A Universal Communication Framework and Navigation Control Software for Mobile Prototyping Platforms

doi:10.3991/ijoe.v6s1.1397

Karsten Henke, Steffen Ostendorff, Thomas Volkert, Andreas Mitschele-Thiel
Ilmenau University of Technology, Ilmenau, Germany

***Abstract*—In our contribution we would like to describe two new aspects of our low-cost mobile prototyping platform concept: a new hardware communication framework as well as new software features for navigation and control of our mobile platform. The paper is an extension of the ideas proposed in REV2009 [1] and is based on the therein used hardware platform and the monitoring and management software. This platform is based on the Quadrocopter concept – autonomous flying helicopter-style robots – and includes additional off-the-shelf parts. This leads to a universal mobile prototyping platform for communication tasks providing both mobile phone and WiFi access. However, the platform can provide these functions far more quickly than a technician on the ground might be able to. We will show that with our concept we can easily adapt the platform to the individual needs of the user, which leads to a very flexible and semi-autonomous system.**

***Index Terms*—Remote Engineering, Remote handling, Remote sensing, Mobile robots, Mobile communication, Rapid prototyping, Navigation.**

I. INTRODUCTION

The Integrated Communication Systems Group, led by Prof. A. Mitschele-Thiel, at the Ilmenau University of Technology is using flying "Quadrocopter" robots[1] to develop self-assembling ad-hoc wireless networks. They can be used in case of disasters, when every wired communication infrastructure is destroyed. For that, the robots create a network of mobile nodes that provide both mobile-phone and standard wireless network access which allow people in distress to call for help. This is all achieved by using off-the-shelf parts.

The system is based on the Quadrocopter concept of autonomous flying helicopter-style models, which are often described as "unmanned aerial vehicles" (UAV) or as "microcopter". This concept was enhanced by different off-the-shelf parts, including VIA's Pico-ITX hardware, an FPGA and a GPS unit, which finally leads to universal mobile communication platforms providing both mobile phone and WiFi access. The platforms can provide these functions far more quickly than non air-borne vehicles might be able to. Furthermore, they can spread themselves around an area and establish a radio network without human intervention.

[1] A Quadrocopter is a four rotor, radio controlled or autonomous operating, electronically stabilized flying platform (for details see next section).

Despite all promising advantages of Quadrocopters there are limitations though. The biggest issue is the power supply because the batteries deliver power for only about 20 minutes. However, these 20 minutes are for flight time, giving the robots enough duration to position themselves and land on high ground or a building somewhere, and meanwhile provide network coverage for several hours [2].

In the Integrated Communications Systems Group at our university this work is used

- to motivate bachelor and master students to study Computer Engineering by increasing their interest in technical issues,
- as integral part of a new Ad-hoc lab to demonstrate different aspects in the area of Mobile Communication as well
- as a basis for universal rapid prototyping nodes to investigate different mechanisms for self-organized mobile communication systems within the International Graduate School on Mobile Communications.

To enhance the functionality achieved by our system, that was presented last year (described in [1]), new software and hardware modules were developed and integrated. One major part is the so called ***Universal Communication Framework*** (UCF). For varying tasks several different configurations with different sensors and actors are desired [12]. With this requirement for flexible configuration a modular and extensible communication architecture is needed because it is not an option to equip the platform with all possible sensors and actors that might be required. That's why an architecture was designed that allows the Quadrocopter's user to apply only the needed external interfaces and equipment modules.

These new hardware aspects were completed by ***new software navigation functions*** which enable the Quadrocopter to work as an autonomous flying prototyping platform. For this purpose, new modules were integrated into the Java based graphical user interface (GUI), which runs on a Laptop on the ground for providing additional monitoring and control elements.

After a short introduction to the basic Quadrocopter concept we will describe these two new aspects of our low-cost mobile communication platform concept in detail: the new hardware communication framework and the new software module for autonomous navigation.

II. THE QUADROCOPTER CONCEPT AS BASIC IDEA

Challenging operational scenarios, partially focusing on cooperative coordination and control of multiple mobile platforms for different missions, have to be simulated and examined using several test beds. These test beds must reflect the complexity expected in these application scenarios. For mobile communication in disaster scenarios we choose a test bed consisting of many (semi-) autonomous (possibly heterogeneous) mobile platforms. The main design philosophy is to use simple platforms with commercial off-the-shelf add-ons, like Quadrocopters in our case. Therefore, many of them can be easily operated at the same time. This provides a good combination of flexibility, agility and mobility and further allows the usage of these test beds in a wide range of applications, as described in [1, 3, 4].

In the following we will briefly describe the architecture of these multipurpose mobile prototyping platforms.

The basis for the mobile prototyping platform is the Quadrocopter concept. It uses a four-rotor aerial platform with two different rotor-rotations (see Fig. 1).

But – how can a Quadrocopter fly at all? The front and back rotor turn clockwise, while the left and right rotor spins counter-clockwise. To hover, all rotors rotate at the same speed. When doing so, the rotation forces between the clockwise rotors on one hand and the counter-clockwise rotors on the other hand, are balanced out. This enables the Quadrocopter to hang “steady” in the air.

In order to fly in one direction, the Quadrocopter will be brought out of balance. The speed of the rotor that opposes the desired direction is increased. This makes the Quadrocopter tip over in a certain direction. For example, to fly forward, the back rotor has to turn faster. This is called “pitch” or “nick”, respectively. The later name is the preferred name for the forward/backward-movement. Left and right movements are usually called “rolls”.

Turning around its vertical axe is called “yaw”. To be able to yaw you need a force to turn the platform around. This is produced by changing the speeds between the forward/backward rotors and the left/right rotors. For example, to yaw clockwise, the forward/backward rotors will turn faster and the left/right rotors will slow down a little bit. This makes the Quadrocopter turning clockwise, while the same height is maintained [5].

Besides these simple movements, controlled by the so called flight control, which is a single microcontroller with some sensors, more control is needed to fulfill all the required application scenarios, as described above. This simple platform had to be enhanced by different hardware and software components which will be described in the following.

III. UNIVERSAL COMMUNICATION FRAMEWORK

The Universal Communication Framework is the central hardware part of the newest version of our Quadrocopter. This framework was developed to fulfill the requirements many applications set for their used platform [10, 11].

Different applications require a different number of interfaces and sensors. To equip the Quadrocopter with the worst case population of modules (all modules ever needed) and a fixed routing is one solution, but causes high weight and a high power consumption of the system. Because power is very limited on mobile devices, especially flying devices, it is a crucial requirement to use the limited power available as efficient as possible.

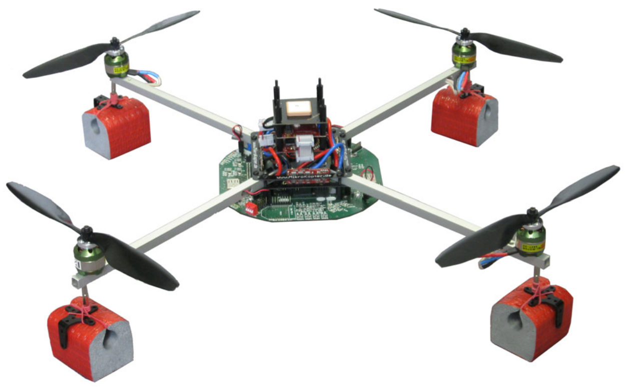


Figure 1. The prototype for the mobile prototyping platform

This primary limitation of the power supply leads to a number of secondary demands for the platform. When power is limited the operation range can be extended by eliminating all unnecessary elements of the platform to save power and weight. This means for every deployment of the Quadrocopter the hardware needs to be adapted to the individual needs with the least or no overhead if possible. This includes removing all unused hardware modules like sensors, controllers or communication interfaces.

Because different configurations need different Quadrocopter setups (available interfaces and communication path between all components) a new platform is needed for every application. This system adaption needs time and is expensive and therefore not practical in many cases.

Our approach of a Universal Communication Framework makes this task a lot easier by providing a hardware independent communication between components. Therefore, only one central platform is needed that can be equipped with different sensors or controllers and allows a data exchange between them independent from the implemented interfaces and protocols. The possibility to use an arbitrary number of different interfaces and protocols makes this concept very flexible.

To fulfill these requirements the Universal Communication Framework consists of basically three modules:

a) bus arbiter,
b) bus device and
c) interface proxy,

which create a central bus structure with a flexible connection of interfaces (see Fig. 2). As hardware we use an FPGA, which allows us to generate the central framework only for the needed interfaces and therefore adapt each system to its individual requirements. The biggest advantage of the FPGA is the ability to implement any interface and to include processing functionality, while e.g. microcontrollers are fixed to a specific interface on certain pins.

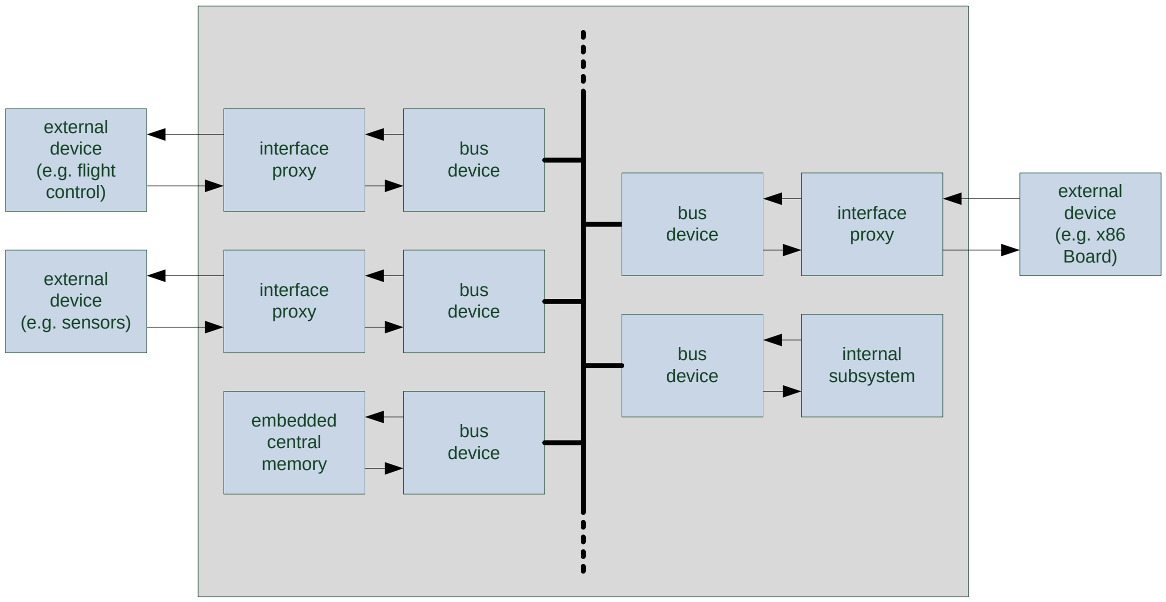


Figure 2. System overview with interface proxies as well as external and internal devices

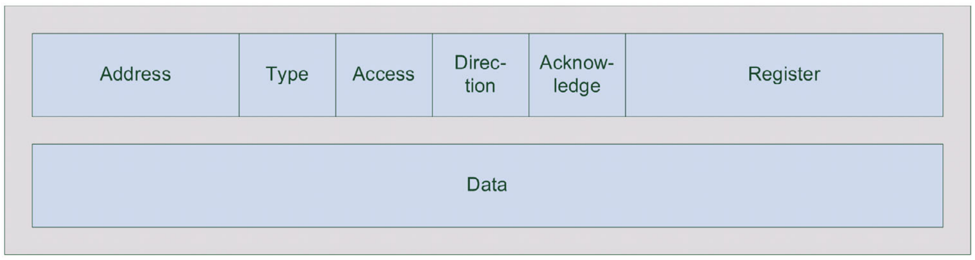


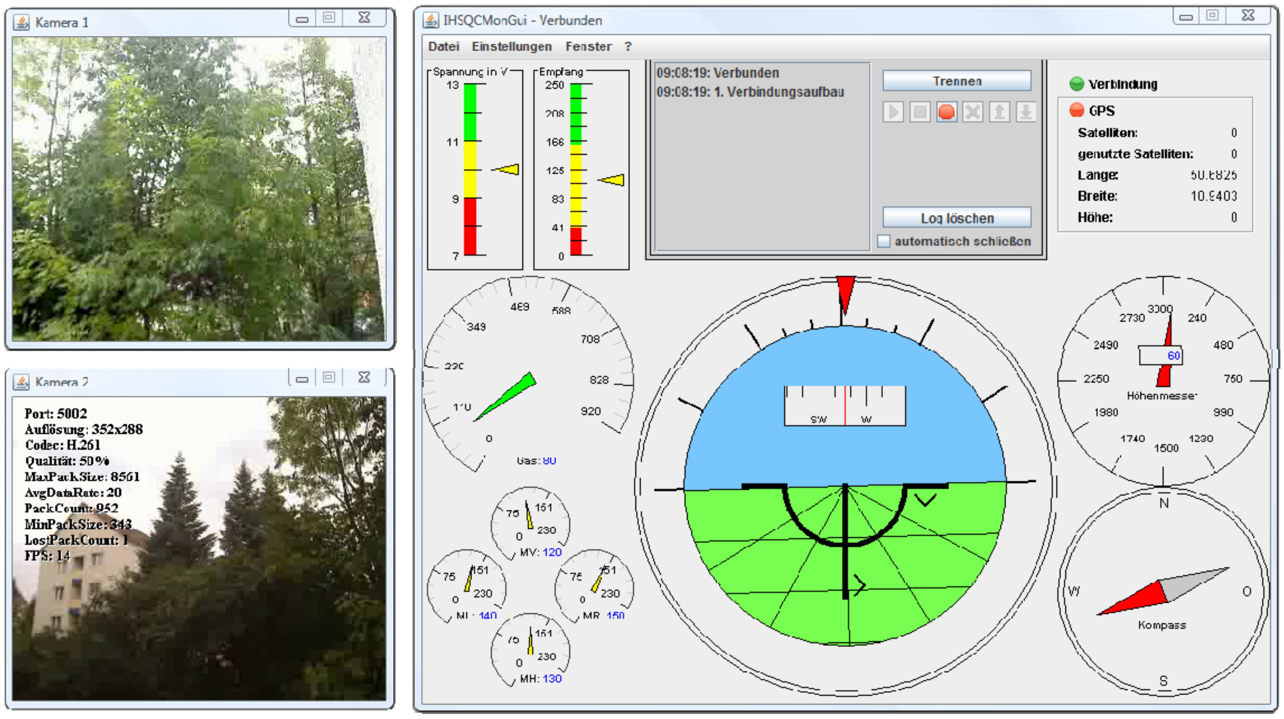
Figure 3. Format for packet based transmission

A. Hardware Abstraction Layer (HAL)

Besides the new hardware concept the Universal Communication Framework consists of a second part: the Hardware Abstraction Layer (HAL). This layer is a collection of C code that allows the software programmer to send and receive data packets by using a common interface which is independent from the currently used physical equipment. The design of the HAL enables a synchronous sending and an asynchronous receiving of data packets. Every packet is transmitted via a defined protocol which is defined inside the FPGA framework. Hence, the HAL library can be ported at minimal cost to a different architecture (e.g. microcontroller) and allows a fast development of communication participants without the need to consider the platform type (different microcontroller etc.), When using this HAL, no knowledge about the internals of the framework is needed. In summary, with the help of the HAL a programmer is able to access interface proxies and read/write to them as well as to request data from inside the framework or other devices on an abstract level. Therefore, if functionality needs to be ported to a different controller with a different interface (for example i2c instead of RS232) only the hardware interface routines need to be exchanged. The only necessary assumption is the demand for an existing Ansi-C compiler for the target hardware.

B. Details of packet based transmission

Every transmission packet consists of seven parts (see Fig. 3). The first field, the ***address***, identifies every receiving/transmitting member of the framework. Only one address field is needed, because the source address is known at the transmitter and we only have one participant on every "entry point" (interface proxy) to the system. Therefore, only data of one participant can be received when the packet is transmitted to the destination. The receiver address is exchanged with the transmitter address by the internal framework. This can be done, because the destination knows its own address and is only interested in the source address (where the packet comes from and the where answer should be addressed to). The ***type*** field indicates if the packet is for interface configuration or if it should be forwarded to the internal bus. Interface configuration packets are processed internally in the proxy. The ***access*** field determines if the message is a unicast or multicast packet. The ***direction*** field selects between read and write, while the ***acknowledge*** field determines if any feedback is needed for the current transmission/request. The ***register*** field selects the register address inside the destination module or the proxy itself, if it is a configuration packet. The ***data*** field consists of the actual payload of the transmission.



Port: 5002
Auflösung: 352x288
Codec: H.261
Qualität: 50%
MaxPackSize: 8561
AvgDataRate: 20
PackCount: 952
MinPackSize: 343
LostPackCount: 1
FPS: 14

09:08:19: Verbunden
09:08:19: 1. Verbindungsaufbau

Satelliten:	0
genutzte Satelliten:	0
Länge:	50.6825
Breite:	10.9403
Höhe:	0

Figure 4. Software for monitoring and navigation

IV. AUTONOMOUS NAVIGATION

In [1], we have presented our software (see Fig. 4) for monitoring our flying communication platforms. With help of this software we are able to observe the following data from the platform:

- height over ground,
- current position in the air,
- global position,
- battery voltage,
- RC receive quality,
- the current gas values for all 4 motors and
- the gas value input from the RC.

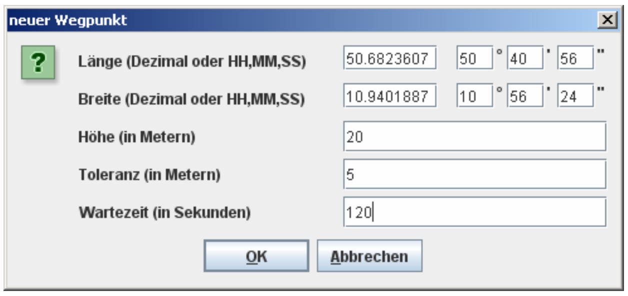
Additionally, we integrated a streaming solution which enables a live transmission of a live camera picture with a resolution of 352*288 pixels, e.g. using H261 or H264 codec. All video processing is based on the ffmpeg [13] software library. Moreover, we added a map view, which displays the position of the flying platform based on map data from GoogleMaps or OpenStreetMap.

In this paper we present our software extensions for autonomous navigation control [14]. In general, the control software is divided into three parts:

- a microcontroller image:
- written in C, running on an Atmega µController, acting as real-time controller for navigation,
- a data fetching server:
- written in C++, running on the flying platform, placed on a x86 compatible board, acting as a bridge between the other two software parts,
- the graphical user interface (GUI) client:
- written in Java and running on the ground station, acting as input/output interface towards the user.

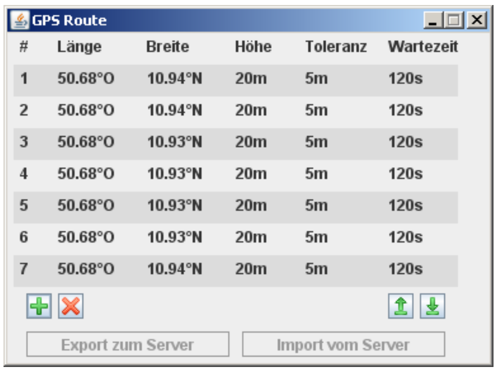
We had to adapt all parts of the existing software to enable autonomous navigation. In the current implementation the microcontroller software is used for storing the set path and automatically routing the flying platform towards the next way point. The used network protocol towards the ground station had to be extended by additional protocol data units. On the opposite, within the GUI we integrated a new configuration dialog for adding new way points (see Fig. 5).

For every way point it is possible to define the GPS coordinates, the height over ground, waiting time and the tolerance (deviation between the real and intended position). When the platform flies along the defined route it moves from way point to way point. However, it doesn't need to reach every point exactly. Smaller drifts from the set values are acceptable. The tolerance range can be defined per way point. Moreover, it is possible to define a time value for every point, which defines how long the platform has to remain at this location.



Länge (Dezimal oder HH,MM,SS)	50.6823607	50 °	40 '	56 "
Breite (Dezimal oder HH,MM,SS)	10.9401887	10 °	56 '	24 "
Höhe (in Metern)	20			
Toleranz (in Metern)	5			
Wartezeit (in Sekunden)	120			

Figure 5. Dialog for inserting new way point



GPS Route

#	Länge	Breite	Höhe	Toleranz	Wartezeit
1	50.68°O	10.94°N	20m	5m	120s
2	50.68°O	10.94°N	20m	5m	120s
3	50.68°O	10.93°N	20m	5m	120s
4	50.68°O	10.93°N	20m	5m	120s
5	50.68°O	10.93°N	20m	5m	120s
6	50.68°O	10.93°N	20m	5m	120s
7	50.68°O	10.94°N	20m	5m	120s

Figure 6. Import and export of navigation data

Fig. 6 shows a graphical view of the defined way points. It is possible to upload a route to the flying platform via the mentioned software parts using the onboard wireless LAN. Besides this, the user at the ground station is also able to download the currently defined navigation route from the platform, modify it and upload it again to the platform.

After the configuration of the navigation route is completed, the map view depicts the way points on the usual map. Fig. 7 shows an example configuration where the route consists of 7 points. They are displayed in the big map view as well as in the mini map view on the bottom.

V. CONCLUSION

In our contribution we described a universal and innovative communication hardware architecture as well as new software features for navigation and control of our mobile communication platform.

- The two most important components of the new concept are:
- the Universal Communication Framework and
- the software modules for autonomous navigation.

The presented Universal Communication Framework is able to serve as a central exchange point for different interfaces and protocols inside the Quadrocopter. Due to the reconfigurability of the used FPGA, the system can be adapted according to the individual needs for every situation depending on the required features and available resources. Additionally it is possible to use an embedded memory component to acquire data centrally and supply it to different destinations e.g. at various rates or with diverse selections/grouping of data or preprocessing. This leads to very flexible and semi-autonomous systems.

By using a library with available hardware descriptions of interfaces as well as functionality provided by the Hardware Abstraction Layer. Therefore it is possible to set up individual Quadrocopter systems without much effort and without high costs. This creates systems with very little overhead.

Now the new version of our mobile platform is able to fly in an autonomous way along a defined route. This route can be configured by the user with help of the new GUI extensions. Additionally, it is also able to let the platform stay at intermediate way points for a predefined time automatically. Moreover, a tolerance range can be defined for each way point. This means, a way point is accepted when the platform reaches the defined area.

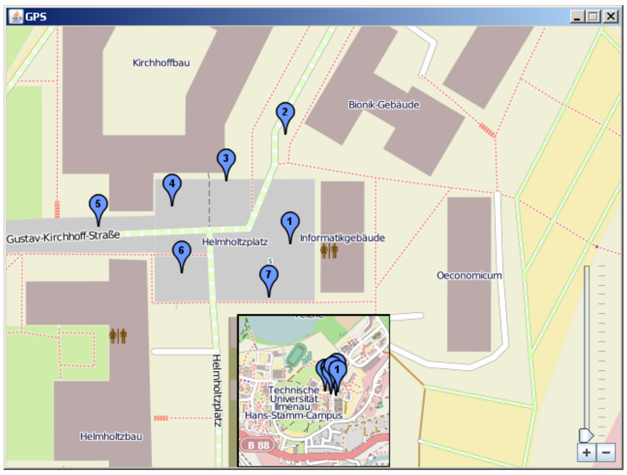


Figure 7. Resulting map view with way points [16]

With these multipurpose mobile prototyping platforms we provide the basics for a flexible application of mobile airborne communication platforms [6 - 9].

Furthermore, these mobile platforms are used within the International Graduate School on Mobile Communications at the Ilmenau University of Technology [15]. Since the communication platform is very robust and configurable, it is possible to execute multiple missions in a short period of time with minimal setup and organization between the experiments.

Besides using the platform for research purposes, it is ideal for student education and motivation. Because of the attractive subject and the large number of different and interesting projects, students can work on many challenging topics during their studies at our department.

ACKNOWLEDGMENT

The authors would like to thank Tobias Simon and Alexander Krause as well as many students which have supported this research, especially Thomas Hertwig, Sven Biegler, Stefan-Wieland Koegel and Kay Wenzel.

REFERENCES

[1] K. Henke, St. Ostendorff, Th. Volkert, A. Mitschele-Thiel, “Mobile Prototyping Platforms for Remote Engineering Applications”, Remote Engineering & Virtual Instrumentation - REV2009, 22 - 25 June 2009, Bridgeport, CT, USA.

[2] D. Fearon, “Flying robots to provide Wi-Fi in disaster zones”, *PC Pro*, http://www.pcpro.co.uk, 3 March 2009.

[3] J.P. How, “UAV SWARM Health Management Project”, MIT Aerospace Controls Laboratory, http://vertol.mit.edu, 2009.

[4] J.P. How, “Multi-Vehicle Flight Experiments: Recent Results and Future Directions”, AVT-146 Symposium on Platform Innovations and System Integration for Unmanned Air, Land and Sea Vehicles, Florence, Italy, May 2007.

[5] Mikrokopter Wiki, http://www.mikrokopter.com, 2010.

[6] S. Pax, “Configuration and monitoring of a mobile communication platform”, student research paper, Ilmenau University of Technology, 2008.

[7] S. Biegler, “Remote monitoring for a mobile communication platform”, student research paper, Ilmenau University of Technology, 2008.

[8] S. Biegler, “Graphical position monitoring for a mobile communication platform”, student research paper, Ilmenau University of Technology, 2009

[9] K. Wenzel, “Possibilities and limitations of GPS auto-navigation”, student research paper, Ilmenau University of Technology, 2008.

[10] Th. Hertwig, “Hardware-independent communication framework”, student research paper, Ilmenau University of Technology, 2009.

[11] Th. Hertwig, “Concept and implementation of a central control unit for mobile aircrafts based on FPGA”, Diploma Thesis, Ilmenau University of Technology, 2010.

[12] F. Herzog, D. Kling, “Sensor system extension of the mobile communication platform”, student research paper, Ilmenau University of Technology, 2008.

[13] FFMPEG project, http://www.ffmpeg.org, 2010.

[14] S.-W. Koegel, “Auto navigation of a mobile communication platform”, Technical report, Ilmenau University of Technology, 2010.

[15] Mobicom – International Graduate School on MOBILE COMMUNICATIONS, Ilmenau University of Technology, http://www.gs-mobicom.de, 2010.

[16] OpenStreetMap, http://www.openstreetmap.de, 2010.

AUTHORS

K. Henke is with the Ilmenau University of Technology, Faculty of Computer Science and Automation, Integrated Communication Systems Group, 98684 Ilmenau, Germany, POB 10 05 65 (e-mail: karsten.henke@tu-ilmenau.de).

St. Ostendorff is with the Ilmenau University of Technology, Faculty of Computer Science and Automation, Integrated Communication Systems Group, 98684 Ilmenau, Germany, POB 10 05 65 (e-mail: steffen.ostendorff@tu-ilmenau.de).

Th. Volkert is with the Ilmenau University of Technology, Faculty of Computer Science and Automation, Integrated Communication Systems Group, 98684 Ilmenau, Germany, POB 10 05 65 (e-mail: thomas.volkert@tu-ilmenau.de).

A. Mitschele-Thiel is with the Ilmenau University of Technology, Faculty of Computer Science and Automation, Head of the Integrated Communication Systems Group, 98684 Ilmenau, Germany, POB 10 05 65 (e-mail: andreas.mitschele-thiel@tu-ilmenau.de).

This work is partly supported by the German Research Council in the context of the Research Training Group program (Graduiertenkolleg) under contract GRK 1487 “Self-organized Mobile Communication Systems for Disaster Scenarios”.

This article was modified from a presentation at the REV2010 Conference at KTH, Stockholm, Sweden in June 2010. Submitted July 15th, 2010. Published as resubmitted by the authors July 29th, 2010.