Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für
Technische Informatik und
Kommunikationsnetze

Ramon Hofer

UAVs in Rescue Mission Networks

Semester Thesis SA-2012-03
February 2012 to May 2013

Tutor: Mahdi Asadpour
Co-Tutor: Dr. Karin Anna Hummel
Supervisor: Prof. Dr. Bernhard Plattner

Abstract

Rescue missions very often take place in environments without or with destroyed communication infrastructure. The SWARMIX project investigates the interaction of heterogenous agents on a search and rescue mission containing human rescuers tracking dogs and Unmanned Aerial Vehicles (UAVs).
Communications comprehends sensor data like positions, images, audio and video sent between each agent and the ground station via an ad hoc network.
For this semester thesis an experimental ad-hoc network is set-up and a quadrocopter is used as an UAV. Some operating figures (throughput, signal quality, delay, etc.) are measured for different scenarios.
The collected data is analysed to provide some parameter data for an outdoor 802.11n ad-hoc wireless network in the 5 GHz band.
To equip the Arducopter drone with WLAN capability a Gumstix minicomputer is used together with a Linksys USB WLAN stick.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

In rescue missions the communication infrastructure can be completely destroyed or not available ab initio. Nevertheless the ability to communicate is crucial to transfer multimedia data, geographical maps with locations of interesting objects and other sensor data. The rescue teams want to communicate with each other and with foreign organisations. Satellite communication is usually very expensive and has low latency. GSM based communication is possibly not available e.g. in avalanche rescue missions or destroyed during an earthquake or a tsunami.
Synergistic Interactions in Swarms of Heterogeneous Agents (SWARMIX) can deploy a very flexible communication infrastructure that can even adapt to different locations and use cases. A voice communication has different Quality of Service (QoS) requirements than latency-insensitive bulk data transfer like sending an image file. This means that for voice communication the latency and jitter of usually connectionless datagram communication (usually UDP) are more important. Whereas the connection-oriented protocol (usually TCP) used for bulk data transfer require reliable and ordered packages otherwise the file will become unusable.
Therefore it could be better to load a file onto the storage on the agent (quadrocopter, deltawing or dog) which is carried to the destination where it's delivered (load-carry-and-deliver paradigm, LCAD [2]). Obviously this is not feasible for voice communication where the delay and jitter is more important.

1.2 The Task

The task of this thesis includes the set-up of an experimental ad-hoc network with the ArduCopter [9], a Linux Overo® gumstix board [5] and Linksys Wi-Fi USB Adapters.
The Arducopter is available ready to fly and a working image for the gumstix board is present. The driver's sourcecode for the Wi-Fi USB adapter is available and easy to compile.
There are two different sources where the data is to be collected from: On one hand it's the link characteristics for which the data collection has to be implemented. On the other hand it's the ArduCopter's log file which contains GPS position, height, orientation, speed, etc. The two data sources have to be combined to get a single dataset per series of measurements.

1.3 Related Work

Henkel and Brown [3] analyse two different approaches of using mobile communication nodes. They differentiate between the chain-relay and conveyor-belt modes and compare the two modes and show what merits they have. The communication data rate is considered a continuous function of node separation distance and a delay-optimal scheduling algorithm is derived.
Figure 1.1 shows the chain relay-model proposed by them. The distance to the neighbour is given by

$$d_{neighbour}(n) = \frac{l - (n+1)r}{n} \tag{1.1}$$

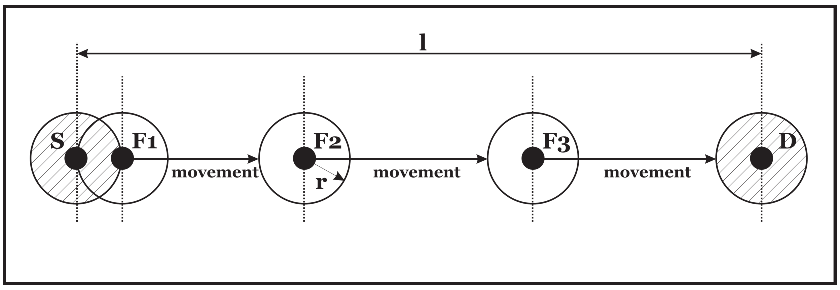


Figure 1.1: Chain-Relay Model [3]

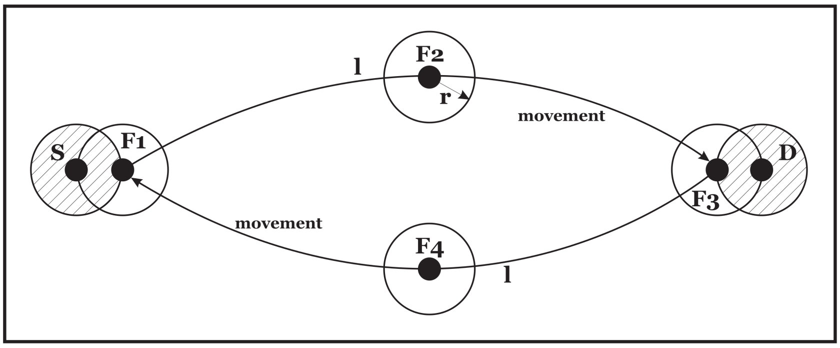


Figure 1.2: Conveyor Belt Model [3]

where l is the total distance, n the number of drones and r the communication radius. Using the speed v, the buffer size b and the load rate R

$$t_{cycle} = t_{travel} + t_{tx} = 2\frac{l-(n+1)\cdot r}{n\cdot v} + 2\frac{b}{R} \tag{1.2}$$

this leads to the throughput

$$T = \frac{b}{2\left(\frac{l-(n+1)r}{n\cdot v} + \frac{b}{R}\right)} \tag{1.3}$$

The conveyor belt model is shown in Figure1.2. The cycle time is given by

$$t_{cycle} = t_{travel} + t_{tx} = 2\frac{l-2r}{v} + 2\frac{b}{R} \tag{1.4}$$

and the throughput

$$T = \frac{n\cdot b}{2\cdot t_{travel} + 2\cdot t_{tx}} = \frac{n\cdot b}{2\frac{l-2r}{v} + 2\frac{b}{R}} = \frac{n}{\frac{2(l-2r)}{v\cdot b} + \frac{2}{R}} \tag{1.5}$$

if the maximum rate at which a node can transmit R is to be taken advantage of throughput $T = R$ and thus

$$n_{opt} = \left\lceil 2\cdot\left(1 + \frac{l-2r}{v}\cdot\frac{R}{b}\right)\right\rceil \tag{1.6}$$

Li et al. [6] focus on the models of 802.11 wireless ad-hoc network. Due to the need of forwarding the packets the throughput can be surprisingly low. They investigate the interference among a chain of nodes and the scalability of such networks.
Another related work where measurement on 802.11g-based Ad-hoc networks in motion is performed by Hummel et al. [4]. They concentrate on Vehicular ad-hoc networks (VANETs) with typical car scenarios like taking over, moving towards, crossing and roundabouts.
Cheng et al. [2] work on Maximizing Throughput of UAV-Relaying Networks with the Load-Carry-and-Deliver Paradigm. Bulk transfer can be faster using the LCAD paradigm than a link over a larger distance. They use model airplanes to perform their measurements.
A larger scaled network is analysed by Brock et al. [1]. They focus on the routing of packets in wireless ad-hoc networks.

1.4 Overview

Chapter 2 describes the testbed including the Arducopter drone, the Gumstix minicomputer and the Linksys USB WLAN stick. It also contains information on how the data is collected, prepared and analysed. In chapter 3 some interesting results are presented. Finally an outlook for future work is given in chapter 4 and chapter 5 summarise the findings.

Chapter 2

Testbed

2.1 Arducopter

2.1.1 Unmanned Aerial Vehicle (UAV)

The UAV is a quadrocopter from the Arducopter open source project [9]. The main controller board APM 2[1] is based on an Atmega2560 microprocessor with a 3-axis gyro, 3-axis accelerometer, a pressure sensor, an USB interface and an external GPS connector. Figure 2.1 shows a ready to fly drone.

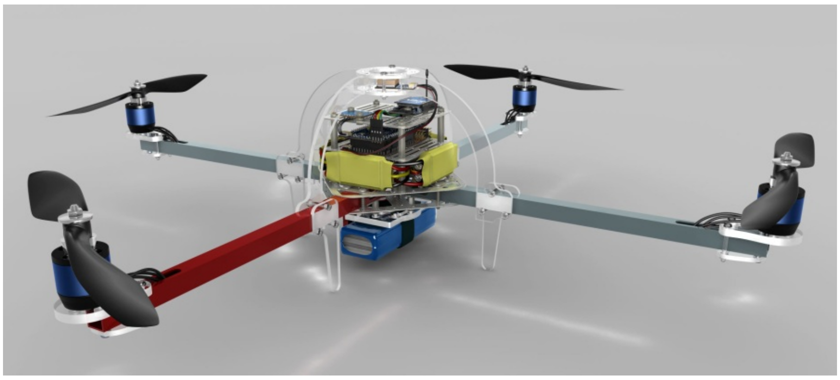


Figure 2.1: Arducopter quadrocopter

The drone has four stepper motors that run pairwise in the same direction. They are controlled by the APM board. Different flight modes are realised:

Stabilize This is the primary operating mode. The drone is automatically leveled and maintains its heading.

Simple An even easier to fly mode. The control is not relative to the drone heading but always on absolute to the compass.

AltHold The throttle is automatically applied to hold the current altitude.

Loiter The current location, heading and altitude is maintained automatically.

Auto The drone follows a predefined script (see chapter 2.1.2).

There are some more flight modes which are described in detail in the Arducopter wiki[2].

[1] http://code.google.com/p/arducopter/wiki/APM2board
[2] http://code.google.com/p/arducopter/wiki/AC2_ModeSwitch

2.1.2 Mission Planner

The mission planner can be used as the name suggests to plan missions for the Arducopter's auto mode. Apart from that it can be used as a full ground station for life monitoring and sending in-flight commands, download mission log files and analyse them (e.g. replay the mission afterwards), configure APM board setting via a GUI or a terminal. The mission planner can even run Python scripts which could be used to send it to a position depending on the link characteristics of the wlan ad-hoc connection statistics.
An example mission is shown in Figure 2.2. It was created by clicking on the Google map in the Flight Planner tab. The parameters of the waypoints can be set individually e.g. a waypoint can be changed to loiter for $5\,\mathrm{s}$. The mission can be saved for later use or directly loaded onto the drone which then can be advices to follow the mission.

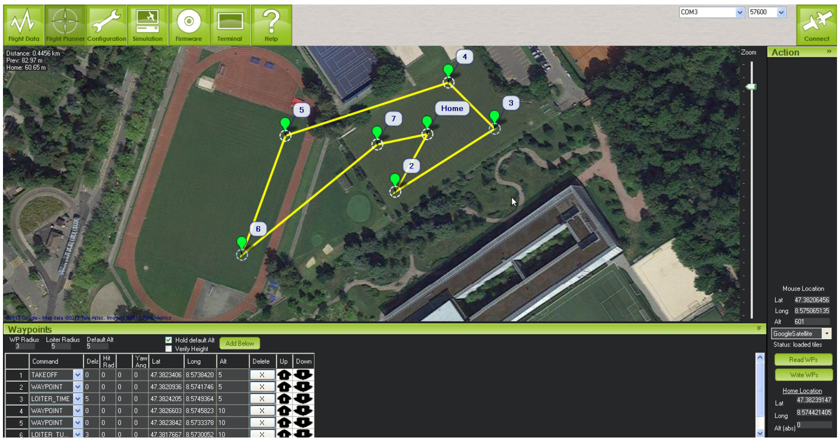


Figure 2.2: Arducopter Mission Planner

The logs can be loaded in the Flight Data tab and replayed. Or they can be exported as a txt file to process externally. Figure 2.3 shows a loaded replay.

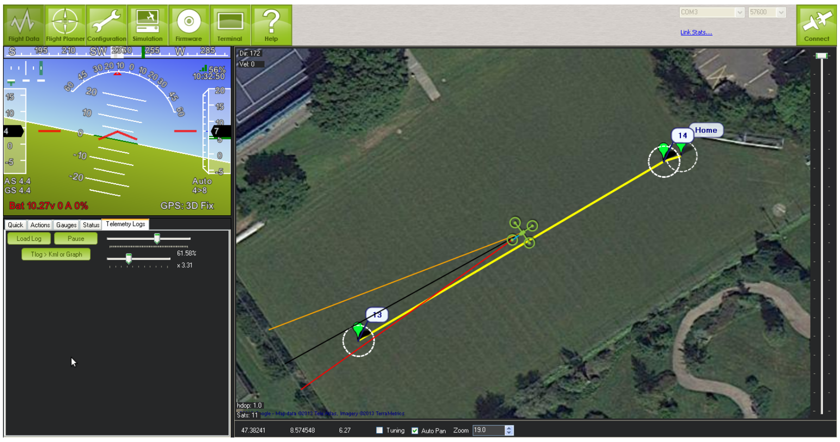


Figure 2.3: Arducopter Mission Planner replay function

2.1.3 Telemetry Logs

The Arducopter telemetry logs can be converted to .txt files by selecting Telemetry Logs then Tlog > Kml or Graph in the Mission Planner's Flight Data tab. The log data is grouped. The two most interesting contain information about the position, altitude, speed and heading of the drone. The groups contain:

- mavlink_gps_raw_int_t
  - lat: The lateral position from the GPS module as decimal integers

  - lon: The longitudinal position from the GPS module as decimal integers
  - satellites_visible : The number of visible satellites

- mavlink_vfr_hud_t
  - alt : The altitude relative to the home altitude
  - heading: The heading of the drone in degrees according to the home heading
  - groundspeed: The speed relative to the ground in $\mathrm{m/s}$

2.2 Gumstix minicomputer

The Gumstix minicomputer [5] is a tiny Linux computer-on-module. The main computer board is proprietary but there exists some Creative Commons Share-alike licensed expansion boards. There exist different versions of the Overo main module with bluetooth, 802.11b/g, OpenGL and extended temperature components. Unfortunately there is no version capable of 802.11n ad hoc communication in the $5\,\mathrm{GHz}$ band. Therefore a Tobi expansion board is used which provides a USB port and other things that are not used for this thesis like a HDMI port, 10/100baseT ethernet and stereo audio. It also possesses a USB serial terminal which comes in handy to do maintenance together with the kermit software for Linux.
To create a bootable microSD card image the Gumstix homepage [5] provides an easy guide with the commands listed to create the partitions and copy the files. It has guides for an own Linux distribution, Ubuntu or even Android. For this work the Gumstix Linux was used. This image uses about $500\,\mathrm{MB}$. To have as few complexity on the drone and also have the timestamps of the measurements from the measurement-toolbox (see chapter 2.4) synchronised with the Arducopter Mission Planner (see chapter 2.1) iperf server is the only additional software needs to be running on the Gumstix.

2.3 Linksys USB WLAN stick

The Linksys AE1000 [7] is 802.11abgn compliant and has a selectable 2.4 or $5\,\mathrm{GHz}$ band. It features two internal antennas and can be plugged into an USB 2.0 port. The chipset is based on the Ralink RT2870.

2.4 Measurement Toolbox

The Measurement Toolbox is a bash script that collects data from iwconfig, ping, iwpriv and iperf. Actually nmap −sP is used instead of ping because it returns the time always in ms in contrary to ping which returns longer RTTs in s. A complete list of the values extracted from the four commands can be found below:

- iwconfig
  - Timestamp
  - BitRate
  - LinkQuality
  - SignalLevel
  - NoiseLevel
  - RxInvalidNWID
  - RxInvalidCrypt
  - RxInvalidFrag
  - TxExcessiveRetries
  - InvalidMisc
  - MissedBeacon
- ping (nmap −sP)
  - Timestamp
  - RTT
- iwpriv (Depends on the used driver)
  - Timestamp
  - TxSuccess
  - TxRetryCount
  - TxFailToRcvAckAfterRetry
  - RtsSuccessRcvCts
  - RtsFailRcvCts
  - RxSuccess
  - RxWithCrc
  - RxDropDueToOutOfResource
  - RxDuplicateFrame
  - FalseCca(OneSecond)
  - RSSI-A
  - RSSI-B
  - RSSI-C
  - WpaSupplicantUP
  - RT2860LinuxStaPinCode
- iperf (UDP)
  - Timestamp
  - Bandwith
  - Jitter
  - LostDatagrams
  - OutOfOrderDatagrams

A pass of measurements always consists of performing the four commands iwconfig, nmap −sP, iwpriv and iperf.
The script extracts the values and stores them in a comma separated format. For each pass a new line is written. Per line there's one measurement per command and for each of them there's a separate timestamp. The timestamp format is date +'%Y%m%d_%H%M%S'.
The script accepts several command line arguments. They can be listed with measurement−toolbox −h:

```
Usage: measurement-toolbox [OPTIONS] SERVER
Log wifi connection information from ping, iwconfig (Linksys), iperf
iperf has to be started with correct protocol (UDP/TCP) setting on server
Depends on bc and is specific to the iwconfig output for Cisco Linksys AE1000

Options:
  -l FILENAME     Filename of logfile (default /home/hoferr/wlink_values_20130302_193314
      .csv)
  -n NUMBER       Number of loops (-1: loop forever; default -1)
  -w TIME         Time to wait after each measurement in seconds (default 1)
  -i INTERFACE    Interface to use (default ra0)
  -b BANDWIDTH    Target UDP bandwidth to n bits/sec (default 200 Mbit/sec)
  -h              Show help
```

Listing 2.1: measurement−toolbox help output

- The SERVER argument stands for the IP address of the device on which iperf server is running.
- The −l FILENAME option can be used to choose a different name for the output file where the log will be saved, e.g. to include more details about the test run. Sometimes it's handy to insert some comments into the same file. This can be done by:

```
echo "Test 1; Start" >> FILENAME
```

Listing 2.2: Add some comment to log file

Where FILENAME stands for the filename which is displayed when the script was started.

- The −n NUMBER option sets the number of loops which the script will perform. If it's set to 1 for example there will be exactly one measurement for iwconfig, nmap −sP, iwpriv and iperf. If set to -1 the script will run until it's manually ended by the user (e.g. by pressing CTRL+C).
- With the −w TIME option the script can be paused after each pass. The sleep command is used which accepts integer and floating point values in s. The TIME option could be used for a future version as time for a complete pass instead of a pause after a pass. But since this value was always 0 for this work I've kept at this simple solution.
- The interface can be specified with the −i INTERFACE option.
- The time iperf transmits for can be specified with the −t TIME option. The value is in s.
- The option −p runs iperf in TCP mode. Please make sure that iperf server is started in the same mode.
- The iperf target bandwidth can be set with the −b BANDWIDTH option. This option is directly passed to iperf. The value is interpreted as $\mathrm{bit/s}$. Although with K or M the value can be changed to $\mathrm{kbit/s}$ or $\mathrm{Mbit/s}$ respectively.
- To change the output of iperf the option −f FORMAT can be used. The default value is Mbit. The values stand for: b bit, k kbit, m Mbit, g Gbit, B B, K kB, M MB, G GB.
- The −o OPTIONS can be used to set further iperf options. Please see the manpage for more details.

2.5 Log Merger

The data in the log files from the Mission Planner and the measurement-toolbox have to be merged together and presented in a way so that they can be analysed easily. For this task a C++ parser was written. It's called slm (Swarmix-Log-Merger). To use it at least a Mission Planner log in .txt format and a measurement-toolbox .cvs have to be passed. The data is merged with respect to the timestamps.
There are some arguments which control the way the logs are merged. They can be listed with slm -h:

```
Usage: slm [OPTIONS] -a ac-log.txt -m mt-log.csv
Analyse swarmix data: WLAN Adhoc link connection and Arducopter flight data.

Allowed options:

Generic options:
  --version              print version string
  -h [ --help ]          produce help message
  -v [ --verbose ]       print used options

Configuration:
  -t [ --title ] arg     title of measurement
  -a [ --at-txt ] arg    arducopter telemetry txt file
  -m [ --mt-csv ] arg    wireless measurement-toolbox csv file
  -o [ --olog ] arg      output log filename
  --from arg             start time [YYYY-MM-DD_hh:mm:ss.nnnnnnnnn]
  --until arg            end time [YYYY-MM-DD_hh:mm:ss.nnnnnnnnn]
  --period arg           time period in seconds
```

Listing 2.3: slm help output

All the passed options can be displayed with -v to validate the intended merging.
The title of the measurement can be specified with the -t arg option. It's used to set the title inside the output file.
The logfile options (-a arg and -m arg) are mandatory and at least one logfile of both types have to be passed. It's possible to specify several files for each type to combine e.g. multiple measurements in one single dataset to analyse more than one measurement at once.
The output filename (-o arg) is also mandatory in the current version 0.1. The data is merged and saved to a file with the title and the log filenames. The start and end times (see below) are also stored next to the title.
The two options --from arg and --**until** arg can be used to limit the start and end times of the data. If e.g. there's data but it's meaningless.
The --period arg can be used to match the periodic times of the log types. The measurement is ignored if the difference to the preceding measurement is less than the provided arg time. A future version of slm could calculate the average of the data for each phase. This was not done since the two data sources both a period of 1 s.
slm depends on some boost libraries: program_options.hpp, filesystem.hpp, algorithm/string.hpp, concept_check.hpp and date_time/gregorian/gregorian.hpp.

2.6 Matlab Analysis

The analysis of the logged data is done in Matlab. There are some allocations and minor calculations done prior creating the graphs. Since there are almost 200 data columns the choosing/filtering and allocation is done prior to the calculations. The title fields are tested against a wishlist of data titles and the corresponding data is chosen.

2.6.1 Distance calculation

On of the few calculations that is done by Matlab on the data is the transformation of the raw GPS data to the planar distance and from there with the altitude to the 3D distance.
The planar distance is calculated using the Haversine function [8]. The Arducopter Mission Planner saves the lateral and longitudinal coordinates as decimal coordinates as integer.

```
for i = 1:numel(A.data(:,d_lat_n))
    gd(i) = haversine([home_lat*1e-7 home_lon*1e-7],[A.data(i,d_lat_n)*1e-7 A.data(i,
        d_lon_n)*1e-7]) * 1e3;
    d(i) = sqrt( gd(i)^2 + (A.data(i,d_alt_n)-home_alt)^2 );
end
```

Listing 2.4: Matlab distance calculation

The variable `i` is used to iterate through the data. `gd` contains the planar ground distance. The two variables `home_lat` and `home_lon` are filled with the first entry of the lateral and longitudinal coordinates since the home coordinates are not contained in the logfile. The `A.data(i,d_lat_n)` and `A.data(i,d_lon_n)` are the `i`-th value of the lateral and longitudinal coordinates respectively to which the planar distance is calculated.
The 3D distance `d` is calculated using the previously computed ground distance `gd` and the difference of the current altitude `A.data(i,d_alt_n)` and the home altitude `home_alt`.

2.6.2 Data grouping for boxplots

The data is grouped into divisions of distances or speeds respectively. The numbers are always rounded towards plus infinity. The following small algorithm performs this grouping:

```
ceil(x/interval)*interval
```

Listing 2.5: Matlab boxplot data grouping

The data `x` is divided by the `interval` rounded towards plus infinity and multiplied again by the `interval`.

2.7 Measurement Scenarios

2.7.1 Scenario 1: Back and Forth

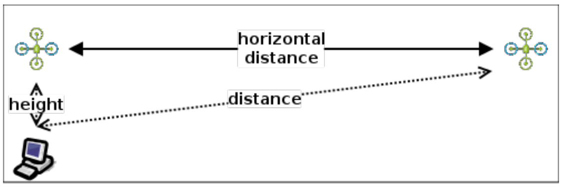
For the first scenario (see Figure 2.4a) one ground station and an Arducopter is needed. The drone starts close to the ground station and flies away until the connection is lost. It then returns on the same path to the ground station. This movement is repeated several times.

2.7.2 Scenario 2: Relaying

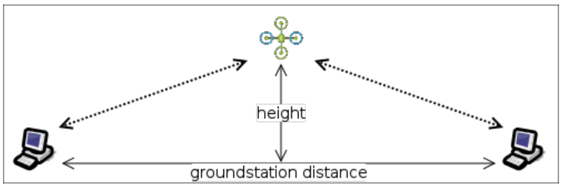
The second scenario (see Figure 2.4b) is a static measurement where two ground stations are as far apart so that they can not talk to each other. The Arducopter is loitering between the two ground stations and forwards the packets. Therefore IP Forwarding needs to be enabled on the gumstix carried by the Arducopter. On the ground station a static routing rule can be added to each of the ground station's routing tables.

2.7.3 Scenario 3: Load-Carry-And-Deliver LCAD

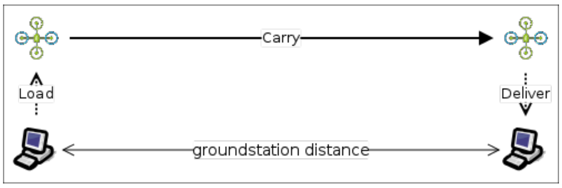
The third scenario (see Figure 2.4c) follows the Load-Carry-and-Deliver paradigm [2]. Two ground stations are needed like in the second scenario but this time the Arducopter does not stay static in between the two stations. Instead it flies close to the first ground station (source station) which loads data to the drone. The UAV flies close to the second ground station (destination station) and delivers the data. This procedure is repeated several times.



(a) Back and Forth



(b) Relaying



(c) Load-Carry-And-Deliver LCAD

Figure 2.4: Scenarios

Chapter 3

Results

The measurements were undertaken on a football court in Zurich near the zoo. Due to severe problems with the auto mode of the Arducopter the measurement flights were performed manually.

3.1 Scenario 1: Back and Forth

Figure 3.1 shows the mission replay of the first measurement. The ground station was located at the right side which means the drone in Figure 3.1 is at the furthest from the ground station.

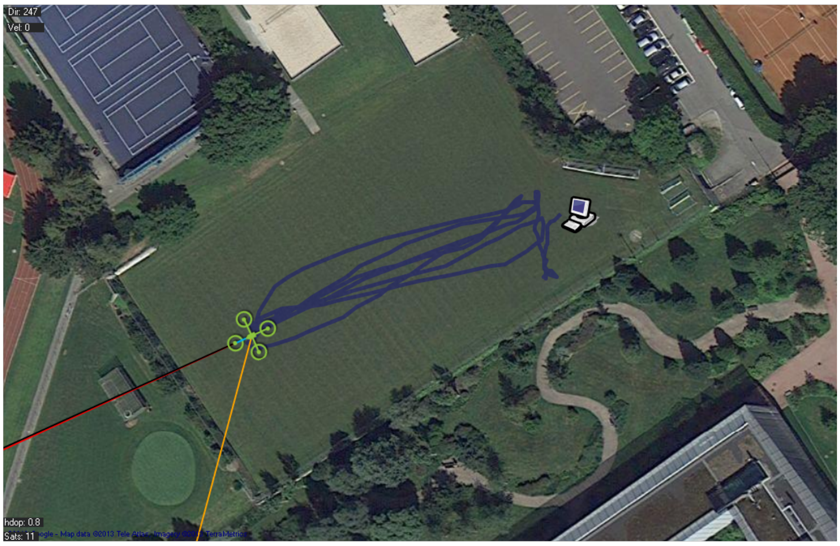


Figure 3.1: Scenario 1 - Back and Forth - Measurement Location

The data from the scenario 1 measurements were combined into a single dataset to do the analysis. The boxplots in 3.1.1-3.1.6 either compare the data with the distance or the speed. To minimize the influence of the distance in the speed plots only speeds close to either turning points are considered and drawn separately. Distances are grouped into $10\,\mathrm{m}$ and speeds to $1\,\mathrm{m/s}$ wide packets. As described in chapter 2.6.2 the distance and speed values are rounded up.
The red lines in the middle of the boxes mark the median values. The boxes extend to the 25^{th} and 75^{th} percentiles and the whiskers to the most extreme datapoints. The red plus mark outliers and the red numbers in brackets stand for the number of samples per group. As described in 2.4 each sample contains one the collected data from iwconfig, nmap −sP, iwpriv and a $100\,\mathrm{ms}$ iperf UDP measurement.

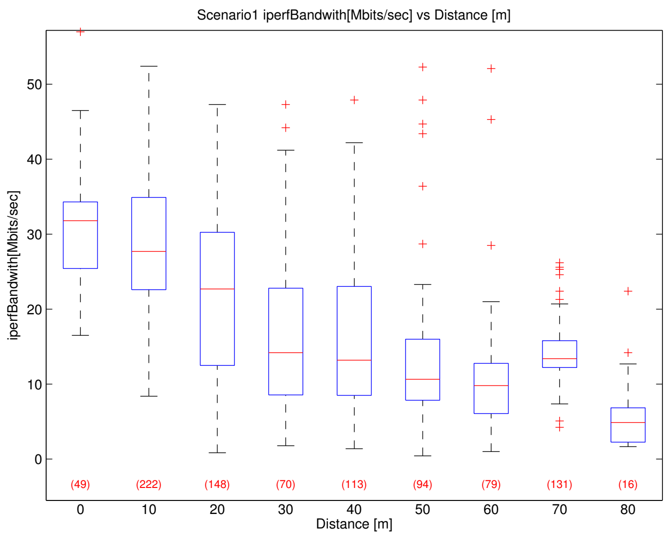
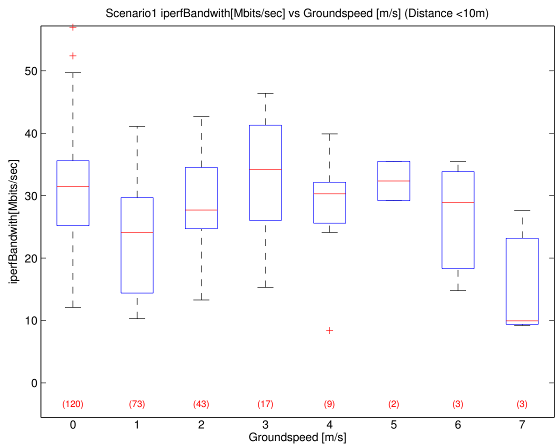


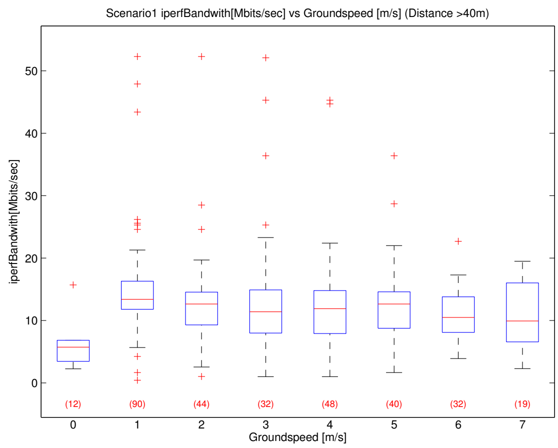
Figure 3.2: Scenario 1 - Back and Forth - Throughput vs Distance

3.1.1 Throughput

The throughput in $\mathrm{Mbit/s}$ recorded by iperf UDP measurements is drawn against the distance (Figure 3.2). The boxplot compares the throughput on the y-axis against the distance. The best throughput of about $32\,\mathrm{Mbit/s}$ is achieved with the drone right next to the ground station. The throughput drops down to about $10\,\mathrm{Mbit/s}$ for distances up to $60\,\mathrm{m}$. The reason for the higher median throughput of about $15\,\mathrm{Mbit/s}$ for the $70\,\mathrm{m}$ group could not be identified. Even though there are more datapoints the worst throughput was not as low as for distances further than $10\,\mathrm{m}$.



(a) $d < 10\,\mathrm{m}$



(b) $d > 40\,\mathrm{m}$

Figure 3.3: Scenario 1 - Back and Forth - Throughput vs Speed

Figure 3.3 shows two boxplots that compare the throughput with the ground speed. The most obvious results are for the data around the far turning point with distances $d \geq 40\,\mathrm{m}$ shown in Figure 3.3b. The slowest speed is when the drone turns around at the furthest distance an has therefore the worst throughput of only about $5\,\mathrm{Mbit/s}$. The medians for the other speeds for the furthest distances have about the same median throughput between $10\,\mathrm{Mbit/s}$ and $15\,\mathrm{Mbit/s}$. The values for speeds closer to the ground station vary more as well as their medians (see Figure 3.3a which suggests that the slow speeds of the Arducopter of less than $10\,\mathrm{m/s}$ has less influence than the relatively small distance the WLAN connection is able to span.

3.1.2 Jitter

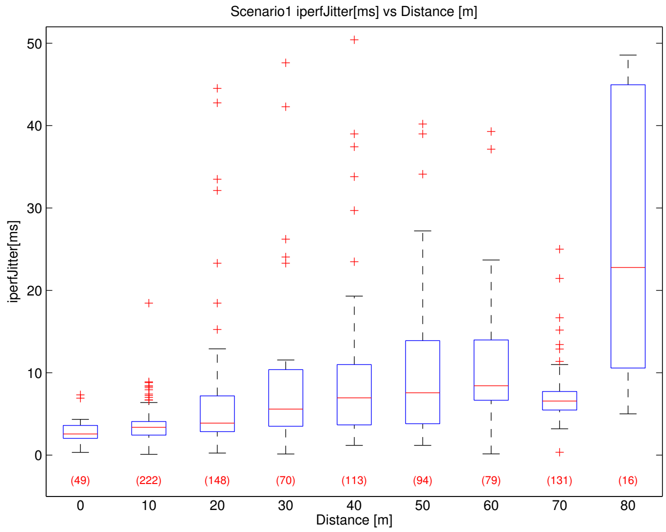
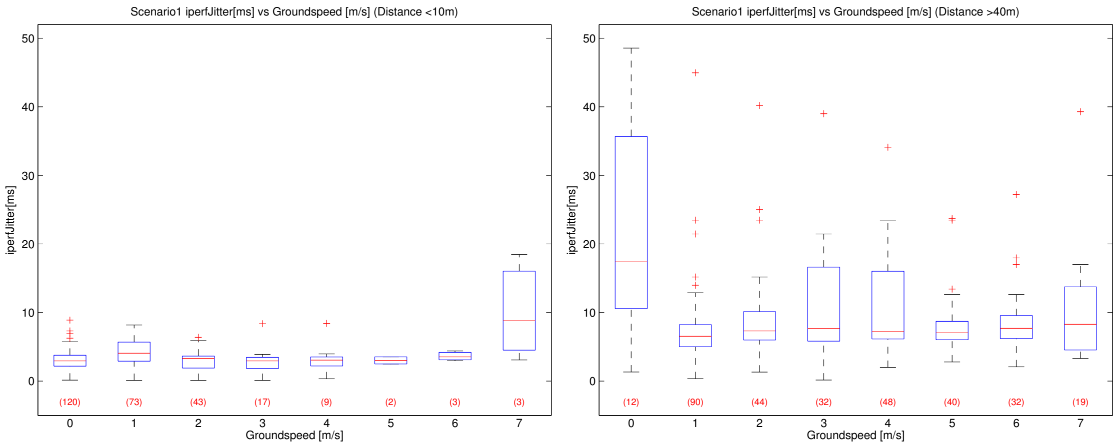


Figure 3.4: Scenario 1 - Back and Forth - Jitter vs Distance

Jitter - reported by iperf - is compared with the distance in Figure 3.4. Right next to the ground station the median is about $3\,\mathrm{ms}$ and grows slowly to about $8\,\mathrm{ms}$ at $60\,\mathrm{m}$. Especially the range of the values grow. Again one can see a little increase in link quality which means a little drop of the median value for $70\,\mathrm{m}$ with fewer variance.



(a) $d < 10\,\mathrm{m}$ (b) $d > 40\,\mathrm{m}$

Figure 3.5: Scenario 1 - Back and Forth - Jitter vs Speed

Plotting jitter against speed (Figure 3.5) shows low medians and little variance for all speeds around the ground station (Figure 3.5a) except for $7\,\mathrm{m/s}$ but this group only contains three values. The speeds around the further turning point (Figure 3.5b) vary more especially at the $0\,\mathrm{m/s}$ group which again is the furthest away from the ground station. The rest of the speed groups share almost the same median which leads to the same conclusion that the tested speeds do not nearly influence jitter as much as the tested distances.

3.1.3 TxRetryCount

The transmit retry count per second reported by iwpriv as a function of distance is shown in Figure 3.6. The best values of around $100'000$ transmission retries per second is at the ground station. The $10\,\mathrm{m}$ group already show 500 thousand retries per second. The 20 - $60\,\mathrm{m}$ group share a significantly bigger median around 1.8 million. The famous $70\,\mathrm{m}$ group has about the

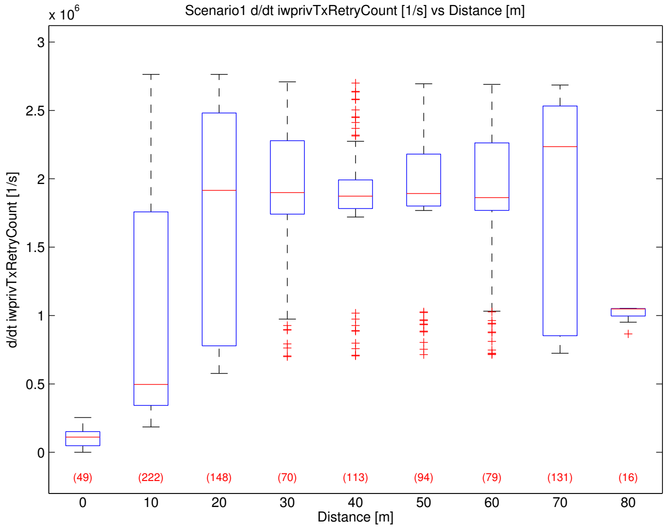
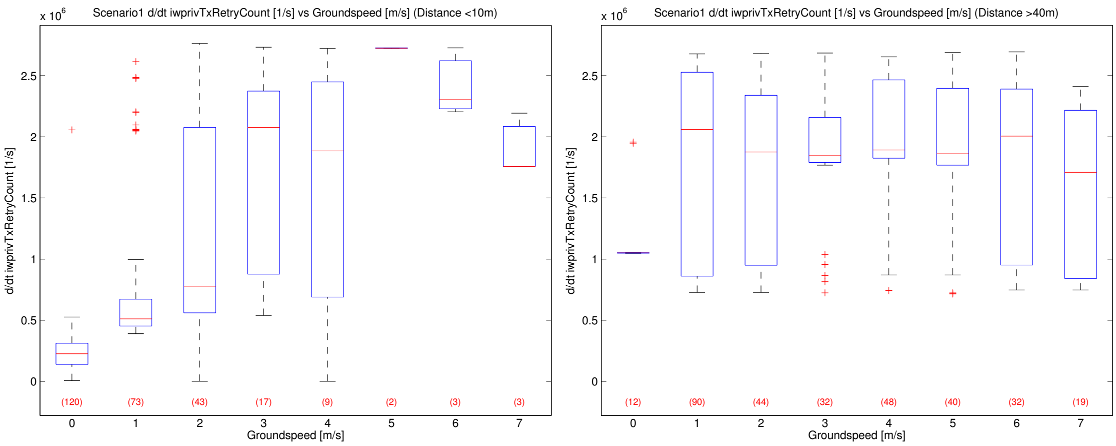


Figure 3.6: Scenario 1 - Back and Forth - TxRetryCount vs Distance

same upper limit as almost all other groups but has a wider variance with the highest median. For once the $70\,\mathrm{m}$ group has the worst median value.



(a) $d < 10\,\mathrm{m}$ (b) $d > 40\,\mathrm{m}$

Figure 3.7: Scenario 1 - Back and Forth - TxRetryCount vs Speed

The closer speed plot (Figure 3.7a) shows a growing median for the groups with more than ten values. Most of the further distance speed groups (Figure 3.7b) share about the same median value of around 2 million transmit retries per second. The slowest group surprisingly has the best median and the smallest variance.

3.1.4 Round-Trip Delay Time

Comparing the round trip time measured by nmap (Figure 3.9a) shows quite stable medians around $4\,\mathrm{ms}$ except for the $80\,\mathrm{m}$ group. The variance stays almost the same until the distance grows beyond $40\,\mathrm{m}$ with an exception of the $70\,\mathrm{m}$ group.
The speed groups around the ground station (Figure 3.9a) have comparable values to the distance groups from the last Figure 3.8. The drone of course does not need $60\,\mathrm{m}$ to accelerate to $7\,\mathrm{m/s}$ thus the plot shows that the round-trip delay times are not significantly influenced by the tested speeds close to the ground station. Looking at the speed groups around the further turning point (Figure 3.9b) the median values stay around 4 - $5\,\mathrm{ms}$. Especially the 0, 3 and $4\,\mathrm{m/s}$ groups show a higher variation with the $0\,\mathrm{m/s}$ group having the highest median and extends to higher round-trip delay times.

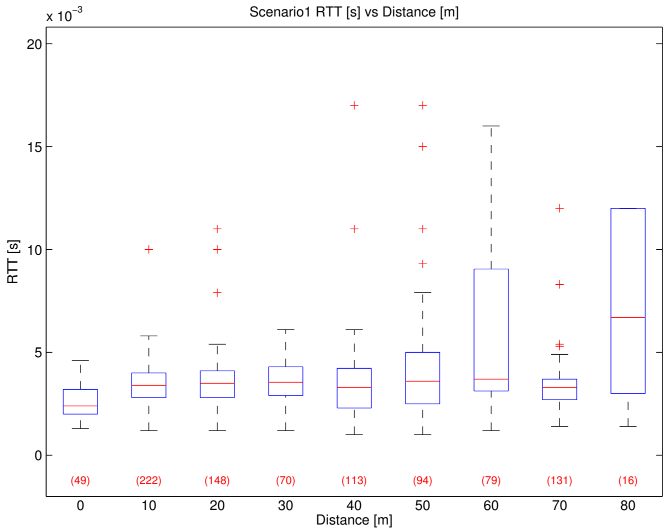
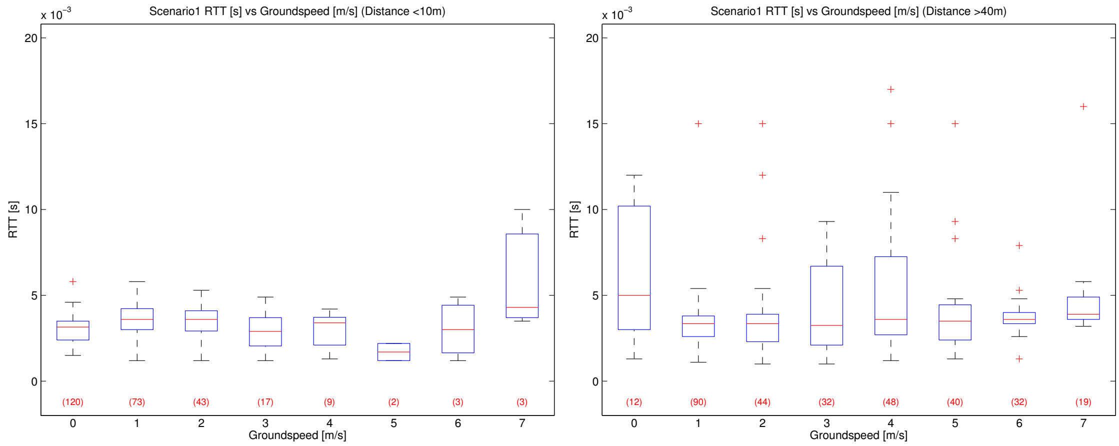


Figure 3.8: Scenario 1 - Back and Forth - Round-Trip Delay Time vs Distance



(a) $d < 10\,\mathrm{m}$ (b) $d > 40\,\mathrm{m}$

Figure 3.9: Scenario 1 - Back and Forth - Round-Trip Delay Time vs Speed

3.1.5 Link Quality

The link quality reported by iwconfig (Figure 3.10) shows a tendency of the median values getting worse from about $80\,\%$ with a variance of more than $30\,\%$ down to $60\,\%$ for distances further away than $50\,\mathrm{m}$. The variance in most groups (especially the 30 - $50\,\mathrm{m}$ groups) make this indicator look not suitable for estimating the signal quality.
The speed plots (Figure 3.11) also show a lot of variance. Speeds around the ground station (Figure 3.11a) have median values around 80 - $90\,\%$ and speeds around the further turning point 60 - $70\,\%$. Statistically the tendency is obvious but not good enough to estimate the expected real link quality.

3.1.6 Radio Signal Strength Indicator

The radio signal strength indicator is reported by iwpriv. Figure 3.12 shows that this indicator, compared with the iwconfig link quality, reports a more pessimistic estimate because the most extreme low values of about $-85\,\mathrm{dB}$ are already recorded for distances further away than $10\,\mathrm{m}$ and stay for distances up to $60\,\mathrm{m}$. The $70\,\mathrm{m}$ group again is very accurate and shows few variance.
As for the other variables the speed has no significant influence which is shown in Figure 3.13.

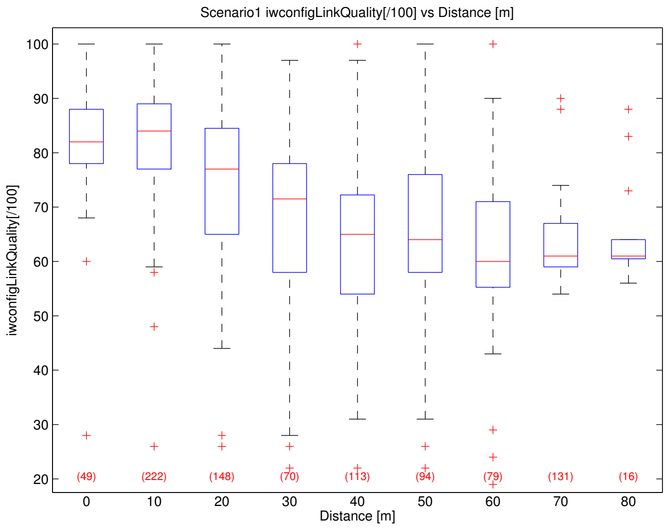
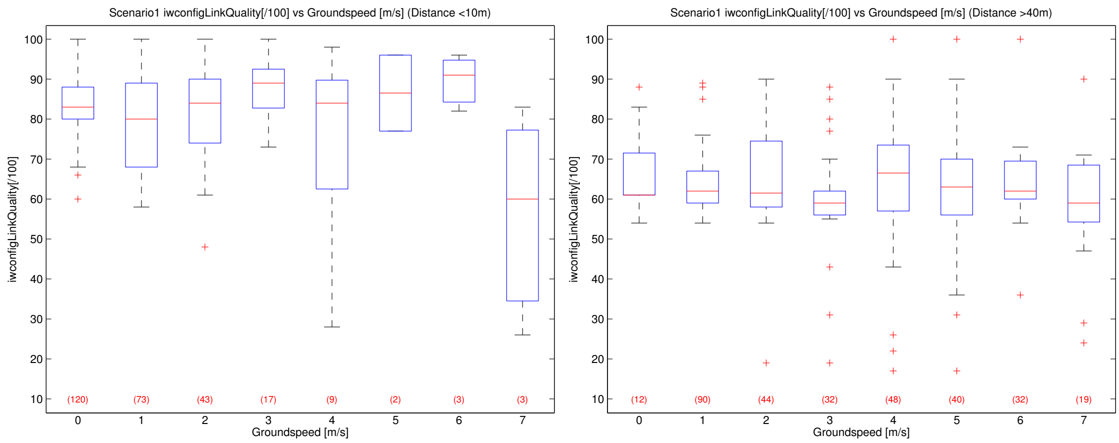


Figure 3.10: Scenario 1 - Back and Forth - Link Quality vs Distance



(a) $d < 10\,\mathrm{m}$ (b) $d > 40\,\mathrm{m}$

Figure 3.11: Scenario 1 - Back and Forth - Link Quality vs Speed

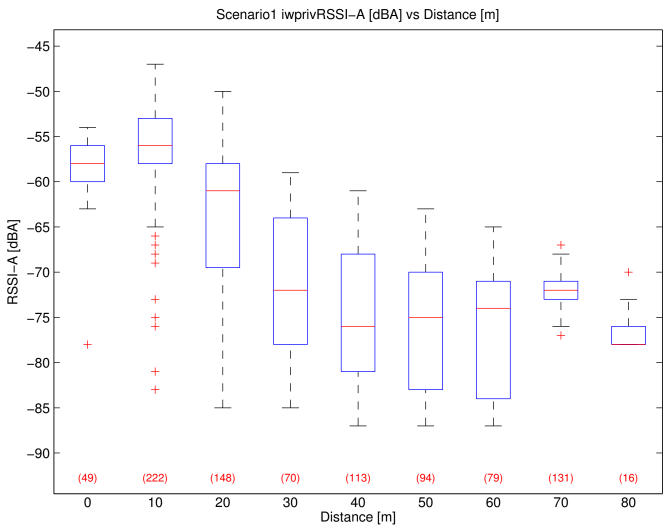
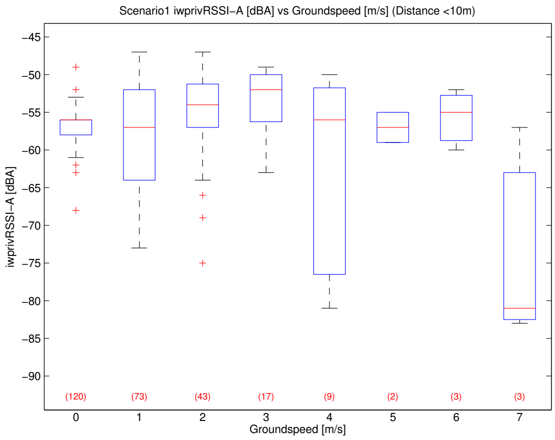
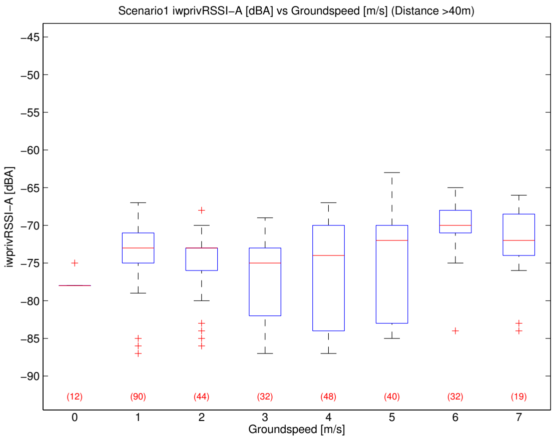


Figure 3.12: Scenario 1 - Back and Forth - Radio Signal Strength Indicator vs Distance



(a) $d < 10\,\mathrm{m}$



(b) $d > 40\,\mathrm{m}$

Figure 3.13: Scenario 1 - Back and Forth - Radio Signal Strength Indicator vs Speed

3.2 Scenario 2: Relaying

The test arrangement can be seen in Figure 3.14. The two ground stations are about $130\,\mathrm{m}$ apart. They can not see each other or communicate directly. The drone hovers in between at an altitude of about $3\,\mathrm{m}$ and forwards the packets.

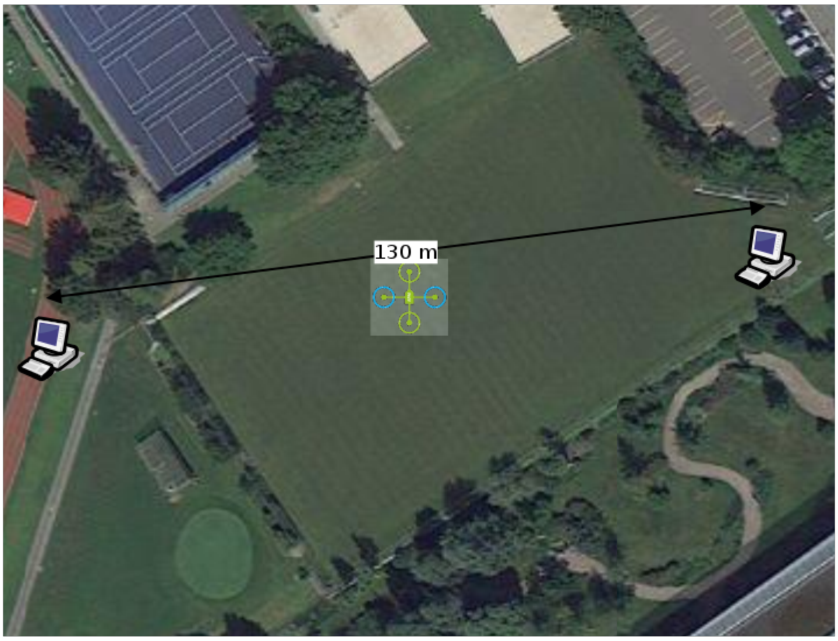


Figure 3.14: Scenario 2 - Relaying - Measurement Location

The blue histogram bars in Figure 3.15 show the number of samples for each throughput. The red line is the normal density function, with the mean value $\mu_{Throughput} = 12.5\,\mathrm{Mbit/s}$ and the variance $\sigma^2_{Throughput} = 50.5\,\mathrm{Mbit/s}$. The same median throughput in scenario 1 was achieved for distances around 30 - $70\,\mathrm{m}$. This means that the forwarding does not significantly influence throughput.
The jitter values are plotted in Figure 3.16. The mean value $\mu_{Jitter} = 10.7\,\mathrm{ms}$ is worse than for all distances in scenario 1 except the $80\,\mathrm{m}$ group which only contains 16 values. This may very well be caused by the packet forwarding. The variance is $\sigma^2_{Jitter} = 174\,\mathrm{ms}$.
The transmit retry count per second is shown in Figure 3.17 and has a mean value $\mu_{TxRetryCount} = 2.82 \cdot 10^5\,1/\mathrm{s}$ and the variance $\sigma^2_{TxRetryCount} = 2.11 \cdot 10^{10}\,1/\mathrm{s}$. The same median transmit retry count per second in scenario 1 was for distances closer than $10\,\mathrm{m}$ to the ground station. This is surprising as one would suggest that the forwarding would increase the number of failed transmits.
The median round-trip delay time $\mu_{RTT} = 49.1\,\mathrm{ms}$. This again seems caused by the forwarding of the drone but why this is about a factor of 10 higher than the round-trip delay times in scenario 1 can not be explained and has to be investigated further. The variance $\sigma^2_{RTT} = 11.2\,\mathrm{ms}$.
The link quality has a median of $\mu_{LinkQuality} = 64.6\,\%$ and a variance of $\sigma^2_{LinkQuality} = 581\,\%$. In scenario 1 the $50\,\mathrm{m}$ group has about the same median.
The mean radio signal strength indicator value $\mu_{RSSI} = -64.8\,\mathrm{dB}$ which is about the same as for the $20\,\mathrm{m}$ group but since the values are either $-80\,\mathrm{dB}$ or $-50\,\mathrm{dB}$ the radio signal strength indicator is in this case not well suited to give prediction of the actual throughput. Thus the variance is $\sigma^2_{RSSI} = 283\,\mathrm{dB}$.
Table 3.1 summarises the results.

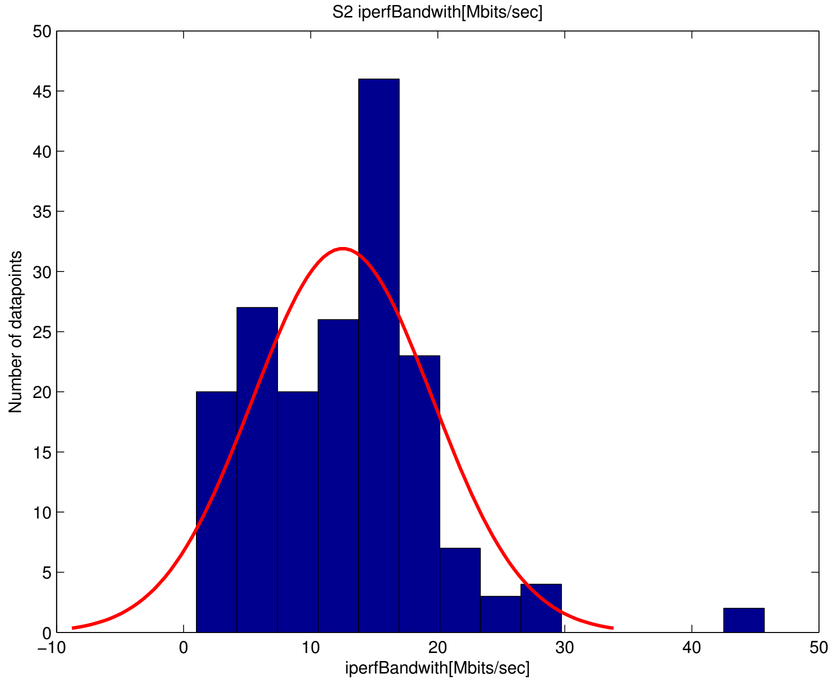


Figure 3.15: Scenario 2 - Relaying - Throughput

Variable	Mean μ	Variance σ^2	S1 Group
Throughput	$12.5\,\mathrm{Mbit/s}$	$50.5\,\mathrm{Mbit/s}$	30 - $70\,\mathrm{m}$
Jitter	$10.7\,\mathrm{ms}$	$174\,\mathrm{ms}$	$80\,\mathrm{m}$
TxRetryCount	$2.82 \cdot 10^5\,1/\mathrm{s}$	$2.11 \cdot 10^{10}\,1/\mathrm{s}$	$10\,\mathrm{m}$
RTT	$49.1\,\mathrm{ms}$	$11.2\,\mathrm{ms}$	-
Link Quality	$64.6\,\%$	$581\,\%$	$50\,\mathrm{m}$
RSSI	$-64.8\,\mathrm{dB}$	$283\,\mathrm{dB}$	$20\,\mathrm{m}$

Table 3.1: Scenario 2 - Results

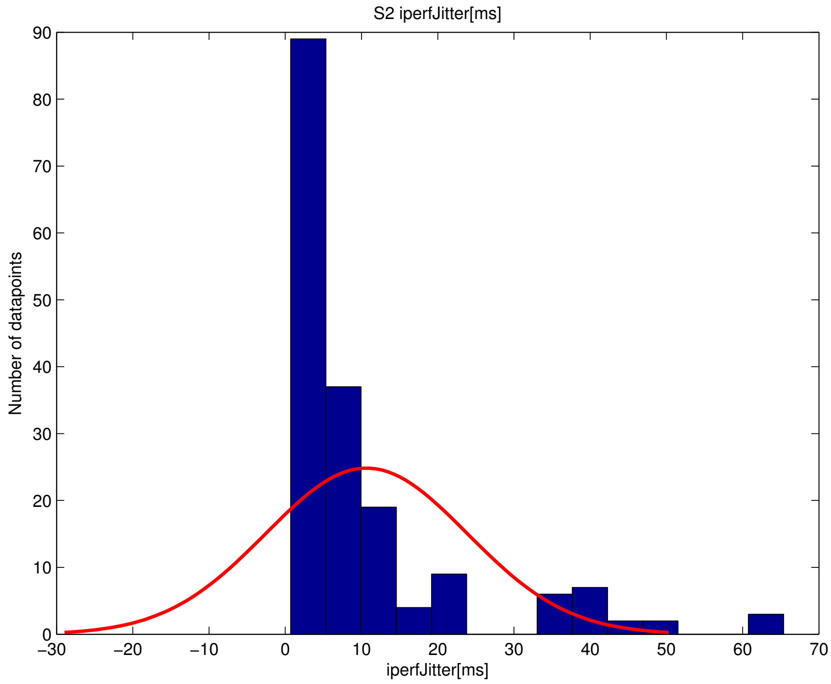


Figure 3.16: Scenario 2 - Relaying - Jitter

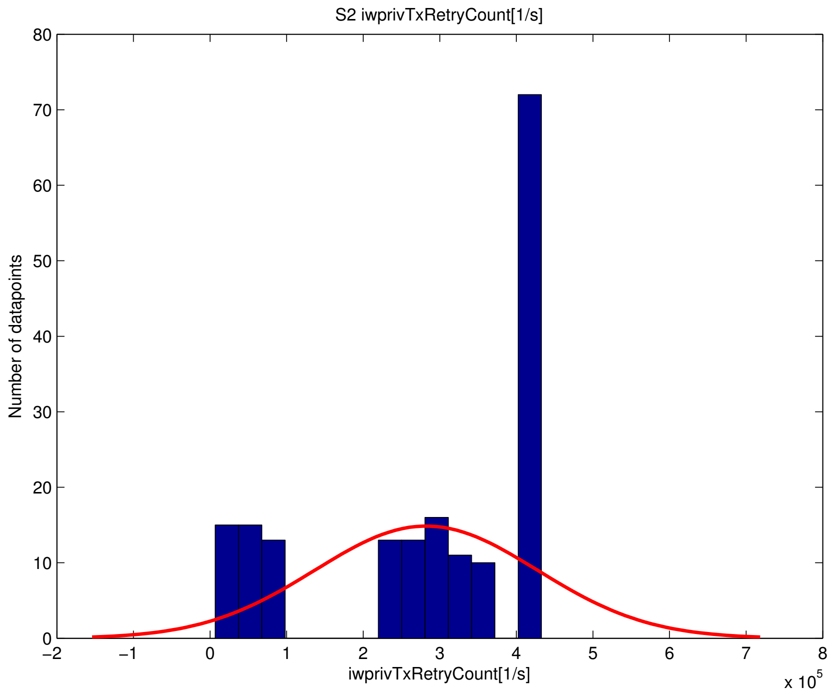


Figure 3.17: Scenario 2 - Relaying - TxRetryCount

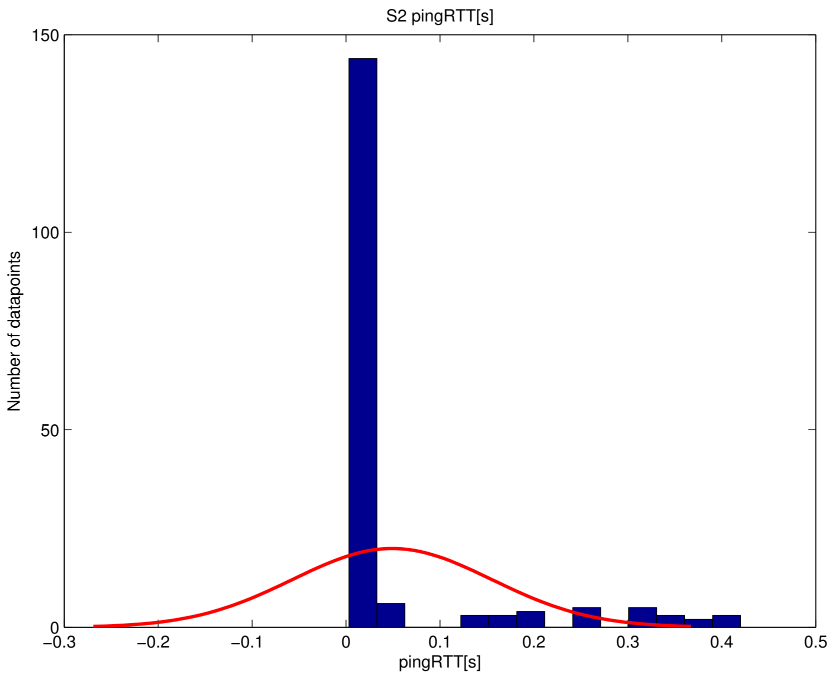


Figure 3.18: Scenario 2 - Relaying - Round-Trip Delay Time

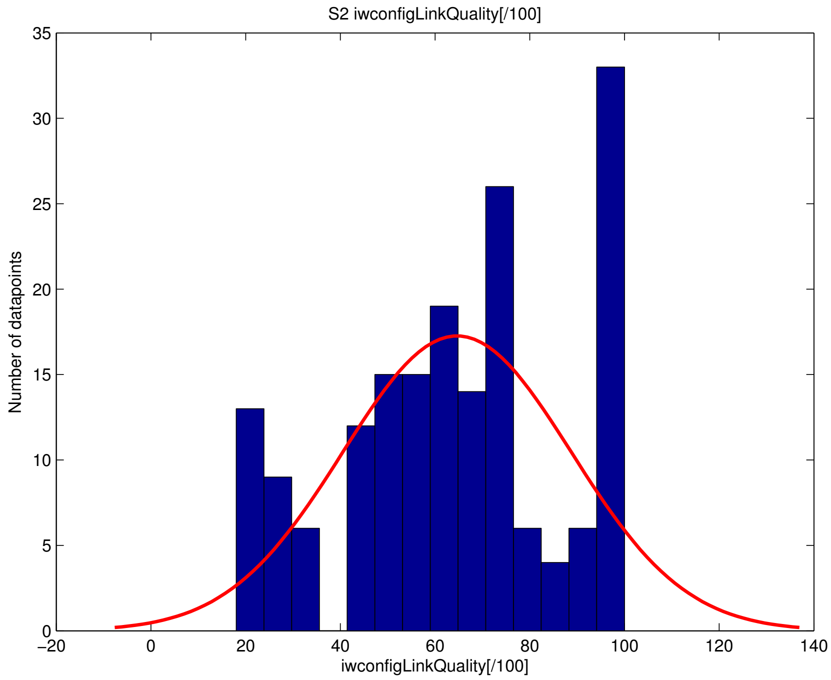


Figure 3.19: Scenario 2 - Relaying - Link Quality

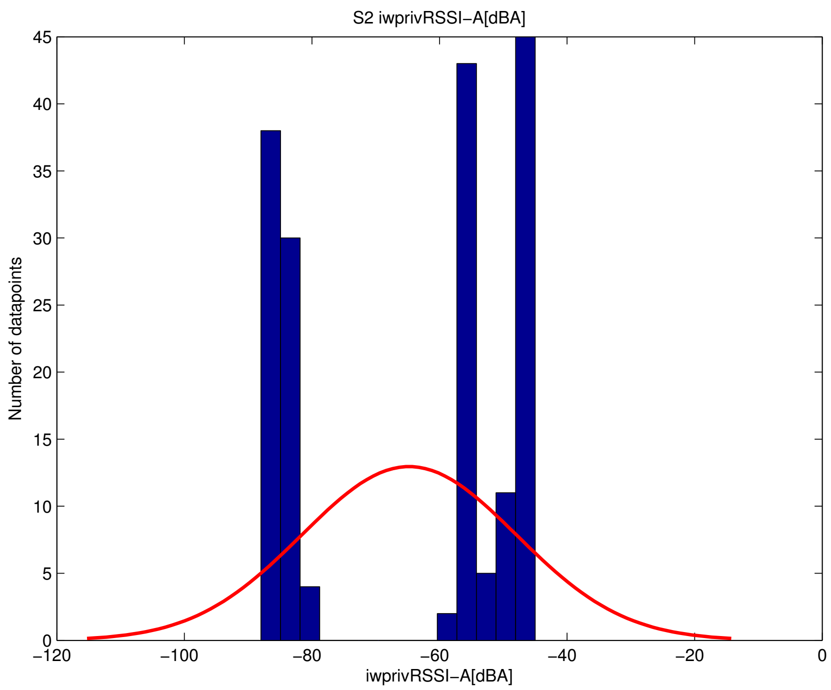


Figure 3.20: Scenario 2 - Relaying - Radio Signal Strength Indicator

3.3 Scenario 3: Load-Carry-And-Deliver LCAD

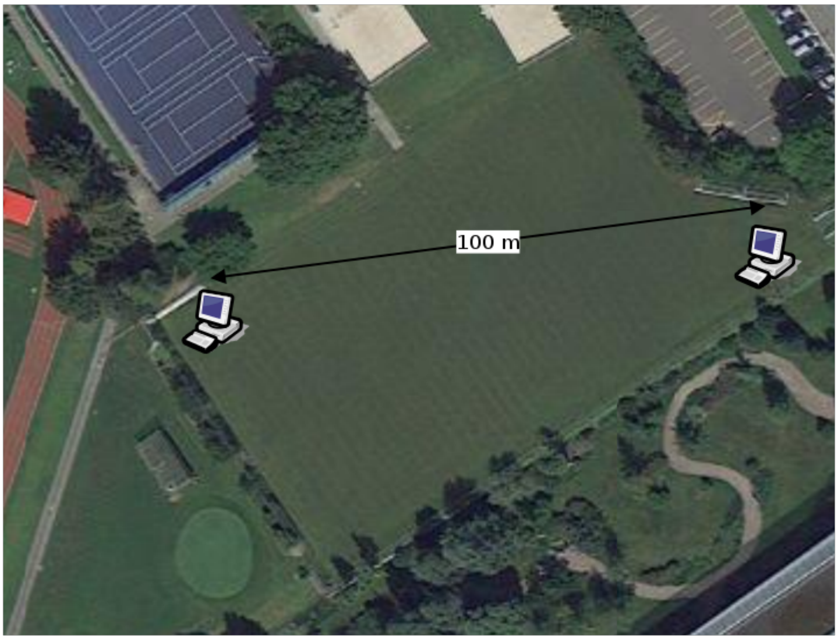


Figure 3.21: Scenario 3 - Load-Carry-And-Deliver - Measurement Location

Scenario 3 was also accomplished at the football grounds near the Zoo in Zurich which is shown in Figure 3.21. The ground stations are $100\,\mathrm{m}$ apart from each other. There is a big fence on the left end of the football ground on the picture which would have made it considerably harder to fly, therefore a distance of $100\,\mathrm{m}$ was chosen.
In this test a file of $b = 10\,\mathrm{MB}$ has been uploaded to the drone, carried to the other ground station and delivered. It took about $t_{tx} = 5s$ to upload the file using the Linux command scp. This results in a load rate $R = 16\,\mathrm{Mbit/s}$.
The real transfer times with the averages are listed in the table 3.2. There are six measurements in total and the average values. The time when the start of each measurement procedure is given. Each procedure consists of running a reference measurement with iperf using the same settings as in the other scenarios thus it is a UDP transfer. These transfer rates are called R_{iperf} in the table below and have an average of $34.18\,\mathrm{Mbit/s}$. Then the actual $10\,\mathrm{MB}$ file is uploaded. scp reports the times which can be found as t_{tx} in the table. The average of these times is $4.51\,\mathrm{s}$. Using this time and the file size the scp rates R_{scp} are calculated and are on average $17.92\,\mathrm{Mbit/s}$. The reason for the difference in transfer rates are due to the overhead of TCP and SCP protocol overheads. The difference in start times is half of the cycle time $t_{\frac{cycle}{2}}$.

Direction	Time	$t_{\frac{cycle}{2}}[s]$	$t_{tx}[\mathrm{s}]$	$R_{scp}[\mathrm{Mbit/s}]$	$R_{iperf}[\mathrm{Mbit/s}]$
Up	17:23:11	-	4.6	17.4	36.6
Up	17:28:40	-	4.8	16.7	36.1
Down	17:29:11	31	3.6	22.2	50.0
Up	17:29:41	20	5.0	16.0	25.8
Up	17:29:58	-	4.8	16.7	66.5
Down	17:30:28	30	4.2	19.0	13.8
Up	17:31:12	44	4.9	16.3	30.8
Down	17:31:38	26	4.2	19.0	13.8
Average		30.2	4.51	17.92	34.18

Table 3.2: Scenario 3 - Transfer Times and Rates

Since this is all done manually (starting the reference measurements, the file transfer and even flying), the shortest time $\min(t_{\frac{cycle}{2}})$ can be used as a parameter for calculations

$$t_{travel} = \min\left(t_{\frac{cycle}{2}}\right) - t_{tx} \approx 15\,\mathrm{s} \tag{3.1}$$

This leads to the drones speed of

$$v = \frac{l}{t_{travel}} \approx 6.7\,\mathrm{m/s} \tag{3.2}$$

With the communication radius $r = 0\,\mathrm{m}$ and the number of drones $n = 1$ equations 1.3 and 1.5 reduce to

$$T = \frac{b}{2\left(\frac{l}{v} + \frac{b}{R}\right)} = \frac{b}{2(t_{travel} + t_{tx})} = \frac{b}{2 \cdot t_{\frac{cycle}{2}}} \tag{3.3}$$

Thus having the distance $l = 100\,\mathrm{m}$, the transmit rate $R = R_{scp} \approx 17.92\,\mathrm{Mbit/s}$ and the buffer size $b = 10\,\mathrm{MB}$ respectively half of the cycle time $\min(t_{\frac{cycle}{2}}) \approx 20\,\mathrm{s}$ the calculated throughput is about $2\,\mathrm{Mbit/s}$.
Since the microSD card is $2\,\mathrm{GB}$ and the size of the operating system only $500\,\mathrm{MB}$ the buffer could be increased to $1.5\,\mathrm{GB}$ and thus in theory have a much better throughput of about $7.6\,\mathrm{Mbit/s}$.

Chapter 4

Future work

Since the measured throughputs with the USB WLAN stick are not optimal it is assumed that using some other setup like a miniPCI WLAN adapter with external antennas could improve the throughput significantly.
There are some results that couldn't be explained like the better connection at the further turning point. This could also be investigated closer to find the cause for this.
With the communication model in place it would be possible to use the actual RSSI value or the monitored throughput, jitter, round-trip delay time or whatever parameter is important for the former use case to calculate the best position of each drone.

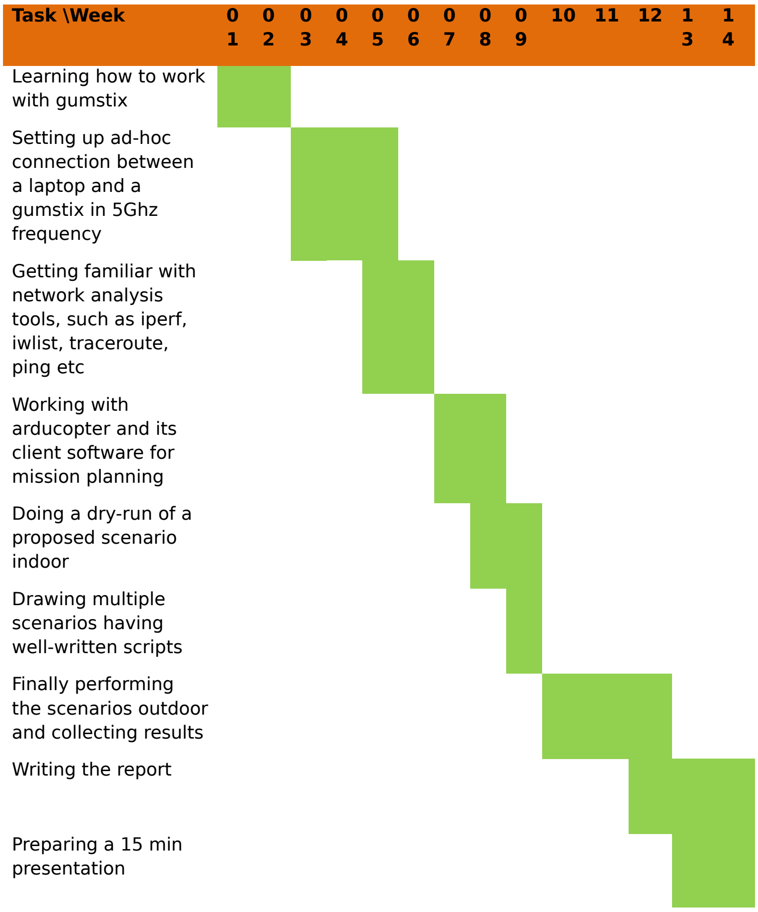
Chapter 5

Summary

This work concentrated on creating a toolbox to provide some parameter data for an 802.11n ad hoc communication in the $5\,\mathrm{GHz}$ band. An Arducopter and a Gumstix minicomputer was provided and commissioned to achieve the goal.
The measured data was analysed and some parameter data listed in the Result chapter 3.
The best estimator for the communication is probably the RSSI (radio signal strength indicator) provided by iwpriv. Also it has been shown that distances below $80\,\mathrm{m}$ have more influence on the connection than speeds below $10\,\mathrm{m/s}$.

Appendix A

Timetable



Appendix B

Original Problem

Institut für Technische Informatik und Kommunikationsnetze
Computer Engineering and Networks Laboratory

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Schweizerischer Nationalfonds zur Förderung der wissenschaftlichen Forschung

UAVs in Rescue Mission Networks

Master or semester thesis for a student in department D-ITET/D-INFK

Rescue missions require timely and flexible communications operating even in lack of infrastructure networks. In the SWARMIX project, we investigate the interactions of heterogeneous agents on a search and rescue mission. A swarm comprises rescue professionals, dogs, and UAVs (Unmanned Aerial Vehicles) cooperating to find a victim as fast as possible. Communication comprises images, voice recordings, positions, and other sensor data sent from each agent back to the ground station via an ad hoc network (see Figure 1).

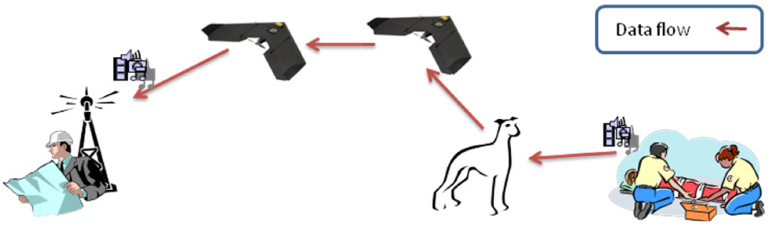


Figure 1. Receiving data wirelessly at a ground station.

In this scenario, the UAVs act as a wireless backhaul to connect all agents with the ground station. Wireless links to UAVs are expected to show different performance characteristics than ground-to-ground links due to movement and missing obstacles such as buildings. To set up a wireless link from the ground to a UAV and to perform measurements, we will use an off-the-shelf quadrocopter called Arducopter (depicted in Figure 2; more information can be found at: http://code.google.com/p/arducopter/)

Figure 2. Quadrocopter/Arducopter (source: UDRONES.COM).

The goal of this thesis is to set-up an experimental ad-hoc network with the Arducopter and to measure and analyze the effect of movement on the performance of the wireless links between multiple nodes. We will define different test scenario cases consisting of one flying robot and a few stationary nodes (laptops). Measurements will include the observation of throughput, signal quality, delay, number of disconnections etc. As one probable scenario, we use an existing ad hoc routing algorithm for the dynamic route discovery, and compare its overhead with static routing in this ground-to-air scenario.

Kind of Work: 70% practical, 30% theory

Requirements: Linux experience (network configuration, scripting), network basics

Contact Persons:

Mahdi Asadpour, mahdi.asadpour@tik.ee.ethz.ch, ETZ G96, +41 44 63 27539

Dr. Karin Hummel, karin.hummel@tik.ee.ethz.ch, ETZ H82, +41 44 632 29 76

Professor: Prof. Dr. Bernhard Plattner

Bibliography

[1] Josh Broch et al. “A performance comparison of multi-hop wireless ad hoc network routing protocols”. In: *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*. MobiCom ’98. Dallas, Texas, USA: ACM, 1998, pp. 85–97. ISBN: 1-58113-035-X. DOI: 10.1145/288235.288256. URL: http://doi.acm.org/10.1145/288235.288256.

[2] Chen-Mou Cheng et al. “Maximizing Throughput of UAV-Relaying Networks with the Load-Carry-and-Deliver Paradigm”. In: *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE*. 2007, pp. 4417–4424. DOI: 10.1109/WCNC.2007.805.

[3] Daniel Henkel and T Brown. “On controlled node mobility in delay-tolerant networks of unmannned aerial vehicles”. In: *International Symposium on Advance Radio Technologies*. Citeseer. 2006.

[4] Karin A Hummel, Alexander C Adrowitzer, and Helmut Hlavacs. “Measurements of IEEE 802.11 g-based ad-hoc networks in motion”. In: *Wireless Systems and Mobility in Next Generation Internet*. Springer, 2007, pp. 29–42.

[5] Gumstix inc. *gumstix®, dream, design, deliver®,* Mar. 2013. URL: https://www.gumstix.com/index.html.

[6] Jinyang Li et al. “Capacity of Ad Hoc wireless networks”. In: *Proceedings of the 7th annual international conference on Mobile computing and networking*. MobiCom ’01. Rome, Italy: ACM, 2001, pp. 61–69. ISBN: 1-58113-422-3. DOI: 10.1145/381677.381684. URL: http://doi.acm.org/10.1145/381677.381684.

[7] Linksys. *Linksys AE1000*. Mar. 2013. URL: http://www.linksys.com/en-eu/products/adapters/AE1000.

[8] Josiah Renfree. *Distance calculation using Haversine formula,* May 2010. URL: http://www.mathworks.com/matlabcentral/fileexchange/27785-distance-calculation-using-haversine-formula.

[9] DIY Drones development team. *Arducopter, Arduino-based autopilot for mulirotor craft, from quadcopters to traditional helis,* Mar. 2013. URL: http://code.google.com/p/arducopter/.