Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics
Study program: Cybernetics and robotics

Stabilization of an autonomous multirotor UAV with doppler radars

Stabilizace autonomní bezpilotní helikoptéry pomocí dopplerových radarů

BACHELOR THESIS

Author: Libor Dubský
Supervisor: Ing. Daniel Heřt
May 2023

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Dubský Libor** Personal ID number: **491834**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Stabilization of an Autonomous Multirotor UAV with Doppler Radars

Bachelor’s thesis title in Czech:

Stabilizace autonomní bezpilotní helikoptéry pomocí dopplerových radarů

Guidelines:

The goal of this thesis is to design and build a compact device with three small doppler radar units, which will be mounted on an autonomous multirotor UAV to measure its velocity. The velocity measurement will be used to stabilize the UAV. The following tasks will be solved:
1. Design and build a printed circuit board which amplifies and processes the signal from a doppler radar module.
2. Design and build a printed circuit board which interfaces with an onboard computer on the UAV and provides power and communication to the individual doppler radar modules and their amplifiers.
3. Design and implement an interface between the ROS [3] based MRS UAV [2] system, and the doppler radar interface board.
4. Implement a state estimator which uses the measurements from the doppler radars along with measurements from the UAVs inertial measurement unit to estimate the current state vector of the UAV.
5. If external factors and the schedule of the MRS group allows, demonstrate the UAV with the doppler radar stabilization in a real-world experiment.

Bibliography / sources:

[1] D. Kellner, M. Barjenbruch, K. Dietmayer, J. Klappstein and J. Dickmann, "Instantaneous lateral velocity estimation of a vehicle using Doppler radar," Proceedings of the 16th International Conference on Information Fusion, 2013.
[2] Baca, T., Petrlik, M., Vrba, M., Spurny, V., Penicka, R., Hert, D., and Saska, M., "The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles", J Intell Robot Syst 102, 26 (2021).
[3] Quigley, Morgan, et al. "ROS: an open-source Robot Operating System." ICRA workshop on open source software. Vol. 3. No. 3.2. 2009.
[4] RFbeam Microwave GmbH, Typical Doppler Signal Amplifier Application Note AN-04 (2012). Retrieved from RFbeam website: https://www.rfbeam.ch/files/products/9/downloads/AN-04%20TypicalSignalAmp.pdf

Name and workplace of bachelor’s thesis supervisor:

Ing. Daniel Heřt Multi-robot Systems FEE

Name and workplace of second bachelor’s thesis supervisor or consultant:

Date of bachelor’s thesis assignment: **11.01.2023** Deadline for bachelor thesis submission: **26.05.2023**

Assignment valid until: **22.09.2024**

Ing. Daniel Heřt	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor’s signature	Head of department’s signature	Dean’s signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

. _____ Date of assignment receipt

_____ Student's signature

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague day .....

.....
Libor Dubský

Acknowledgements

I am grateful to my thesis supervisor Ing. Daniel Heřt, for his feedback and valuable advice regarding this work.

Libor Dubský

Title:
Stabilization of an autonomous multirotor UAV with doppler radars

Author: Libor Dubský

Thesis supervisor: Ing. Daniel Heřt

Abstract: This thesis deals with measuring the velocity of an Unmanned Aerial Vehicle (UAV) using three Doppler radars. K-LC5 radars from RFbeam Microwave GmbH are used. A suitable method for processing the radar signals is chosen, and two types of Printed Circuit Boards are created for the measurement. The boards are assembled into a compact device. The layout of the boards defines the calculation of the measured velocities. A ROS node is created for communication between the device and the Multi-robot Systems Group (MRS) UAV system. Finally, the device is tested in flight.

Key words: Doppler radar, velocity measurement, UAV, ROS

Název práce:
Stabilizace autonomní bezpilotní helikoptéry pomocí dopplerových radarů

Autor: Libor Dubský

Vedoucí práce: Ing. Daniel Heřt

Abstrakt: Tato práce se zabývá měřením rychlosti autonomní bezpilotní helikoptéry (dronu) pomocí třech dopplerovských radarů. Jsou použity radary K-LC5 od společnosti RFbeam Microwave GmbH. Je zvolena vhodná metoda zpracování signálů z radarů a pro měření jsou vytvořeny dva typy desek plošného spoje. Desky jsou složeny do kompaktního zařízení. Rozložení desek definuje výpočet naměřených rychlostí. Pro komunikaci mezi zařízením a systémem drona Multi-robotické skupiny je vytvořen program s využitím ROS. Na závěr je zařízení podrobeno testovacímu letu.

Klíčová slova: Dopplerův radar, měření rychlosti, autonomní bezpilotní helikoptéra, ROS

Contents

List of abbreviations

UAV	Unmanned Aerial Vehicle
MRS	Multi-robot Systems Group
GNSS	Global Navigation Satellite System
SLAM	Simultaneous localization and mapping
PCB	Printed Circuit Board
SLB	Solder-less Breadboard
SMD	Surface Mount Device
PCBR	PCB with radar
EB	Evaluation board
MPU	Micro-Processing Unit
HSE	High-Speed External
HSI	High-Speed Internal
RCS	Radar coordinate system
UCS	UAV coordinate system
RUCS	Rotated UCS
ROS	Robot Operating System
FOV	Field of view

List of Figures

Introduction

The Unmanned Aerial Vehicle (UAV), commonly known as a drone, is a relatively new technology. There are many types of UAVs, fixed-wing aircraft, single-rotor UAV, multi-rotor UAV, and more. In this thesis, we will talk about quadcopters that are not seen much in everyday life. They are primarily used in special cases but are slowly becoming mainstream in various industries. UAVs are used, for example, in the army, to find people, fight fires, deliver packages, make movies, and much more. A pilot with remote control is required for most of these tasks, but in the future, UAVs may do these tasks automatically without a pilot. For automatic control, it is necessary to know the position and the velocity of the UAV. Usually, the estimation from the Global Navigation Satellite System (GNSS) is used for this purpose. GNSS covers almost the whole world. Connecting to GNSS is easy and works only in open spaces. However, in the case of UAVs in areas where GNSS cannot reach, or GNSS is limited by weather, it is not possible to estimate the velocity from the GNSS. The UAV should have a backup system for estimating velocity for these cases.

The Multi-robot Systems Group (MRS) is a research group specializing in autonomous UAVs. A suitable velocity estimation method must be selected for MRS autonomous UAVs. There are many methods of measuring velocity, for instance, estimating from SLAM computed from 3D lidar data or using a camera to estimate velocity from optical flow. Although estimating from SLAM is very accurate, it is computa-tionally intensive. Furthermore, the 3D lidar is quite expensive. Estimation from the optical flow uses a camera that may be cheaper than 3D lidar, but its accuracy depends on light and the type of surface the camera scans. This Thesis will discuss the method of measuring velocity using K-LC5 Doppler radars. Three radars will be used to create a compact device attached to the UAV. This method is still accurate enough, cheaper, and does not need as much computing power.

The main objective and partial tasks

The main objective of this thesis is to design and build a compact device with three small doppler radar units, which will be mounted on an autonomous multirotor UAV to measure its velocity. Partial tasks of this thesis are:

- Design and build a printed circuit board which amplifies and processes the signal from a doppler radar module.

- Design and build a printed circuit board which interfaces with an onboard computer on the UAV and provides power and communication to the individual doppler radar modules and their amplifiers.

- Design and implement an interface between the ROS [1] based MRS UAV [2] system, and the doppler radar interface board.
- Implement a state estimator which uses the measurements from the doppler radars along with measurements from the UAVs inertial measurement unit to estimate the current state vector of the UAV.
- If external factors and the schedule of the MRS group allow, demonstrate the UAV with the doppler radar stabilization in a real-world experiment.

Chapter 1

Technical background

1.1 Doppler radar functionality

Doppler radar is a special radar that measures a moving object's velocity using the Doppler effect. Doppler effect describes a change in the frequency of a wave in relation to an observer moving relative to the wave source. If the wave source moves towards the observer, the observer records a higher frequency than the source transmits, and when the source moves away, the observer records a lower frequency. The same principle would apply if the observer were moving, while the wave source was not moving. These frequency differences are used to measure a moving object's velocity accurately. The Doppler radar for measuring velocity transmits a microwave signal toward the object and analyzes how the object's velocity changes the frequency of the reflected signal.

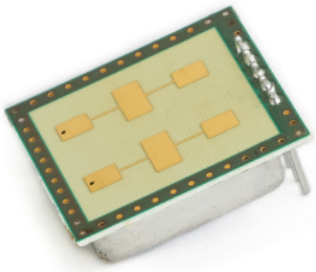

Figure 1.1: Figure of doppler radar K-LC5 [3]

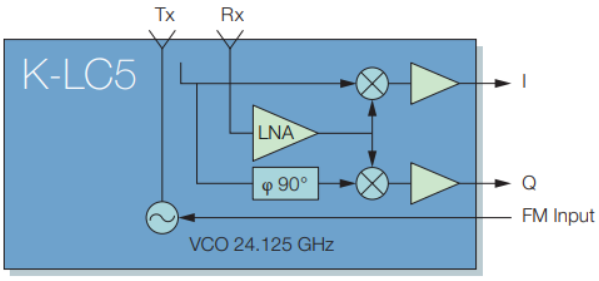

Figure 1.2: Doppler radar K-LC5-RFB-00x block diagram [3]

1.2 The use of Doppler radars

Three Dopplers radar K-LC5 will be attached to the UAV. The radar is shown in Figure 1.1 and its block diagram in Figure 1.2. The radars will be inclined to measure velocity relative to the ground.

The radar has two output signals marked as I and Q. The signals are sinusoidal and have the same frequency and amplitude, but the signal Q is phase shifted. The velocity will be estimated from the frequency of the signals, as the estimated velocity is linearly dependent on the frequency. The direction of motion will be determined from the phase shift. When the phase shift is -90°, the radar approaches a measured

object, and when the shift is 90°, the radar recedes from the measured object. The same shifts apply when the measured object is approaching/receding from the radar.

The voltage of the signals range from less than $100\,nV$ to some mV [4]. Therefore, it is necessary to design a Printed Circuit Board (PCB) to amplify these signals. Further, for better processing and to reduce some noise, we convert the signals to rectangular signals, which are simpler to process.

1.3 Signals processing

We will design another PCB for signal processing, which will be called the Evaluation board. The Evaluation board will include a Micro-Processing Unit used to process the rectangular signals and calculate the velocities from them. We define that positive values of velocities mean approaching the measured object and negative values mean receding from the measured object. After estimating the velocities, they are sent to the UAV via USB.

Chapter 2

Design of the printed circuit board for the Doppler radar

The principle of operation of the K-LC5 radar is explained in section 1.2. In this chapter, we will discuss the design of the PCB for amplification and digitization of radar signals.

2.1 Amplification circuit

The manufacturer provides a recommended amplification circuit for the radar, and Figure 2.1 illustrates the circuit schematic. The circuit receives a signal from the radar, which is amplified through two operational amplifiers, and at the output is the amplified signal U_{zes}. The operational amplifiers are in the inverting bandpass filter configuration. Two operational amplifiers are used so that the circuit has sufficient Gain-Bandwidth product and high gain and maintains the orientation of the amplified signal. This configuration amplifies and passes only specific frequencies. The values C1, R1 and C3, R3 determine the low pass limit, while C2, R2 and C4, R4 determine the high pass limit. Capacitors C1 and C3 also remove the DC component from the radar signal, the amplification of which would set a large offset. The circuit is powered by voltage from 0 V to 5 V which does not allow to amplify the negative voltage of the radar signal. Therefore there is a voltage divider that sets the voltage offset of both operational amplifiers at 2.5 V which is in the middle of the power supply voltage range. The offset ensures that the entire signal waveform is preserved during amplification.

For testing on the SLB, we used LM358 operational amplifiers. These amplifiers were sufficient for testing, but there was a problem with their output voltage swing because the outputs were not rail-to-rail. That means the maximum output voltage is lower than the supply voltage. The maximum output voltage of these operational amplifiers is decreased by 1.4 V. For testing, it was necessary to change the values of resistors R10 and R11 shown in Figure 2.1. Otherwise, there was no need to modify the schematic. The output of the test amplification circuit is in Figure 2.2. The figure shows that the output sine waveform is noisy and slightly clipped, however, it is possible to read the frequency.

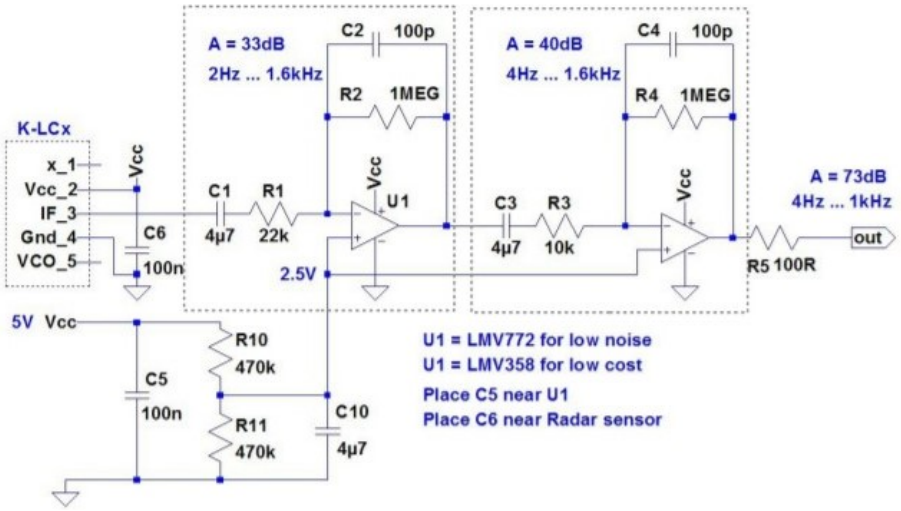

Figure 2.1: Recommended amplification circuit schematic [4]

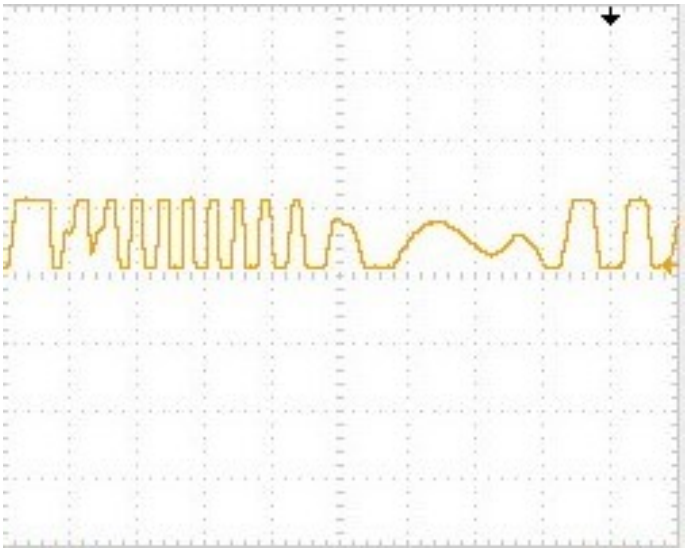

Figure 2.2: Amplified voltage from the test circuit

For the design of the PCB, we used the TL974IPWR operational amplifier was used, which in contrast to the LM358, contains four operational amplifiers inside the case, and the outputs are rail-to-rail. However, it was necessary to adjust the values of resistors R10 and R11 from the recommended schematic. During the commissioning of the PCB, a problem occurred as the voltage divider showed a periodic waveform of charging and discharging the capacitor instead of the stable 2.5 V. The simplest solution we devised was to reduce the resistance of the resistors of the voltage divider. In this way, the charging of the capacitor will take less time, and the capacitor will not be able to discharge completely. We changed the values of resistors R10 and R11 from 470 $k\Omega$ to 715 Ω. With these values, the voltage was already stable enough for the amplification circuit to work.

The radar and the amplifier circuit need the cleanest possible supply voltage to work best. Any noise will be amplified through the circuit, and the resulting signal will be degraded. Therefore, it is necessary to filter the noise from the supply voltage.

2.2 Power supply circuit

2.2.1 Radar PCB connectivity

For radar output signals and power supply, a 4-pin header 2.54 mm and 4-pin JST-GH were used. Since the connectors are parallel, both can be used without distinction and the choice is up to the user's preference. We use the outermost pins of the connectors for the power supply and the two inner pins for the output signals Q and I that are converted to rectangle signals. These connectors are stable enough to stay connected during a flight of the UAV.

2.2.2 Power supply voltage filtering

As the power supply, 5 V voltage from USB is used. The voltage is a stable 5 V, but we cannot rely on the purity of the voltage. If we were to supply the radar with this voltage, we would burden output signals with noise. Therefore, the input voltage has to be filtered.

We used the primary circuit of the capacitance multiplier for the filtering, shown in Figure 2.3. The circuit works similarly to an RC integrator circuit, which is shown in Figure 2.5, with only two differences. The first one is a voltage drop across the transistor, so the output voltage will always be lower than the input voltage, depending on the transistor throughput. The second is that the circuit externally behaves as if the capacitance of the capacitor is multiplied by the transistor current amplification factor β. Hence the circuit is called a capacitance multiplier. The actual principle of operation of the circuit is somewhat more complicated. The circuit behaves like an emitter follower with an RC integrator circuit on the base of the transistor, where the input current to the cell is β times less than it would be without the transistor. For this reason, it looks like the capacitor is β times larger. However, the time constant τ does not change, so the circuit has better-filtering properties for frequencies higher than the cutoff frequency f_0 of the integrator RC circuit.

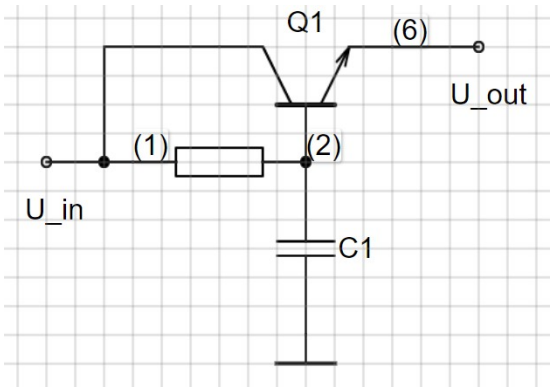

Figure 2.3: Fundamental connection of the capacitance multiplier

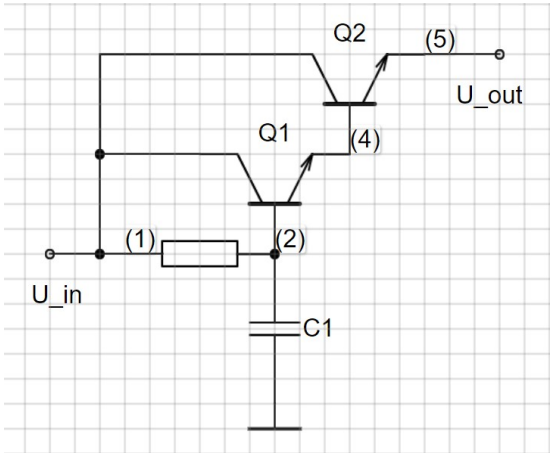

Figure 2.4: Fundamental connection of a capacitance multiplier with a Darlington pair

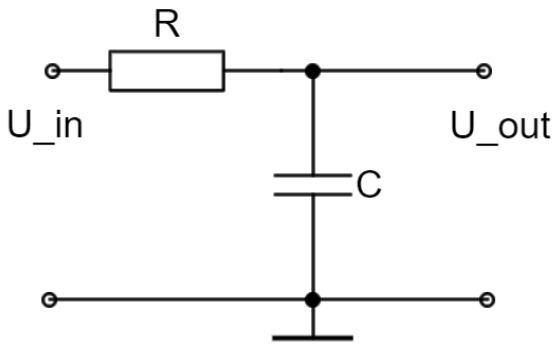

Figure 2.5: RC integrator circuit schema

For testing, on a Solder-less Breadboard (SLB), we chose the transistor BC337-40 with current gain factor $\beta_{min} = 250$ and $\beta_{max} = 630$ [5], while for the PCB design, we used its SMD equivalent BC817-40. For the resistor and capacitor values, we chose R = 100 $k\Omega$ and C = 10 μF. From the chosen values, we can then calculate by formulas:

$$\tau = RC = 100 \cdot 10^3 \cdot 10 \cdot 10^{-6} = 1\,\mathrm{s} \tag{2.1}$$

$$f_0 = \frac{1}{2\pi\tau} \doteq 0.159\,\mathrm{Hz} \tag{2.2}$$

The output voltage of the capacitance multiplier does not have a constant voltage value under load. We have modified the schematic using a Darlington pair. We show the modified circuit in Figure 2.4. Using the Darlington pair, we have obtained a stable output voltage value of about 3.6 V. The voltage is lower because of the passage of two transistors. At the same time, We have further improved the filtering because a single transistor can replace the Darlington pair with a current gain factor given by a formula:

$$\beta_D = \beta_1 \cdot \beta_2 + \beta_1 + \beta_2 \tag{2.3}$$

We can see the difference before and after filtering in 2.6 and 2.7. We can see that the noise voltage in 2.7 is smaller than in 2.6.

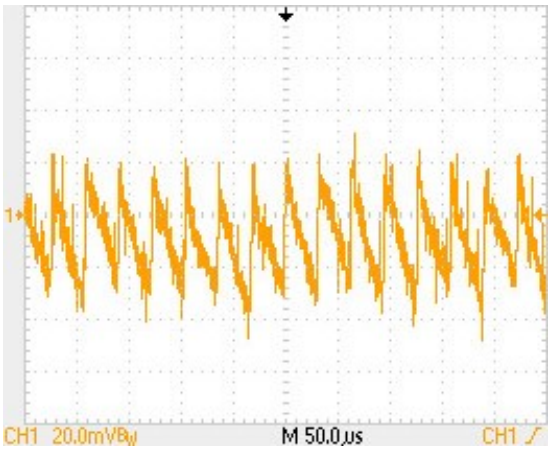

Figure 2.6: The noise voltage waveform without filtering

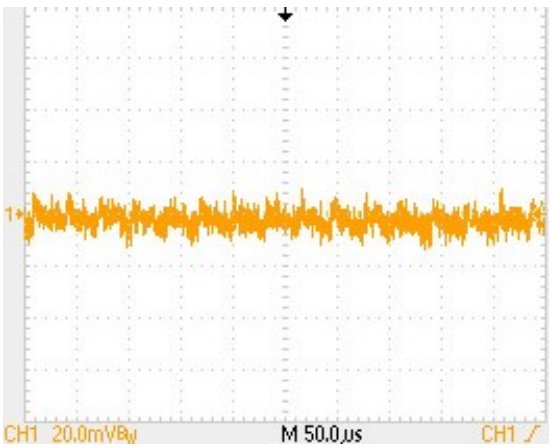

Figure 2.7: The noise voltage waveform after filtering

2.2.3 Input voltage amplification

After completing the filtering, we obtained a stable voltage to power the radar and amplification circuit. However, the selected version of the radar (K-LC5-RFB-00D) requires a supply voltage of 5 V [3], and after filtering, the voltage remained at only 3.6 V. Based on the fact, that we cannot increase the defined input voltage, another solution is needed.

For this reason, we decided to include a step-up converter on the plate before filtration. For testing, we used the HW-637 converter and set it to compensate for the voltage drop during filtering, approximately 6.4 V. For PCB design, we used the converter AP3012. Its wiring is shown in Figure 2.8. The input voltage $V_{in} = 5\ V$ and the output voltage of the converter can be defined by the following formula:

$$U_{out} = 1.25\left(1 + \frac{R_1}{R_2}\right) = 1.25\left(1 + \frac{32 \cdot 10^3}{5.1 \cdot 10^3}\right) \doteq 9.1\,\mathrm{V}\ [6] \tag{2.4}$$

The following section describes the reason for choosing such a high output voltage.

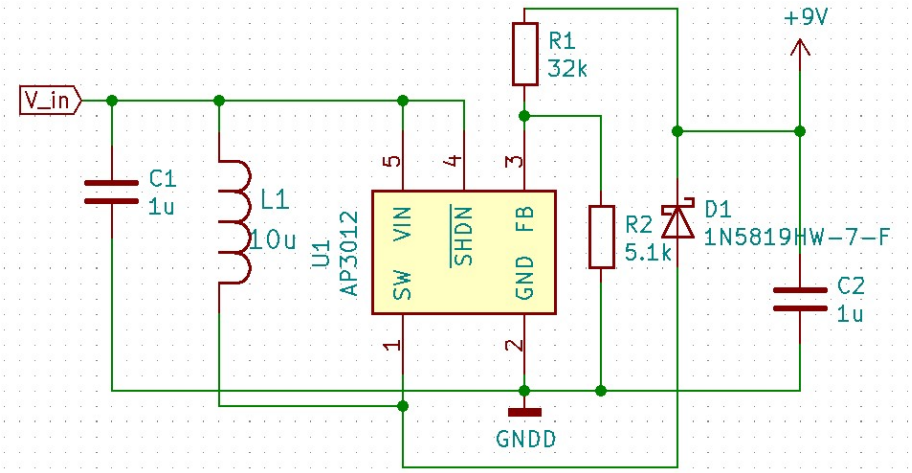

Figure 2.8: wiring of step-up converter AP3012

2.2.4 Voltage stabilization

After connecting the converter, we found that the radar is very sensitive to exceeding the supported voltage and quickly starts to heat up. For this reason, we decided to use a 5 V linear voltage regulator.

For testing and PCB design, we used the regulator L78L05. The wiring is shown in Figure 3.2. We connected the regulator's input to the filter circuit's output and set the output voltage from the step-up converter to 9 V. The reason for choosing such a high voltage was the minimum input voltage of the regulator, which is 7 V. After filtering, the voltage should decrease to about 7.6 V, which is sufficient voltage for the regulator.

The result is a stable voltage of 5 V without much noise, which is suitable for powering the radar and amplification circuit. In these parts of the PCB, the supply voltage must contain as little noise as possible. If the noise is too high, distortion will occur, and the resulting signals will not be usable.

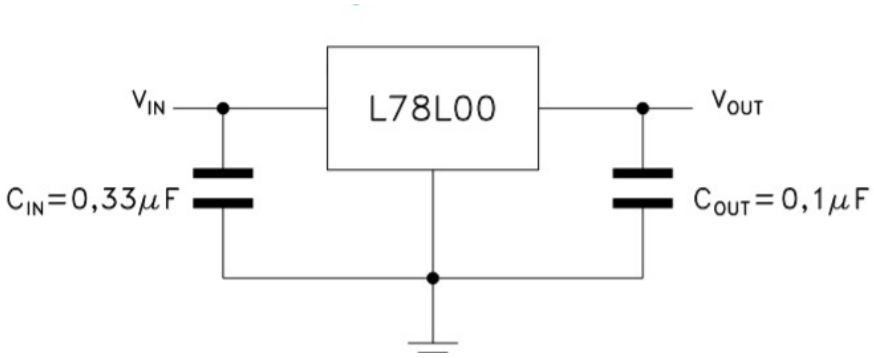

Figure 2.9: Wiring of the L78L05 regulator [7]

2.3 Conversion to rectangular signal

To determine the velocity, it is necessary to know the frequency of the amplified signal. Measurement of the frequency of a sinusoidal signal is difficult. It would need a lot of computing power, and if the signal contains some amplified noise, it would significantly affect the measurement. Therefore, the amplified signal will be converted to a rectangular signal while the frequency of the signal is preserved, and less computing power is needed to measure the frequency. Additionally, the conversion also removes some of the noise.

2.3.1 Recommended circuit: Window comparator

The manufacturer also has a recommended circuit for the radar to convert the amplified signal to a rectangular signal and it is called a window comparator. The circuit does not need to be powered by a filtered voltage. Here all voltages are high enough so that noise in the supply voltage should not affect the resulting signal too much. The schematic of the circuit is in Figure 2.10. The window comparator is powered similarly to the amplification circuit by voltage from 0 V to $Vcc = 5$ V. That means the window comparator can be set to values 0 V or Vcc. 0 V is marked as GND.

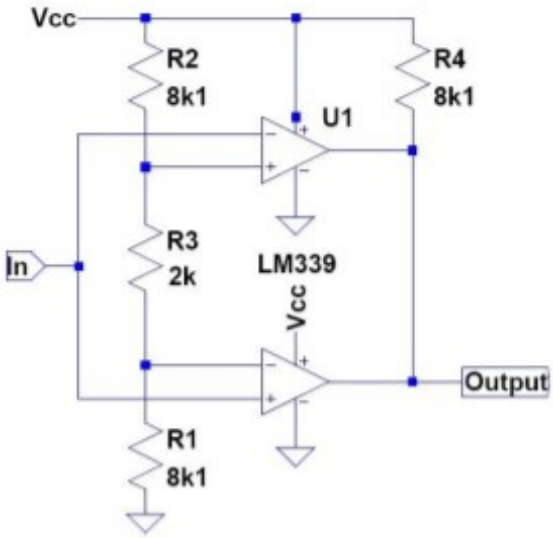

Figure 2.10: Window comparator schematic [4]

The window comparator consists of two comparators. The first is in an inverting comparator circuit, which compares the input signal U_{zes} with the reference voltage U_{max}. The second is in a non-inverting comparator circuit, and it compares the input voltage to a reference voltage U_{min}. We calculated the reference voltage from values in Figure 2.10 using the formulas:

$$U_{max} = Vcc \cdot \frac{R_3 + R_1}{R_2 + R_3 + R_1} = 5 \cdot \frac{2000 + 8100}{8100 + 2000 + 8100} \doteq 2.77\,\mathrm{V} \tag{2.5}$$

$$U_{min} = Vcc \cdot \frac{R_1}{R_2 + R_3 + R_1} = 5 \cdot \frac{8100}{8100 + 2000 + 8100} \doteq 2.23\,\mathrm{V} \tag{2.6}$$

If the input voltage is between U_{min} and U_{max}, the comparator output is set to positive saturation, i.e., to the value of the supply voltage Vcc. If the signal is out of the range U_{min} and U_{max}, the comparator output is set to negative saturation, i.e., to the value of GND.

The window comparator is sensitive and provides twice the frequency for counting rising/falling edges. Double frequency is sufficient for future processing, but the input signal U_{zes} always contains noise, and it is necessary to design a hysteresis for the circuit. [4]

2.3.2 Window comparator modifications

It was required to modify the circuit for better resistance to noise in the amplified signal. For better tuning the modifications, we simulated the circuit on the falstad.com website in the circuit simulator program. We tried different settings for the voltage divider values and resistors for the hysteresis of the two internal comparators until we finally designed the circuit shown in Figure 2.11.

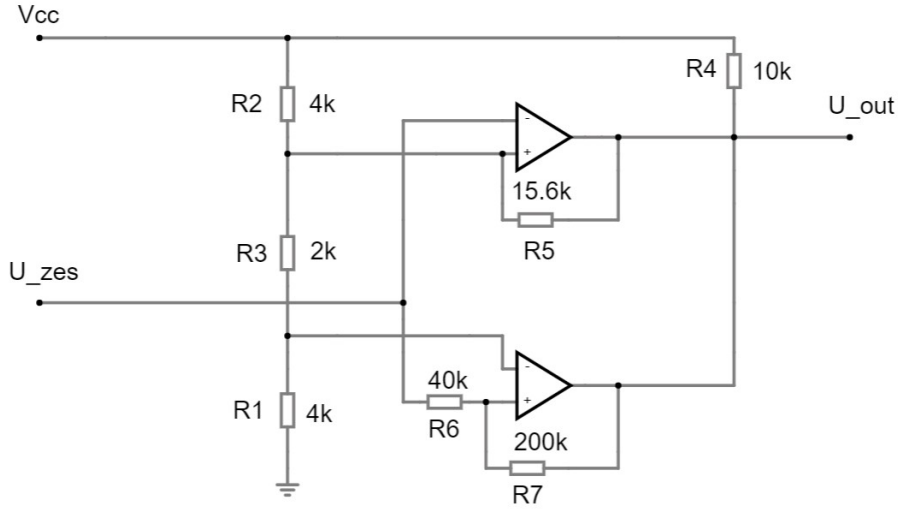

Figure 2.11: Window comparator schematic with hysteresis

The upper comparator, from Figure 2.11, is an inverting comparator with hysteresis. We computed its comparator voltage levels by using the formulas:

$$U_{TH} = Vcc \cdot \frac{(R_1 + R_3) \parallel R_5}{(R_1 + R_3) \parallel R_5 + R_2} = 5 \cdot \frac{[(4 + 2) \parallel 15.6] \cdot 10^3}{[(4 + 2) \parallel 15.6 + 4] \cdot 10^3} = 2.6\,\mathrm{V} \tag{2.7}$$

$$U_{TL} = Vcc \cdot \frac{R_1 + R_3}{R_2 \parallel R_5 + R_1 + R_3} = 5 \cdot \frac{(4 + 2) \cdot 10^3}{(4 \parallel 15.6 + 4 + 2) \cdot 10^3} \doteq 3.27\,\mathrm{V} \tag{2.8}$$

If the output of the inverting comparator is set to GND and the voltage U_{zes} drops below U_{TH}, the comparator output is set to Vcc. If the output is set to Vcc and the voltage U_{zes} is higher than U_{TL}, then the comparator output is set to GND. The R5 resistor and the modified voltage divider ensure that the hysteresis is greater than 0.6 V. This circuit, therefore, provides a relatively high immunity to noise in the amplified signal.

The lower comparator is in a non-inverting comparator circuit with hysteresis. The comparator voltage levels depend on the setting of the inverting comparator. When the inverting comparator is in positive saturation, we compute the comparative voltage levels of the non-inverting comparator by formulas:

$$U_{ref1} = U_{TL} \cdot \frac{R_1}{R_1 + R_3} = 3.27 \cdot \frac{4 \cdot 10^3}{(4+2) \cdot 10^3} = 2.18\,\mathrm{V} \tag{2.9}$$

$$U_{TH2} = U_{ref1} \cdot \left(\frac{R_6}{R_7} + 1\right) = 2.18 \cdot \left(\frac{40 \cdot 10^3}{200 \cdot 10^3} + 1\right) \doteq 2.62\,\mathrm{V} \tag{2.10}$$

$$U_{TL2} = (U_{ref1} - U_{s+}) \cdot \frac{R_6}{R_7} + U_{ref1} = (2.18 - 5) \cdot \frac{40 \cdot 10^3}{200 \cdot 10^3} + 2.18 \doteq 1.62\,\mathrm{V} \tag{2.11}$$

When it is in negative saturation, we compute the comparative voltage levels:

$$U_{ref2} = U_{TH} \cdot \frac{R_1}{R_1 + R_3} = 2.6 \cdot \frac{4 \cdot 10^3}{(4+2) \cdot 10^3} \doteq 1.73\,\mathrm{V} \tag{2.12}$$

$$U_{TH3} = U_{ref2} \cdot \left(\frac{R_6}{R_7} + 1\right) = 1.73 \cdot \left(\frac{40 \cdot 10^3}{200 \cdot 10^3} + 1\right) \doteq 2.08\,\mathrm{V} \tag{2.13}$$

$$U_{TL3} = (U_{ref2} - U_{s+}) \cdot \frac{R_6}{R_7} + U_{ref2} = (1.73 - 5) \cdot \frac{40 \cdot 10^3}{200 \cdot 10^3} + 1.73 \doteq 1.08\,\mathrm{V} \tag{2.14}$$

U_{s+} is the positive saturation output voltage of the non-inverting comparator. U_{ref1} and U_{ref2} are the reference values against which the voltage U_{zes} is compared. If the voltage U_{zes} exceeds U_{TH2}, the comparator output is set to Vcc, and if the input voltage drops below U_{TL2}, the comparator output is set to GND. The same behavior applies to U_{TH3} and U_{TL3}.

From these comparator voltage levels, we can determine that the window comparator is in positive saturation from U_{TH} to U_{TL2} for the falling edge of the input sinusoidal signal. For the rising part, the window comparator is in positive saturation from U_{TH3} to U_{TL}. So, we can see that the window comparator will be in positive saturation only around the set offset, and it is set to GND in other input signal levels. The output signal simulation is in Figure 2.13.

Each rising edge of the output signal indicates the beginning of the half-period. By counting edges, we get twice the frequency the manufacturer wrote about in the documentation. The same properties apply to counting the falling edges.

2.3.3 Signal processing from window comparator

In section 1.2, it was mentioned that the radar signals are phase shifted. The same shift is also applied to the output signals from the window comparators. The shift of signals is shown in Figure 2.12.

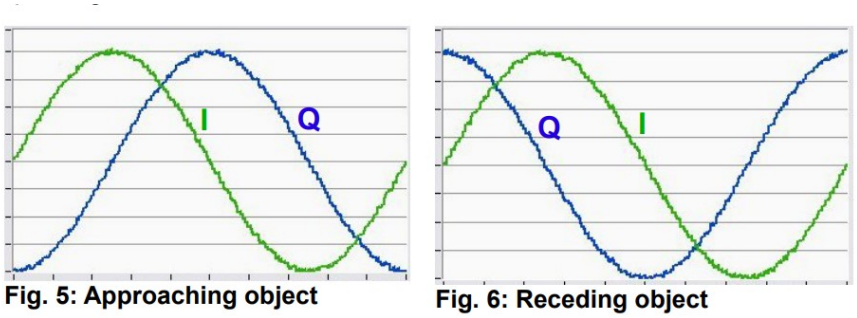

Fig. 5: Approaching object

Fig. 6: Receding object

Figure 2.12: The phase shift of radar output channels [4]

For testing on SLB, we used the recommended LM339 [4] comparator, which we also used on the first version of PCB. It contains four comparators, which both radar channels can use.

To test the first version of PCB, we connected the outputs of the window comparators to the interrupt pins of the Arduino Uno and measured the frequency, and determined the direction of motion by detecting the falling edges. If the circuit were ideal and contained no noise, then the output signals, for approaching in, would look like the simulation in Figure 2.13. The figure shows that the only way to determine the direction of motion is to measure the time difference between the rising/falling edges of each signal I and Q.

The physical circuit will always contain some noise, especially when it flies on the UAV, which despite the hysteresis setting, can cause an unwanted pulse in the output of the window comparator. We are unable to determine if the pulse is valid or not. So any noise-induced pulse may reverse the direction of motion when edges are detected. Thus the use of the window comparator is inappropriate for this task.

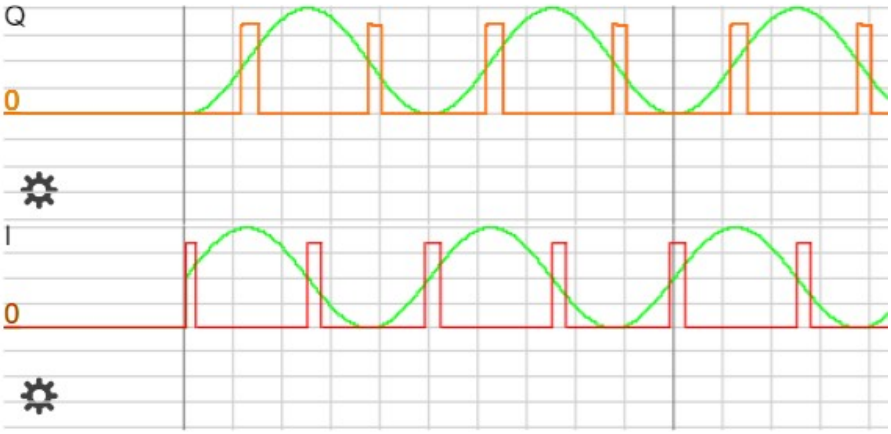

Figure 2.13: Simulation of output signals of window comparator with hysteresis for approaching

2.3.4 Window comparator alternative

For greater noise immunity, we replaced the window comparator with a separate inverting comparator with hysteresis. The schematic is in Figure 2.14, and a simulation of its output is in Figure 2.15. The comparator voltage levels can be described by equations (2.7) and (2.8), which were for the inverting comparator with

the edit that R1 + R3 = R13. We can see from the simulation that the inverting comparator has only half the resolution of the window comparator. We counted only the rising/falling edges for the window comparator to get twice the frequency. We will simultaneously count the rising and falling edges with the inverting comparator. This way, we can get the same frequency as the window comparator but with half the wiring. The inverting comparator needs fewer comparators than the window comparator, so we replaced the LM339 with an LM393.

After closer examination of the simulation in Figure 2.15, we can notice that the signals look like the output signals of a rotary encoder. So the direction of motion we specify as we would with a rotary encoder. We will watch the I signal and use the phase of the Q signal to determine the motion for approaching or receding. If there is any invalid pulse, it will be easy to filter out.

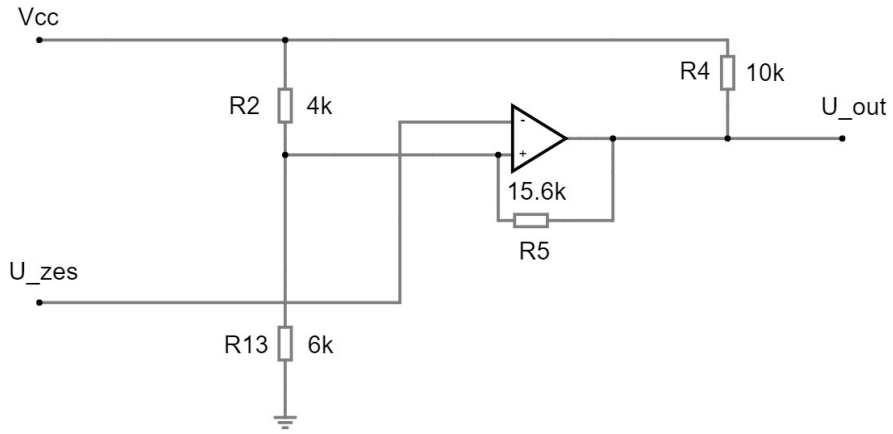

Figure 2.14: Inverting comparator with hysteresis

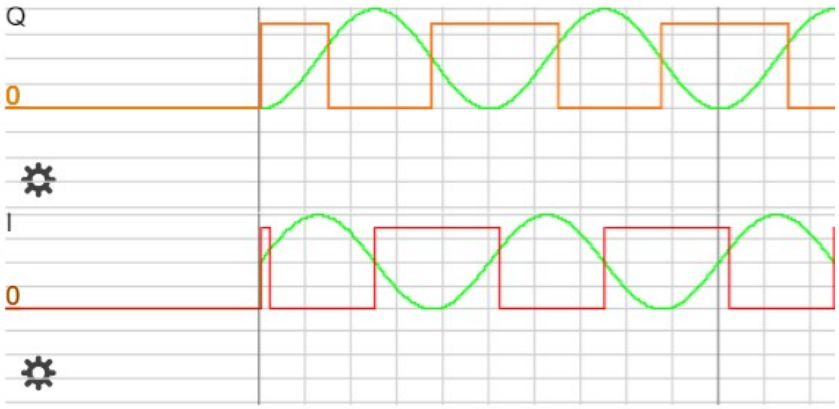

Figure 2.15: Simulation of output signals of inverting comparators with hysteresis for approaching

2.4 PCB design and implementation

This section will combine the circuits from the previous sections and integrate them on a PCB. As mentioned in section 2.1, some circuits are susceptible to noise. Therefore, we create an area on the PCB shielded from noise by a shielding box.

This area contains the entire amplification circuit (2.1), the voltage regulator (2.2.4), and the radar pins. For mounting the PCB, there are holes in each corner with a diameter of 3.2 *mm*. We implemented the PCB in Kicad version 5.15.

The first version of the PCB still contained the window comparator. We mainly used this version to test the circuits' functionality and find bugs we could not detect on SLB. While testing this PCB, we discovered the problem described in subsection 2.3.3, so a new version was needed.

The second version of PCB differs from the first one by a better arrangement of components in the shielded part and replacing the LM339 with LM393. The implementation of the second version of PCB is presented in the Figures 2.16, 2.17, and 2.18. The attached file **radar.zip** contains the overall circuit, PCB layout, and components used.

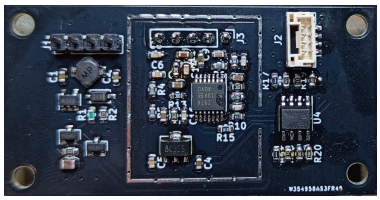

Figure 2.16: The back side of PCB without shielding box

Figure 2.17: The back side of PCB with shielding box

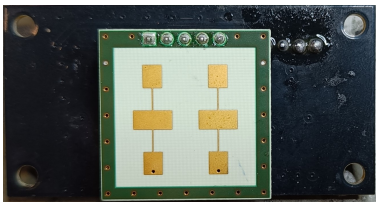

Figure 2.18: The front side of PCB with radar

Chapter 3

Implementation of the Evaluation board

PCB with radar (PCBR) must be powered and their output signals processed. Therefore, we needed to design an Evaluation board (EB). The PCBRs are powered through the EB, and the EB calculates desired velocities from the output signals of PCBRs. The EB communicates with the UAV and sends the velocity to it.

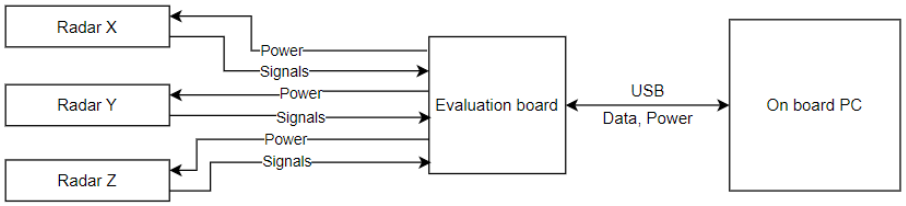

Figure 3.1: block diagram of the Evaluation board communication

3.1 Evaluation board connectivity

PCBR has the connectors described in subsection 2.2.1. The EB contains both connectors for all PCBRs, which means there will be six connectors on the board to connect the three PCBRs.

We used a 5-pin 2.54 mm header for programming the EB and connected ST-link V2 to this connector. It has a similar layout to the SWD connector on the Nucleo-F446RE.

Then, we used a USB mini connector for communication between EB and the UAV. The USB mini also supplies a 5 V supply voltage to power the EB and the connected PCBRs.

3.2 Connecting and setting up the MPU

The Micro-Processing Unit (MPU) STM32G030F6P6 from STMicroelectronics was used for data processing. This MPU has more than enough resources for this task and contains, for example, 2X UART, 8 timers, and 2 timers that can switch their outputs to encoder mode [8].

3.2.1 MPU power supply

The MPU has a required supply voltage ranging from 2 V to 3.6 V [8]. We used a voltage regulator XC6206P331MR-G. This regulator can stabilize a supply voltage of 5 V at 3.3 V. Its wiring is in Figure 3.2.

3.2.2 Pins Configuration

We configured the MPU pins in the STM32CubeMX application. The chosen configuration is in Figure 3.3, which shows the pins for the power supply (VDD, VSS), crystal oscillator connection, and UART. After finding the error described in section 3.4, we configured the oscillator's pins. Two PCBRs we connected to timers TIM1 and TIM3. We connected the third PCBR to pins PB7 and PA4. The remaining pins (SYS_SWCLK, SYS_SWDIO, NRST) together with 3.3 V and GND we connected to the 5-pin header described in section 3.1.

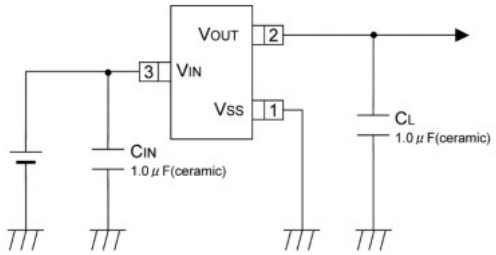

Figure 3.2: Wiring of the XC6206P331MR-G regulator [9]

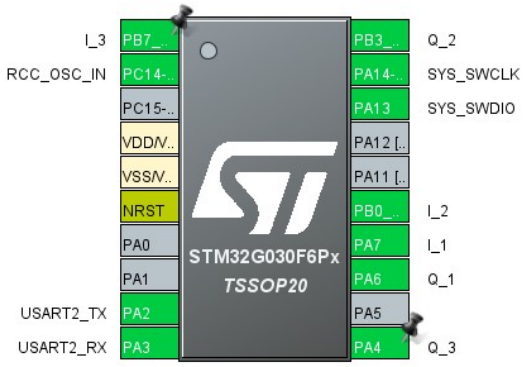

Figure 3.3: STM32G030F6P6 MPU pin configuration

3.2.3 MPU Wiring

The wiring of the MPU is in Figure 3.4. After finding the error described in section 3.4, we created the oscillator wiring. We chose the values of capacitors and resistors according to the documentation [8]. We used an external crystal oscillator with 16 MHz. It is not shown in Figure 3.4, but for MPU protection, we connected the outputs of PCBRs to the MPU via 470 Ω resistors.

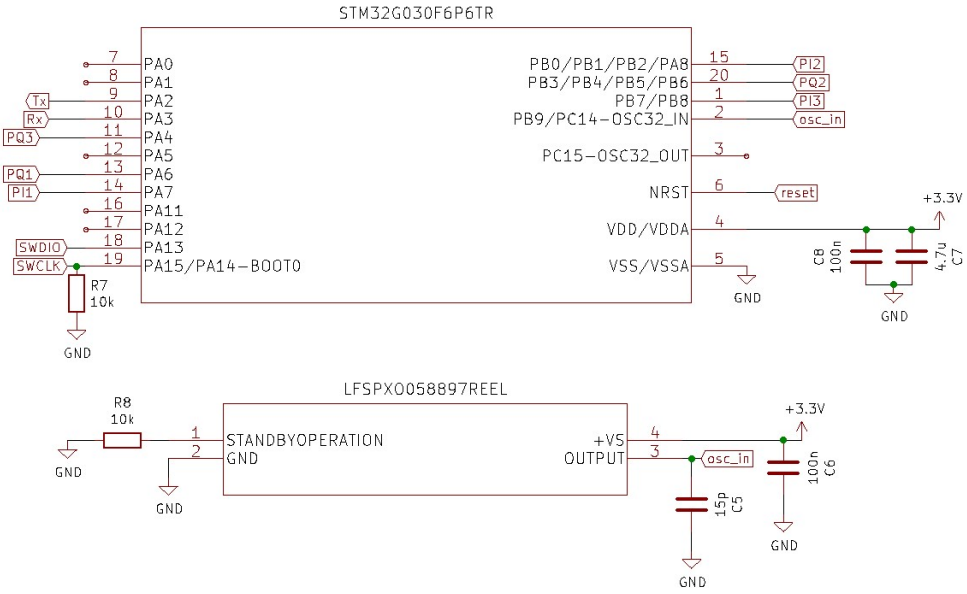

Figure 3.4: STM32G030F6P6 MPU wiring

3.3 UART to USB converter

In section 3.1, we mentioned using The USB mini for communication between the UAV and the MPU. However, the UAV communicates via USB and the MPU via UART. So a USB to UART converter must be added on the EB between the USB mini and the MPU. So we used the FT232 chip, its power supply via USB is in Figure 3.5. After connecting the converter and the MPU via UART, it was possible to communicate with the UAV.

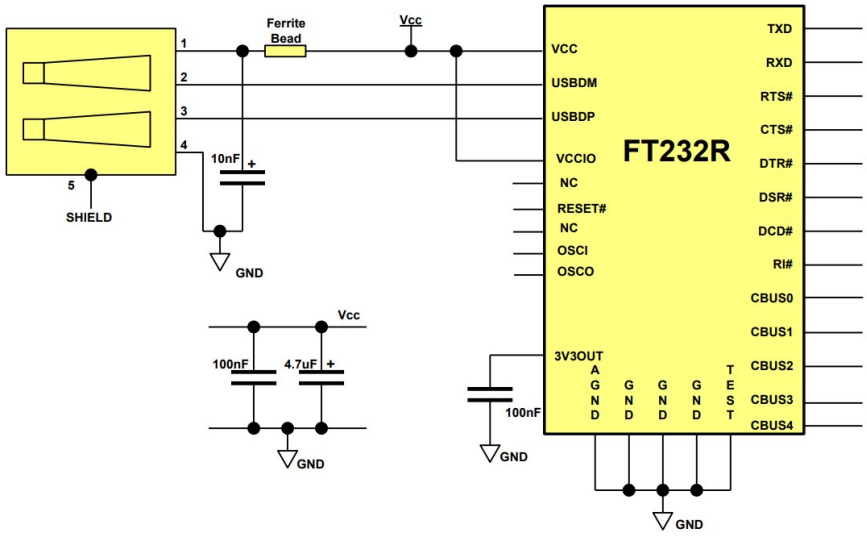

Figure 3.5: FT232 converter power supply [10]

3.4 Evaluation board design and implementation

The design of the EB is just like the PCBR from section 2.4 made in Kicad 5.15, and for mounting the EB, there are holes in each corner with a diameter of 3.2 *mm*.

During the flight, the cable could fall out of the USB connector, so we added holes around the connector for a zip tie. The zip tie helps to keep the cable connected. The holes allow us to use a zip tie up to 4 *mm* wide.

The first version is presented in Figure 3.6. After soldering, we discovered problems we did not consider or overlooked during the design.

The first problem is that we designed the board to include an HSE crystal resonator, but this MPU in HSE mode only supports an oscillator. The EB can work, but only on an internal clock source (HSI), which has some error rate compared to an external source.

The second problem is the protection of the EB when powered via ST-link. We can power the MPU via ST-link for programming, but the power supply is directly connected to the output of the regulator. Through this, the voltage reaches the whole PCB. If the PCBRs are not connected, we program the MPU without a problem. However, when PCBRs are connected, the voltage will start to feed them, and the total current may burn the regulator on the EB.

A preview of the second version is in Figure 3.7. This version differs only by adding or modifying components to fix bugs. To protect the regulator, we connected a Schottky diode in front of its input, which in case of power supply via ST-link prevents the voltage from reaching the rest of the board, and in case of power supply via USB mini, the diode has only a small voltage drop. We replaced the external resonator with an oscillator controlled by resistor R8, shown in Figure 3.4. The oscillator will work if R8 is not wired because it contains an internal pull-up resistor on the Standby Operation pin. When R8 is wired into the circuit, the pin will be connected to GND, and the oscillator will turn off. The attached file **eval_board.zip** contains the overall circuit, PCB layout, and components used.

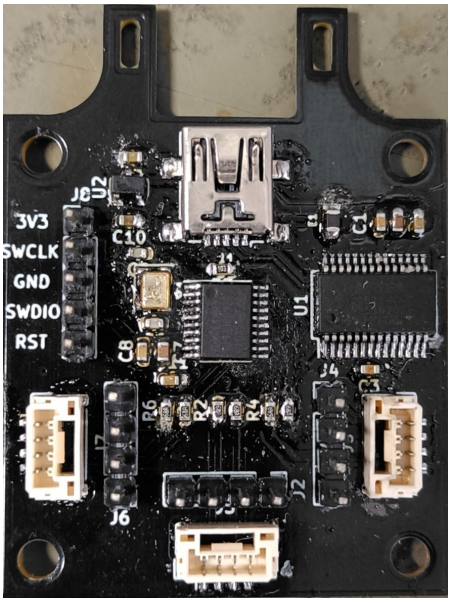

Figure 3.6: The first version of the evaluation board

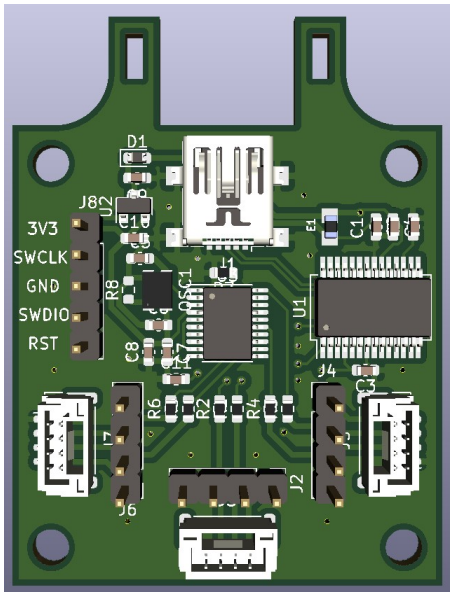

Figure 3.7: Preview of the second version of the evaluation board

Chapter 4

Design the mount for PCBs and attachment to a UAV

We designed all the needed PCBs. Now, we need to work out the PCBs layout and design the mount. The layout determines the parameters to tune the program for the EB. The radars will be inclined at an ω angle to measure velocity relative to the ground. A sketch is shown in Figure 4.1. The velocity measurement will work at lower heights independent of the surrounding environment.

The mount must be designed to attach to the UAV. We used Fusion 360 to lay out the PCBs. We imported 3D models of the PCBs from the Kicad into Fusion 360 and created the mount for them. Then we print the created mount on a 3D printer.

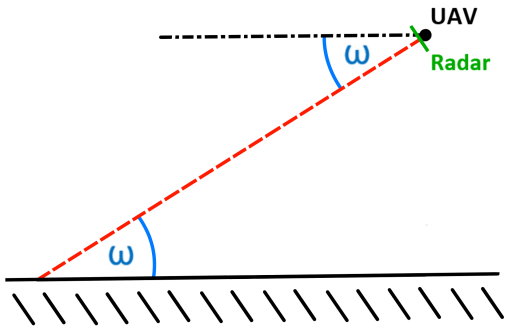

Figure 4.1: Sketch of radar inclination

4.1 Design of the PCBs mount

The mount needs to be solid, without taking too much space, and easy to mount and connect all PCBs. We decided to model objects with a minimum thickness of 2 *mm* to maintain strength. The resulting design is in figures 4.2 and 4.3.

In the figures, we arranged the PCBRs into an equilateral triangle with an inclination $\omega = 30°$ to the ground. The layout and tilt allow the radars to measure the velocity relative to the ground, and the EB converts the measured velocities to the UAV coordinate system (UCS). The EB is placed in the middle between PCBRs, but because the EB is larger than the centre space, it is moved a few millimetres lower. This placement allows simple wiring of all PCBs and ST-link wiring for program modification. The mount is attached from the bottom to the UAV through four holes with a diameter of 3.2 *mm*. There is a hole in the frame of

the mount for easier connection of the USB cable. The EB is not lowered enough to connect the USB cable without the hole. The orientation of the mount is such that the PCBR opposite the USB cable hole is in the direction of movement of the UAV.

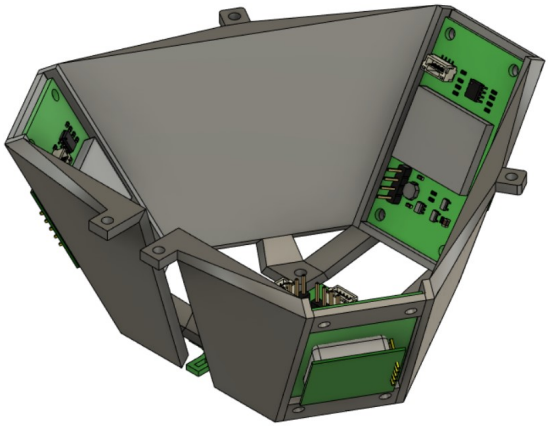

Figure 4.2: Side preview of the mount with PCBs

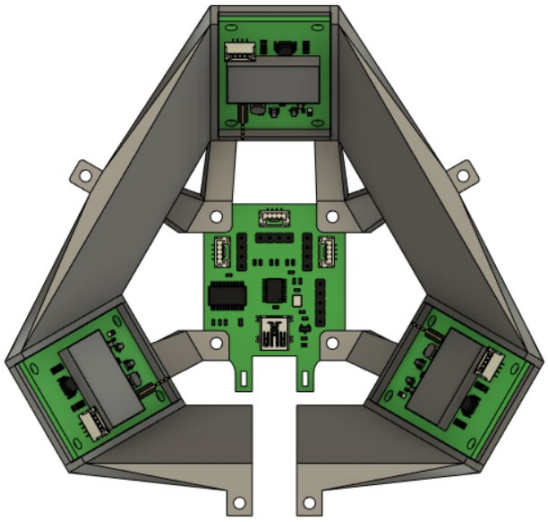

Figure 4.3: Top preview of the mount with PCBs

4.2 Modification to the battery cage

The battery cage is at the bottom of the UAV quadrocopter. The cage is shown in figures 4.4 and 4.5. It only has holes to attach to the UAV. Therefore, this cage must be modified to attach the PCBs mount.

The modified battery cage is in figures 4.6 and 4.7. After these modifications, it is possible to attach the mount to the battery cage, but it is more difficult to attach it to the UAV.

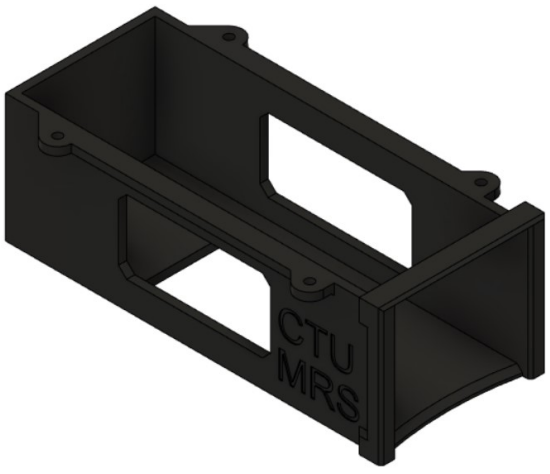

Figure 4.4: Preview of the back of the battery cage

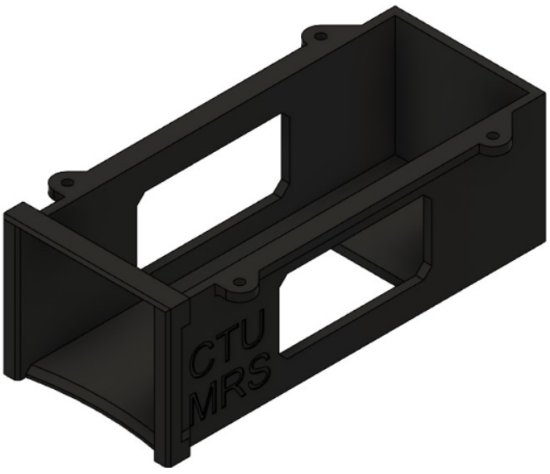

Figure 4.5: Preview of the front of the battery cage

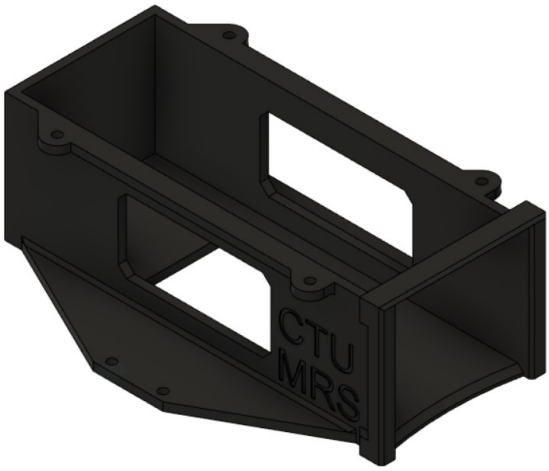

Figure 4.6: Preview of the back of the modified battery cage

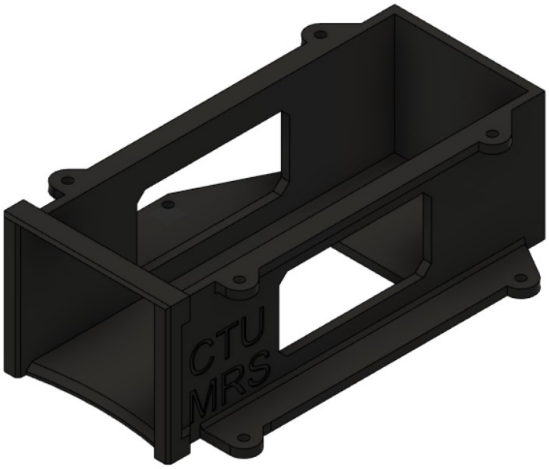

Figure 4.7: Preview of the front of the modified battery cage

4.3 3D printing

We used the Original Prusa i3 MK3S printer to print the mount and the modified battery cage. To print them, we had to export both designs from Fusion 360 to PrusaSlicer. In PrusaSlicer, we set the print parameters such as printer type, nozzle width, material, fill, and supports.

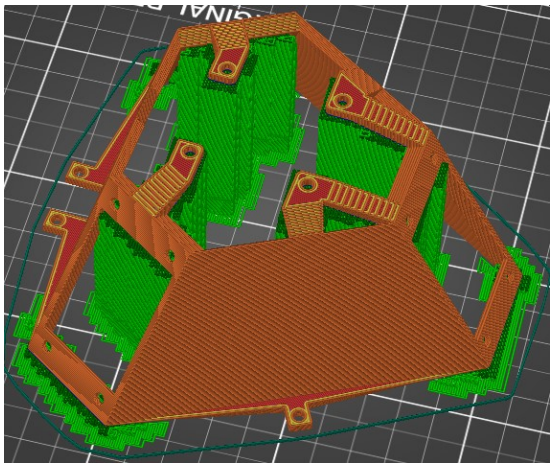

Figure 4.8: Preview of model for 3D printing of the PCBs mount

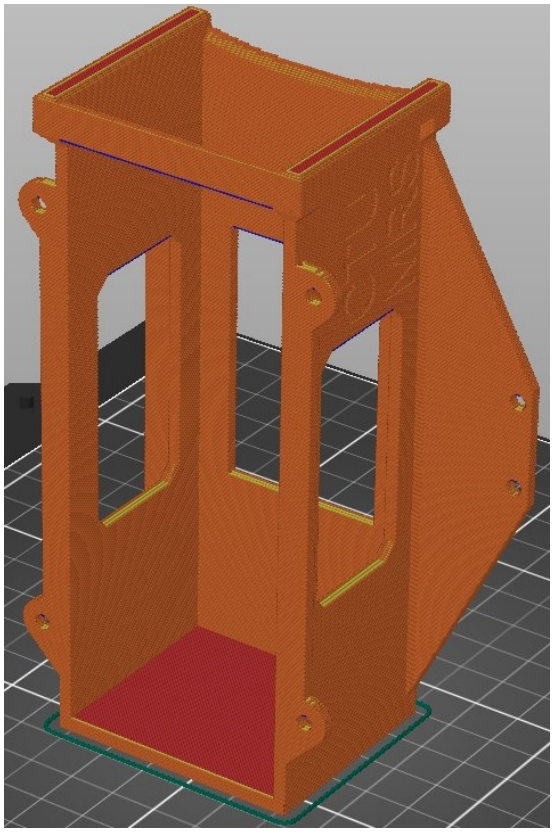

Figure 4.9: Preview of model for 3D printing of the modified battery cage

A preview of the model for printing the modified battery cage is in Figure 4.9. Even after adjustments for printing, no supports are needed. It is better to print the cage for more durability under battery load, as shown in Figure 4.9.

A preview of the model for printing the PCBs mount is in Figure 4.8. The mount has such a complex shape that the printer cannot print it without support. That is why we used a print setup with supports on the pad.

After setting all the parameters, we generated G-code for each 3D model in PrusaSlicer. The G-code is a file that defines a material and where the printer should print. The 3MF files of both mounts are in the attached file **mounts.zip**. These files can be opened in PrusaSlicer and set to print.

Chapter 5

Software

We set up the hardware. All that is left is to create the software. The software will ensure the correct reading of signals from the PCBRs, calculate velocities, and communicate the EB with the UAV.

5.1 Firmware for the Evaluation board

We wrote the firmware for the EB in C99 in the STM32CubeIDE application, which we can link with the STM32CubeMX application. This interfacing allows importing the MPU pin configuration directly into the code and thus simplifies the creation of the firmware.

We decided that the firmware will send a message to the UAV with a frequency of 5 *Hz*. That means the EB process signals from PCBRs measured in period $T_1 = 200\,ms$.

5.1.1 The MPU clock configuration

The clock signal determines how fast the MPU will process operations. The best capture the signals change from PCBRs, we have to set the MPU clock signal to the maximum supported frequency (64 *MHz* [8]). However, HSI can provide only 16 *MHz*. In the STM32CubeMX, we modified the clock configuration. The modified clock configuration is in Figure 5.1. A similar setup would work for the HSE oscillator. Depending on the oscillator frequency, some scaling parameters need to be adjusted.

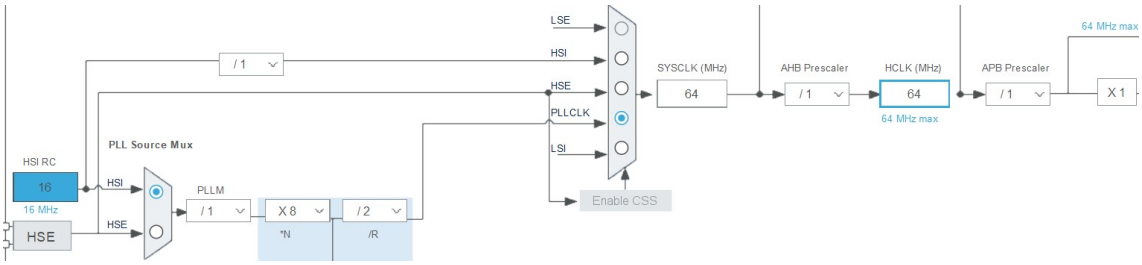

Figure 5.1: Modified MPU clock configuration

5.1.2 Frequency measurement

In subsection 2.3.4, we mentioned that the signals from PCBR look like the output signals of the rotary encoder. In subsection 3.2.2, we determined the con-

nection of each signal of PCBRs. With this information, we can set the MPU pins correctly.

We set the TIM1 and TIM3 timer pins to encoder mode. In this mode, the timer with hardware support processes two input signals and, according to them and the setting, counts pulses up/down and stores them in the internal counters, and the encoder mode filters out most of the noise. We set pins PB7 and PA4 to external interrupt mode, which will respond to both the rising and falling edges. We define a global variable type of *volatile int16_t* for these pins in the firmware, which will store the counted pulses. If an interrupt occurs on the PB7 pin, the firmware will ask what the logic level is on the PA4 pin and add or subtract one to the global variable accordingly. This setting should also filter noise, but it does not do it as well as encoder mode.

With this signals processing method, we can get a positive or negative number of pulses. A positive amount means the radar approaches the object from which the radar signal reflects, and a negative amount means receding from the object.

Pulses counting in encoder mode works similarly to interrupt counting, but the internal counter is of type *uint16_t*, but to calculate the rate, we need to read the value as *int16_t*. To increase sensitivity, we set the MPU to count pulses on the rising and falling edges of the rectangular I-signals. So the amount of counted pulses should be equal to double of frequency. The more number of samples ensures that the resulting frequency will be less affected by errors introduced by the counting.

5.1.3 Velocity calculation

Each PCBR represents a coordinate axis of the Radar coordinate system (RCS). In this system, we calculate the velocities from the counted pulses. The inclination ω will not be included in the calculation. As mentioned in section 1.2, the velocity is linearly dependent on the frequency of the radar signal. At velocity $v = 1\ m/s$ the Doppler frequency f_d of the output signal will be $158\ Hz$ [4]. The linear dependence between frequency and velocity is given by the formula:

$$f_d = v \cdot \frac{158Hz}{\mathrm{m/s}} \cdot \cos\alpha \text{ [4]} \tag{5.1}$$

The angle α expresses the difference from the perpendicular incidence of the reflected signal on the radar. In this case, $\alpha = 0$ because the reflected signal from the ground will be perpendicular to the radar, as is indicated in Figure 4.1. Next, we express the velocity from the formula:

$$v = \frac{f_d}{158} \tag{5.2}$$

We modified the formula (5.2) according to the defined program parameters. As mentioned in subsection 5.1.2, the measured frequencies are two times bigger than the actual ones, and the measurement period is T_1. So the measured frequencies must be approximated to a whole second and divided by two. The resulting formula:

$$v = \frac{f_d}{158} \cdot \frac{5}{2} \qquad [\mathrm{m/s}] \tag{5.3}$$

In the code, we replaced numbers with constants:

$$TIME\_FOLD = \frac{ONE\_SEC\_MS}{PERIOD} = \frac{1000}{T_1} = \frac{1000}{200} = 5 \tag{5.4}$$

$$v = \frac{f_d}{158} \cdot \frac{TIME\_FOLD}{DIVIS\_RAD} \tag{5.5}$$

With this modification, we can modify the calculation by changing the *PERIOD* constant. We assemble the vector $\overrightarrow{v_R} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$ from the calculated velocities of each PCBR.

5.1.4 Data sending

For the communication between the EB and the UAV, the MRS created the llcp [11] library. This library simplifies and standardizes communication with low-level devices based on UART [11]. To send the vector $\overrightarrow{v_R}$, we defined the message *data_msg*, which contains a variable *id* of type *uint8_t* to identify the message and three variables of type float for the value of the vector $\overrightarrow{v_R}$ components. The MPU sent this message via UART. The UAV expects UART with a specific profile - 115200 baud rate, 8 data bits, 1 stop bit, and no parity.

5.1.5 Periodic measurement execution

The timer callback function is executed when the timer counter counts up to a predefined value. We can set the timer to run the callback function repeatedly in a defined period.

In the function, we will load the measured values for period T_1, calculate the velocities and then send a message to the UAV. After sending the message, the function resets the variables in which we stored the measured frequencies of each PCBR for the next measurement period.

For activation of the callback function, we used the timer TIM16, which we have activated and set to global interrupt in the STM32CubeMX application. The timer counts according to the clock signal defined in subsection 5.1.1. For better timer setting, we set the prescaler to 64000 - 1. With this prescaler, the timer will not count according to the frequency 64 *MHz* but according to 1 *kHz*. With this frequency, the counter counts every millisecond. In subsection 5.1.3, we have already defined a constant defining the period T_1 in milliseconds, so we set the counter period to *PERIOD* - 1.

Since the MPU timers count from zero and not from one, we had to subtract one from the desired value of the prescaler and counter period. By using the constant *PERIOD* for the calculation and the callback function, we ensured easy modifiability of the code if we wanted to adjust the measurement period. The firmware is in the attached file **eval_board_fw.zip**.

5.2 Onboard computer software

The MRS has developed its system for their UAVs based on Robot Operating System [1] (ROS). We created the **radar_velocity** package that will work in this

system. The package receives the message, calculates the state vector, and sends it to another package in the UAV.

5.2.1 Receiving messages

For communication between low-level devices such as the EB and the MRS system, we use the llcp library mentioned in subsection 5.1.4 and the **mrs_llcp_ros** [12] package. This package converts the received message from the EB into a ROS message that the MRS system can handle.

Launching the **mrs_llcp_ros** package creates a node **/$UAV_NAME/llcp**, which creates the topic **/$UAV_NAME/llcp/received_message**. From the topic, we will read messages received from the EB. **$UAV_NAME** is the name of a particular UAV. In the case of testing on a computer, the name is set to **uav1** by default.

Our package reads a message from the topic. The message is an array of bytes. The first element in the array is the identification number, which we read as type *uint8_t*, the rest of the array are parts of the vector $\overrightarrow{v_R}$, and we read as type *float*.

5.2.2 Expression of the state vector in the UAV coordinate system

The vector $\overrightarrow{v_R}$ we must convert to the state vector $\overrightarrow{v_U}$, which is in the UCS. The RCS and the UCS we show in Figure 5.2. We used bright colours for the UCS and light colours for the RCS. To convert $\overrightarrow{v_R}$ to $\overrightarrow{v_U}$, we rotated UCS about the Y-axis by the angle $\beta = \omega = 30°$ and then about the X-axis by $\gamma = -135°$. The Rotated UCS (RUCS) is in Figure 5.3.

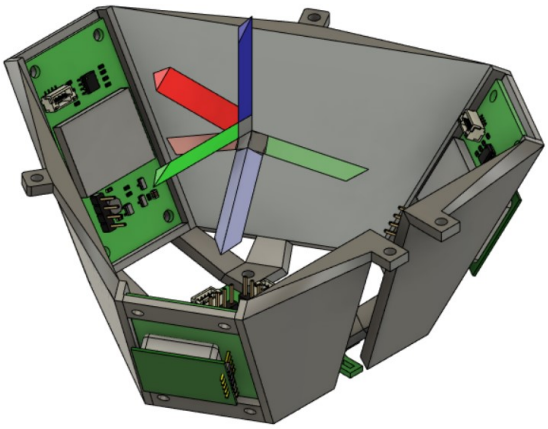

Figure 5.2: RCS and UCS relative to PCBRs

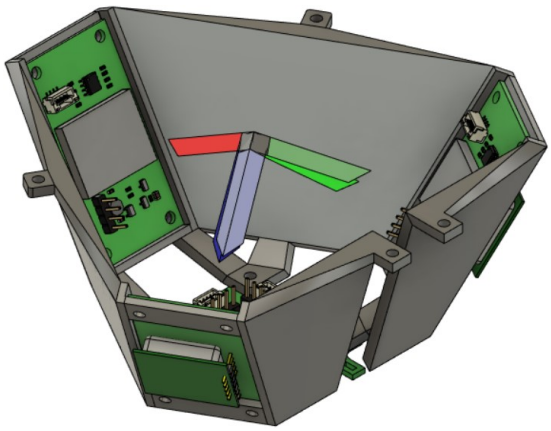

Figure 5.3: RCS and RUCS relative to PCBRs

From Figure 5.3, we can see that RCS has no 90° angles between the axes. To determine the angle δ between the axes of RCS, we create an equilateral triangle with vertices in the middle of PCBRs. From subsection 4.1, we know that PCBRs have an ω inclination. Combined with the equilateral triangle, we get the pyramid shown in Figure 5.4.

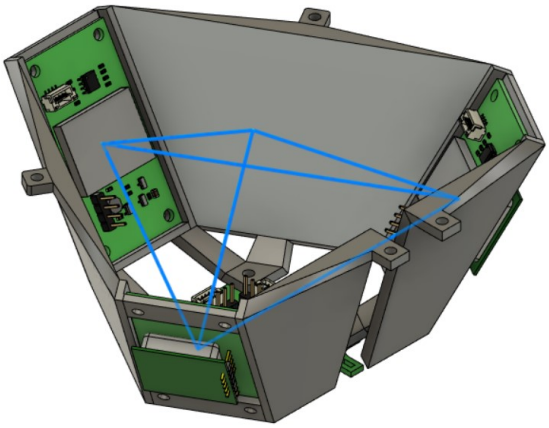

Figure 5.4: Illustration of a pyramid for angle δ computation

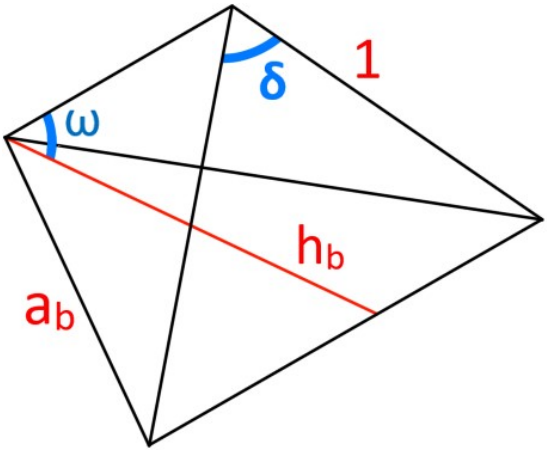

Figure 5.5: Sketch of a pyramid example

To calculate the angle δ, use the similarity, and we adjust the scale of the pyramid's edges. For example, the edges outside the pyramid's base are 1 cm long. A sketch of the example is in Figure 5.5. We calculate the angle δ:

$$\frac{2}{3} \cdot h_b = 1 \cdot \cos\omega = \cos 30^\circ = \frac{\sqrt{3}}{2} \tag{5.6}$$

$$h_b = \frac{3}{2} \cdot \frac{\sqrt{3}}{2} = \frac{3 \cdot \sqrt{3}}{4} \tag{5.7}$$

$$a_b = \frac{h_b}{\sin 60^\circ} = \frac{\frac{3\cdot\sqrt{3}}{4}}{\frac{\sqrt{3}}{2}} = \frac{3}{2} \tag{5.8}$$

$$a_b^2 = 1^2 + 1^2 - 2 \cdot 1 \cdot 1 \cdot \cos\delta \tag{5.9}$$

$$\cos\delta = -\frac{\frac{9}{4} - 2}{2} = -\frac{1}{8} \rightarrow \delta \approx 97.181^\circ \tag{5.10}$$

Since we know the angle δ, and we can determine the transformation matrix $\mathbf{T}$, which transforms from the RCS to the RUCS. For the matrix, we determine the setting:

$$T = \begin{bmatrix} \overrightarrow{t_1} & \overrightarrow{t_2} & \overrightarrow{t_3} \end{bmatrix} \tag{5.11}$$

$$\|\overrightarrow{t_1}\| = \|\overrightarrow{t_2}\| = \|\overrightarrow{t_3}\| = 1 \tag{5.12}$$

$$\overrightarrow{t_1} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \overrightarrow{t_2} = \begin{bmatrix} t_{2x} \\ t_{2y} \\ t_{2z} \end{bmatrix} \qquad \overrightarrow{t_3} = \begin{bmatrix} t_{3x} \\ t_{3y} \\ t_{3z} \end{bmatrix} \tag{5.13}$$

In Figure 5.3, we can see that the X-axis of the RUCS and the RCS overlap. The X-axis does not need to be changed. The angle between vectors of the matrix $\mathbf{T}$ is the same as the angle between the axes of the RCS. From this, we calculate the following:

$$\cos\delta = \frac{\overrightarrow{t_1} \cdot \overrightarrow{t_2}}{\|\overrightarrow{t_1}\| \cdot \|\overrightarrow{t_2}\|} = \frac{\overrightarrow{t_1} \cdot \overrightarrow{t_3}}{\|\overrightarrow{t_1}\| \cdot \|\overrightarrow{t_3}\|} = -\frac{1}{8} \tag{5.14}$$

$$t_{2x} = t_{3x} = -\frac{1}{8} \tag{5.15}$$

Since the X-axis is not transformed, there is a symmetric transformation of the Y-axis and Z-axis, therefore:

$$t_{2y} = t_{3z} \tag{5.16}$$
$$t_{2z} = t_{3y} \tag{5.17}$$

From the equation for the angle between the vectors $\overrightarrow{t_2}$ and $\overrightarrow{t_3}$, we express the variable t_{2z}:

$$\frac{\overrightarrow{t_2} \cdot \overrightarrow{t_3}}{\|\overrightarrow{t_2}\| \cdot \|\overrightarrow{t_3}\|} = -\frac{1}{8} \tag{5.18}$$
$$\frac{1}{64} + t_{2y} \cdot t_{3y} + t_{2z} \cdot t_{3z} = -\frac{1}{8} \tag{5.19}$$
$$t_{2y} \cdot t_{2z} + t_{2z} \cdot t_{2y} = -\frac{9}{64} \tag{5.20}$$
$$t_{2z} = -\frac{9}{128 \cdot t_{2y}} \tag{5.21}$$

We insert variable t_{2z} into the equation for the size of the vector $\overrightarrow{t_2}$:

$$\|\overrightarrow{t_2}\| = 1 \tag{5.22}$$
$$\sqrt{\frac{1}{64} + t_{2y}^2 + t_{2z}^2} = 1 \tag{5.23}$$
$$\frac{1}{64} + t_{2y}^2 + t_{2z}^2 = 1 \tag{5.24}$$
$$t_{2y}^2 + t_{2z}^2 - \frac{63}{64} = 0 \tag{5.25}$$
$$t_{2y}^2 + \left(-\frac{9}{128 \cdot t_{2y}}\right)^2 - \frac{63}{64} = 0 \tag{5.26}$$
$$16384 \cdot t_{2y}^4 - 16128 \cdot t_{2y}^2 + 81 = 0 \rightarrow s_1,\ s_2,\ s_3,\ s_4 \tag{5.27}$$
$$s_1 = -\frac{3}{8}\sqrt{\frac{7}{2} - 2\sqrt{3}} \tag{5.28}$$
$$s_2 = \frac{3}{8}\sqrt{\frac{7}{2} - 2\sqrt{3}} \tag{5.29}$$
$$s_3 = -\frac{3}{8}\sqrt{\frac{7}{2} + 2\sqrt{3}} \tag{5.30}$$
$$s_4 = \frac{3}{8}\sqrt{\frac{7}{2} + 2\sqrt{3}} \tag{5.31}$$

From equation 5.27, we get four possible solutions. From Figure 5.3, we can see that the RUCS and RCS have a similar direction to the Y-axis and Z-axis. Therefore the correct solution is the following:

$$t_{2y} = s_4 = \frac{3}{8}\sqrt{\frac{7}{2} + 2\sqrt{3}} \tag{5.32}$$
$$t_{2z} = -\frac{9}{128 \cdot \frac{3}{8}\sqrt{\frac{7}{2} + 2\sqrt{3}}} = -\frac{3}{8}\sqrt{\frac{7}{2} - 2\sqrt{3}} = s_1 \tag{5.33}$$

The resulting transformation matrix **T** is the following:

$$T = \begin{pmatrix} 1 & -\frac{1}{8} & -\frac{1}{8} \\ 0 & \frac{3}{8}\sqrt{\frac{7}{2}+2\sqrt{3}} & -\frac{3}{8}\sqrt{\frac{7}{2}-2\sqrt{3}} \\ 0 & -\frac{3}{8}\sqrt{\frac{7}{2}-2\sqrt{3}} & \frac{3}{8}\sqrt{\frac{7}{2}+2\sqrt{3}} \end{pmatrix} \approx \begin{pmatrix} 1 & -0.125 & -0.125 \\ 0 & 0.99 & -0.071 \\ 0 & -0.071 & 0.99 \end{pmatrix} \tag{5.34}$$

We express the rotations in terms of the angles β and γ using rotation matrices:

$$R_Y = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} = \begin{pmatrix} \cos(30°) & 0 & \sin(30°) \\ 0 & 1 & 0 \\ -\sin(30°) & 0 & \cos(30°) \end{pmatrix} \tag{5.35}$$

$$R_X = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(-135°) & -\sin(-135°) \\ 0 & \sin(-135°) & \cos(-135°) \end{pmatrix} \tag{5.36}$$

To calculate the state vector $\overrightarrow{v_U}$, we use the formula:

$$\overrightarrow{v_U} = R_Y \cdot R_X \cdot T \cdot \overrightarrow{v_R} \tag{5.37}$$

5.2.3 Sending message

Launching the **radar_velocity** package creates a node **/$UAV_NAME/radar**. The package creates two topics. For each topic, we send a message type *Twist-Stamped*. In the topic **/$UAV_NAME/radar/radar_velocity** we send the vector $\overrightarrow{v_U}$ and in the topic **/$UAV_NAME/radar/radar_raw** we send the vector $\overrightarrow{v_R}$. The vector $\overrightarrow{v_R}$ is used to check the data sent by the EB. The package is in the attached file **uav_software.zip**.

Chapter 6

Functionality testing

An MRS UAV [2] with 3D lidar was used for testing. We attach the PCBs to the UAV for verifying the in-flight functionality of the wiring and the software. The measured data will be compared with the estimation from SLAM computed from 3D lidar data.

6.1 PCB with radar inclination testing

When the second version of the PCBR was ready, we tested under which inclination it worked best. We connected the PCBR to an Arduino Nano for testing. The Arduino Nano sent messages to the UAV with a frequency of 10 Hz and recorded rising and falling edges using an external interrupt similar to the EB.

The PCBR was mounted in the X-axis direction of the UAV, under various inclinations ω to the ground. The resulting data were compared with estimated velocities from lidar data. We multiplied the measured data from the PCBR to get the X-axis velocity by $\cos(\omega + \delta_p)$. The angle δ_p expresses the inclination of the UAV around the Y-axis, which changes with different movements of the UAV during flight. The best result was an inclination of $\omega = 30°$. The measurements are shown in Figure 6.1. The black line is measured velocity from the lidar and red from the PCBR. We can see that the PCBR is more sensitive to UAV shaking than the lidar. Otherwise, the velocities are similar, so this inclination was used in the mount design in section 4.1.

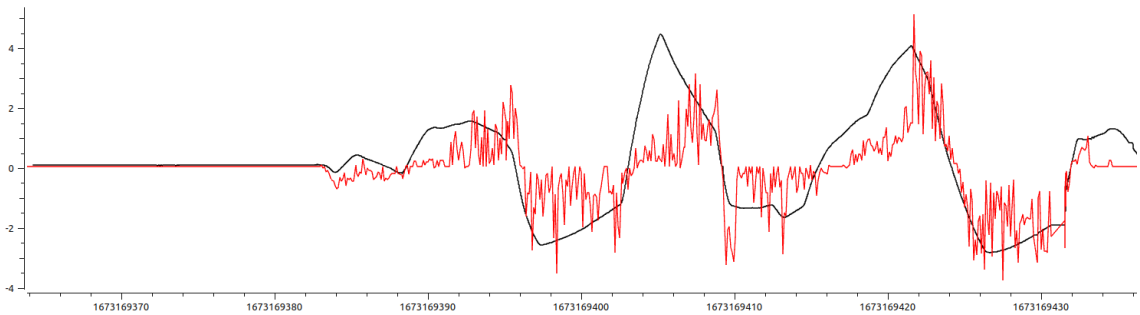

Figure 6.1: Velocity comparison between PCBR and lidar in the X-axis

6.2 The state vector computation test

6.2.1 adjustment of velocity calculation

The UAV with the lidar for the test flight has a different design than the UAV for which the 3D models have been designed in subsections 4.1 and 4.2. The battery cage can be attached to the UAV with lidar, but it is rotated by $-90°$ about the Z-axis. Therefore, we have to modify the formula 5.37 using the rotation matrix:

$$R_Z = \begin{pmatrix} \cos(-90°) & -\sin(-90°) & 0 \\ \sin(-90°) & \cos(-90°) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6.1}$$

$$\overrightarrow{v_U} = R_Z \cdot R_Y \cdot R_X \cdot T \cdot \overrightarrow{v_R} \tag{6.2}$$

6.2.2 Test results

The measured values are in Figures 6.3, 6.4, and 6.5. Black lines are measured velocities from lidar and other colours are components of the state vector $\overrightarrow{v_U}$. Radars are more sensitive to in-flight shaking of the UAV than lidar, so their waveforms are wavier than lidar waveforms. Otherwise, we can see that the Y-axis and Z-axis state vector waveforms are similar to the lidar waveforms. Values in these two axes could be used to stabilize the UAV, but values in the X-axis could not. The values from radars on the X-axis are significantly smaller than those from the lidar. So this solution cannot be used to stabilize the UAV because the inaccuracies in the X-axis measurements would result in a UAV crash.

We tried to add different shielding around the radars to reduce the Field of view (FOV). We tried a shielding that was around the whole radar up to 3 cm above the surface of the radar or less. Alternatively, a shielding that was only on two adjacent sides of the radar. All the test flights with these shieldings had worse results than the flight without shielding.

Many factors, like high FOV, interference between radars, or inefficient radar layout, could have affected functionality. The K-LC5 radar has quite a large FOV, which can cause the evaluation of unwanted environmental effects. A sketch of how the radar with high FOV could detect multiple objects simultaneously is shown in Figure 6.2. In flight, one of the detected objects may approach the radar, and the other one recedes, which may affect the frequency of the radar output signal. With a smaller FOV, the radar will not scan as many objects, and the resulting output signal frequency will not be affected as much. There could be interference between radars, which can also be caused by a high FOV. The solution would be to replace the K-LC5 radar with another radar that has a smaller FOV, for example, the K-MC1 radar [13]. Another factor that could affect the functionality is the distribution of the radars. From the testing in section 6.1, we know the radar can return fairly accurate values if it is below an inclination ω in the direction of the UAV's axis of motion. It is possible that the radars are not designed for the current layout and do not measure quite correctly. The solution would then be to choose a different layout in which the radars would work better.

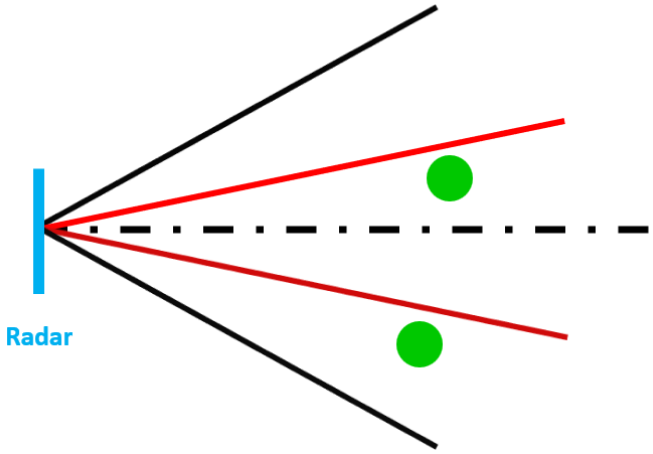

Figure 6.2: various FOVs sketch

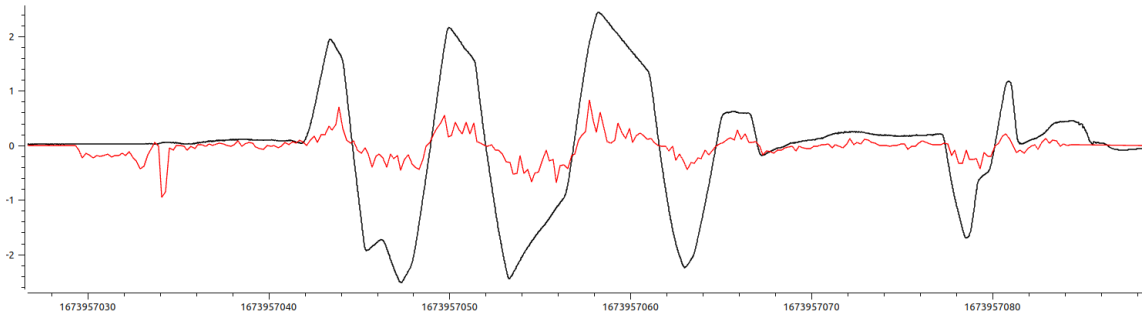

Figure 6.3: Velocity comparison between lidar and radars in the X-axis

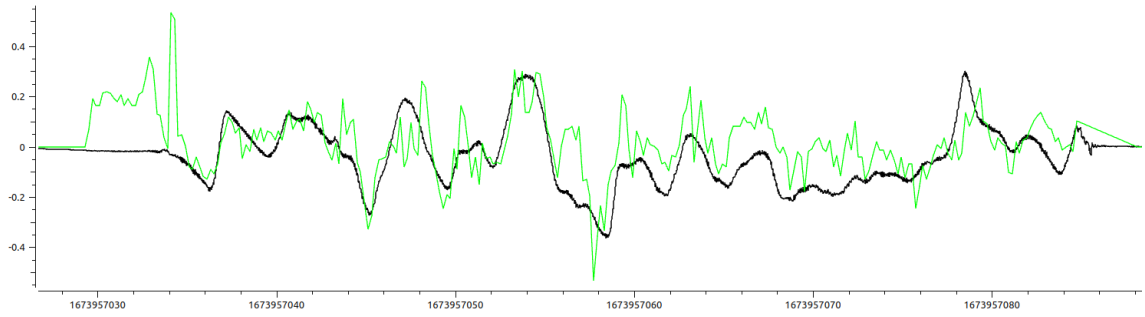

Figure 6.4: Velocity comparison between lidar and radars in the Y-axis

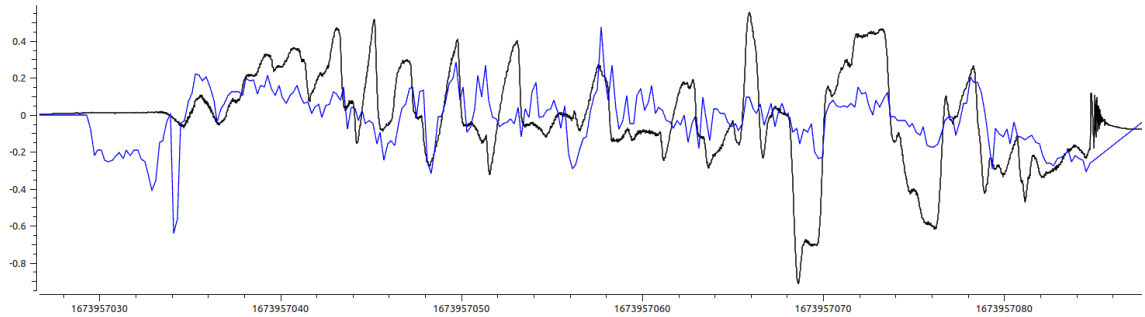

Figure 6.5: Velocity comparison between lidar and radars in the Z-axis

Conclusion

In this thesis, we developed a compact device with three doppler radars that can be attached to the UAV and measured its velocity. The device is composed of two types of PCBs and the mount for PCBs.

The first type of PCB (PCBR) is connected to one doppler radar and amplifies its signals and converts them to rectangle signals. These signals are evaluated by the second type of PCB (EB). The EB uses Micro-Processing Unit STM32G030F6P6 for the evaluation and computes velocities in the Radar coordinate system (RCS). When the signals from all three PCBRs are evaluated, it sends velocities to the UAV.

The mount is designed to determine the layout of the radars that are pointed at the ground at a certain angle. The most effective inclination angle was chosen by test flight with one PCBR and an Arduino. The layout of PCBRs sets the RCS. The mount attaches to the bottom of the UAV to the battery cage, so the cage had to be modified. Both the mount and the battery cage are 3D printed.

A ROS node was created for communication between EB and the UAV. The node computes the UAV state vector from received velocities. Two rotations and one transformation matrix are used for the computation. The node sends the state vector to another ROS node.

A test flight verified the functionality of the device. The test flight showed that the device does not work as it should. The values of the calculated state vector match the lidar reference values in only two axes. The device in this state cannot stabilize the UAV. It would cause the UAV to crash.

The functionality of the device could have been affected, for example, by high FOV, interference between radars, or inefficient radar layout. In the future, the MRS could design a new layout for the current radars or use different radars with smaller FOV. Alternatively, design a new device to be assembled with more radars and more Micro-Processing Units.

Fulfilment of partial tasks of the thesis

The partial tasks of the thesis are:

- Design and build a printed circuit board which amplifies and processes the signal from a doppler radar module - elaborated in chapter 2.

- Design and build a printed circuit board which interfaces with an onboard computer on the UAV and provides power and communication to the individual doppler radar modules and their amplifiers - elaborated in chapter 3, firmware in section 5.1.

- Design and implement an interface between the ROS [1] based MRS UAV [2] system, and the doppler radar interface board - elaborated in section 5.2.
- Implement a state estimator which uses the measurements from the doppler radars along with measurements from the UAVs inertial measurement unit to estimate the current state vector of the UAV - elaborated in subsection 5.2.2.
- If external factors and the schedule of the MRS group allow, demonstrate the UAV with the doppler radar stabilization in a real-world experiment - the experiment was not performed.

All tasks can be considered fulfilled except for the stabilization in a real-world experiment. There was no time left to order better radars (K-MC1). Therefore it was not possible to do the experimental flight.

Bibliography

1. QUIGLEY, M.; CONLEY, K.; GERKEY, B.; FAUST, J.; FOOTE, T.; LEIBS, J.; BERGER, E.; WHEELER, R.; MG, A. ROS: an open-source Robot Operating System. In: *ICRA Workshop on Open Source Software*. 2009, vol. 3. No. 3.2.
2. BACA, T.; PETRLIK, M.; VRBA, M.; SPURNY, V.; PENICKA, R.; HERT, D.; SASKA, M. The MRS UAV System: Pushing the Frontiers of Reproducible Research, Real-world Deployment, and Education with Autonomous Unmanned Aerial Vehicles. *Journal of Intelligent & Robotic Systems*. 2021, vol. 102, no. 26, pp. 1–28.
3. RFBEAM MICROWAVE GMBH. *Data sheet K-LC5 radar transceiver* [online]. 2021 [visited on 2023-04-12]. Available from: `https://rfbeam.ch/wp-content/uploads/dlm_uploads/2022/11/K-LC5_Datasheet.pdf`.
4. RFBEAM MICROWAVE GMBH. *Typical Doppler Signal Amplifier* [online]. 2014 [visited on 2023-01-21]. Available from: `https://rfbeam.ch/wp-content/uploads/dlm_uploads/2022/10/AN-04-TypicalSignalAmp.pdf`.
5. ON SEMICONDUCTOR. *BC337 - Amplifier Transistors NPN Silicon* [online]. 2007 [visited on 2022-09-15]. Available from: `https://www.onsemi.com/pdf/datasheet/bc337-d.pdf`.
6. BCD SEMICONDUCTOR. *datasheet AP3012* [online]. 2010 [visited on 2022-01-07]. Available from: `https://www.diodes.com/assets/Datasheets/AP3012.pdf`.
7. STMICROELECTRONICS. *datasheet L78L* [online]. 2021 [visited on 2022-01-07]. Available from: `https://www.st.com/resource/en/datasheet/l78l.pdf`.
8. STMICROELECTRONICS. *STM32G030x6/x8* [online]. 2022 [visited on 2022-06-10]. Available from: `https://www.st.com/resource/en/datasheet/stm32g030k8.pdf`.
9. TOREX SEMICONDUCTOR. *XC6206* [online]. [N.d.] [visited on 2022-06-11]. Available from: `https://www.torexsemi.com/file/xc6206/XC6206.pdf`.
10. FTDI CHIP. *Future Technology Devices International Ltd. FT232R USB UART IC Datasheet* [online]. 2020 [visited on 2022-06-10]. Available from: `https://ftdichip.com/wp-content/uploads/2020/08/DS_FT232R.pdf`.
11. HERT, D. *GitHub - ctu-mrs/mrs_llcp*. 2020-04. Available also from: `https://github.com/ctu-mrs/mrs_llcp`. [visited on 2022-10-12].
12. HERT, D. *GitHub - ctu-mrs/mrs_llcp_ros*. 2022-04. Available also from: `https://github.com/ctu-mrs/mrs_llcp_ros`. [visited on 2022-10-12].

13. GMBH, RFbeam Microwave. *data sheet K-MC1 radar transceive* [online]. 2022 [visited on 2023-05-14]. Available from: `https://rfbeam.ch/wp-content/uploads/dlm_uploads/2022/11/K-MC1_Datasheet.pdf`.

Appendix

Schematic, PCB design, list of components

radar.zip - PCB with radar
eval_board.zip - Evaluation board

3D models

mounts.zip - Models of the mount and the battery cage

Software

eval_board_fw.zip - Firmware for the Evaluation board
uav_software.zip - Onboard computer software