NOVIA
UNIVERSITY OF APPLIED SCIENCES

Development of a Quadcopter Sensor System for Measuring Thermal Profiles of the Lower Atmosphere

Samuel Kekäläinen

Bachelor's Thesis

Degree Programme in Energy and Environmental Engineering

Wolffskavägen 2016

BACHELOR'S THESIS

Author: Samuel Kekäläinen

Degree Programme: Energy and Environmental Engineering

Specalization: Energy Technology

Supervisors: Kendall Rutledge

Title: Development of a Quadcopter Sensor System for Measuring Thermal Profiles of the lower Atmosphere

Date 27.4.2016 Number of pages 39 Appendices 0

Summary

This thesis was made for the R&D department of Novia, Wolffskavägen. A temperature measurement system for measurements in the lower atmosphere (0-150m) was devised. By the use of a temperature sensor and a miniature computer strapped onto a quadcopter capable of independent flight this was achieved.

The properties of the resulting temperature measurement system were tested in both lab and real conditions. The process of putting together the temperature measurement system physically was described as were various minor software solutions to process and summarize the data as well as to operate the temperature system. Implications of flying quadcopters in the conditions required of the system were explored.

Language: English Key words: Temperature measurement, quadcopter, temperature profile

Examensarbete

Författare: Samuel Kekäläinen

Utbildningsprogram och ort: Energy and Environmental Engineering, Wolffskavägen

Inriktningsalternativ/Fördjupning: Energiteknik

Handledare: Kendall Rutledge

Titel: Development of a Quadcopter Sensor System for Measuring Thermal Profiles of the lower Atmosphere

Datum 27.4.2016 Sidantal 39 Bilagor 0

Abstrakt

Detta slutarbete gjordes för Novias FOU avdelning vid Wolffskavägen. I slutarbetet planerades och skapades ett temperaturmätningssystem för att mäta temperaturer i lägre atmosfären (0-150m). Detta var uppnått genom att sätta fast en minidator och en temperatursensor på en quadcopter kapabel till att flyga automatiskt.

Egenskaperna på detta mätningssystem undersöktes i labb och i fält. Processen hur temperaturmätningssystemet lades ihop beskrivs, och ett antal mindre program och skript menade att hantera data och styra temperaturmätningssystemet likaså. Praktiska saker man bör betänka då man flyger quadcopters i förhållanden som systemet är tänk att flyga i behandlas.

Språk: Engelska Nyckelord: Temperature measurement, quadcopter, temperature profile

Acknowledgements:

I'd like to thank the people at Quadcopter Forum for providing helpful information on quadcopters early on, and Ari Kutvonen for actually getting me off the ground, as well as just providing me with advice whenever something went wrong. The always helpful people in the Cheerson CX-20 thread over at RC Groups forums deserve special mention for being especially helpful when it comes to troubleshooting problems with the quadcopter.

Big thanks to Hans Lindén for helping me and enduring my questions about all things electrical. Equally big thanks to Dennis Bengs for his help with any questions I had about IT, often providing me with superior solutions. Another big thanks to Mats Borg for help with thermodynamics and analysis of time constants.

And of course, thank you everyone over at Novia R&D department for being helpful and friendly whenever I asked for help.

Contents

1 Introduction

The goal of this thesis work is to deliver a functioning temperature sensor for taking temperature measurements in different heights of the atmosphere. The way this task was solved was to attach a temperature sensor system onto a quadcopter and using the quadcopter as a mean to get the temperature sensor up to the desired heights.

This temperature measurement system is to be used to validate a weather model (WRF) as a part of the WindSoMe project. By comparing these temperature measurements with predicted temperatures from the weather model it is possible to see how accurate the weather model is. This is significant for WindSoMe in that the propagation of sound is greatly affected by weather conditions, and being able to use this weather model could make it much easier to take atmospheric conditions into account when modeling sound, improving the accuracy of sound modeling of for instance wind power parks.

1.1 WRF model

The WRF model (Weather Research and Forecasting model) is a numerical model for weather prediction and research. It can perform simulations of the atmosphere using either real data or idealized conditions. [1]

1.2 Difficulties faced in the project

There was several difficulties involved in creating this system, primarily when it comes to learning to operate the quadcopter itself. Nobody in the project team had any experience with quadcopters at the beginning of the project. This lack of know-how also posed the problem of the quadcopter which was chosen in the planning stages of the project not being an optimal solution.

Another large part in this project were also planning and figuring out the electronics of the sensor system. As a novice to electronics this posed a problem, but luckily easily understood guides found on the internet as well as local expertise within the school provided excellent support in creating a functioning system. The principle of keeping things simple and favoring existing solutions rather than custom-made ones also significantly reduced the workload on this front.

2 Quadcopters

Quadcopters (and multicopters in general) are unmanned aerial vehicles powered by multiple propellers which provide downward thrust. Typically they have fixed propellers, and steer themselves with the help of increased and decreased thrust. Due to the inherently unstable aerodynamics they require a flight controller and sensors (gyroscopes, accelerometer and more depending on model) to fly in a stable fashion.

[2] A commonly used term for quadcopters are drones, but drone is a term reserved for unmanned vehicles capable of autonomous flight.

2.1 Role of quadcopters in science

Quadcopters have exploded in popularity in recent years as prices have been significantly reduced, not to say that their capabilities have been increased as well. The fact that multicopters in general have become cheap, small and maneuverable also means that they have become attractive for use in certain scientific applications, one example of which being this very thesis.

Quadcopters are already used for many different purposes in different industries. In one case in the construction business a 13 story construction was leaking precast. Inspection would normally require erection of a scaffolding or to suspend someone to take pictures of the side of the building manually, but instead they used a quadcopter to take images of the area in question. By doing this an estimated $15000 was saved. [3]

Within the energy sector drones have been used for boiler and stack inspections. Usually these inspections, which are important to monitor the health of power plants, would have to be done by hoisting down personnel, who would have to contend with cramped spaces, sooty walls and a generally unpleasant work environment. A drone can provide photographs or real time footage of the inside of a boiler, one particular model climbing using a set of arms to keep it away from walls and to smoothly climb up along it. Similarly drones could also take samples or measurements of emissions from the top of a stack, which otherwise require people to climb up maybe even hundreds of meters along it to take a sample. [4]

Another place where drones are making large headway is within logistics; Amazon for instance is working on making drones capable of delivering goods straight to the doorstep. A Dutch student has designed a drone capable of travelling up to 100 km/h that is fitted with a defibrillator. He hopes that a network of drones like these could be used to reduce the time taken for defibrillators to reach their patients from 10 minutes down to 1, which might improve the survival chance of a cardiac arrest from 8% up to 80%. [5]

The use of drone in science would be similar to all of the above. Aerial photography and pictures of terrain is much cheaper with drones than from a helicopter or a satellite, measurements or samples from hard to get places can often be gotten using a drone provided that it the equipment necessary to take these can be carried by that drone, and delivering equipment and supplies up a hill or mountain or even just through rough terrain might very well be done most cost effectively with a drone. The sky is the limit when it comes to use in drones, and as more people and sectors become aware of the versatility of drones the list of uses of them will keep on expanding.

2.2 Cheerson CX-20

The model of quadcopter used in this thesis work is the Cheerson CX-20, which can be seen in picture 1. Its intended use as a camera platform along with the relative ease of

use, the open source nature of its firmware and low price made it the chosen model in this thesis work. It was also the cheapest possible option which had the capability of autonomous flight, with the possible exception being using an even cheaper quadcopter kit. However, this was not something that expertise existed for within the project.

Despite being the cheapest and simplest option work on it was not without problems. This particular is popular and therefore has information on it available online from hobbyists, but official documentation for the quadcopter was next to non-existent. Finding solutions to problems therefore meant looking at seeing what other hobbyists have done and proven to work. This made modification somewhat difficult. As is true for most quadcopters its size it cannot be flown in moderately high winds. The Cheerson CX-20 is very much a hobby-grade drone, made for intermediate users rather than complete beginners like everyone involved in the project was at the beginning of the project.

Picture 1: Picture of Cheerson CX-20, front view.

2.2.1 Technical specifications

Some notable features of the Cheerson CX-20 contains are a 6-axis gyroscope, transmitter range of approximately 800 meters, a maximum speed of 8 m/s, a return to home function and an automatic function, which can be used to carry out a preprogrammed flight routine. [6]

The CX-20 also contain a flight controller (a modified Arudpilot Mega 2.5 flight controller) with an integrated barometer, an electronic compass, a GPS receiver and four electronic speed controllers, one for each motor. These come standard in any quadcopter capable of autonomous flight but the reason why these are notable is due to the fact that many of these require calibration at some point or another.

2.2.2 Calibration of quadcopter sensors

Calibration of the sensors used in quadcopters (specifically quadcopters which use the arducopter firmware, which is explained in more detail in chapter 3.2) can be done either manually by putting in various inputs on the transmitter, or in a so called ground station software. A ground station program has capabilities of modifying quadcopter control system parameters, change the firmware, set a mission for the quadcopter and download flight logs. Mission Planner was the ground station of choice in this project. Any unstable flight which is not caused by faulty component is likely to be able to be corrected by recalibration; although this is not the place for a detailed troubleshooting an example of this would be that if the quad is flying around in a circle (a term called toiletbowling) when set to hover in place it is likely a compass issue.

For autonomous flight (and especially when an additional weight is added to the quadcopter, like in this case) it is important that two additional calibrations are done in addition to the standard ones above; autotune and autotrim.

Autotrim is a process which automatically tries to find the correct trim to ensure that the quadcopter is flying stable. This is done by activating the mode, then keeping the quadcopter as straight as possible manually. Autotune sets the P and D (proportional and derivative) parameters in the control system. It is activated by flying around in altitude hold mode then sending a high signal through the transmitter. After this the quadcopter will begin to make sharp turns in different directions as to see where the safe limits for the P and D parameters are, and will save these limits automatically. It is highly recommended that both of these processes are to be done on a windless day. [7] [8]

2.3 Practical considerations of flying a quadcopter

There are several considerations that need to be taken into account when doing measurements. First off is the weather. The ability for a quadcopter to fight against the wind is different from model to model, but a rule of thumb is that you should not fly in wind speeds higher than the quadcopters max speed. After further taking into account that the wind speeds significantly increase as the altitude increases due to a diminishing ground effect it is only during relatively mild winds that it is possible to fly up high to take measurements with the CX-20. This unfortunately limits the conditions in which data can be collected.

Another thing to be taken into consideration for year-round measurements is the fact that moisture can damage electronics. In the case of the CX-20 it is quite protected by its plastic body, with the exception of the electronic speed controllers who got open air slots immediately over them, which are highly sensitive to moisture. It is possible to use different protective sprays for electronics that protect from moisture and has proven to protect electronics used in quadcopters and model vehicles in general, but nevertheless it is by no means recommended to fly in mist, rain or snow. Moisture should be kept in mind during wet conditions (primarily spring and fall) but also during winter, where a landing or crash in deep snow could cause snow to go into the body through exposed parts and then melt quickly against hot electronics.

During winter cold temperatures are also a problem. For the operator keeping your fingers warm is actually more than just a matter of comfort as most of the steering is done through tactile sense, and if the operator's fingers go numb they lost the ability

to judge how they are holding the control sticks. Furthermore batteries also lose power in cold weather as the reactions of the battery happen more slowly with a lower temperature, which translates into shorter flight times. By keeping the battery warm before use this can be counteracted to a large extent, which will then keep warm during use. Although this is to be kept in mind year round in different weather conditions visibility can be an issue during winter, especially if it is snowing [9]. The stock color of the CX-20 is white which makes it more difficult to keep track of the quadcopter on a bright day with snow, especially if on top of that the sky also got a white hue as opposed to a blue one.

Although the CX-20 is able to fly at least half a kilometer up into the air (though possibly some modification is required for that), there are significant challenges involved in flying up high. Judging at what points that the quadcopter is losing altitude (whether it is intentional or not) becomes difficult with only visual line of sight, but is not as big a problem as judging which direction the quad is moving in. Orientation, that is which way is forwards in the view of the quadcopter is important to keep track of as it not possible to tell with the naked eye once the distance becomes large enough. If you lose track of orientation in high altitude flights it is next to impossible to see whether the quadcopter is moving backwards or forwards, though moving to the sides is easier to spot, but with high winds at high altitudes there is also often limited time to regain control of the craft before it drifts off out of range and/or line of sight. There are automatic fail safes and the return to launch function that can be used in situations like these, and much of the difficulties of flying in high altitudes will be reduced by letting the quad fly itself, but a backup plan is required for the times where something goes wrong and you cannot rely on these. It is vital that you must be able to reel the quadcopter back in with manual flight.

Vortex ring state (VRS) is another issue to keep in mind. The effect can happen to more or less any aerial vehicle which uses helicopter-like propellers for thrust. VRS occurs during a vertical descent, and what happens is that as you descend into the turbulent air under the propellers they lock up, and then no matter how much vertical thrust you add the quadcopter will inevitably descend and crash. For quadcopters this can be seen as a wobble as it descends. To counteract this state you can instead give the quad full forward pitch and thrust, which hopefully will move you away from the turbulent air underneath the propeller. [10] In automatic flights VRS is a non-issue as quadcopters controlled by arducopter only descends slowly by default, which it does to eliminate the risk of vortex ring state.

2.3.1 Operational flight envelope for the sensor system

Although the real limits of the system is untested due to the obvious risks of pushing the envelope, a rough flight envelope can be summarized as follows: in the current state the quadcopter cannot be safely used at high altitudes, the legal limit of 150 meters providing a convenient upper limit (which can be read more about in chapter 2.3.1). The system may not be used in any and all precipitation or mist, and deep snow should be avoided. Winds of over 8 m/s at the altitude of flight should be avoided, the system having only been tested in conditions of roughly 3 m/s or lower at ground level with an unknown wind speed at higher altitudes. Weather conditions with poor visibility should not be flown in.

2.3.2 Legal

In Finland the use of quadcopters and other remotely piloted aircraft systems (RPAS) are under regulation. The Finnish regulation makes a distinction made between model aircraft which are used for hobby, and unmanned vehicles with professional use. The same quadcopter can be defined as one or the other depending on use. The most important legislation for our use is the Aviation Act (853/2014) and Trafi's response on it, OPS M1-32. [11]

All flights must be based on visual line-of-sight (vlos), and the vision must be good enough for air traffic and other obstacles to be identified in time to dodge it. The maximum height you are allowed to fly is 150 m above the surface or water level, though it is possible to gain permission to fly higher as well as to fly outside of vlos, provided you can do so safely. Permission is also necessary to fly above crowds of people. Model aircraft and unmanned vehicle always gives way to other air traffic. [12] [13]

In flights exceeding 150 meters height it is necessary to reserve airspace. However, as this is a lengthy process (requiring a minimum of 10 weeks to process) as well as having a significant fee to inscribe a maximum of 2 weeks of reserved airspace this was determined to be unfeasible. This, along with the technical challenges of flying too high meant that the maximum flight altitude was limited to 150 meters, meaning that no special permissions were required to make the flights. Similarly the flights cannot reasonably be qualified as professional to their extent as there is nothing done in excess of what is allowed in hobby flights.

3 Sensor system hardware

To take the measurements a sensor system was developed. Temperature measurements were taken with a MCP 9808 temperature sensor, using a Raspberry Pi B+ (a miniature computer) as a microcontroller by an I2C connection. To measure the altitude of the quadcopter the internal logging capabilities of the quadcopter was utilized. These measurements are in turn taken by the BMP 85/180 sensor included in the APM2 flight controller inside the Cheerson CX-20 [14]. To get the hardware ready for use in the temperature system it was necessary to both figure out the electric connections as well as devise a way to hold the system as a whole onto the quadcopter.

3.1 Accuracy of sensors

MCP9808 is a high accuracy sensor, which is in the application given is important given that the temperature differences that are to be measured are quite low. In its data sheet its typical accuracy is +-0.25°C (from -40°C to +125°C), a maximum deviation of +-0.5°C from-20°C to 100°C and a maximum deviation of +-1°C from 40°C to +125°C. [14]

A BMP 85 or a BMP 180 pressure sensor is used by the quadcopter to calculate its current altitude [15]. This is done by comparing the pressure at sea level with the current pressure. Its accuracy is reported to typically be -1.0+-1 hPa. The international barometric formula can be used to calculate altitude:

$$altitude = 44330 * (1 - (\frac{P}{P_0})^{\frac{1}{5.255}})$$

where P is the current pressure and P_0 is the pressure at sea level. This corresponds to a curve of altitude as a function of barometric pressure which looks like Figure 1. [16]

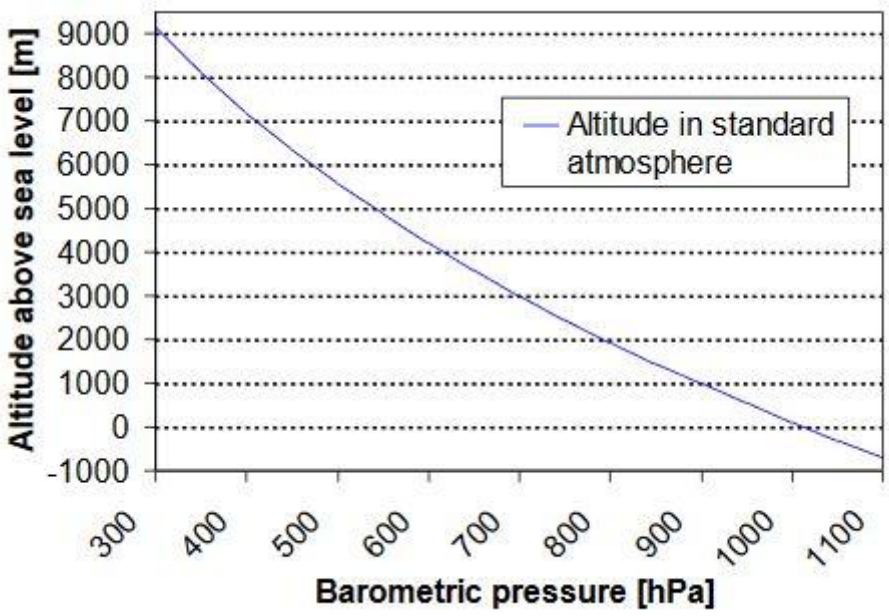


Figure 1: A graph of the relationship between altitude and barometric pressure. [16]

3.2 Electrical connections, power

A micro-USB port is used to provide power to Raspberry Pi's. The developers used this connection as a power supply which has the advantage of being a common and documented standard with some comparability between other USB standards. According to the USB standards the voltage supplied through an USB connection is 5V, which meant it was possible to supply the Raspberry Pi with power in the air using a custom connection between the micro-USB port and a 5V power/ground pin connection on the underside of the quadcopter. To do this a micro-USB cable was cut to a suitable length, the power and ground cables identified after which they were soldered onto a connector compatible with the pin connection on the quadcopter according to the labels next to the pins.

This setup has the issue that if multiple consecutive runs are desired it creates a problem with continuous power; the batteries used cannot safely be used to power more than one flight before it needs to be recharged and as the raspberry pi require a continuous supply of electricity to keep a session running (not to say there is a risk of corruption when the raspberry pi is not shut off in a safe manner) it creates the necessity to shut down the Raspberry pi after each run. To solve this, a secondary input source for power was introduced in the form of a USB port. But simply soldering another wire in parallel is not advisable when you have two regulated sources of power as both strive to achieve a balance and are affected by each other, leading to unpredictable results. To fix this a diode were added to each power line to isolate the two sources of power, the resulting electric diagram being able to be seen in Figure 2, and pictures of the power cable itself can be seen in picture 2 and 3. The regulated power sources then simply keeps itself at 5V and

whatever power the Raspberry Pi needs is drawn from either or both of the power sources.

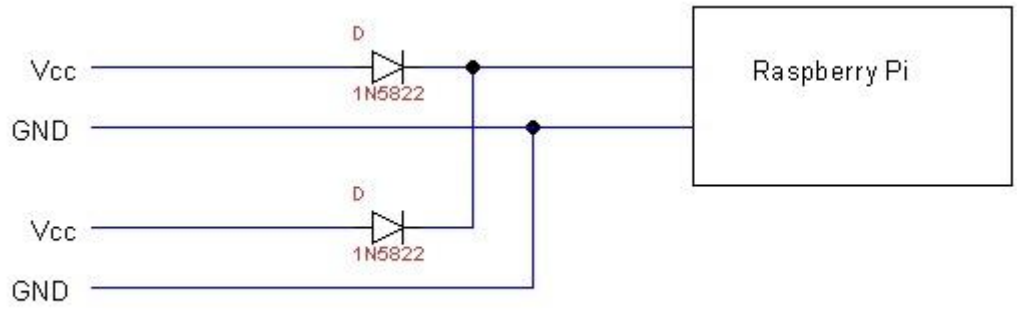
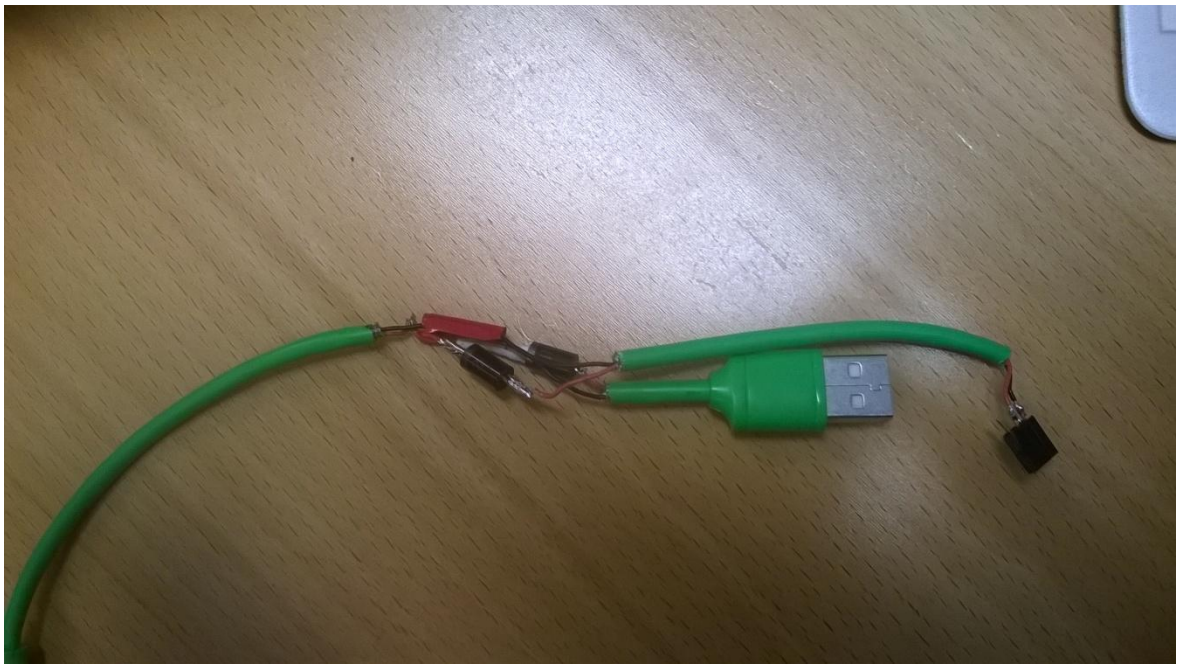


Figure 2: A schematic diagram of the power cable. On left are the entrances for power.



Picture 2: The power cable used to supply the raspberry pi with power.

Picture 3: The finished Raspberry Pi power cable. Heat shrink has been added to protect the soldering.

3.3 Electrical connections, MCP9808

The MCP 9808 uses the I2C standard to communicate with microcontrollers, which the Raspberry Pi has built in connector pins for in its GPIO (General Purpose Input/Output). A set of pins were soldered to the holes for connections on the MCP 9808 and ~20 cm long wires soldered onto the pins, which were in turn soldered onto a connector which fit onto the GPIO pins on the Raspberry Pi (see Figure 3 and picture 4).

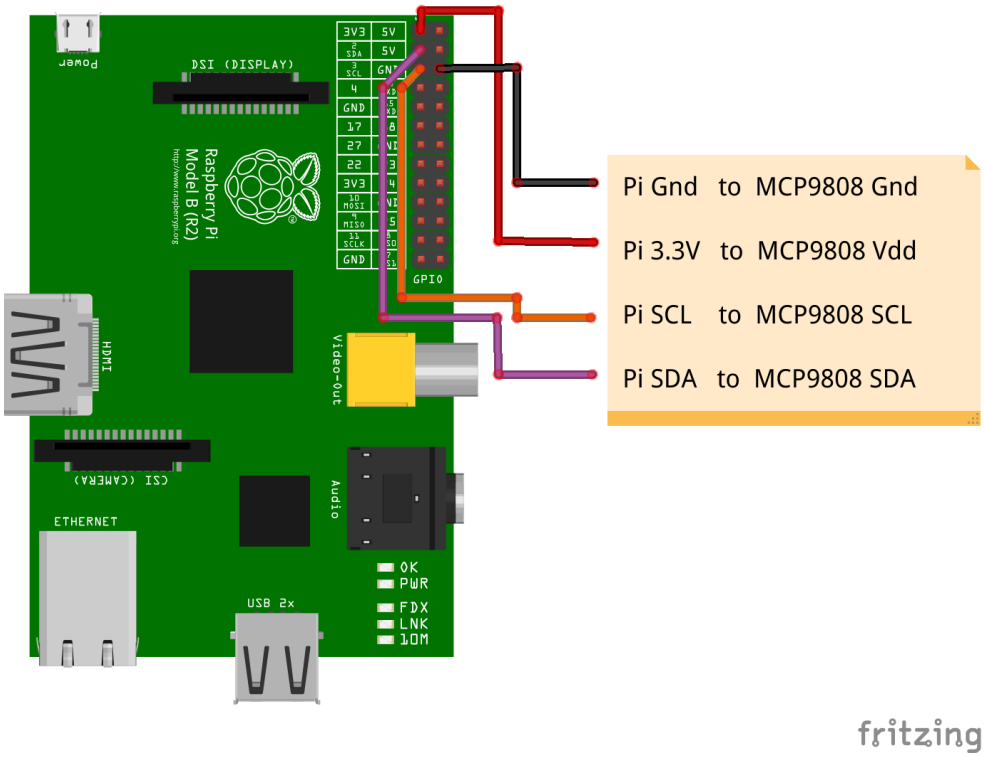
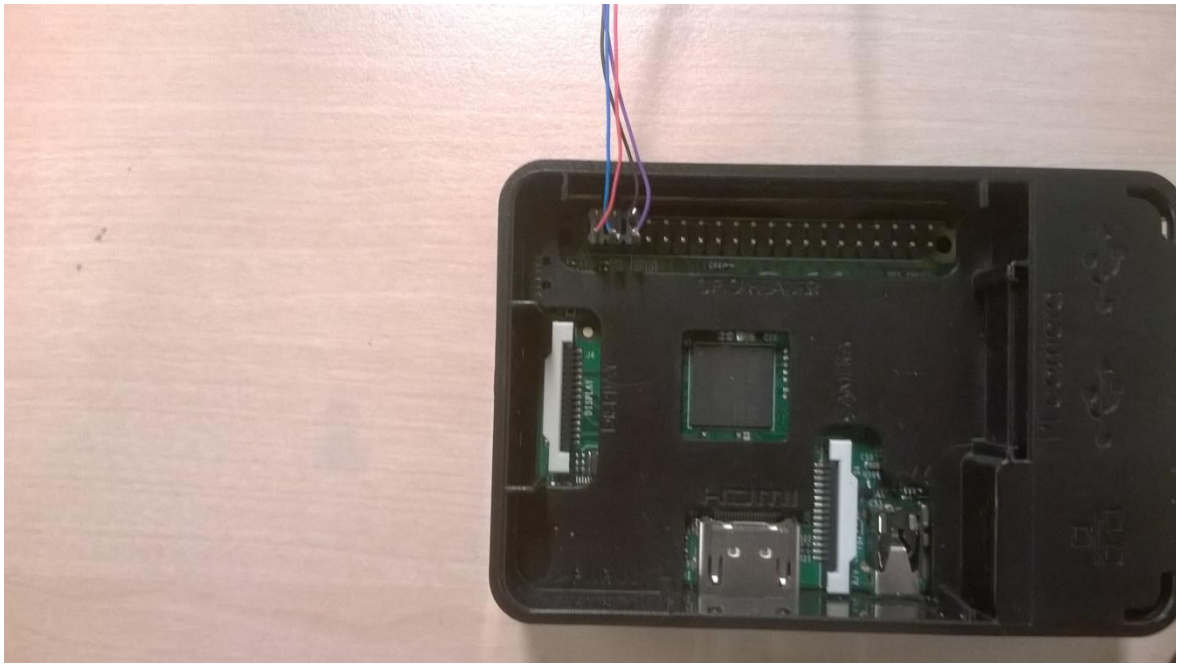
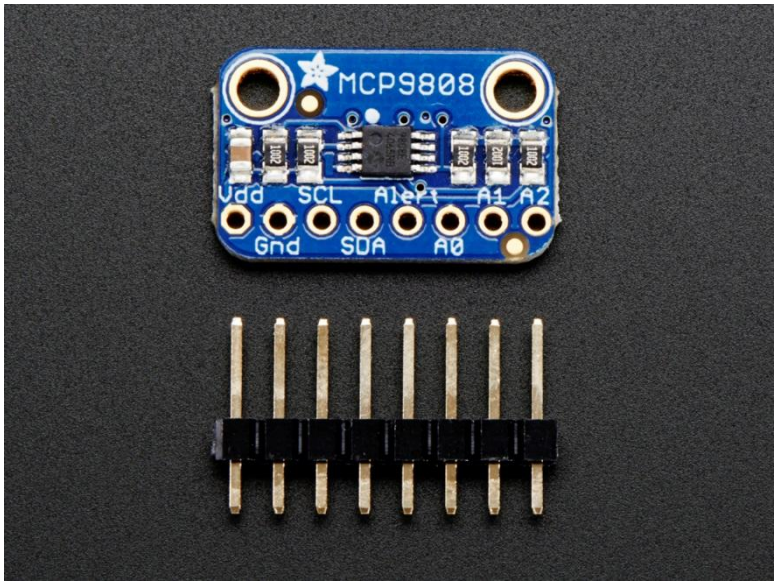


Figure 3: Schematic of the pin configuration of the raspberry pi. [17]

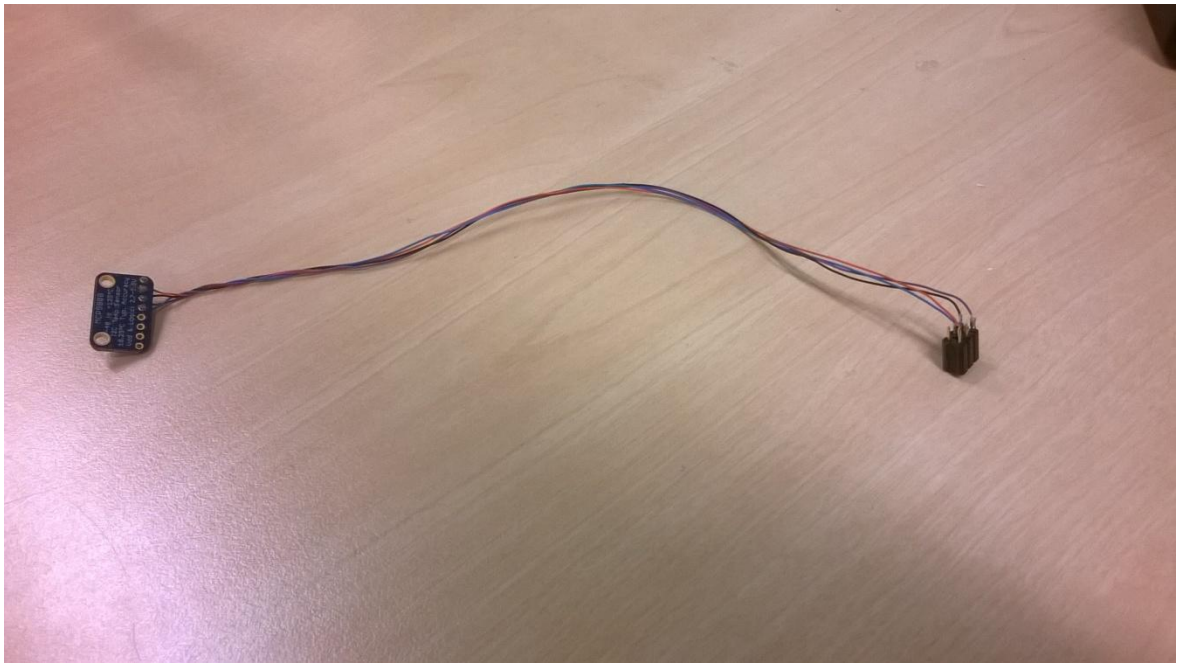


Picture 4: Detail of the connection between the Raspberry Pi in a protective case and the MCP9808 sensor.

The connectors are labeled on the MCP9808, and the equivalent GPIO pins on the Raspberry Pi were used; Vdd (power) with GPIO pin 1, Gnd with pin 6, Scl with pin 5 and Sda with pin 3. [17] See picture 5 for the sensor itself and picture 6 for the finished sensor and connector.



Picture 5: Picture of the version of MCP9808 used. [18]



Picture 6: MCP9808 with wires and connector soldered to it.

3.4 Mechanical setup

Velcro was used to hold the Raspberry Pi in place on the underside of the quadcopter during the flight, due to the ease of attachment and detachment whilst still being strong enough not to risk the Raspberry Pi falling off. This Velcro was attached to the underside of a specialized Raspberry Pi B+ case, rather than the computer card itself. To avoid vibrations and other movement from weakening the soldering that holds the sensor in place over time a small plastic tube is used to hold the wires in place under a propeller; having the sensor be in the airflow decreases the time constant for the measurements, which means that an accurate measurement can be taken quicker.

This also means that the actual temperature measured is of the air slightly above the position of the sensor (as the air above it is what is actually hitting the sensor), but as the air which takes the place of the air sucked down into the propeller tends to come from the sides rather than from straight above the effect of this is negligible [19]. For further information on the response time of the temperature system see Chapter 6.

4 Sensor system drivers and related software

In the sensor system there are two vital software systems that were worked with; Raspbian (the operating system on the Raspberry Pi) and software related to getting the temperature sensor running, and Arducopter, the firmware that was installed onto the quadcopter.

4.1 Time synchronization

As the log from the quadcopter and the log from the temperature measurements are from two discrete systems they will inevitably operate on two different timescales. Although time is not as sensitive in this application as many other there is still a need

to synchronize the two sets of data as well as to verify that the resulting synchronization is sufficient.

Arducopter logs are recorded in GPS time, which operates under slightly different rules than UTC time (see the next section). This is corrected in the Arducopter log retrieval program by converting the GPS time stamps in the Arducopter logs (in GPS weeks and GPS milliseconds) into UTC time, and recording the temperature measurements with time stamps in UTC time. The synchronization between the two log files were tested in chapter 5.

4.1.1 TAI, UTC, GPS time

TAI (from the French temps atomique international) or international atomic time is the basis of UTC time. As the name implies, it is a weighted average of over 200 atomic clocks, most of which are cesium based, which are compared using GPS satellites. [20] The TAI time scale is based on the passing of time, meaning that any inaccuracies caused by differences between how long the year is mundanely considered to be and how long it actually is are not corrected.

UTC (Coordinated Universal Time) fixes this problem of errors accumulating over time by introducing leap seconds as needed. UTC was originally formalized in 1963, but it was not until several years later, in 1972, when leap seconds were first introduced, 10 leap seconds being added at this point. Traditionally leap seconds have been added whenever UTC is mismatched from UT (Universal Time, which is based on the rotation of earth and is considered the "correct" time for UTC), but there has been discussions of abolishing leap seconds altogether. As of June 30th 2015 23:59:60 UTC 36 leap seconds have been added. [20] [21]

GPS time is mostly used in GPS satellites, and on its inception in 1980 it was set to match UTC time exactly, and is maintained to be within a microsecond of UTC time. [20] Since then additional leap seconds have been added to UTC, but not to GPS time. As of June 30th 2015 23:59:60 UTC the difference between UTC and GPS time is 17 seconds. [22]

4.2 Sensor program

The MCP9808 sensor is not plug-and-play compatible, but both guides and drivers to the sensor are available online. However, it was still necessary to create a program which would record the measurements taken.

The python library "Adafruit Python MCP9808" by Tony Dicola was used in the program, which provided all the necessary drivers and functions to communicate between the sensor and the raspberry pi [23]. The program records the measurement number (for ease of analysis within excel and in general), the temperature as well as the (UTC) time in "years-months-days hours:minutes:seconds.milliseconds" in a comma separated values format. A point is measured and recorded every second.

Due to the nature of the system it is likely that the system suffers a power failure at one point or another, which if done in the middle of writing to a file could possibly corrupt the file. In other words it is important to ensure that it is impossible to have

the program be shut off during the middle of writing which in our case is achieved by having the writing be done in a single step, so called atomic writing. Atomic writing is achieved by temporarily saving measurements in a temporary log file, and having another permanent log file. In the temporary log file 10 measurements are recorded, after which the writing to the permanent file takes place. The contents of the permanent log file is copied to a second temporary file, after which the 10 measurements from the first temporary log file is appended. This second temporary log file, which contains all previous measurements as well as the 10 new ones, is then used to overwrite the old permanent log file by using the os.replace function in python.

This achieves atomic writing by using the fact that in POSIX systems (which Debian and in extension its derivative Raspbian, the latter of which the raspberry pi runs, should be compatible with) file replacement in an operating system is done in such a way that no inconsistencies may occur [24] [25].This method has the disadvantage of being quite heavy computation-wise as all old measurements are copied over to the new file instead of just appending 10 new measurements, a problem which will only get worse as the measurement run goes on. However, in a test run of over 15 minutes the effects of this proved to be unnoticeable, and with an expected battery time of 10-12 minutes on the quadcopter the effect is not going to be seen at all, provided that the measurement system is not running for too long before actual measurements take place.

4.3 Arducopter

The firmware of the APM2 flight controller used in the Cheerson CX-20 is called Arducopter. Arducopter is a part of the open source program ArduPilot which has firmware for land and air-based unmanned vehicles, Arducopter being the version for multicopters [26]. Arducopter can with the help of a GPS receiver and sensors (accelerometer, magnetic compass and pressure sensor) control the quadcopter autonomously according to the instructions given through a so-called ground station, a program which can communicate with the quadcopter (see Chapter 2.3, measurement process) . The version of Arducopter we used was initially version 3.1.2 but later it was updated to version 3.2.1.

Arducopter also has in depth logging capabilities, logging various information useful for troubleshooting quadcopter issues, performance, flight route and the state of the various systems of the quadcopter. This all is stored onto a log file in the comma separated values format. The control system checks and records different parts of the control system at different intervals as well as dynamically according to need, rather than looping through the operations in a set order and moving on to next as soon as the previous one is finished. This makes it necessary to identify each row of values in the log file with header at the beginning of the row. We are interested in height as well as the time of the measurement, which can be found by the header “GPS” in value 3, 4 (time in GPS milliseconds and weeks respectively) and 9 (height in meters), or value 2, 3 and 8 if you exclude the header. [14]

4.3.1 Arducopter log retrieval program

To get the data we need out of the log file another short program was devised, once again in python.

In the Arducopter log file the value for time is stored in GPS time, in this case in GPS weeks and GPS milliseconds. These values are the amount of weeks since midnight the night between 1.5.1980-1.6.1980 and the amount of milliseconds since the start of the week. A notable difference between this time and UTC time is the fact that no leap seconds are added to GPS time. [27]

The principle of the program is to search for a line with the header "GPS," and with the help of a standard python package for use in csv files, copy the 3rd, 4th and 9th values for any line where it found "GPS,". The time is converted into UTC with the format of "years-months-days hours:minutes:seconds.milliseconds" and then written into a csv file along with the altitude.

To convert the time back to UTC time the function "UTCFromGps" from the python package glue.gpstime was used. The GPS milliseconds value was divided by 1000 to change it into the GPS seconds which is what the function takes as a parameter. The return value of this function is UTC time down to the second, but by taking modulo 1000 of GPS milliseconds the milliseconds part of the GPS time was retrieved (e.g. if the gps milliseconds is 120150 or 120 seconds and 150 milliseconds, then by taking 1000 of 120150 you would get 150). This was then able to be reintroduced into UTC time after it was converted. This function also takes leap seconds as a parameter, the default value being 14. The function does not automatically update or adjust the amount of leap seconds making it important for the user themselves to insert the appropriate amount of leap seconds. [28]

To make it easier to keep track of when the quadcopter has stopped hovering and moves on to its next command an additional variable was kept. Whenever the log recorded that the quadcopter moved on to the next command it was checked whether the last command was a hover. If so, the variable was set to 1 for the next recorded point in the GPS header. Under all other conditions the value of the variable was 0.

4.4 Data compilation

The data from the temperature sensor program as well as the data extracted from the log program are both in .csv format. Initially this was processed using Microsoft Excel where the data was synchronized manually and with the help of a few simple Excel functions. Later the data synchronization was automated in a Python script. As the temperature data contains a continuous series of all points of interest the program looped through those points. The time stamp of each time in the series was compared to all time stamps of the series from the arducopter log, and the point where the absolute time difference between the two were the smallest (the point from the arducopter log closest to the temperature point) was picked out. If the time difference was less than 50 milliseconds the points were taken as a match and time, temperature, altitude, whether the point was within half a second of having stopped hovering at a point and the time difference between the two were appended into a third csv file. Because arducopter records its altitude every fifth of a second (at 1.00, 1.20, 1.40 seconds etc.) and because the temperature logging program records its temperature whenever a second ticks over to the next there should not be any problem with points being recorded out of sync (in practice, for the stretches of time where

both systems are measuring, every fifth arducopter log point is connected to a temperature log point).

4.5 Operation of the temperature system

The sensor system needs to be initialized manually, but as the measurements are to be done in remote locations bringing a monitor, mouse and keyboard to use the operating system was not an option. Instead an SSH connection via an Ethernet cable is used to initialize the temperature measurement system. A set of scripts were made to simplify the initialization of the temperature program.

5 Quadcopter and sensor system syncronization test

5.1 Introduction

The quadcopter temperature sensor system used in the WindSoMe project records data in two different sources; altitude measurements from the quadcopters (Cheerson CX-20) internal system and firmware (arducopter) and temperature measurements from a Raspberry Pi and a MCP9808 temperature sensor. Due to the technical difficulties involved in either connecting a MCP9808 sensor to the quadcopter (which requires modification of the quadcopters control system) or alternatively to get the raspberry pi to take signals from the quadcopter (which requires writing custom software for the purpose and most likely making modifications to the quadcopters control system as well) it was decided to keep these two systems separate, despite the difference in timing systems between them. This experiment tests the difference between the two systems.

5.2 Theory

In the system the Raspberry Pi uses UTC as its time scale, and the quadcopter operates under GPS time. Although both use averages of atomic clocks to achieve high accuracy they are slightly different from each other. The most significant difference is of course the fact that whilst UTC time uses leap seconds and GPS time does not. This necessitates keeping track of and adding or removing the appropriate amount of leap seconds when converting between UTC and GPS time. As of June 30th 2015 23:59:60 UTC this difference is 17 seconds, meaning that GPS time is 17 seconds before UTC time. [22]

The quadcopter uses a stock GPS module to keep track of time. This is demonstrated by the fact that in the arducopter logs there are no timestamps present until a proper GPS signal is in place. Although no information on this particular model of GPS has been found e.g. Garmin brand GPS receivers time display should be “within a second” as noted by Garmin. However, the internal clock of GPS receivers is actually a lot more accurate (within a few nanoseconds) as updating the display is a low priority action. [29] It is hard to say whether or not the time stamps in the log could be as accurate as the internal clock of the GPS or whether some accuracy is lost due to the time needed to process all information from different sensors.

In the arducopter log the time parameter is stored in GPS weeks (weeks since epoch, midnight between the 5th and 6th of January 1980) and GPS milliseconds (milliseconds since midnight between last Sunday and Monday). The python package glue.gpstime has been used to convert these into UTC time.

Being a computer the Raspberry Pi can output time data in virtually any format imaginable but UTC time was chosen due to its precise and standardized nature. One thing to note about the Raspberry Pi is the fact that it has no hardware clock, but rather uses a "fake hardware clock" which is a program which keeps track of time as the Raspberry Pi is powered on; although the current time is saved whenever the Raspberry Pi is turned off it has no way of knowing how long it has been turned off.

Luckily the Network Time Protocol (NTP) package can remedy this by taking the appropriate time from the internet. By restarting the NTP package at any time when you have an internet connection it is possible to force it to update to current time (something which it might not do if there was no internet connection at boot) and this is the method used to set the time in the Raspberry Pi.

The accuracy of NTP is relatively high, a study reporting an offset of less than 100 ms between reference clocks and devices synchronized by NTP in 99.5% of the times [30]. This all is of course affected by network connections, higher latencies producing more inaccurate results but highly variable latencies being a much bigger problem than high yet stable latency. The reason behind this is the iterative process of NTP where it takes the average of all round-trips to the server and back [31].

5.3 Method

An Ethernet cable was plugged in between the raspberry pi and a laptop, and the raspberry pi was then connected to by using the Putty program. By restarting the NTP service time on the raspberry pi was updated to the current time, and was then displayed on screen by using the command 'watch –p -n 0.1 TZ=UTC date +%T%1N' in the terminal. This command displays the current UTC time in hours, minutes and seconds down to a decimal.

After the quadcopter had gotten GPS contact, which allows it uses to set its time, the experiment began. The quadcopter was held horizontally, then by hand quickly tilted over 90◦ just after every even 10 second mark (i.e. 12:56:10, 12:56:20 etc.), after which it was held horizontally again. This tilting motion is measured and logged by the quadcopters internal sensors (the accelerometer and magnetic compass in particular) whose outputs were later used to check synchronization.

After a few minutes of testing the quadcopter was shut off to ensure the test was saved in a discrete log, and after this the log was downloaded.

5.4 Results

In arducopter the log file is divided into different headers with different sets of parameters. This poses a challenge in that although the time and date is recorded precisely under the GPS-header the magnetic compass records time in milliseconds, but without any reference point; the first logged point does not even start from zero. To remedy this, a point of data under GPS-time was chosen, and the closest magnetic compass data point was taken to have happened simultaneous to this point. By

adding the difference between this first point's time value and later points a time scale for the magnetometer points was added. Note that by taking the first point as having happened simultaneously as the GPS-time point an offset of up to +-0.1 seconds (the measurement rate of the magnetic compass) is added to every point of measurement.

Aside from the time value there are three main values for the magnetic compass, MagX, MagY and MagZ. MagZ was picked out due to having the clearest difference between the horizontal and vertical state.

The results were converted into UTC time after which they were exported into Microsoft Excel. The points where the quadcopter started to be tilted were identified manually and picked out, see figure 4. This was defined as the first point which started a continuous trend of points going down to under an output of -100 in the MagZ value.

The points which were picked out as the start of tilting had all zero as the last whole digit of zero. They were in other words all within one second of the expected value of having the last digit of their number being zero (after correction with leap seconds.) A graph of the points can be seen in figure 5.

5.5 Discussion on synchronization tests

This result shows that there is a small difference to be expected between the exact UTC time (as provided by the Raspberry Pi) and GPS time. However, as this difference is under a second it will in our case prove to be mostly negligible for the purposes used in the temperature system as highly precise timing simply is not necessary.

The results received in this experiment were relatively consistent despite the human factor, but it should be kept in mind that there is also a delay in form reaction time mixed into the result. As a result of that delay the real timing should be earlier than in this experiment, but to what extent depends entirely on the reaction time of the human in question. Regardless of the actual accuracy of measurements the difference in time between UTC as provided by the raspberry pi and GPS time from the quadcopter is small enough not to provide any problems for the system as a whole.

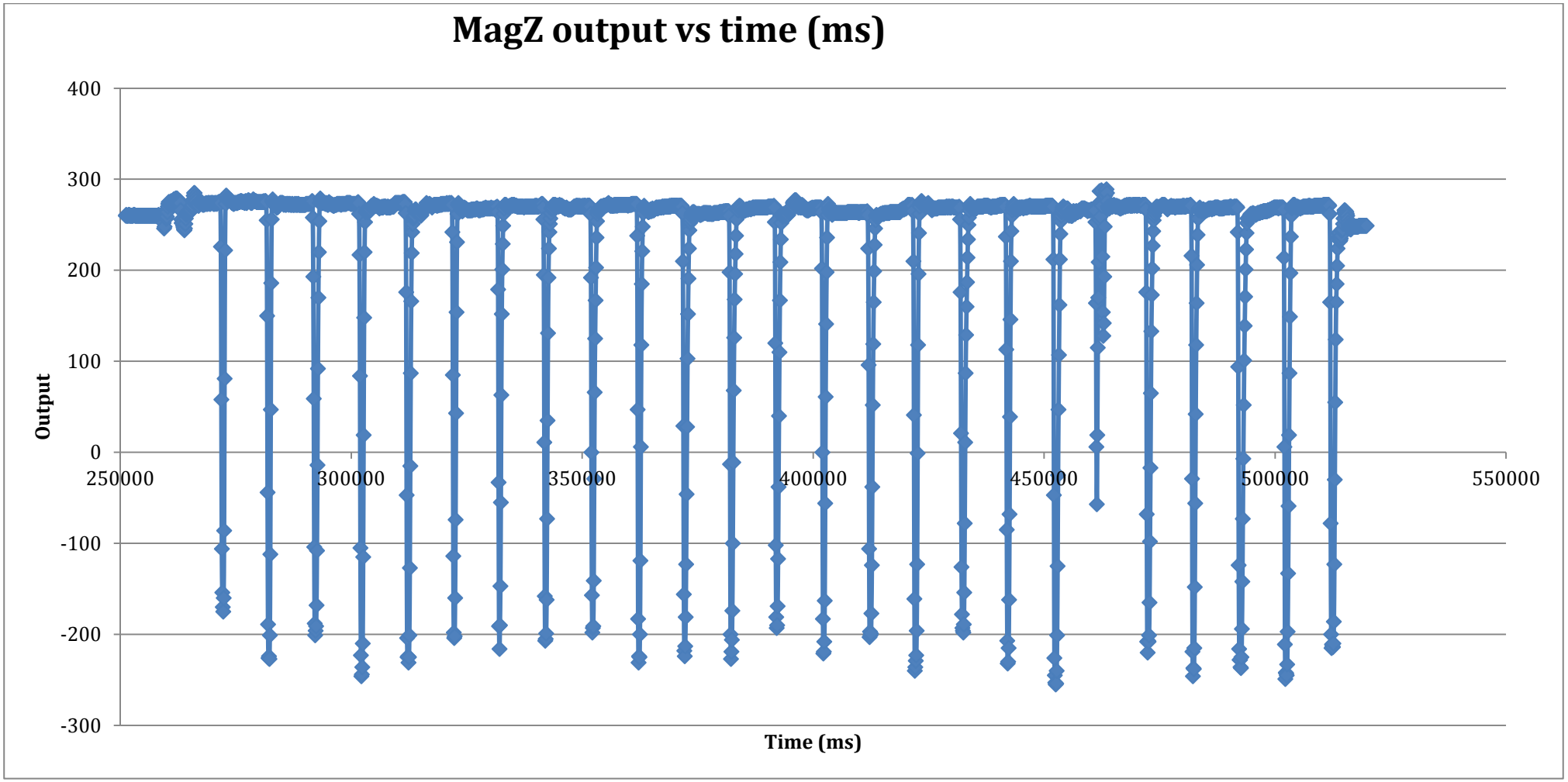


Figure 4: The output of the magnetic compass during the experiment. Note the valley that did not reach -100 (just after 450000 ms) which was omitted from the test result due to human error.

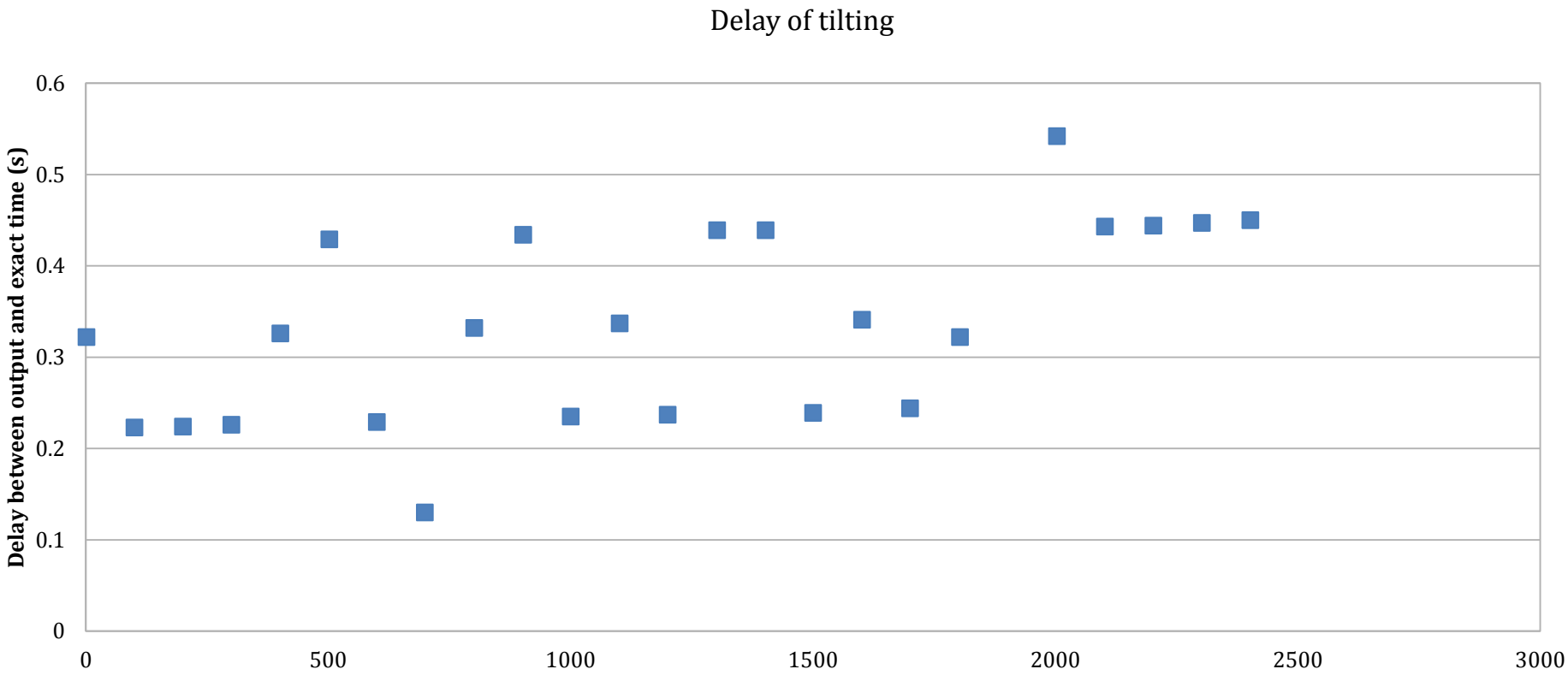


Figure 5: The decimal numbers of the picked out points in seconds. This is also how far the points are off from the exact time. Average is approximately 0.3 seconds from exact time (as provided by the Raspberry Pi.

6 Quadcopter sensor system testing

6.1 Introduction

This experiment was conducted to evaluate the properties of the temperature sensor system. Without experimentally testing the sensor system it is unknown what the accuracy of the system could be, or even if the system can be trusted to provide a correct result at all. Other than investigating the sensor system itself it was important to determine the response time of the sensor, so that it can be determined how long the quadcopter should hover in place to achieve an accurate reading. In other words the time constant for the temperature sensor needed to be determined.

6.2 Theory

The response time of a temperature sensor is typically dependent on the rate which it cools down and warms up, as most temperature sensors essentially measure their own temperature (with the exception of, for instance, infrared thermometers). In the case of the temperature measurement system detailed in this thesis work the temperature sensor is cooled/heated with the help of convection.

Convection is transfer of heat through fluids in motion. Heat is conducted to or from a fluid in contact with an object that is hotter/cooler than the surrounding. In case the object is hotter the fluid will then warm up and in turn become hotter than its immediate surroundings, and as a hot fluid (in our case air) is less dense than cold air this hotter fluid will ascend above colder fluids, which will in turn move in to take its place. As the cycle begins anew the object is then slowly cooled down to ambient temperature. This is called natural convection.

Natural convection is a slow process and takes and causes slow changes. A way to achieve equilibrium with ambient room temperature more quickly is to force fluid over the object in question, called forced convection. This greater volume of air passing over the object is much more effective at transferring heat and the protective layer of the heat gradient is also broken up by the fluid constantly flowing over the object. This significantly decreases the time taken for the object to reach ambient temperature. This process is described in Newton's law of cooling which goes as follows

$$\dot{\boldsymbol{Q}}_{\boldsymbol{conv}} = hA_s(T_s - T_\infty)$$

where h is the convection heat transfer coefficient, $A_s$is the surface area where convection takes place, T_s is the surface temperature of the object and T_∞ is the temperature of "the fluid sufficiently far from the surface" (i.e. ambient temperature). One thing to note here is that h is dependent on a number of things, such as the surface geometry, properties of the fluid and of course the bulk velocity of the fluid. [32]

With the limited flight time of the quadcopter getting an accurate measurement quickly is much preferred, and so forced convection was deliberately used in the system by placing the temperature sensor under a propeller. This increases the convection significantly by increasing the bulk velocity of the fluid over the temperature sensor.

Another formula for heat transfer in a cooling process goes as follows

$$Q = -mc\Delta T$$

Where m is the mass of the object, ΔT is temperature change of the cooled object, c the specific heat and Q the heat transferred. By derivation with the respect for time we can get a formula for heat transfer over time

$$\dot{Q} = -mc\frac{dT}{dt}$$

When an object is cooled or heated up the rate of temperature change slows down as the object which changes temperature approaches ambient temperature (see figure 6). If we combine the formula above with Netwon's law of cooling we can get an expression the temperature at any point in time of this process as we first get the following formula

$$\frac{dT}{dt} = \frac{-hA_S}{mc}(T - T_\infty)$$

The solution of this differential equation is

$$T = T_\infty + (\,T_{start} - T_\infty)e^{-\frac{hA_S}{mc}\cdot t}$$

In the above equation for the time constant (τ) of a temperature cooling process over time is

$$\tau = \frac{mc}{hA_S}$$

with this inserted in the formula above and after some further processing you end up with

$$\ln(T_{end} - T) = \ln(T_{start} - T_\infty) - \frac{1}{\tau}\cdot t$$

where T is the temperature at the calculated point, T_∞ is the temperature at the end point (or ambient temperature in this case), T_{start} is the temperature at the start, t is time and τ is the time constant.

This equation is only valid for processes where the ambient temperature is constant, in which case the graph of the process corresponds to a line. To get the equation of this line we set $\ln(T_{end} - T)$ to equal to y, $\ln(T_{end} - T_{start})$ equal to k, t equal to x, and $-\frac{1}{\tau}$ equal to a constant, m. We can then from the equation of m extract τ like this

$$-\frac{1}{\tau} = m \Leftrightarrow \tau = -\frac{1}{m}$$

In other words, if we know the line, we can then numerically calculate m, which we can then use to calculate the temperature constant τ. Although this particular equation assumes that the end temperature is lower than the start temperature, it is possible to calculate the reverse situation by reversing the sign of y and m.

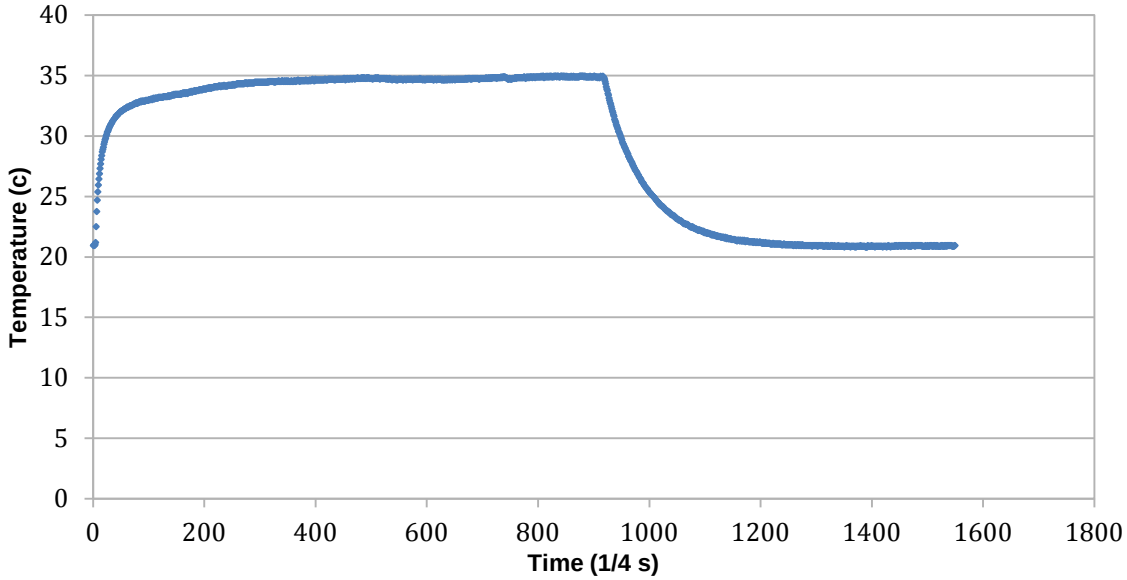


Figure 6: A typical temperature curve from the experiment using the method described below.

6.3 Method

The laboratory tests were made in multiple iterations. The equipment used was, aside from the sensor system (raspberry pi, MCP9808 sensor with a plastic tube as support, quadcopter, mouse, keyboard and monitor as interface) a tripod and a hotwire anemometer (Testo 435-4).

The quadcopter was affixed to the tripod using straps after which the raspberry pi was attached to the quadcopter's landing gear using plastic straps. The sensor was attached to the plastic tube using tape, which in turn was taped onto the quadcopters wing to affix the sensor under the propeller blades, in the location where the wind was at its strongest.

The experiment was improved upon as it went along, meaning that there are some differences in methods during different iterations. The basics remain the same; first off the measurement of the temperature was started and the sensor was heated using a finger. When the temperature was stabilized with that of the finger the sensor was let go, and fanning with a propeller or a fan was started. In case of control tests no fanning was applied. The measurement data was recorded for some time after the temperature was stabilized to ensure that no further changes were going to happen to the temperature.

As mentioned earlier the experiment underwent several different iterations where a series of tests were done; the first few tests were done by holding the hotwire anemometer manually. After that it was rested in the same place on top of a flat surface, and in the iteration after that it was held in place by taping the anemometer against a wooden board. The legs were unstable during all of these tests however, and so in the iteration after that the legs were held in place using tape (the setup can be seen in picture 7 and 8). In this iteration the wind was not measured at all, rather the thrust of the quadcopter was set to a constant output instead.

Picture 7: Picture of a later stage of the experimental setup, front view.

Picture 8: Picture of a later stage of the experimental setup, top view.

After the laboratory test a field test was made in conditions that the system would face in real measurements. Using automatic flight the quadcopter was made to fly up to a height of 100 meters where it would hover for 2 minutes. After that it would descend down to 50 meters where it hovers for an additional 2 minutes. This should ensure that the sensor is given enough time to reach ambient temperature.

6.4 Results

To calculate the results numerically, the natural logarithm of the difference between ambient temperature and the current temperature for each measured point during the cooling process was graphed. The resulting graph had slower cooling at the start

of the curve and what seems to be noise at the end of it, an example of which can be seen it figure 7. The delay in cooling was caused by the fact that fanning started ever so slightly after the heat source was removed from the sensor, and the noise at the end is because of the limited resolution of the sensor at 0.0625°C, which along with the slower rate of cooling causes the measured temperature curve to be step-like. The linear curve in between those two extremes contain the actual cooling curve of the temperature sensor during the given conditions, and by removing these points at either end of the spectrum the correct value for *m* can be calculated numerically. See figure 8 for a curve corrected like this.

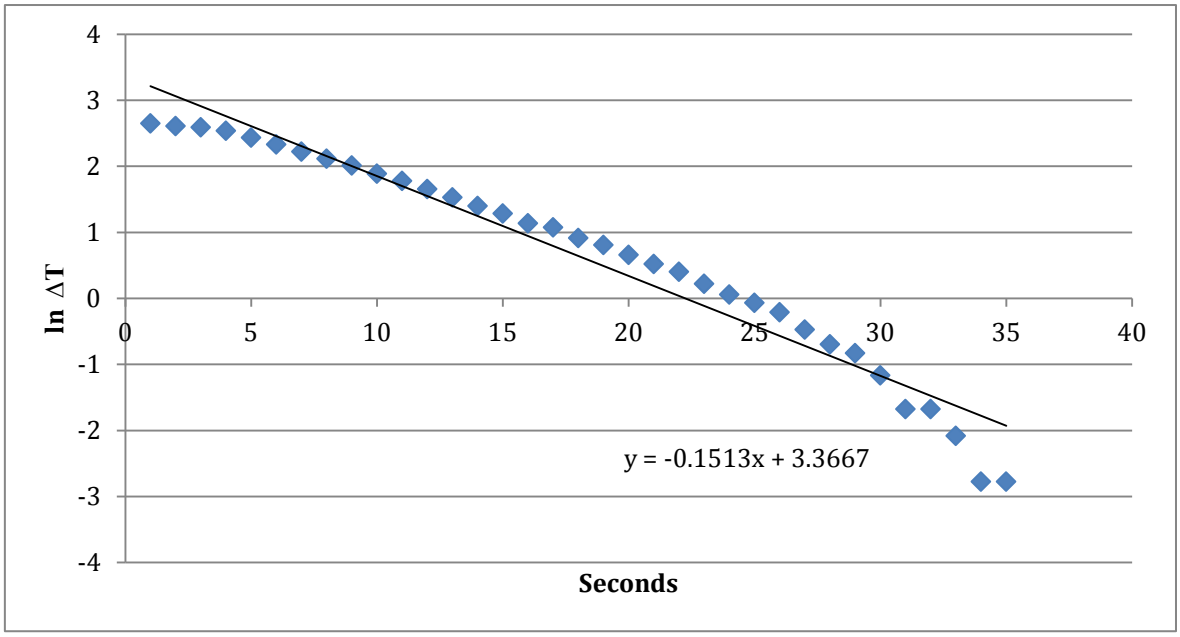


Figure 7: A logarithmic curve of the cooling process, unadjusted.

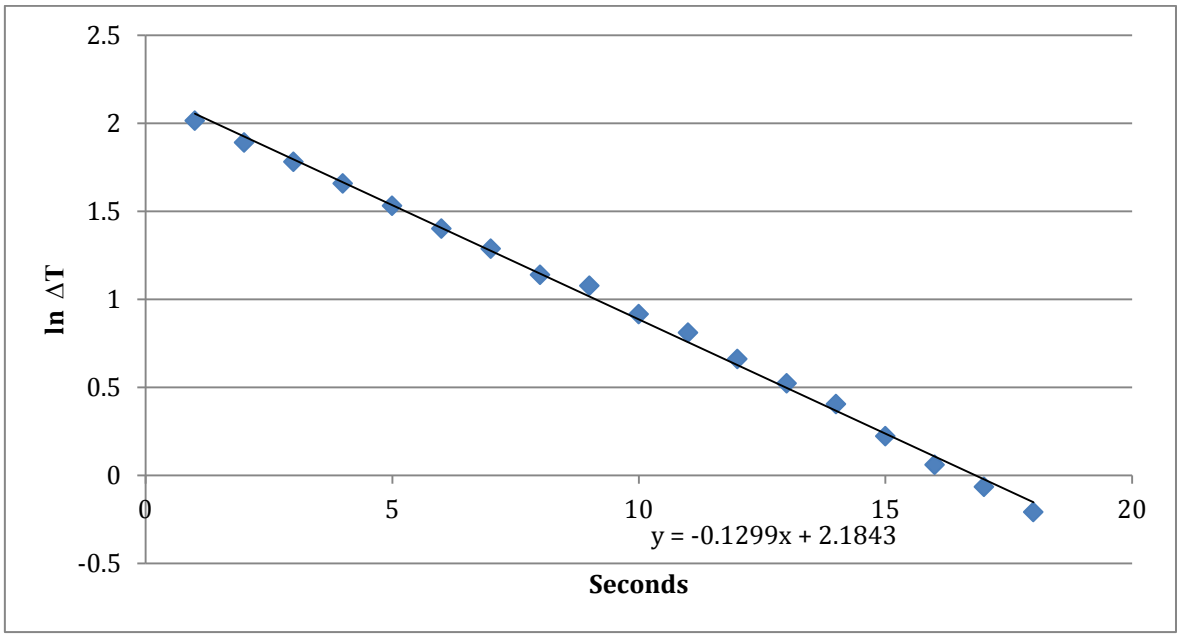


Figure 8: The same logarithmic curve of a cooling process, adjusted by removing non-conforming points.

The calculated time constants during different conditions for a few different test runs can be seen in figures 9 through 13 (the control tests being done without the propellers being ran at all).

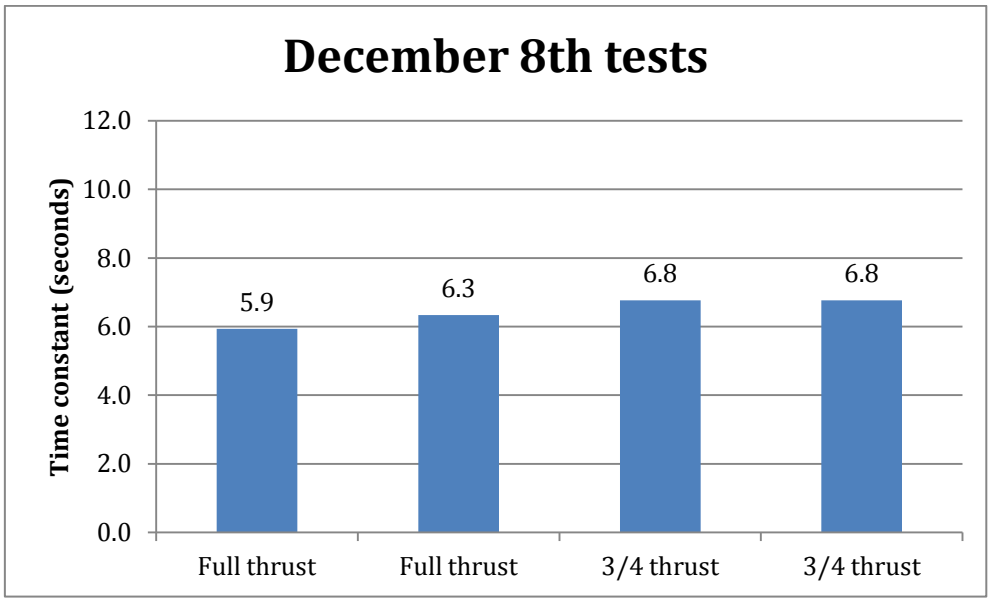


Figure 9: Calculated temperature constants of experiments done on December the 8th, 2015.

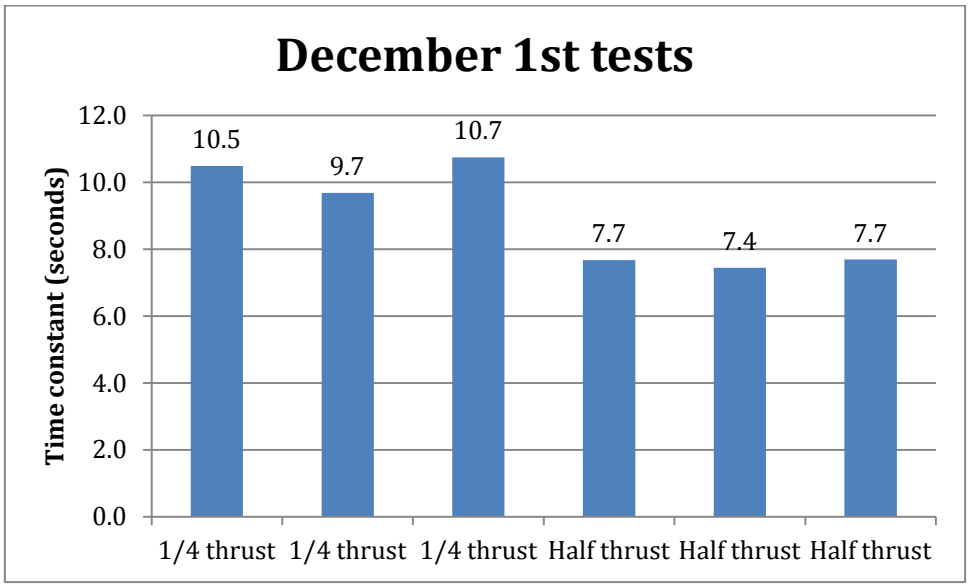


Figure 10: Calculated temperature constants of experiments done on December the 1st , 2015.

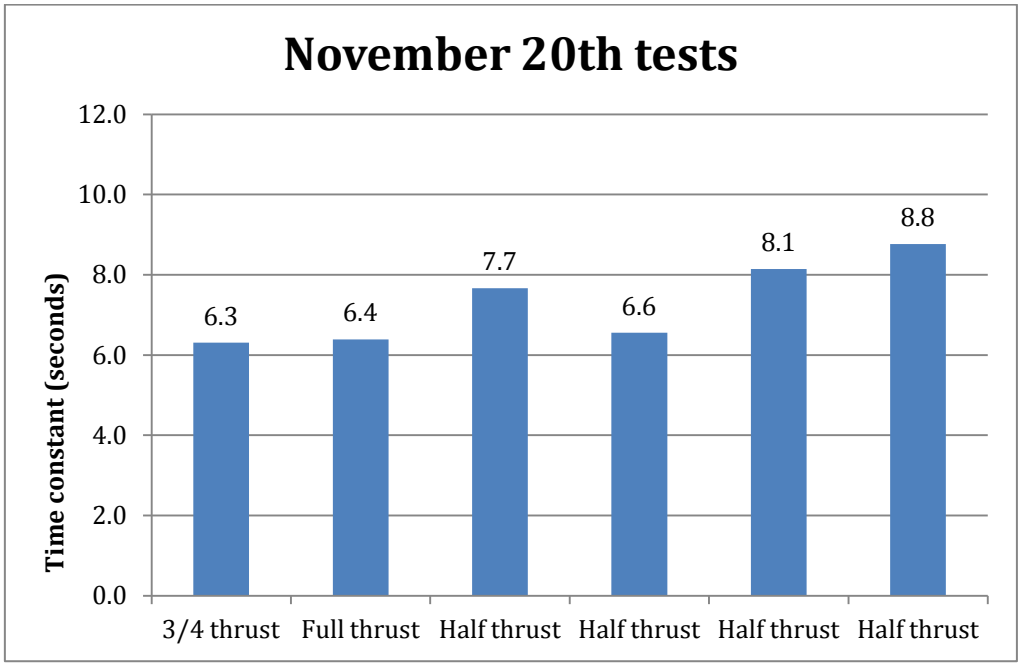


Figure 11: Calculated temperature constants of experiments done on November 20th, 2015.

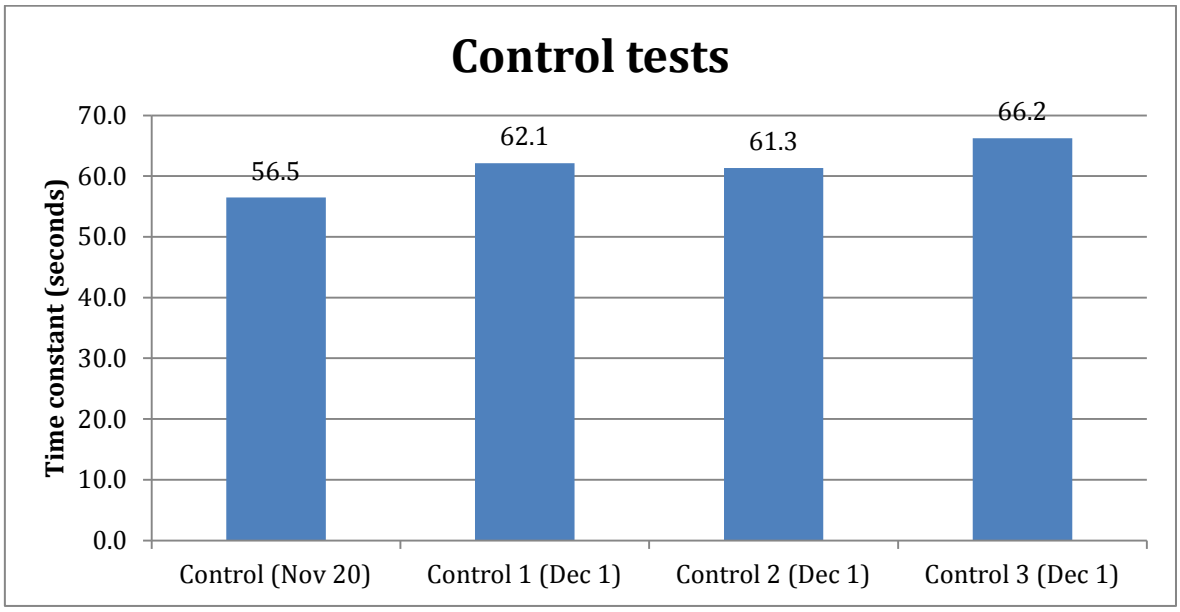


Figure 12: Calculated temperature constants of control tests done December the 1st and November the 20th.

Although the results look good (with the exception of November 20th where the half thrust tests seem to have unusually large variance, which can partly be explained by the fact that the legs were not properly stabilized at this point) it was difficult to achieve consistency in wind measurement tests. See table 1 for approximate measured wind speeds measured during the tests made in November 20th; note that large variance in wind speed was observed even beyond the approximate ranges given.

November 20		
Thrust	Time constant (seconds)	Measured wind speed (m/s)
3/4 thrust	6,3	~11,5
Full thrust	6,4	13,5-14,0
Half thrust	7,7	~10,0
Half thrust	6,6	10,0-11,0
Half thrust	8,1	8,0-9,0
Half thrust	8,8	9,5-10,0

Table 1: Calculated temperature constant and measured wind speeds on tests done in November 20th.

What this all leads up to is the fact that there is an inherent difficulty in trying to measure the quadcopters ability to fan air under different conditions; the fact that the motors are affected by a control system which tries to regulate the quadcopter to achieve the most stable possible flight. This means that even when giving the same amount of thrust straight upwards the control system will try to achieve a 90° angle towards the earth (or whatever it considers to be that), and with the tripod used not being completely flat compared to the bottom of the quadcopter the thrust given out by the motors will be different for each setup of the experiment, even if the quadcopter is strapped in securely. The fact that it is strapped in is also a variable which is hard to predict the result of; the control system is intended to control flight after all, not any other circumstances. In fact, often it was possible to feel the difference between how much the motors were spinning just by holding your hand under the propeller, something which was also confirmed by measuring with the anemometer.

Instead it makes more sense to see how it would actually perform under real conditions, flying and hovering in the air. This is done by using the same process as described in the theory part, but due to the slight temperature differences in the field along with the limitations in resolution it is difficult to observe the characteristic temperature curve. Therefore the quad is also to hover at the same altitude for 2 minutes, after which the temperature sensor has reached ambient temperature for certain. Then you compare the ambient temperature with the rest of the curve, and see how long it took the quadcopter to reach ambient temperature.

Unfortunately the method described above couldn't be executed fully due to a crash in the middle of the testing. However, it was possible to recover the data which can be seen in figure 13.

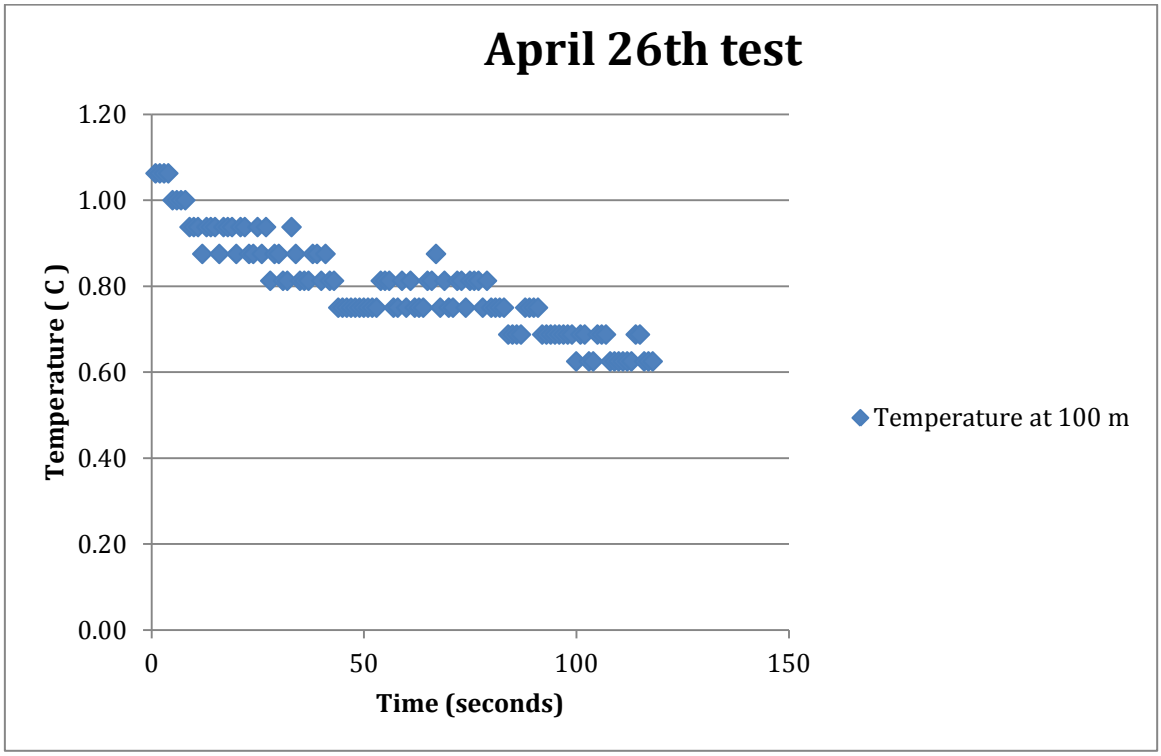


Figure 13: The temperature measurement results of hovering at 100 meters, measured approximately 8:28-8:30 local time.

These results go against the lab testing which has been done in that the time constant seem to be very high. Theory dictates that the time constant shouldn't be dependent on the time difference being high, which leads me to believe that it might be a combination of noise and the resolution being too low to see the real value, which is supported by the fact the graph isn't as linear as the earlier results. On the other hand it isn't clear if the temperature change has actually stopped or not at the last few points. It might be that the fluctuations are a part of a natural temperature change process.

6.5 Discussion

Achieving consistency in measuring the temperature constant in lab conditions proved incredibly hard, but doing these tests still gave us an idea of about how large the time constant should be.

Measurements in the field under real conditions gives a hint to the actual time constant, but it is necessary that it should be kept in mind that it is the real time constant under the particular set of circumstances during the duration of the test. Wind will affect the streams of air as well, which of course will affect the result.

Wind also affects the control system which strives to keep in place, and therefore will increase or decrease thrust to different motors to achieve that. Even when wind is taken out of consideration the control system itself has variability to it, such as the small variations in thrust to different motors when it tries to keep an even altitude during hovering, and variability of the quadcopters own sensors and how it adjusts itself to reflect all of that. It is therefore not reasonable to assume that the sensor will have a constant bulk fluid washing over it in real test conditions.

In the end the most important thing is that a value for the time constant with a good marginal is chosen so that a correct temperature measurement is gotten every time. Once that value is chosen a good rule of thumb would be to multiply that time by 3 to get the final hover time, as after 3 time constants the value has reached 95% of the real one (as compared to the starting temperature).

7 Atmospheric conditions and sound

The atmosphere, being the medium of sound, directly affects how sound travels through it. Wind has a large impact, but so does temperature differences.

7.1 Temperature inversions

If air temperature increases instead of decreases as altitude gets higher it is called a temperature inversion. This is caused by warmer air trapping colder air, and is actually a very stable situation. As discussed later, these affect the path sound takes through the atmosphere, and they also have significant effect on the spread of airborne pollution by trapping the polluted air underneath the temperature inversion.

There are three kinds of temperature inversions, radiational, subsidence and frontal. [33]

7.1.1 Radiational inversion

Radiational inversions happen at low altitudes, 50 – 200 meters. It happens as a consequence of the ground cooling off rapidly by heat radiation, after which the colder ground will cool down the air closest to it, leaving a layer of air with higher temperature on top of the cold air. These kinds of inversions happen during the night and persist until next day when the ground is heated up again. An illustration of this can be seen in Figure 14. [33]

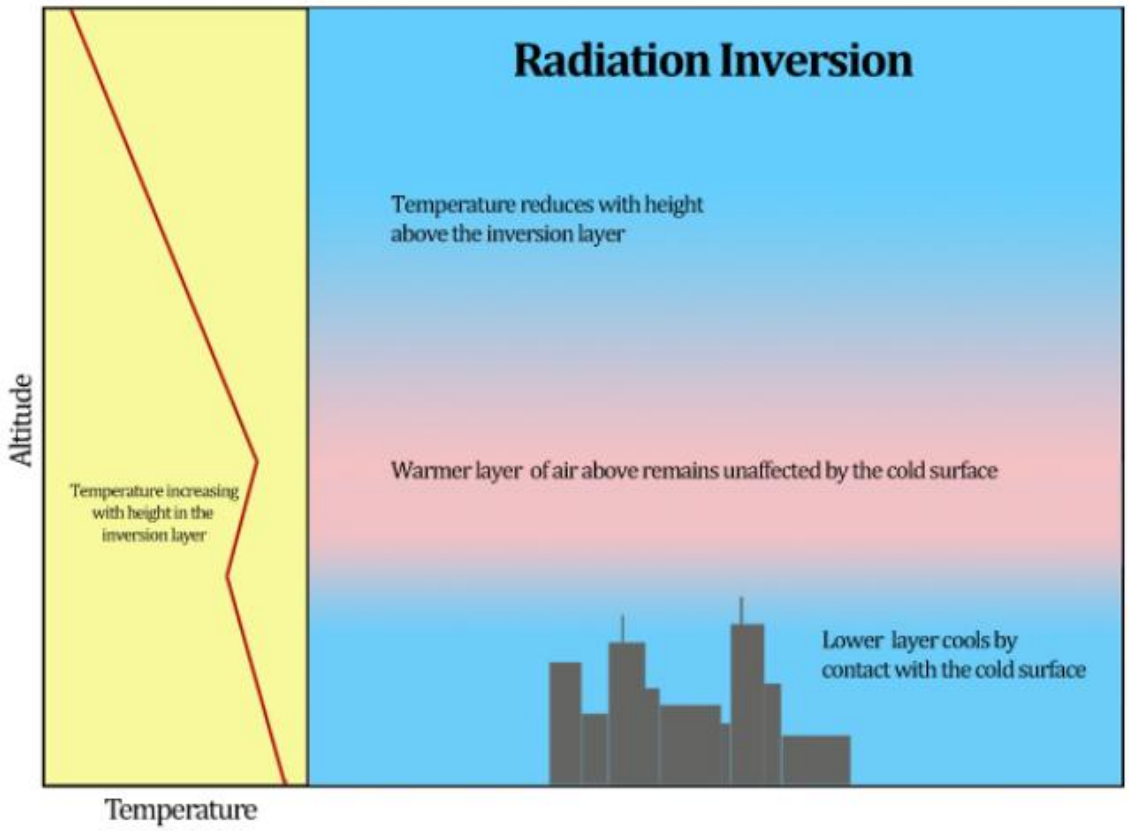


Figure 14: A figure of radiation inversion. [33]

7.1.2 Subsidence inversion

Subsidence inversions are caused by a rise of hot air and a descent of cold air taking its place. Subsidence inversions can be caused by a stagnant high pressure or cold air from the sea or the ocean flowing into an area of where it is trapped by mountains. The warm temperature layer is maintained as descending air is compressed against the colder air and heated adiabatically, stopping its descent and making it ascend again. A figure of this situation can be seen in Figure 15. [33]

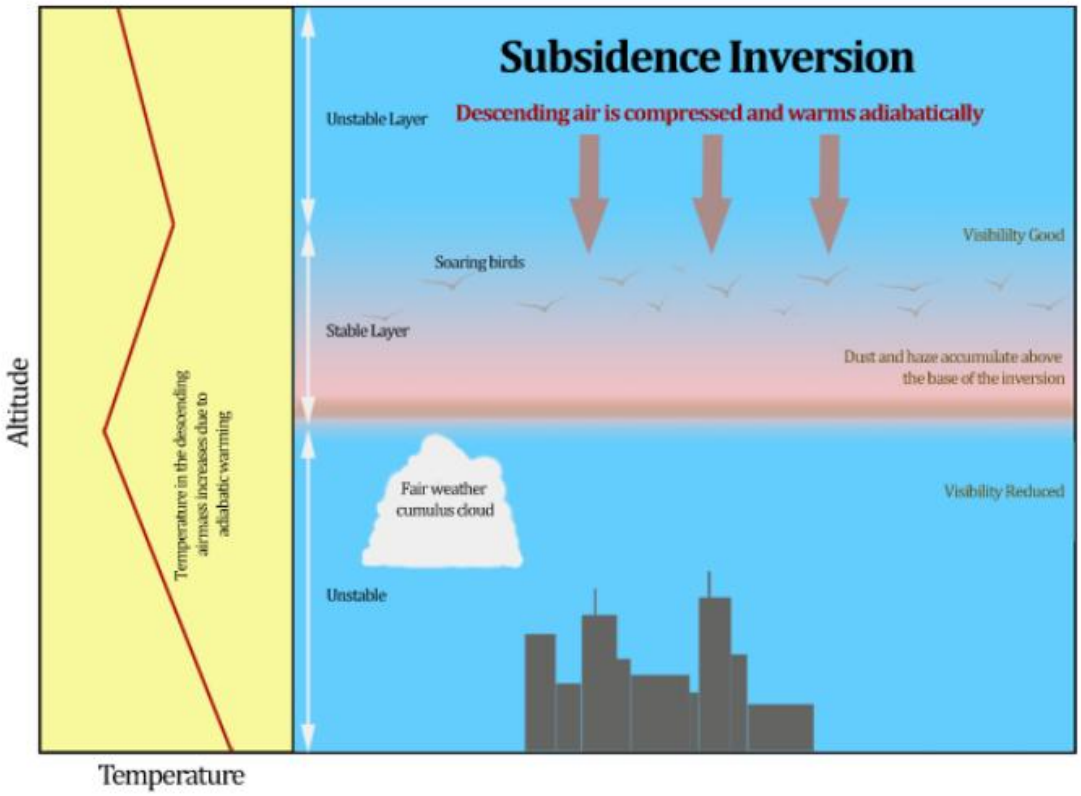


Figure 15: Schematic picture over a subsidence inversion. [33]

7.1.3 Frontal inversions

Frontal inversions, as the name implies, occur when a front of warm air envelops a mass of cold air. Any warm air that rises is stopped as soon as it hits the mass of even warmer air, trapping it along the boundary layer. Note that there is no layer of warm air involved, rather the entirety of the front is warmer than the cold air beneath it., which is pictured in Figure 16. [33]

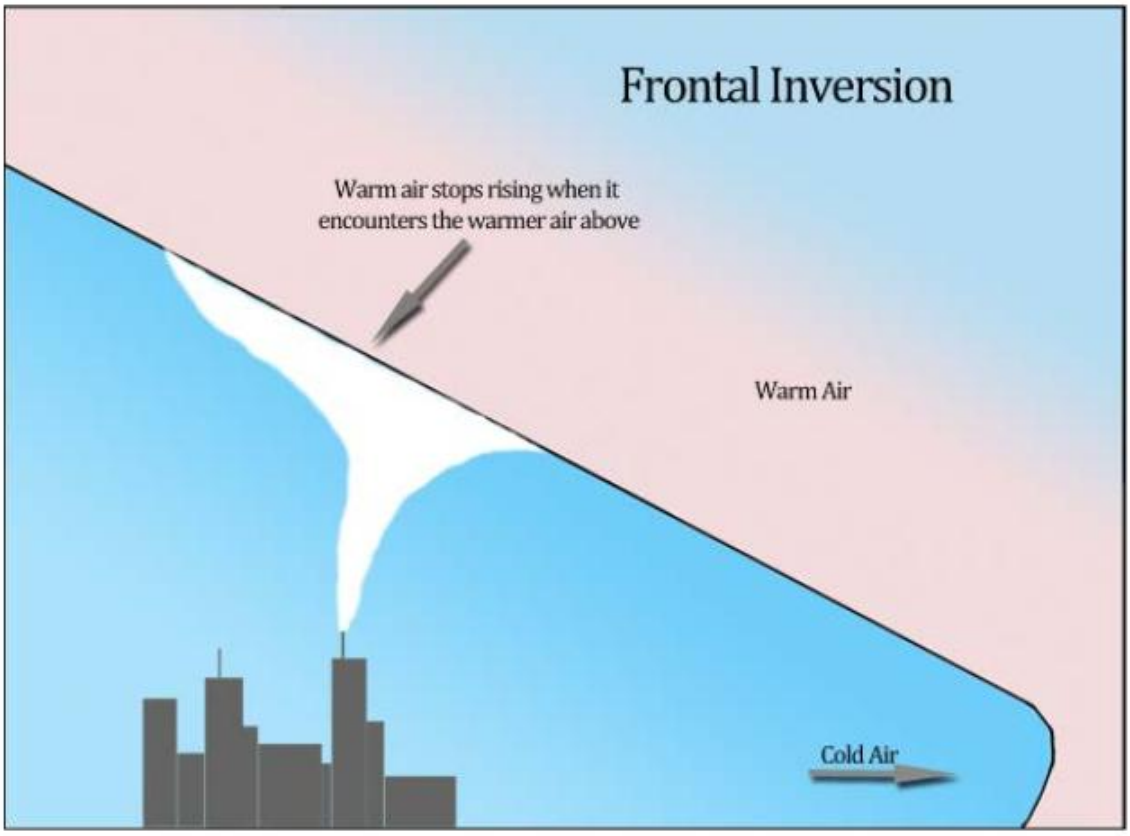


Figure 16: Schematic picture of a warm front moving in over cold air in a frontal inversion situation. [33]

7.2 Sound propagation

It is vital that atmospheric conditions be taken into consideration when modeling sound, which as mentioned before is what the project aims to help with in the end. In the following section are a few ways in which sound is affected by atmospheric conditions as well as ground geometry.

7.2.1 Air absorption

There are two different ways sound is absorbed by the atmosphere; molecular relaxation and viscosity effects, molecular relaxation having the bigger impact of the two. How much the sound is dampened by the atmosphere depends on the frequencies of the sound and the temperature and humidity of the atmosphere. For instance, sounds at 2 kHz are typically absorbed by the rate of 0.25 dB/100m at 20 °C and 30% relative humidity, and 5 dB/100m at 20 °C and 10% relative humidity for sounds at 8 kHz. Absorption is has little impact on sound levels unless the distances involved are long or there are high frequencies involved, which are more readily absorbed than sounds with lower frequencies. [34]

7.2.2 Surface effects

Sound is also affected by how the ground level looks like, a few major effects being ground absorption and attenuation through different barriers.

In ground absorption, sound is attenuated (reduced in magnitude) as it reflects off the ground. Not all sound that reaches a location travels straight to it, some is reflected from the ground before it reaches its source, an illustration of which can be seen in figure 17. A soft surface absorbs more than a hard one, thick grass being able to reduce sound levels by 10dB every 100 meters for instance. If both the source of the sound as well as wherever the sound is measured both are close to the ground it is also possible the reflected wave will interfere with the direct sound wave destructively in the range of 200-600 Hz. This is called ground effect.

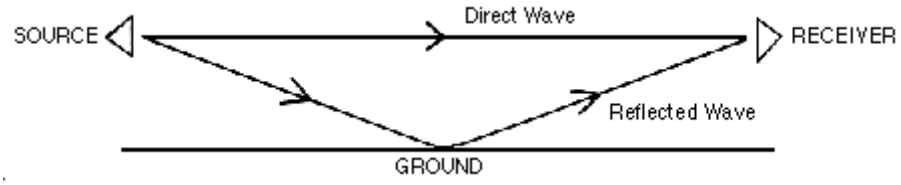


Figure 17: Schematic picture of ground absorption. The reflected wave loses some of its magnitude when it reflects from the ground. [34]

Solid barriers can attenuate sound in a significant manner, provided that line of sight to the source of the sound is obscured by it. When sound waves hit an obstacle they diffract at the edges of it, sending some of the sound waves around the obstacle as in Figure 18. This still means that a significant amount of the sound is blocked by the wall, of course depending on things like the size and position of the barrier; barriers close to either the source or the recipient of the sound are more effective. [34]

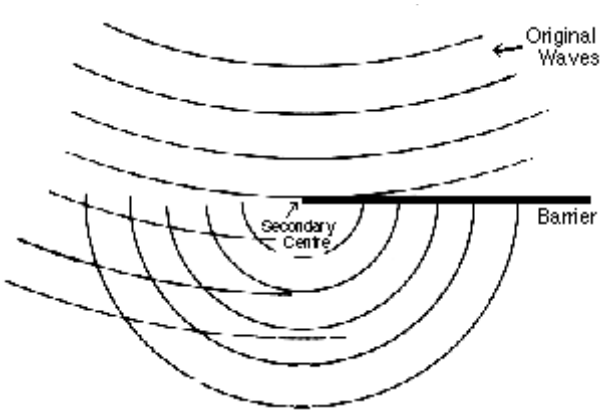


Figure 18: A schematic image of a barrier. Sound bending around the edge of the obstacle from the secondary centre is dampened, making the area directly behind the barrier quieter than its surroundings. [34]

7.2.3 Wind and temperature gradients

Both wind and temperature can affect the direction sound takes through the atmosphere. These effects can be significant, accounting for differences in sound level up to 20 dB. These effects are especially important when estimating sound more than a few hundred meters away.

Whenever wind blows there is a wind gradient; the layer of air next to the ground is always stationary [35]. Sound propagates more easily in the direction of wind, but sound waves propagating against the direction of the wind are instead bended upwards by it, as illustrated in Figure 19.

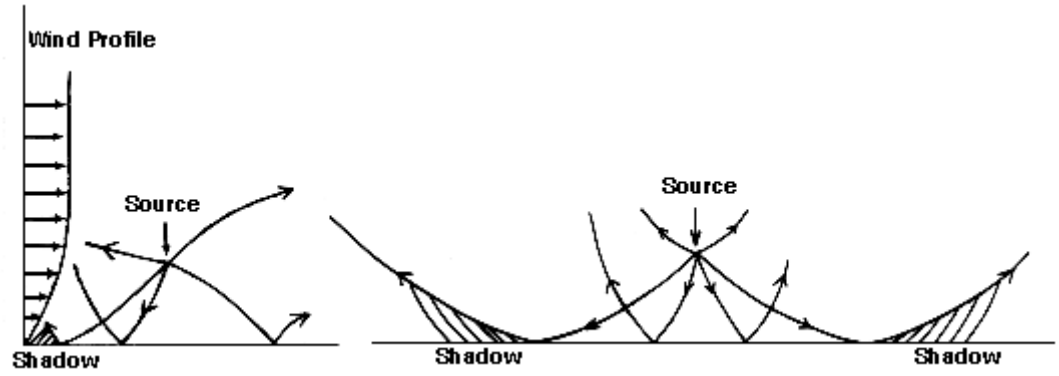


Figure 19: Schematic figure of winds affect on sound. The right figure shows how sound travel without wind, whilst the left figure has wind blowing in from the left, bending the sound waves moving against the wind up and sound waves moving along the wind along its direction. [34]

Sound waves behave similarly to light when it transitions from one medium to the next in that it refracts, or bends. Depending on whether it slows or speeds up when it changes medium determines the direction it refracts. If you look at a "piece of sound" as being a square or a rectangle which transitions from one medium to the next through an angle, if it travels more slowly in the new medium the rectangle will slow down in the corner hitting the new medium first and change angle towards it, and if sound travels faster through the new medium it instead speeds up in the corner hitting the new medium, angling itself away from the new medium. The formula describing refraction is:

$$\frac{\sin \alpha_1}{\sin \alpha_2} = \frac{V_1}{V_2}$$

where $\sin \alpha_1$ is the incoming angle of the wave and $\sin \alpha_2$ the refracted one, and $V_1$is the velocity of the incoming wave and $V_2$is the wave of the refracted wave. [36]

The atmosphere usually get colder as height increases, and as sound travels more slowly through cold gases than hot ones this means that sound usually bends upwards into the atmosphere and disappears. However, if a temperature inversion has taken place, it will instead bend back down towards the ground (as described above), refracting back down to earth (see Figure 20). This also causes so called sound shadows, where loud sounds such as explosions sometimes have large areas of silence between areas where the sound is audible. [34] This is caused by the fact that a majority of the sound is sent up into the atmosphere, where it meets a layer of hotter air. There it refracts back down to earth outside the usual audible range of the sound, where it might even reflect back off the ground and even do the whole journey again, creating another area where the sound is audible. An example of this can be seen it Figure 21. [37]

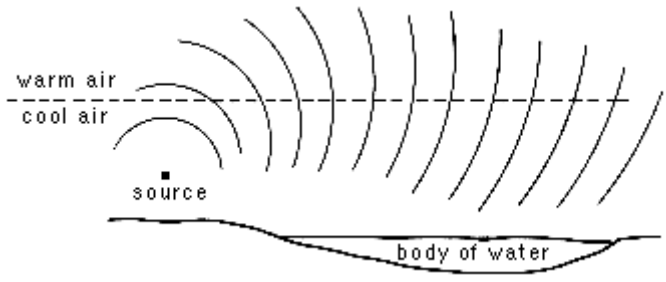


Figure 20: Schematic figure of sound being refracted by a boundary layer of hot air. [34]

Figure 21: An explosion in Kummersdorf (South of Berlin) on 18th December 1925 had sound propagating unusually far, the crossed regions signifying where the explosion was heard. The pattern is a clear result of sound waves hitting a temperature inversion and then being refracted back to the ground, which it in some cases reflected off of, sometimes even several times. [37]

7.3 Temperature profiles

The temperature cycle over a day has enormous impact on weather and climate. Although the true extent of its importance is beyond this thesis it is important to what is talked about in Chapters 7.2.1 to 7.2.3 as many of these phenomena are affected by or a direct consequence of temperature and differences in temperature.

Just like temperatures are different during the day so are the shape of temperature profiles. In mornings before the sun is above the horizon the atmosphere gets some amount of sunlight on it which is reflected. This heats up the atmosphere. When the sun rises up high enough to heat the ground, the ground will typically warm up more quickly than air, which means that the air closer to the ground will heat up more quickly in turn. That means that the shape of this profile changes as the day goes on.

This is of course also highly dependent on the day and the time of year. The quickest changes can be found in mornings at civil twilight and after it has risen, and at the evening before the sun sets and after it has set. [38]

At the temperatures measured by the system in the thesis the temperature gets colder as altitudes becomes bigger (and will continue to do so up to the end of the stratosphere at 10 km altitude), if no temperature inversions occur.

8 Diurnal measurement

The final use of the quadcopter is to take temperature measurements over a whole day, or in other words to measure the diurnal cycle. Measurements are to be taken one day each month, several times during the day. The day which measurements are actually to take place is chosen according to weather (days without rain, snow and wind being preferred).

8.1 Measurement process

During measurements the quadcopter is set to fly itself with the help on the on-board firmware called arducopter. With the use of a so called ground station program, in our case Mission Planner, it is possible to give commands to the quadcopter which it can later execute when automatic mode is engaged.

The first steps to the measurement process is setting up the flight route as well as updating the time on the Raspberry Pi and starting the measurements. The flight features a takeoff command, a single way point, a set amount of stations on different heights where the quadcopter is hovers in place for 60 seconds, after which the quadcopter returns to the launch location and lands. The Raspberry Pi in contrast to most other computers does not contain a hardware clock, and instead relies on NTP, which pulls the current time from atomic clock data over the internet, to update it to the correct time. By restarting the NTP service it is ensured that the Raspberry Pi got the correct time initiated and measurements can begin.

After automatic mode is engaged and the quadcopter goes through the flight route, the temperature sensor taking measurements during this time. Once this is done measurements are stopped and temperature logs are transferred, either manually or by using FTP (File Transfer Protocol). After this the raspberry pi is shut off unless further measurement runs are desired. Flight logs from the quadcopter are downloaded and then the quad is shut off as well.

8.1.1 Measurement cycle during diurnal tests

In the proof of concept trial measurements were to be taken four times, once at dawn before sunrise, once after sunrise, once in the afternoon before sunset and a last time after sunset. Temperatures at 120 meters and all the way down to 20 meters (using 20 meter steps) are measured, the temperature at ground level taken as the temperature that the sensor settles at as the quadcopter is idle on the ground. The quadcopter is set to hover for 60 seconds at each point to ensure that an accurate temperature has been reached. These are unfortunately not available in this thesis due to technical difficulties and weather conditions making taking these tests impossible. An example of earlier tests can be found in Figure 22.

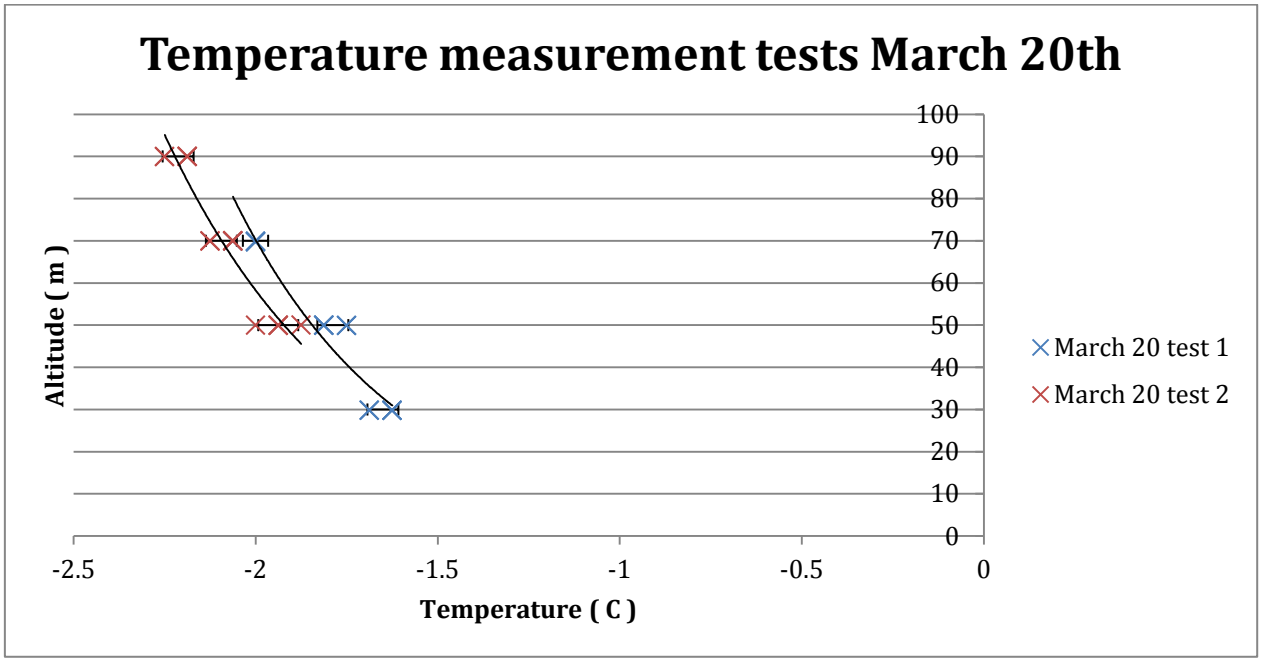


Figure 22: Graph over 2 test runs done in March 20th. In these two tests the quadcopter hovered at each station for 30 seconds and the last 5 points of data at each station were picked out and plotted. Confidence intervals use the t-distribution.

9 Conclusions

The result of this thesis work is a working system to measure temperatures of the atmosphere between 0 and 150 meters altitude, but the system is in many ways limited. Fact of the matter is that as a complete beginner when it comes to quadcopters it would be only by luck that a completely suitable quadcopter model was chosen. If I were asked to do it over again I believe that I would have the knowledge of what to look for and be able to pick out a more suitable quadcopter for the needs of the system. In the case of the CX-20 its limited ability to withstand wind is particularly limiting. Nevertheless the Cheerson CX-20 proved useful in learning about quadcopters and was in fact the cheapest option found that fit the requirements. However, its limitations were noticeable and the data that can be gathered by it is intrinsically biased towards conditions that are good to fly in. These limitations are also the largest contributing factor to the diurnal measurements not being finished in time of this thesis.

The Cheerson CX-20 also did not prove to be the most reliable during the course of the project. Many of these difficulties were caused by inexperience, however. Soldering was not an optional skill. Some spare parts proved difficult to find within Europe which caused delays due to long shipping times, and although broken part are to be expected in quadcopters the build quality was not as good as expected either.

In hindsight a hexacopter of some sort would have handled wind conditions a lot better than the Cheerson CX-20. The Cheerson CX-20 being firmly classified as a hobby-grade quadcopter is not a beginner quadcopter, toy-grade quadcopters being recommended as a first beginner quadcopter instead, due to them being cheaper, more durable and with their lower mass also less dangerous. A hexacopter in the hands of a beginner - with six motors, a LiPo battery for each of these motors, increased mass and carbon-fiber propellers – has the potential to cause serious damage. The price of mistakes on a hexacopter is also greater due to increased cost of replacement parts compared to the Cheerson CX-20.

On the other hand a hexacopter might not be cost effective for its intended use in the temperature sensor system, as its measurements are planned to be made once a month and doing them in high wind speeds is not a requirement. A slightly more powerful quadcopter could fly in higher wind speeds and would probably be the ideal choice in terms of performance per cost.

Looking at the background of the thesis, an alternative way to measure temperatures in the atmosphere at different altitudes without having to use very expensive equipment, this thesis has proven to be at least partially successful. To replicate this system and buy equipment needed for the quadcopter you would have to spend roughly around 500€ for the quadcopter and an additional 100€ on the Raspberry Pi and sensor. With some knowhow I find it likely that an ideal quadcopter platform for the uses in this project (a quadcopter with parts of higher quality and longevity, reliability, better resistance against weather and wind, longer flight times) could be built for under 1000€, maybe a bit more if a truly solid system is required.

The temperature sensor also proved itself to be able to use some improvement. Although small its response time still proved to be somewhat slow, and its resolution could have been higher. Using, for instance, a platinum resistance temperature detector (RTD), which is much smaller (and as an added bonus also more accurate) could reduce response time down to being nonexistent. However, as opposed to the digital temperature sensor used in this system a platinum RTD is affected by electromagnetic interference, meaning that having it anywhere close to the power lines to the motor would cause faulty readings. This can be reduced by keeping it as far away from electronics as possible as well as using an analog to digital converter to reduce the length where the signal can be interfered. Nevertheless it is likely that there are many challenges to attaching a temperature system like this to a quadcopter that I am simply not aware of.

The work proved itself quite multidisciplinary, which meant that it was varied and interesting. On the other hand it also meant that it was difficult to decide what parts should be focused on, and what parts to actually research in further detail. This meant that there was also no one person who could advice me on all of the parts of the project, but rather I had to ask help on different topics from different people.

The project work was done over a long time in low intensity, which I would say worked against it. It was no schedule that could be made with any sort of reasonable accuracy (due to inexperience about the subjects at hand and work times being at times when other studies did not interfere), and therefore regrettably no real attempt at making a unified schedule or time plan, which I consider a mistake no matter how broken it would turn out to be. Another issue was that there was no requirements set

out at the start, which although it could be explained by the fact that at the start no one really knew what was possible it is simply put poor practice.

References

[1] WRF Organization, "Weather Research and Forecasting model," No date. [Online]. Available: http://www.wrf-model.org/index.php. [Accessed 24 March 2016].

[2] Ardupilot Dev Team, "Ardupilot," 2016. [Online]. Available: http://ardupilot.org/copter/docs/what-is-a-multicopter-and-how-does-it-work.html. [Accessed 24 March 2016].

[3] S. Reed, "Building Design Construction," 16 January 2014. [Online]. Available: http://www.bdcnetwork.com/blog/quadcopters-save-project-team-15k-warranty-work. [Accessed 23 March 2016].

[4] J. S. Cavote, "Power," 4 January 2014. [Online]. Available: http://www.powermag.com/drones-promise-faster-easier-inspection-of-boilers-stacks-towers-and-more/?pagenum=7. [Accessed 23 March 2016].

[5] T. Mogg, "Digitaltrends," 29 October 2014. [Online]. Available: http://www.digitaltrends.com/cool-tech/check-out-the-ambulance-drone-that-could-one-day-save-your-life/. [Accessed 23 March 2016].

[6] Cheerson, "Cheerson," No date. [Online]. Available: http://www.cheersonhobby.com/productShow.asp?id=587. [Accessed 24 March 2016].

[7] Arducopter Team, "Autotrim," 2016. [Online]. Available: http://ardupilot.org/copter/docs/autotrim.html. [Accessed 24 March 2016].

[8] Arducopter Team, "Autotune," 2016. [Online]. Available: http://ardupilot.org/copter/docs/autotune.html. [Accessed 24 March 2016].

[9] "Glitchy647", "Reddit," 30 December 2014. [Online]. Available: https://www.reddit.com/r/Multicopter/comments/2qu8gn/the_winter_multicopter_flying_guide/. [Accessed 12 April 2016].

[10] Quadcopter 101, "Quadcopter "Wobble of death": VRS recovery and Avoidance," 1 August 2014. [Online]. Available: https://www.youtube.com/watch?v=LCret4rv0HE. [Accessed 12 April 2016].

[11] Trafi, "Miehittämätön ilmailu," N.d. [Online]. Available: http://www.trafi.fi/ilmailu/miehittamaton_ilmailu. [Accessed 25 January 2016].

[12] Trafi, "Flygning av fjärrstyrda luftfartyg och modellflygplan (OPS M1-32)," 10 September 2015. [Online]. Available: http://www.trafi.fi/filebank/a/1444309380/e7774ceb949ed2f7c708171276d31674/18720-OPS_M1-32_sve.pdf. [Accessed 4 February 2016].

[14 Microchip, "Microchip," 2011. [Online]. Available:

] http://ww1.microchip.com/downloads/en/DeviceDoc/25095A.pdf. [Accessed 12 April 2016].

[15 Arducopter team, "Arducopter source code 3.1.2," Arducopter team, 2014.
]

[16 Bosch Sensortec, "Adafruit," 5 April 2013. [Online]. Available: https://cdn-
] shop.adafruit.com/datasheets/BST-BMP180-DS000-09.pdf. [Accessed 12 April 2016].

[17 T. Dicola, "Hardware," 2015. [Online]. Available:
] https://learn.adafruit.com/mcp9808-temperature-sensor-python-library/hardware. [Accessed 2015].

[18 Adafruit, "MCP9808 High Accuracy I2C Temperature Sensor Breakout Board,"
] N.d. [Online]. Available: https://www.adafruit.com/products/1782. [Accessed 5 February 2016].

[19 M. Borg, *Oral source,* 2016.
]

[20 NAVSTAR-GPS, "US Coast Guard Navigation Center," September 1996. [Online].
] Available: http://www.navcen.uscg.gov/pubs/gps/gpsuser/gpsuser.pdf. [Accessed 24 March 2016].

[21 No author, "Time and date," No date. [Online]. Available:
] http://www.timeanddate.com/time/leap-seconds-future.html. [Accessed 24 March 2016].

[22 Homeland Security National Cybersecuirty and Communications Integration
] Center, "US Coast Guard Navigation Center," 26 May 2015. [Online]. Available: http://www.navcen.uscg.gov/pdf/cgsic/Leap_Second_Best_Practices_20150526_Intrl_Version.pdf. [Accessed 24 March 2016].

[23 T. Dicola, "Github," 21 November 2014. [Online]. Available:
] https://github.com/adafruit/Adafruit_Python_MCP9808. [Accessed 12 April 2016].

[24 "barnert", "Stupid Python Ideas: Getting atomic writes right," 2014. [Online].
] Available: http://stupidpythonideas.blogspot.fi/2014/07/getting-atomic-writes-right.html. [Accessed 3 February 2016].

[25 Debian, "The Debian GNU/Linux FAQ: Chapter 4 - Compatibility issues," 2015.
] [Online]. Available: https://www.debian.org/doc/manuals/debian-faq/ch-compat.en.html. [Accessed 2 February 2016].

[26 ArduPilot, "ArduPilot," N.d.. [Online]. Available: http://ardupilot.com/. [Accessed
] 2 February 2016].

[27 E. Powers, "Applications of GPS Provided Time and Frequency and Future," 14
] August 2012. [Online]. Available:

http://www.gps.gov/governance/advisory/meetings/2012-08/powers.pdf. [Accessed 4 February 2016].

[28] B. P. Bruegger, "Source code for module glue.gpstime," 2002. [Online]. Available: http://software.ligo.org/docs/glue/glue.gpstime-pysrc.html. [Accessed 3 February 2016].

[29] J. Mehaffey, "How accurate is the TIME DISPLAY on my GPS?," No date. [Online]. Available: http://gpsinformation.net/main/gpstime.htm. [Accessed 7 April 2016].

[30] P. R. T.-J. P. M. Cristina R. Murta, "NTP Survey," No date. [Online]. Available: http://www.ntpsurvey.arauc.br/globecom-ntp-paper.pdf. [Accessed 6 April 2016].

[31] D. L. Mills, "Network Time Protocol (NTP) daemon," No date. [Online]. Available: http://doc.ntp.org/4.1.0/ntpd.htm. [Accessed 7 April 2016].

[32] J. M. C. R. H. T. Yunus A. Çengel, "Convection," in *Fundamentals of Thermal-Fluid Sciences Fourth Edition in SI Units*, McGraw-Hill, 2012, pp. 644-645.

[33] J. C. A. v. Werner, *Finnish wind data management and usage for Gauss modeling with SoundPlan-7.2 software,* Tampere: Tamperen Ammatikorkeakoulu, 2015.

[34] B. Truax, Handbook for Acustic Ecology, second edition, Cambridge Street Publishing, 1999.

[35] J. M. C. R. H. T. Yunus A. Çengel, "The No-Slip Condition," in *Fundamentals of Thermal-Fluid Sciences Fourth Edition in SI Units*, McGraw-Hill, 2012, p. 424.

[36] M. K. I. P. L. K. P. M. Raimo Seppänen, Maols Tabeller, Helsinki: Schildts Förlag Ab, 2006.

[37] T. B. Gabrielson, "Refraction of Sound in the Atmosphere," *Acoustics Today,* April 2006.

[38] K. Rutledge, Vasa, 2016.