms. We have also implemented a watchdog functionality which immediately stops the robot, if no more command messages are received. The timeout for the watchdog can be dynamically set via a single parameter. This is an essential safety feature. Based on the teleoperation interface, that publishes new speed commands in a fixed interval, and the velocity of the robot, the watchdog should act properly to detect operation timeouts and to stop the robot as fast as possible, especially for the Fast Mover. Based on this core functionality the RC robot can be controlled by all ROS supported human machine interfaces, like graphical user interfaces, web browsers, tablets / smart phones or simple joystick devices. Optional sensor functionality for webcams or anything else can be included via their corresponding ROS nodes. We have written a launch file, which is also part of the rc_bot node, that starts the driver and also loads a connected webcam. The source code is avialable in our git hub at: https://github.com/roblab-wh-ge/roblab-whge-ros-pkg.

As mentioned above, the RC cars are only one example for using this core component. The modular framework, see **Figure 4**, provides the opportunity to attach the computational sensor to any RC model. For example the PI can be mounted on a RC boat for water operations. This can provide video information from a water flooded area, which is unaccessible for humans, or it can extend the network between two riverbanks by driving the boat to the center of the river.

There are several videos, which show the mobile robots in action. The videos can be found at the YouTube channel of the University of Applied Science Gelsenkirchen (http://www.youtube.com/user/RoblabFhGe)

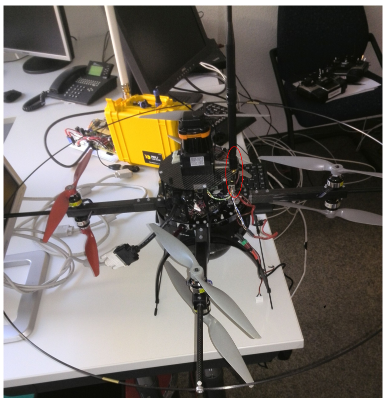

Figure 5: First test scenario: UAV with a bullet connected via WLAN to the netserver box.

3 Results: Network evaluation

3.1 Performance test of the netserver box

What is the performance of the network infrastructure based on the netserver box? To answer this question, we define typical mission scenarios. The first scenario is a mission with a drone where no network infrastructure is available. The drone is connected via WLAN with the netserver box and the box via ethernet with the command center (**Figure 5**). This reflects the situation at the earthquake area in Mirandola 2012 [2]. For the test we transfer ten times a 500MB file in both directions and measure the throughput in Mbit/s. For the second and third scenario, we connect an external network to the LAN2 socket, i.e., internet is available at the side. This scenario is needed, e.g., if an infield rescuer has a tablet and uploads pictures to a cloud in the internet or a robot send its sensor data to the cloud. Furthermore (third scenario), the operator control unit has access to the internet and can download / upload data. Since the PI has to route the packets, it generates some load. These scenarios are also a typical developers' scenario, when the code on all robots/computers has to be updated from an external repository. The next three scenarios simulate connections if additional services, e.g., a local cloud is running on the PI. All scenarios show that the PI is nearly completely loaded with this task. Our tests have also shown that the speed of the SD card (class 10 and better) is critical since all the data has to be stored on the card and the PI stops transmitting data over the network when it writes data to the card. **Table 1** show the results of the different tests for the scenarios.

Table 1: Performance evaluation. A 500MB file is transfered 10x via scp. The numbers are mean values. The last column shows the load of the RaspberryPI (standard deviation 5%). The maximal deviation for the data rate was +/- 5 MBit/s. The WLAN connections was optimal. In real situation the WLAN connection depends on the distance to the robot and will be lower.

Scenario	data rate (Mbit/s)	CPU load (%)
Scenario 1: robot -> operator control unit		
LAN1 to WLAN	55	0
WLAN to LAN1	54	0
Scenario 2: infield rescuer -> internet		
LAN2 to WLAN	35	60
WLAN to LAN2	39	43
Scenario 3: infield rescuer -> internet		
LAN2 to WLAN	39	55
WLAN to LAN2	42	74
Scenario 4: operator control unit -> box (local cloud)		
LAN1 to WLAN	23	90
WLAN to LAN1	23	98
Scenario 5: internet -> box (local cloud)		
LAN1 to WLAN	26	95
WLAN to LAN1	22	98
Scenario 6: robot -> box (local cloud)		
LAN1 to WLAN	23	90
WLAN to LAN1	24	98

3.2 Network topology

Another challenging question for USAR environments is to handle scenarios where communication is either lost or degraded during a mission. On one hand the robot should move autonomously, on the other hand we need communication with the robot to get the data from the field. So, one task is to test new network architectures, e.g., mesh networks instead of infrastructure networks with repeaters, which are currently used to enhance communication capabilities.

For the first test, we extend scenario 1 from above and install also the mesh firmware OpenWrt with olsr at the bullet routers (at 2.4 GHz). Furthermore, we connect the command computer also via WLAN to the robot. **Figure 6**:top shows the basic test setting at the Fraunhofer Campus at Sankt Augustin. AP (blue) is the netserver box in the field. C1 is the computer of the robot in the field and C2 the computer of the operator. The distance between the robot and the netserver box is about 35 m and from the box to the command computer about 50 m. The scenario is equivalent to the set up at the NIFTi project review in April 2012. It has to be noticed that the computers C1 and C2 have no direct connection to each other. The blue bars show the performance of the up-to-date version of the original router firmware (airOS) and the red bars the patched router firmware with OpenWrt [13, 15]. For this experiment, airOS shows the better performance (**Figure 6**:middle). For the case that computer C1 and computer C2 have a direct connection, OpenWrt shows a little better performance than airOS because the communication over the netserver box (AP) reduces the transfer rate (**Figure 6**:bottom). This leads to the result: if possible put the AP in the center of the scenario.

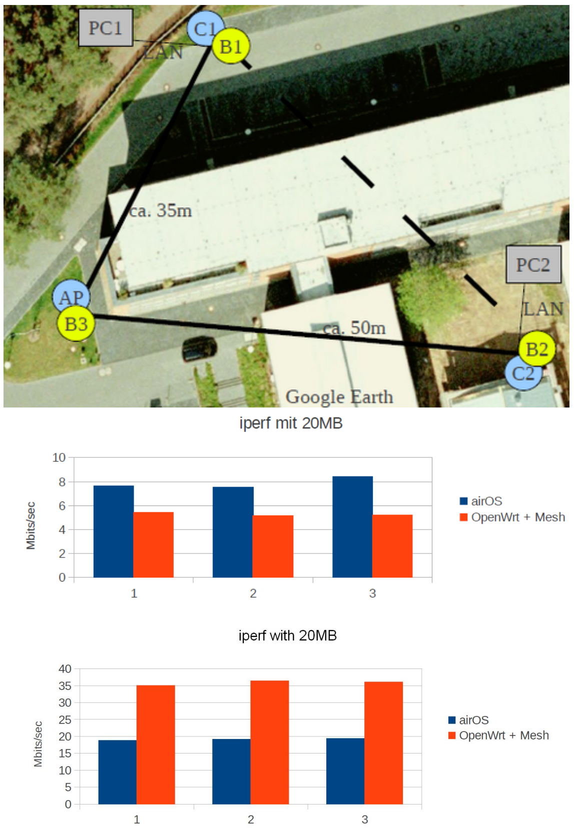

Figure 6: Top, scenario 1: AP (blue) is the netserver box in the field. C1 is the computer of the robot and C2 the computer of the operator. Both are connected via WLAN. Middle: C1 and C2 do not have a direct connection. For the test we have transferred one hundred times a 20MB file at three different days to test the influence of other networks in the environment. The blue bars show the performance of the up-to-date version of the original router firmware (airOS) and the red bars the patched router firmware with OpenWRT. The standard deviation is 5%. Bottom: Computer C1 and C2 have a direct connection.

The optimal scenario for mesh networks is defined in the last test given in **Figure 7**. The computer C1 and C2 indicate two robots in the field, e.g., C2 the drone and C1 the Fast Mover or Rock Crawler. One of the simple robots C1 has a connection to the netserver box but C2 is to far away from it. As a result, C2 is lost in the infrastructure mode and has no connection to the netserver box AP and can not transmit data to the operator. The mesh network (or also a repeater) architecture has nearly the same performance as in the scenario above but the bandwidth has to be shared between both robots. As a result, mesh networks with their reduced throughput are more suitable here.

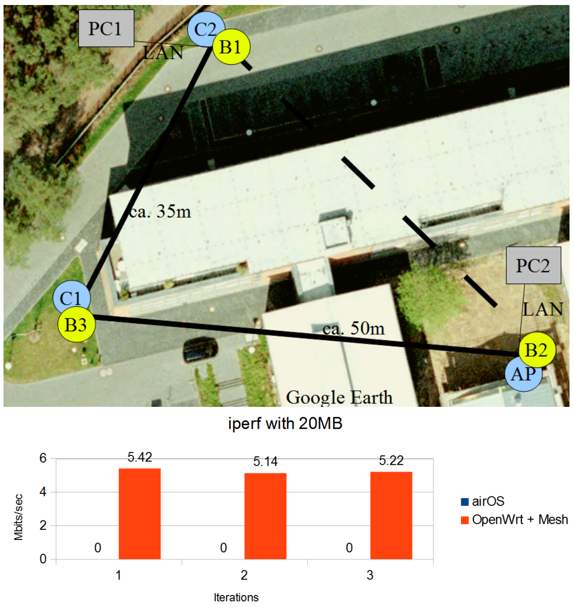

Figure 7: Top, scenario 2: Two robots are in the field (indicated as C1 and C2). C2 has no direct connection to the netserver box AP but C1 has a connection. Bottom: The performance of the mesh network is similar to the first scenario but C2 is lost and has no connection. Also 20MB files are transferred hundred times at 3 different days with a similar standard deviation of 6%

4 Conclusion

What kind of network architecture is suitable for rescue missions? This paper shows one approach to answer this

question. We presented a waterproof netserver box to provide LAN and WLAN capabilities in field, consisting of a bullet M(2/5) and a RasberryPI as a static access point with several options and connectors. Furthermore, we extended the network nodes and made them mobile by adding RC cars. The servo motors of the RC cars are controlled by the RasberryPI over the GPIOs. The bullet routers convert the cars to mobile access points, which can serve as bridges or mesh nodes to extend the network infrastructure. We defined typical in-field scenarios and evaluated the performance of the network architecture.
Future work will concentrate on the integration of new hardware, e.g., more powerful computer boards than the PI (RIoTboards) and to the improvement of the capabilities of the RC cars, e.g., adding autonmous features and on-board computer vision. The use of drones for example, the AR Drone from Parrot would also be an interesting option to adjust the network infrastructure to the realities in the field. Furthermore, we would like to add a low bandwith safety and maintenance channel between the netserver box and the robots, e.g., by using citizens' band radio or other frequencies.

Acknowledge
This work has been partly funded by the European Commission under contract number FP7-247870-NIFTI and FP7-609763-TRADR. Thanks to all colleagues from Fraunhofer and the other partners in the consortiums for their support.

References

[1] NIFTi project website: *www.nifti.eu*, 2014.

[2] Kruijff, G.-J.; Janicek, M.; Keshavda, S.; Larochelle, B.; Zender, H.; Smets, N.J.J.M; Mioch, T.; Neerincx, M.; van Diggelen, J.; Colas, F.; Liu, M.; Pomerleau, F.; Siegwart, R.; Hlavac, V.; Svoboda, T.; Petricek, T.; Reinstein, M.; Zimmermann, K.; Pirri, F.; Gianni, M.; Papadakis, P.; Sinha, A.; Balmer, P.; Tomatis, N.; Worst, R.; Linder, T.; Surmann, H.; Tretyakov, V.; Corrao, S.; Pratzler-Wanczura, S.; Sulk, M: *Experience in System Design for Human-Robot Teaming in Urban Search and Rescue*. In: Proceedings of 8th International Conference on Field and Service Robotics. July 16-19, Japan, STAR, Springer Verlag, 2012.

[3] TRADR project website: *http://www.tradr-project.eu/*, 2014.

[4] Dong, Jiawei; Kim, Won-jong: *Experimental analysis and implementation of a multiscale wireless/wired networked control system*, International Journal of Control, Automation and Systems, pp 102-110, 2014.

[5] Jennifer Yick, Jennifer; Mukherjee, Biswanath; Ghosal, Dipak: *Wireless sensor network survey*. In Computer Networks, vol. 52, no. 12, pp. 2292-2330, 2008.

[6] ROS documentation website: *http://wiki.ros.org/*, 2014.

[7] Raspberry Pi documentation website: *http://www.raspberrypi.org/*, 2014.

[8] UBiQUiTi bullet routers / documentation: *http://www.ubnt.com/airrouter*, 2014.

[9] ALFA network: WLAN adapter/documentation: *http://www.alfa.com.tw/products.php*, 2014.

[10] Monstertruck website: *http://www.conrad.de/ce/de/product/237300/Reely-110-Elektro-Monstertruck-Titan-Brushless-4WD-EM-04MT-RtR-40-MHz-FM*, 2014.

[11] Rock-Crawler website: *http://www.conrad.de/ce/de/product/235973/Reely-18-Elektro-Rock-Crawler-4WD-CR-300-RtR-40-MHz-FM*, 2014.

[12] Jon Bennett: The Wifi Robot project. *http://www.jbprojects.net/projects/wifirobot/*, 2008.

[13] OpenWrt website: *https://openwrt.org/*, 2014.

[14] Pi-blaster github repository *https://github.com/sarfata/pi-blaster/*, 2014.

[15] olsrd: an adhoc wireless mesh routing daemon: *https://olsr.org/*, 2014.

[16] Quigley, Morgan; Gerkey, Brian; Conley, Ken; Faust, Josh; Foote, Tully: *ROS: an open-source Robot Operating System*. In: ICRA workshop on open source software, 2009.

[17] DARPA Robotics Challenge website: *www.darpa. mil/Our_Work/TTO/Programs/DARPA _Robotics_Challenge.aspx*, 2014.

[18] Murphy, Robin; Tadokoro, Satoshi; Nardi, Daniele; Joacoff, Adam; Fiorini, Paolo; Choset, Howie; Erkmen, Aydan: *Chap. 50.6 Search and Rescue Robotics, Fundamental Problems and Open Issues* in Handbook of Robotics, eds. Siciliano, Bruno; Khatib, Oussama, Springer, 2008.

[19] Ogren, Petter; Fiorelli, Edward; Leonard, Naomi Ehrich: *Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment*. Automatic Control, IEEE Transactions on, 2004, 49. Jg., Nr. 8, S. 1292-1302.

[20] Linder, Thorsten; Tretyakov, Viatcheslav; Blumenthal, Sebastian; Molitor, Peter; Holz, Dirk; Murphy, Robin; Tadokoro, Satoshi. and Surmann, Hartmut: *Safety Rescue robots at the Collapse of the municipal archive of Cologne City: A field report*, 8. IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR2010), 2010.