UAVRC, A GENERIC MAV FLIGHT ASSISTANCE SOFTWARE

M. Israel[a], M. Mende[a], S. Keim[b]

[a] Remote Sensing Technology Institute, German Aerospace Center, 82234 Oberpfaffenhofen, Germany - (martin.israel, manuel.mende)@dlr.de
[b] German Remote Sensing Data Center, German Aerospace Center, 82234 Oberpfaffenhofen, Germany - stefan.keim@dlr.de

KEY WORDS: UAV, MAV, Multicopter, Mission Planning, Flight Assistance Software, Ground Control Station, Remote Control, RPAS

ABSTRACT:

In this paper we present our multicopter flight assistance software uavRC, which bears on a distributed system with interchangeable MAV-drivers and a browser-based user interface that can be used on any computer, tablet or smartphone. The software components can be distributed on different computers and are even executable on a Raspberry Pi. The components communicate over a well-defined interface. One module is the browser based user interface, another module is the MAV driver. There are additional modules like a task-scheduler, a path-planer and many more. Currently the software is in beta stage, so there is still a lot of work in progress. With this paper we focus on the software architecture.

1. INTRODUCTION

In the last few years unmanned aerial vehicles (UAV) and especially rotorcraft driven micro aerial vehicles (MAV) gained in importance. There are a lot of different MAVs for prices from 300 to 30000 €. And there are lots of different applications, where such systems could be used. Beside the security/surveillance sector they can help in finding hot spots in photovoltaic modules or cracks in rotor blades of wind engines (Hallermann and Morgenthal, 2013). They are used to create 3D models for geodetic purposes (Neitzel and Klonowski, 2011) or they can help to rescue roe deer fawns from being killed by mowing machines (Israel, 2011). For many tasks a camera on a MAV is observing a defined area systematically. Beside the sensors for the observation these MAVs are usually equipped with sensors for a GPS based waypoint navigation. In contrast to a manual flight, an automated waypoint navigation based flight often leads to a faster and more reliable coverage of the observation scene. Usually the MAV manufacturer has its own waypoint navigation software, but it is often not optimized for the observation task and is only suitable for a special MAV. Moreover it is often restricted to one operating system. The ones with a Web Mapping Service (WMS) based background map often require an internet connection and the others with a fixed background image are not as comfortable. We want to fill these gaps with our flight assistance software uavRC. We call it flight assistance software instead of ground control station, because we don't want to create the impression, that one can fly without the manual remote control (for emergency cases), when using this software.

2. SOFTWARE ARCHITECTURE

The software system is divided into submodules that are only able to interact via networking. For the communication between the modules the Publish-Subscribe-Pattern (Tarkoma, 2012, Eugster et al., 2003) and Remote Procedure Calls (Birrell and Nelson, 1984) are used. This leads to decoupled modules that can be modified and managed independently from the other system components. As a consequence, these subcomponents of the software can run on different computers, on different operating systems and they can moreover be written in different programming languages. The glue between the modules is a well-defined communication protocol, that uses the JavaScript Object Notation (JSON). Instead of the comparable XML format, JSON is more compact and thus needs less bandwith for transmitting (Bray, 2014).

Especially the intensive use of the Publish-Subscribe-Pattern enables a step-by-step extension of the software. Security aspects for example could be supervised by a special module (a later extension), that listens on the uav-position-topics and interacts when critical behaviour of an uav occurs (collision prevention, ...). With this architecture approach in mind we can focus now on the basic functionality, while not running in a dead end. This software architecture as a distributed system forces us to separate different logical components into different modules.

To enable third party developers to extend the systems functionality, we build the system on standardized protocols like websockets (Fette and Melnikov, 2011) and Web Application Messaging Protocol (WAMP). WAMP is a registered subprotocol for communicating via websockets (Mende, 2014). It especially supports the previously mentioned communication-pattern Publish/-Subscribe and Remote-Procedure-Calls. To support these protocols in a convenient way, the system uses the Autobahn framework(Tavendo, 2014a). Autobahn combines WAMP and websockets and establishes a easy to use programming interface.

The Publish/Subscribe Communication and the Remote-Procedure-Calls are registered and routed by a central router. For this purpose crossbar (Tavendo, 2014b) was used, which also supports websockets and WAMP. Figure 1 shows the systems architecture. Many modules can communicate via the autobahn framework, in a well defined but separated manner.

So it is quite simple to use a decoupled Publish/Subscribe pattern. First of all a connection to the router by using websockets is established. In a second step a topic by defining a URI like *my.sample.topic* is registered on the router while connecting a callback to it. The callback is the function that is being executed when the topic is received. After that, any publication can be handled that is issued by a component on the topic *my.sample.topic*. Same applies for remote-procedure-calls. Once again a URI like *my.sample.rpc* gets registered on the router with a specified callback. For remote-procedures a callback can also throw exceptions and/or have a return value. A rpc-result gets transmitted to the calling instance.

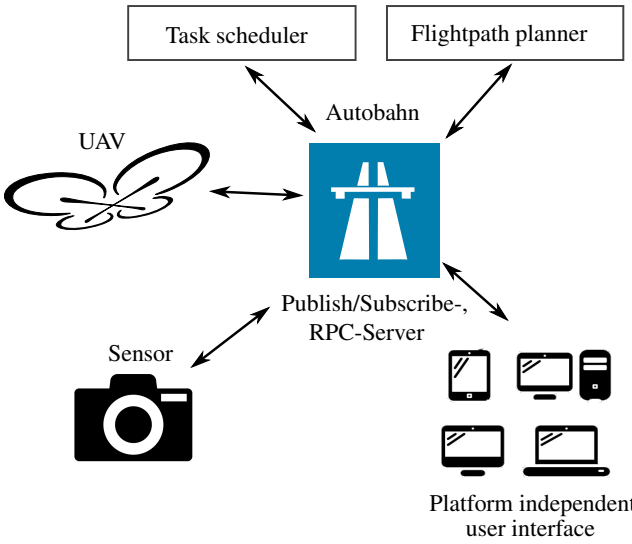

Figure 1: Software Architecture

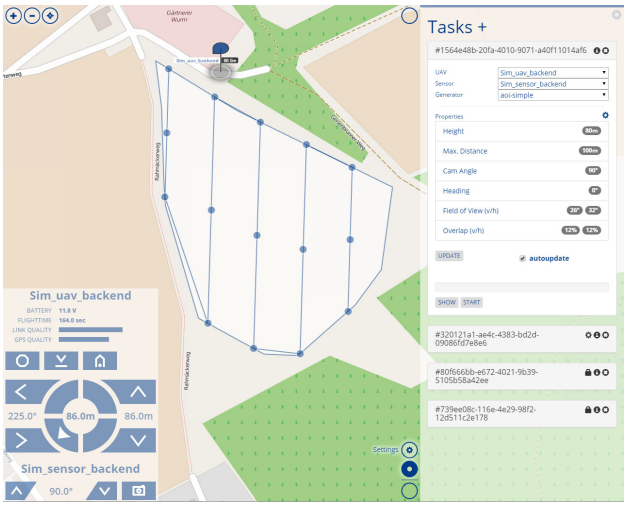

Figure 2: The browser based user interface

The Robot Operating System (ROS) provides the Publish/Subscribe Pattern and Remote-Procedure-Calls (Quigley et al., 2009) just as Autobahn and Crossbar. It is widespreadly used even for multicopters and their payload sensors. We planned to extend the uavRC with one ROS driver for UAV and one ROS driver for Sensors.

In aerial navigation it is important to react on events with only a small delay. Therefore the overhead created by well-defined interfaces and using established protocols might reduce the suitability of the software system for flight assistance purposes. Nevertheless the design seems to be suitable to support the task it was designed for (Mende, 2014) and is furthermore extendable to more complex tasks with even more overhead.

3. MODULES

3.1 User Interface

The user interface is based on web technologies (HTML5, CSS3 and JavaScript) and utilizes a map visualization library with broad browser support, that allows a cross-device usage. The web-application gives the user direct control over multiple MAVs and access to the functionality of other modules. As single-page-application (SPA) applying a model-view-controller pattern (MVC) the frontend is mainly data-driven, establishing a reliable binding between the backend and the visualization, propagating any value changes directly to the display. The background map is important for the users general orientation and critical for task planning. It is possible to feed in any OGC compliant WMS or TMS as map source, but one can also use a stored orthophoto to deploy a local map service with preprocessed tiles on the fly. This allows a convenient usage like Google Maps, even without internet connection.

3.2 UAV Driver Module

There are many different manufactures for MAVs and most of them provide their own communication protocol. Many of the open source MAVs are using the MAVLink Micro Air Vehicle Communication Protocol (Meier et al., 2013). Main goal for the UAV-driver is to unify those independent protocols to guarantee access to a wide range of MAVs via the uavRC software system. Hence the UAV-driver serves as a translator between the commands specified among the uavRC system and the commands used by the devices. The interface to uavRC is divided into commands to control the UAV on the one hand (Table 1 lists the minimum requirement commands) and data-providing publications from the UAV on the other hand (Table 2 lists the minimum requirement topics).

URI	Corresponding action
uRC.uav.GOTO	Send UAV to the requested position
uRC.uav.COMEHOME	asking UAV to head for a set homeposition
uRC.uav.HOLD	stop the current movement and hover
uRC.uav.LAND	landing the UAV at the current position
uRC.uav.ROTATE	rotate UAV about the vertical axis
uRC.uav.HOME_GET	requesting homeposition
uRC.uav.HOME_SET	setting the homeposition
uRC.uav.POSITION_GET	requesting current position
uRC.uav.ORIENTATION_GET	requesting current orientation
uRC.uav.STATUS_GET	requesting the current status
uRC.uav.PROPS_GET	requesting device-specific parameters
uRC.uav.PROPS_SET	setting device-specific parameters

Table 1: Command-interfaces to control a UAV via websockets

Currently there are two different UAV-drivers, a Simulator UAV and an adaptor to UAVs from Ascending Technologies GmbH (Asctec). A MAVLink driver and a driver for Mikrokopter UAVs (Buss and Busker, 2015) are in preparation.

3.3 The Task Scheduler

The Task Scheduler module is designed to handle any kind of automated flight execution. There are several generic operations on tasks that are supported like starting, pausing, resuming and aborting a task. In addition, tasks can handle further data like flight paths, devices and metadata for using the parameters for flight automation and reflecting the data to support e.g. a distributed user-interface. Each task has a specific set of default parameters which are also provided by the task-module.

URI	Information-content
uRC.uav.POSITION	current position of the UAV
uRC.uav.ORIENTATION	current orientation of the UAV
uRC.uav.STATUS	current state of the UAV (is moving etc)
uRC.uav.PROPS	device-specific parameters like position accuracy
uRC.uav.TARGET_REACHED	UAV reached target
uRC.uav.TARGET_APPROACHING	UAV started approaching target
uRC.uav.TARGET_ABORTED	UAV aborted approaching target

Table 2: Topics UAVs publish information on

The implemented module offers a generic way to manage tasks in order to facilitate adding new task types. Currently only *Wildretter*-tasks are supported, which means having an area of interest observed by one MAV carrying a single sensor. Nevertheless the module can be extended easily to provide variable tasks like letting a swarm of MAVs explore a defined area simultaneously or observing a building from different point of views in order to generate a 3D point cloud.

To enhance the workflow for flight preparation, it is possible to generate multiple tasks and executing them on demand. In practice this means one can plan the next flight while the MAV is still busy completing a previous task. Also it is possible to observe two UAVs executing two different tasks at the same time.

3.4 Flight Path Planner

There exists a flight path planner for vertical image acquisition with whole area coverage. It guarantees full coverage even at hillside areas because the calculation is based on a digital elevation model. The flight altitude will be adapted for constant acquisition distance to avoid gaps in the coverage (see fig. 3). Due to the fact,

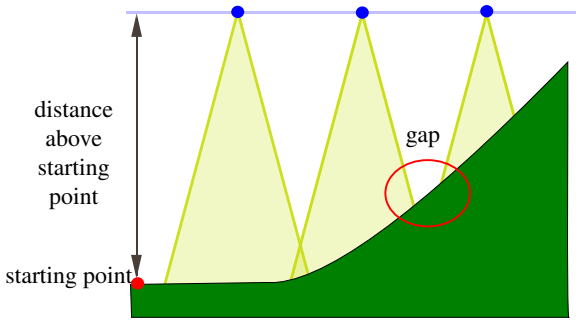

Figure 3: Scanning a hillside area while flying on a constant distance above the starting point can lead to gaps in the coverage.

that the flight duration of MAVs is limited to few minutes, a time-optimized flight path is of crucial importance to achieve high area capacity. An automatic calculation of such a flight path is not a trivial task, especially when the area is not rectangular, a special MAV orientation is desired or if there are restricted areas where the MAV is not allowed to fly. Our flight path planner module requires a boundary polygon with geo coordinates in form of a GeoJSON (Butler et al., 2008), the cameras field of view, the desired overlap and distance to the ground and the launch position as input parameters.

It is common to scan an area line by line. At the reversal point there are two waypoints side by side to guarantee a parallel fligh path. The UAV has to break and accelerate again, which takes much time. Therefore the flight path planner tries to minimize the amount of such short distant waypoints. For convex polygons, a good approach is to find the longest extend of the polygon and setting this direction as the main flight direction. For many cases then the amount of scanlines is minimized. Finding the minimum amount of scanlines in concave polygons is way more complicated and not yet implemented.

Because of the modular software concept, it is easy to replace the vertical image acquisition flight path planner with another flight path planner for a different task.

4. APPLICATION

This software was developed during our work on a system to detect roe deer fawns in meadows, to rescue them from being killed

by mowing machines (Israel et al., 2010). Under different sensor/plattform configurations we found the best detection results with a thermal camera mounted on a MAV that is looking vertical down (Israel, 2011). Since 2013 (Wimmer et al., 2013) we could increase the flight altitude from 50 m to 80 m, due to optimized image processing algorithms. With our for this application optimized flight path planner we achieve now an area capacity of maximum 7 hectars per flight with a Falcon-8 from Asctec. The duration of one flight is about 12 min. The swath overlap is still 30% but in contrast to (Israel, 2011) we don't stop for each photo. We rather define supporting waypoints each 100 m, so that the uav is not drifting too far away from the desired path, which is mainly caused by wind. During the flight the camera is time-triggered instead of position-triggered. This yields to a less accurate georeferencing, but enables an area coverage of about the double. At 80 m distance to the ground an trigger intervall of one second leads to an overlap of minimum 43% with our thermal camera (FLIR Tau640, 19mm) at the maximum possible flight speed of 20 m/s[1]. But to guarantee sharp images we restrict the flight speed to 5 m/s. Taking more than just one image per scene (as usual by position triggering) allows us to enhance the detection reliability.

In the whole process of finding roe deer fawns our flight assistance software uavRC helps to fasten the subprocesses mission planning and flight guidance. The mission planning with our software takes only few seconds, even on hillside areas and difficult area geometry.

5. OTHER MISSION PLANNER

There exists lots of flight assistance software or ground control stations with very different features. The focus of this compari-

[1] As per Asctending Technologies the Falcon-8 can cope with a maximum wind speed of 10 m/s. If the Falcon-8 flies at maximum wind speed with tailwind this leads to 20 m/s

son section is just on the mission planning part of some popular ground control stations.

Ground Control Stations for MAVLink based UAVs that allow a waypoint based flight are for example QGroundControl, IDroneCtrl, HK Ground Control Station, ArduCopter Mission Planner, APM Planner 2.0 and many more. QGroundControl and APM Planner 2.0 are built with a multiple-vehicle architecture like our uavRC. Of the MAVLink based ground control stations just the mission planner is picked up, because it is the most complex one.

Our flight assistance software currently only focuses on a fast mission planning and easy to use flight guidance. There exists no 3D engine for virtual first person view like in the HK Ground Control Station, and there is no tool for calibrating the imu-sensors as in the Ascending Technologies Autopilot Control Software.

5.1 Mission Planner

The Mission Planner brings the most complex flight path planning tool of all ground control station in this comparison. It has much more features than our flight path planner. But nevertheless we can not use it, because it does not support our Asctec UAV, and the hillside problem is not took into account. The Features: It has several different geometry templates for area scans with a comfortable preconfiguration for different cameras and a reasonable image overlap. All parameters can be adjusted individually. A polygon can be defined to mark a scan area. The Survey (Grid) of the Mission Planner brings nearly all expected features for fast full area coverage vertical image acqusition. The flight orientation will be adjusted automatically in the direction of the longest polygon extend. The altitude of all waypoints are set to the same value. An elevation model of the ground is not incorporated. The hillside problem as described in 3.4 can only be avoided by manually adapting the elevation of each waypoint. Figure 4 shows a screenshot of the polygon based Survey (Grid) submenu. The background map is zoomable and loaded from a internet based or custom WMS Server. Offline maps could be used when a WMS-Server is installed on the local machine. The Mission Planner can handle just MAVLink based UAVs.

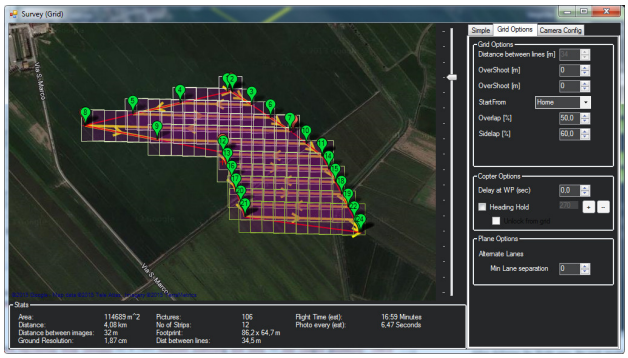

Figure 4: After defining a polygon, a right click on the map enables to start the *Survey (Grid)* widget (under *auto wp*).

5.2 Asctec Autopilot Control

For UAVs from Ascending Technologies there exists the Autopilot Control Software. From http://wiki.asctec.de it can be downloaded. It's a single UAV application with many special features for configurating and calibrating the UAV. It includes the so called *GPS Mission*, a waypoint based mission planner. Figure 5 shows a screenshot of the Autopilot Control with the GPS Mission window.

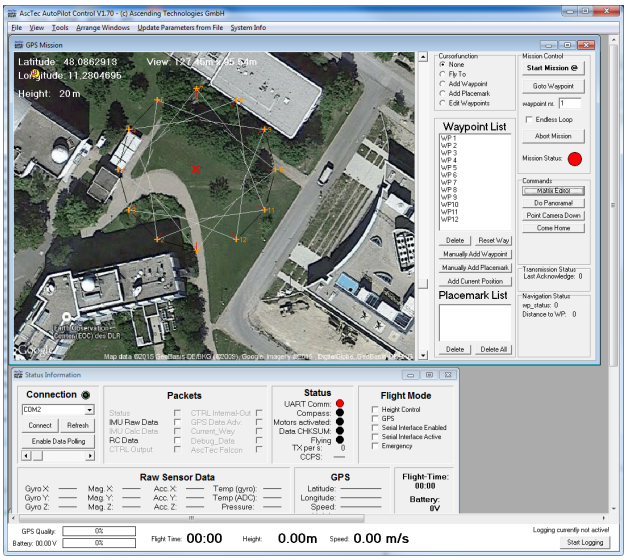

Figure 5: The AscTec Autopilot Control Software.

It allows to set a fixed size background image, that has to be in the *jpg* format. The georeferencing of the image is done by an additional file with the same name as the image, but with the extension *ini*. This file contains the geo position of the top left pixel and the bottom right pixel in WGS84 format. An example of such an ini-file is given in the following listing:

```
[Top_Left]
Latitude=48.08681614064547
Longitude=11.27921998500824
[Bottom_Right]
Latitude=48.08595789681314
Longitude=11.280933916568756
```

The waypoints can be placed manually or via the *Matrix Editor*, which allows to set a rectangle scan area with a precalculated spacing of the waypoints based on the cameras fied of view, or a point of interest circle. A polygon based scan area creation is not included. The least cumbersome way to plan a mission for a non-rectangle scan area is to use the *Matrix Editor* and use as much colums and rows, that the desired area is completely covered by the matrix grid. After drawing the matrix, all single waypoints outside of the polygon have to be deleted manually.

5.3 Mikrokopter Tool

Mikrocopter UAVs can be controlled by the *Mikrokopter Tool*. The waypoint navigation and map based flight control is called in this software *OSD* (abbrev. for On Screen Display). Like the *Asctec Autopilot Control* this software uses a fixed background image. Mikrokopter facilitates the image loading step by a link to a browser based map-tool, that generates the proper image (with geodata included) of the desired area of interest. Figure 6 shows a screenshot of the *OSD* window of the *Mikrokopter Tool*. The *Waypoint Generator* allows to create a flight path for area acquisition and point of interest circle like the *Matrix Editor* from Asctec. The setting options for those two are nearly the same. Additionally there is a panorama flight path generator and a tool to draw a raster and circles, which support the manual waypoint placing. A polygon based scan area creation is not included.

5.4 DJI PC Ground Station

DJI has the PC Ground Station and the IPad Ground Station to control a DJI UAV. The PC Ground Station uses the Google Earth

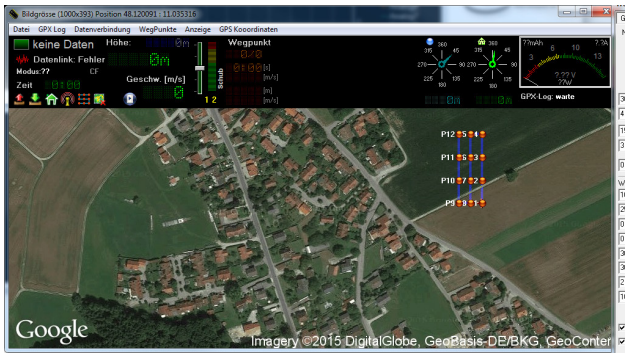

Figure 6: Mikrokopter OSD

Plugin. Due to the 3D engine the viewing direction is not limited to nadir. The software caches the google earth satellite images, so that operating offline is possible. The mission planning module of the PC Ground Station has different templates for creating flight paths. The first step is to create a boundary rectangle for the flight path. Then it is possible to choose between point, line, triangle, rectangle, circle and scan. Scan is a line by line scan with waypoints only at the reversal points. In contrast to the Matrix of Asctec there are no waypoints between the reversal points (see figure 7). All waypoints have the same elevation, which has to be specified in this step. Changing the altitude of single waypoints is possible after leaving the route template toolbox.

Figure 7: Mission Planning with the DJI PC Ground Station.

6. CONCLUSION

We presented the concept and the first implementations of our uavRC flight assistant software. It is still in progress and yet in beta-stage. Our goal was to use the same planning and control software for all of our UAVs from different manufacturers. Beside the uavRC, such a software does not exist. We've learned from our roe deer fawn application, that for many multicopter based applications it's of crucial importance to optimize (accelerate) the flight path planning and the flight itself due to the short flight duration. The distributed architecture enables the use of any kind of smartphone or tablet as user interface. The connection to a uav (e.g. XBee module) is plugged into a different computer (e.g. a Raspberry Pi).

ACKNOWLEDGEMENTS

The project is supported by funds of the Federal Ministry of Food, Agriculture and Consumer Protection (BMELV) based on a decision of the Parliament of the Federal Republic of Germany via the Federal Office for Agriculture and Food (BLE) under the innovation support programme.

REFERENCES

Birrell, A. D. and Nelson, B. J., 1984. Implementing remote procedure calls. ACM Transactions on Computer Systems (TOCS) 2(1), pp. 39–59.

Bray, T., 2014. The javascript object notation (json) data interchange format. http://tools.ietf.org/html/rfc7159.

Buss, H. and Busker, I., 2015. mikrokopter. http://www.mikrokopter.de/en/home.

Butler, H., Daly, M., Doyle, A., Gillies, S., Schaub, T. and Schmidt, C., 2008. The geojson format specification. http://www.geojson. org/geojson-spec. html.

Eugster, P. T., Felber, P. A., Guerraoui, R. and Kermarrec, A.-M., 2003. The many faces of publish/subscribe. ACM Computing Surveys (CSUR) 35(2), pp. 114–131.

Fette, I. and Melnikov, A., 2011. The websocket protocol. https://tools.ietf.org/html/rfc6455.

Hallermann, N. and Morgenthal, G., 2013. Unmanned aerial vehicles (uav) for the assessment of existing structures. In: IABSE Symposium Report, International Association for Bridge and Structural Engineering, pp. 1–8.

Israel, M., 2011. A UAV-based roe deer fawn detection system. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVIII, pp. 1–5.

Israel, M., Schlagenhauf, G., Fackelmeier, A. and Haschberger, P., 2010. Study on wildlife detection during pasture mowing. In: 68. Internationale Tagung LAND.TECHNIK, Braunschweig.

Meier, L., Camacho, J., Godbolt, B., Goppert, J., Heng, L., Lizarraga, M. et al., 2013. Mavlink: Micro air vehicle communication protocol. Online. http://qgroundcontrol. org/mavlink/start.

Mende, M., 2014. Flugassistenzsoftware für unbemannte kleinfluggeräte. Bachelor's thesis (Studienarbeit), Duale Hochschule Baden-Würtemberg Mannheim, Germany.

Neitzel, F. and Klonowski, J., 2011. Mobile 3d mapping with a low-cost uav system. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci 38, pp. 1–6.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R. and Ng, A. Y., 2009. Ros: an open-source robot operating system. In: ICRA workshop on open source software, p. 5.

Tarkoma, S., 2012. Publish / Subscribe Systems: Design and Principles. Wiley Series on Communications Networking & Distributed Systems, Wiley.

Tavendo, 2014a. autobahn project homepage. http://autobahn.ws.

Tavendo, 2014b. crossbar specification. http://www.crossbar.io.

Wimmer, T., Israel, M., Haschberger, P. and Weimann, A., 2013. Rehkitzrettung mit dem Fliegenden Wildretter: Erfahrungen der ersten Feldeinstze. Bornimer Agrartechnische Berichte.