TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Department of Mechanics
Chair of Mechatronics Systems

MHK70LT

Martin Karmin

DEVELOPMENT OF THE ANTENNA TRACKER DEVICE FOR THE GROUND STATION OF THE UNMANNED AERIAL SYSTEMS

MEHITAMATA ÕHUSÕIDUKITE MAAPEALSE JUHTIMISKESKUSE ANTENNISUUNAMISSEADME VÄLJAARENDAMINE

MSc Thesis

The author applies for
The academic degree
Master of Science in
Engineering

Tallinn 2016

AUTHOR'S DECLARATION

I declare that I have written this graduation thesis independently.
These materials have not been submitted for any academic degree.
All the works of other authors used in this thesis have been referenced.

The thesis was completed under .... supervision
“.......”....................201….

Author
.... signature

The thesis complies with the requirements for graduation theses.
“.......”....................201….

Supervisor
.... signature

Accepted for defence.

.... chairman of the defence committee
“.......”....................201… .
.... signature

TUT Department of Mechatronics
Chair of Mechatronics Systems

MASTER'S THESIS TASK

Year 2016 Spring Semester

Student:	Martin Karmin, 144236MAHM
Curricula:	MAHM 02/13
Specialty:	Mechatronics
Supervisor:	Early-stage researcher Märt Juurma
Consultants:	Kristjan Tiimus, CTO Threod Systems, kristjan@threod.com

MASTER'S THESISTITLE:

(English)	Development of the antenna tracker device for the ground station of the unmanned aerial systems.
(Estonian)	Mehitamata õhusõidukite maapealse juhtimiskeskuse antennisuunamisseadme väljaarendamine.

Assignments to be completed and the schedule for their completion:

Nr	**Description of tasks**	**Timetable**
1.	Determining the design requirements. Research of existing solutions and methods to find out the technical requirements for the new solution.	14th of March
2.	Choosing a concept, components and designing a test model for the solution, preparing experimental work.	28th of March
3.	Testing the suitability of the actuators and component integration.	Week 14&15 (of solar calendar)
4.	Evaluating the design and listing possible improvements. Preparation for additional testing.	18th of April
5.	Compiling and formatting thesis.	16th of May

Engineering and economic problems to be solved:

Creating a new antenna tracker design that would be more compact than predecessors. Reducing the price for obtaining the antenna tracker.

Additional comments and requirements: None

Language: English

Thesis application submitted no later than: 16.05.2016

Deadline for submitting the thesis: 20.05.2016

Student	Martin Karmin	date
Supervisor	Märt Juurma	date

TABLE OF CONTENTS

EESSÕNA

Käesolev töö teostati Tallinna Tehnikaülikoolis ning koostöös Threod Systems OÜ-ga. Tegemist on iseseisva projektiga mille eesmärgiks oli mehitamata õhusõidukite maapealse juhtimiskeskuse juurde kuuluva kompaktse raadiotransiiveri antennisuunamisseadme väljaarendamine. Käesolev projekt on ajendatud Threod Systems OÜ praktilisest vajadusest kaasaskantava antennisuunaja väljaarendamiseks.

Antud seade on mõeldud tagamaks optimaalset raadiosidet mehitamata õhusõiduki ja maapealse juhtimiskeskuse vahel. Selleks orienteeritakse raadiotransiiveri suundantenni peavihk mehitamata õhusõiduki suunal kogu selle lennu ajal sõltumata õhusõiduki asukohast ja asendist transiiveri suhtes. Käesoleval hetkel on enamus turul pakutavatest antennisuunajatest mõeldud kasutamiseks mobiilsetes juhtimispunktides, mis baseeruvad enamasti motoriseeritud platvormil. Seega on valdav osa turul pakutavaid seadmeid suhteliselt suured ja rasked ning seetõtu on nende kasutamine jalgsi liikuva ekipaaži poolt raskendatud. See ongi üheks ajendiks uue lahenduse väljatöötamiseks. Teiseks ajendiks võib lugeda turul pakutavate lahenduste integreeritavust ettevõtte olemasolevate süsteemide ning lahendustega. Praegusel hetkel tuleb sisseostetud lahendusi tugevalt modifitseerida, et need ühilduksid kasutuses olevate seadmetega. Kolmandaks ajendiks võib välja tuua sõltumatuse kommertslahenduste pakkujatest. Nimelt kuna olemasolevate lahenduste tootjad asuvad üle maailma laiali, kaasneb toodete ja varuosade hankimisega arvestatav ajaline ja rahaline kulu transpordi, tollimaksude ja muu säärasega. Firmasisese lahenduse loomine aitaks paindlikumalt opereerida tootmise ja tarnimise kulude ning ajaga.

Tehtud töö käigus arendati välja antennisuunaja mehhaaniline kontseptsioon ja lahendus ning selle juhtimise põhimõtted, seadmed ja võimalikud komponendid. Antud kontseptsiooni loomisel järgiti põhimõtet hoida seadme mõõtmed ja kaal minimaalsed, seejuures tagades funktsionaalsuse kriteeriumid. Antud töö kahes esimeses peatükis defineeritakse konstrueerimise kriteeriumid ja vaadeldakse turul pakutavaid lahendusi. Järgmises peatükis pakutakse välja kontseptsioon. Neljandas peatükis kirjeldatakse konstrueerimise protsessi ja koostatakse esmane mehaaniline kavand ning elektrisüsteemide lahendus. Valitakse ka komponendid ning andmeside lahendus.

Luuakse ka kavand juhtprogrammile. Viiendas peatükis katsetatakse esmase kavandi põhjal süsteemi komponentide toimimist ning integratsiooni. Kuues peatükk tutvustab süsteemi lõplikku lahendust. Lõplikku lahendusse on viidud sisse muudatused katsetamise käigus ilmsiks tulnud probleemide lahendamiseks. Lõplikule versioonile on koostatud kolmemõõtmeline CAD mudel, toodetavate komponentide joonised, komponentide loend ja elektrisüsteemi eskiis koos juhtprogrammidega.

LIST OF FIGURES

LIST OF ABBREVIATIONS AND ACRONYMS

DC - *Direct Current*

GCS - *Ground Control Station*

ISR - *Interrupt Service Routine*

LH- *Left Hand*

PC - *Personal Computer*

PCB - *Printed Circuit Board*

RH - *Right Hand*

SPI- *Serial Peripheral Interface*

UAS - *Unmanned Aerial System*

UDP - *User Datagram Protocol*

USB - *Universal Serial Bus*

1 INTRODUCTION

This thesis was conducted at the Tallinn University of Technology. The thesis would result with development of the design of antenna tracker device for the ground station of the unmanned aerial systems. This problem was presented by Threod Systems OÜ and is driven by the practical need for customized, lightweight, portable and low-cost tracker solution for directional antennas. The antenna trackers are used to ensure an optimal radio link between the ground control station and the unmanned aerial system. The tracker points the main beam of the directional antenna at the aerial system during the whole duration of the flight, irrespective of the position or the attitude of the aircraft. By keeping the antenna's main beam focused on the aircraft, the tracker ensures the optimal radio visibility and thus the optimal radio connectivity. The tracker would need to be man carried, tripod mounted device that would be compatible with other relevant systems and components that the company uses.

The majority of currently available solutions are designed to be used in conjunction with the mobile command centers which usually are based on motorized platforms. That makes the current solutions regarding antenna trackers either too bulky and heavy to be suitable for man carrying. This is one of the driving motives to develop a more compact solution. As the second driver, compatibility can be pointed out. With the solutions currently used, there exists the need to modify the existing devices heavily to achieve compatibility with the other systems that the company uses. By using bought-in solutions there is also an aspect of dependability on the manufacturing companies in the cases of breakdowns and malfunctions. And since the manufacturers of current solutions are scattered around the globe, the time and cost factor of acquiring the products and spare parts becomes evident. Therefore it is reasonable to develop an in-house solution, tailored for the needs of Threod Systems.

The problem shall be addressed by following the common stages of the engineering design process. The first two chapters focus on determining the design requirements and the research of existing solutions and methods. The third chapter offers a conceptual design. In the fourth chapter the design process is described and the preliminary mechanical and electric designs are being created; also the components selected, the data communication specified and the architecture for the code created. In the fifth chapter the integration of the electric devices and the performance of the actuators is tested. In the sixth chapter the detailed design is described. For the detailed design the Siemens NX9 three dimensional assembly model is created along with two dimensional manufacturing schematics for the parts needed to be manufactured. The electrical design is drafted with Eagle Cad. The control codes used in this thesis are composed with MATLAB and Arduino IDE software. All the documentation mentioned is included in the appendixes. The thesis has served its purpose and can be regarded as successful if the prototype of the tracker can be built based on this work.

2 THE DESIGN REQUIREMENTS

2.1 Specification of design requirements

Since the development of the antenna tracker has been initiated by Threod Systems OÜ, the main design requirements are driven by the needs of the company. To determine the main design criteria, the requirements were inquired from the representative of Threod Systems OÜ. One of the requirements is the cost of one unit. The cost should be competitive compared to the other trackers available at the market. The second criteria is portability, the device should be compact and easy to carry by one person along with other items necessary to operate the UAS. The tracker device is likely not the only piece of equipment the person is carrying, therefore the desired weight of the tracker device should not exceed 5 kg and the overall volume should be in a magnitude of 1,5…3 dm^3. In those constraints, the tracker device could be easily carried in a patrol backpack for example. The device is meant for mounting the MARS Systems MA-WA24-2X High Gain Subscriber Antenna and/or the Chushcraft S2403B omnidirectional antenna, thereby the mounting and actuation capacity of the tracker could to be up to 2 kg. The usual UAS mission profile dictates the requirements for the actuation range of the tracker. The device should be able to move 360 degrees in the vertical axis and from -10 degrees to +90 degrees in the horizontal axis. The rotation speeds have to be in a magnitude of approximately 36 degrees per second and resolutions in both axis 10 steps per degree. The accuracy of the device has to be less than 1 degree. The tracker must be able to receive its commands through an Ethernet cable using one of the common Ethernet protocols.

2.2 Products on the market

In order to investigate the existing solutions and methods used in the industry, a background research was carried out. The datasheets and inquires were gathered from the following companies regarding their relevant products: Optimum Solutions Engineering LLC: *OS-PT-25*; Latitude Engineering: *Basic Antenna*; Marcus UAV Corp.: *Long Range Tracking Antenna System* and Steatite Limited: *Wave Relay Tracking Antenna System*. The image below pictures two of the antenna trackers available [1] [2]. In general, most of the observed solutions used the following layout: the trackers consist of two part housings - the lower ones were static, attached to the tracer mounting device (tripod vehicle etc.) and the upper housings have the ability to be rotated around a vertical axis and they mount a horizontal axis, which actuate the antenna mounts.

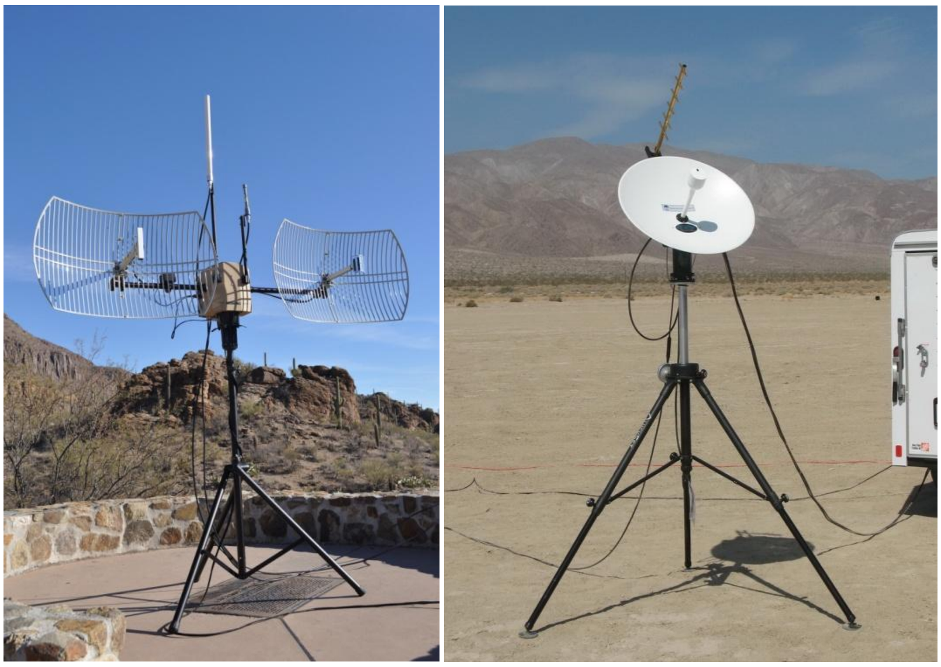


Figure 2.1. Latitude TA-i20 and Optimum OS-PT-25 antenna trackers

Both axis are actuated either by open loop controlled stepper motors or closed loop controlled direct current (DC) motors. In either solutions the output motions of the motor shafts are strongly reduced by planetary-, or spur gears. In most of the solutions there were no independent position detectors embedded to the antenna trackers. The orientation commands of the tracker are provided by the ground control station and they rely on the GPS data received from the UAS. The existing solutions used a wide range of data communication protocols: RS-232 (TA-i20), RS-485(OS-PT-25), USB (Marcus UAV), Ethernet (Seatite Wave Relay). The cost of the trackers varies in the range of 8995 USD (Marcus UAV) to 33000 USD (Latitude Engineering). The general dimensions of the tracker bodies ranged in an overall magnitude from 286 mm height, 199 mm width, 107 mm diameter for the OS-PT-25 tracker to 265 mm height, 244 mm width and 200 mm diameter for the Wave Relay antenna tracker. The corresponding volumes range from 6,08 liters (dm^3) to 12,93 liters accordingly. The corresponding weights of the trackers are 8,7 kg for the OS-PT-25 and 7,2 kg for the Wave Relay antenna tracker. The supply voltages of the observed tracker range from the most common 12 V direct current to 220 V alternating current. The power consumptions vary in the range of 3 A to 5,5 A. The usual range of the actuation in the vertical axis is full 360° rotation, but the tilt around the horizontal axis varies by the tracker. For example, the Wave Relay antenna tracker has a tilt range of 180° (+-90°) and the OS-PT-25 has a tilt range of 140° (-10°...+130°). The rotation speeds of the trackers vary from 10 degrees per second (OS-PT-25) to 25 degrees per second (Wave Relay). The resolution was mentioned only in the datasheet of the OS-PT-25 tracker: +-0,5 degrees.

Taking into account the design requirements that were specified above and comparing them to the parameters of the existing solutions available on the market, it can be conjectured that the development of the compact version of the antenna tracker with similar operation characteristics is possible. Since the payload capacity for the current design (2 kg) is less than Wave Relay (7,2 kg) for example, the tradeoff can be made - downsizing motors and transmission and thereby reducing the weight and the volume of the tracker device.

3 CONCEPTUAL DESIGN

3.1 System layout

The antenna tracking system consists of an external power source, a Ground Control Station (GCS) and an antenna tracker device. The control signals are sent from the GCS to the antenna tracker device via Ethernet. The Ethernet signal is being converted to a serial signal that the microcontroller can process by an Ethernet to serial converter. The onboard microcontroller controls the stepper-, or DC motor drivers, which in turn are providing step pulses or driving current to the motors. If the device is actuated by the DC motors, then the closed loop feedback of the motion is being acquired by the rotary encoders. In case of the stepper motors, the encoders may be necessary if additional position determination is required. To avoid potential damage to the device, both axis are equipped with end-switches or hall sensors to determine stop positions or neutral points. The system is being powered by an external 12...24V power source and the tracker device has an embedded regulated power supply to provide the required voltage to the Ethernet to serial converter and to the microcontroller, also providing the necessary power to the motor drivers. The overview of the layout is shown in the figure below.

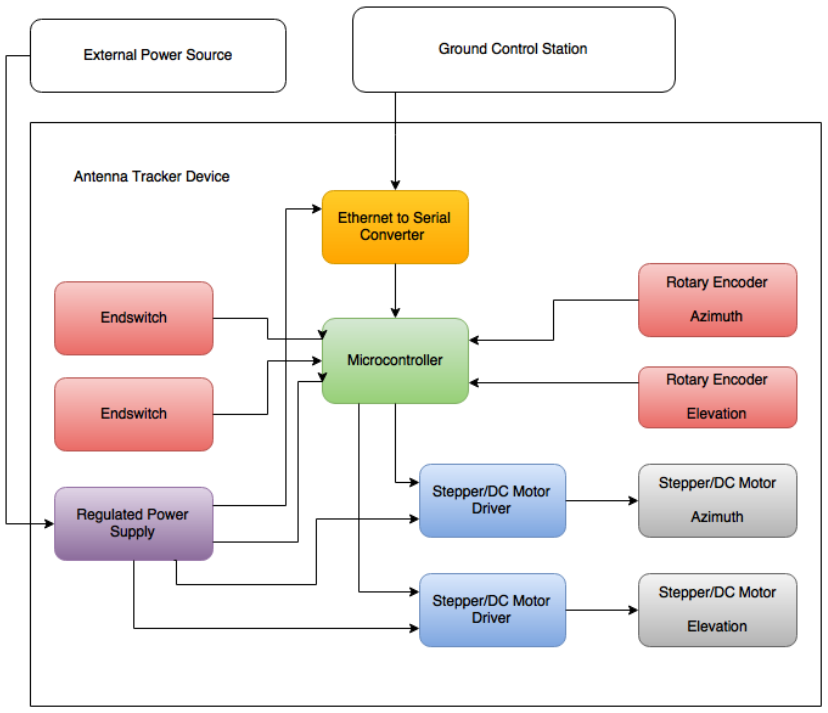


Figure 3.1 Antenna tracker system layout

3.2 Mechanical concept

In the initial mechanical concept the antenna tracker device consists of two separate housings: the upper housing and the lower housing. The cylindrical lower housing is fixed to the mounting device (tripod, vehicle, etc) and the upper housing is attached to the lower housing by a rotating vertical shaft. The vertical shaft is supported by a pair of angular contact bearings. The vertical shaft is being actuated by the motor situated in the lower housing via a power transfer mechanism, like spur gears or timing belt and pulleys. The upper housing is formed by joining two cylinders in T-shape. The upper housing holds the horizontal axis motor and it actuates the horizontal shaft by a similar power transfer mechanism. The horizontal shaft is mounted to the upper housing by a pair of bearings. The antenna mount is attached to the horizontal shaft from both ends.

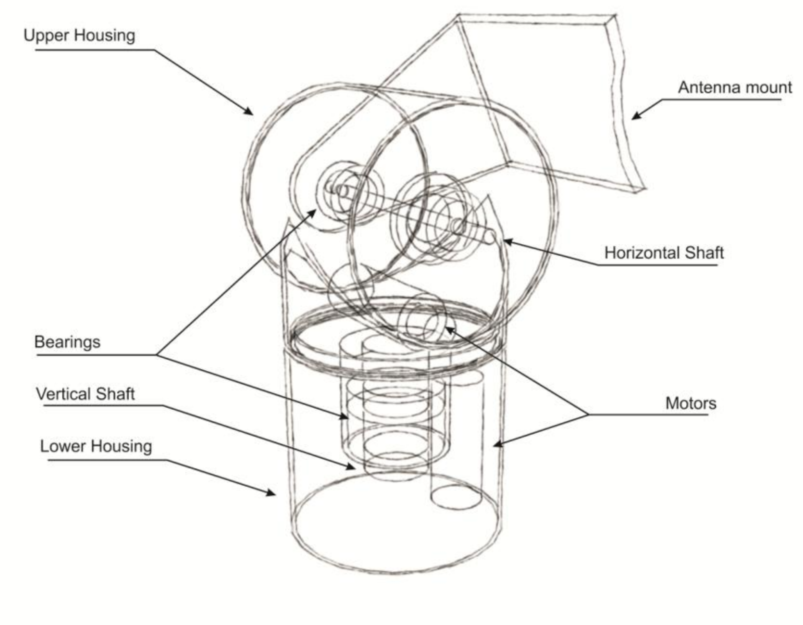


Figure 3.2. Mechanical concept

4 PRELIMINARY DESIGN

4.1 Choosing the actuators and gears

In order to choose the drive-train solution and select suitable components, it is necessary to investigate the requirements that constrain the selection of components. The maximum allowed weight of the antennas that the tracker can carry is approximately 2 kg, the antennas would be mounted on the mounting frame, which will create a lever arm of approximately 0,1 m in relation to the horizontal rotation axis. At the current state, the mass of the antenna is considered as point mass and the moment of inertia is calculated by the formula $J = m * r^2$ [3]. This results in the maximum load resistance torque of 1,962 Nm and moment of inertia of 0,020 kgm^2 to the horizontal axis caused by the antennas when the arm is aligned to the horizon. The estimated mass of the upper housing with its contents is roughly estimated to be 1.5 kg. The estimated diameter of the cylindrical upper housing is 130 mm. The approximate moment of inertia of the upper housing can be calculated by the formula $J = \frac{1}{2} m * r^2$ [4], based on the assumption that the distribution of the matter is uniform in the cylinder. This yields that the moment of inertia of the upper housing itself is 0,029 kgm^2; with antenna added, the moment of inertia acting relative to the vertical axis will be 0,049 kgm^2. The requirement for the maximum actuation speed is approximately 36 deg/s and the maximum rotational speed is estimated to be reached in 2 s, this yields to the estimated acceleration of 18 $\mathrm{deg/s}^2$.

To calculate the required motor torques the following formula is used [5].

$$T = T_R + J_{eq} \frac{\omega_{max}}{\Delta t}$$

Where

T_R- is the net resistance torque [Nm]

J_{eq}- is the equivalent moment of inertia [kgm^2]

ω_{max}- is the maximum operating speed [rad/s]

Δt- is the time taken to accelerate to the maximum speed, starting from the rest [s]

The torque calculation for the motor actuating horizontal axis is following: $T_{hor} = 1{,}962 + 0{,}020\frac{0.628}{2} = 2{,}025$ Nm

In the calculation of the necessary torque for the vertical axis, the sum of antenna and upper housing moments of inertias are used:

$$T_{ver} = 0 + 0{,}49\frac{0.628}{2} = 0{,}154 \text{ Nm}$$

With added safety factor of 0,3 corresponding torques are $T_{hor,max} = 2{,}633$ Nm for horizontal axis and $T_{ver.max} = 0{,}2$ Nm for vertical axis.

For choosing the motors, there are two main motor types in consideration that are capable to deliver the necessary speed and torque while fitting in to the desired room and weight constraints: the stepper motor and the brushed direct current (*DC*) motor. The stepper motors are driven by fixed angular increments, each step responding to the input pulse of the digital command. Thus it is possible to achieve fairly accurate control over the motion of the motor without using the control feedback loop. Stepper motors deliver considerable amount of torque at relatively moderate rotational speeds, due to that it is possible to drive the device coupled directly with motor shaft or by using low or moderate ratio reduction gears. There also exists a risk that the speed of the stepper motor may not be sufficient for the current task or the stepper motor may slip some steps. One benefit that the stepper motors have against DC motors, is the ability to provide full holding torque at standstill. The stepper motors are relatively reliable, since there are no contact brushes in the motor and the only wearable parts are bearings. The stepper motors may be preferred when a larger torque is required while actuation distances are shorter and movements are slower and more frequent.

The second option in consideration is the DC motor. DC motors in general run at higher speeds and lower torques than stepper motors in the same category, thus they need larger reduction ratios to be able to cope with the torque requirements. This can lead to larger mechanical losses in gearboxes and increased inertia of the system. The control of the DC motors is somewhat more complex. To successfully control the position of the movement, closed loop control is required. Some more accurate control solutions have multiple feedback loops (position feedback, motor speed feedback and current control feedback). This is a factor to be considered when the controller may be already loaded with running other tasks. The DC motors are more favored when dealing with longer actuation distances that require larger speeds.

For the current motor selection process some options from both previously mentioned categories were chosen as candidates. The considered options are listed in the table below [6] [7] [8] [9] [10].

Table 4.1. Motor options

Planetary geared stepper motors	Holding torque [Ncm]	Current/ phase [A]	Voltage/ phase [VDC]	Phase inductance [mH]	Gear ratio	Max. load [Nm]
NEMA8 (8HS15-0604S-PG90)	4,0	0,6	6,0	5,5	90:1	0,9
NEMA11 (11HS20-0674S-PG51)	12,0	0,67	6,2	7,2	51:1	4,0
NEMA11 (11HS20-0674S-PG100)	12,0	0,67	6,2	7,2	100:1	4,0
DC Motors			Stall torque [Nm]	Stall current [A]	No load speed [rpm]	Gear ratio
Pololu 25Dx54L Gearmotor 12V HP			2,1	5,6	100	99:1
Pololu 37Dx73L Gearmotor			1,5	5,0	100	100:1
Pololu 37Dx73L Gearmotor			1,8	5,0	80	131:1

First the DC motors were considered. In order to determine the suitability of options, the MATLAB script, provided by the manufacturer was used to plot the motor performance curves [11]. Since the manufacturer recommends to utilize the motors up to 25% of the stall current, the maximum performance for each motor was determined at 25% level of stall current as follows [12].

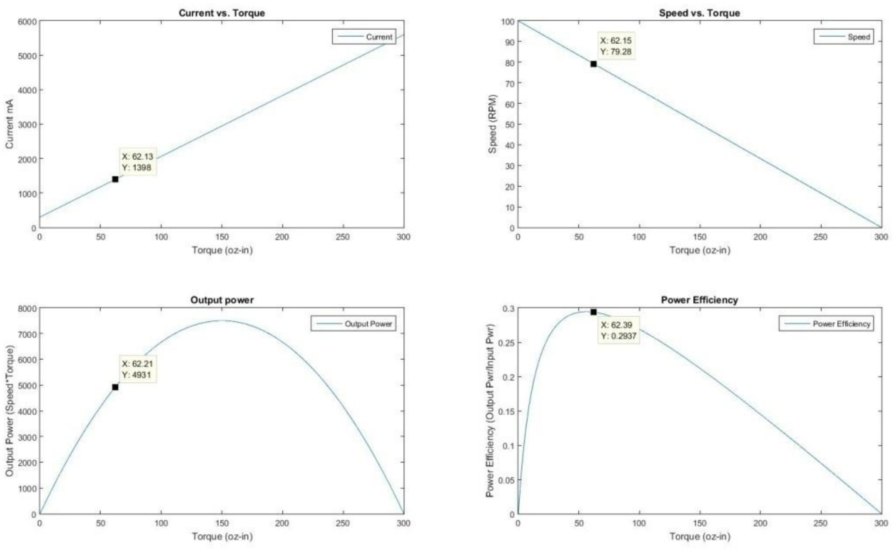


Figure 4.1. Pololu 25DX54L 90:1 performance plot

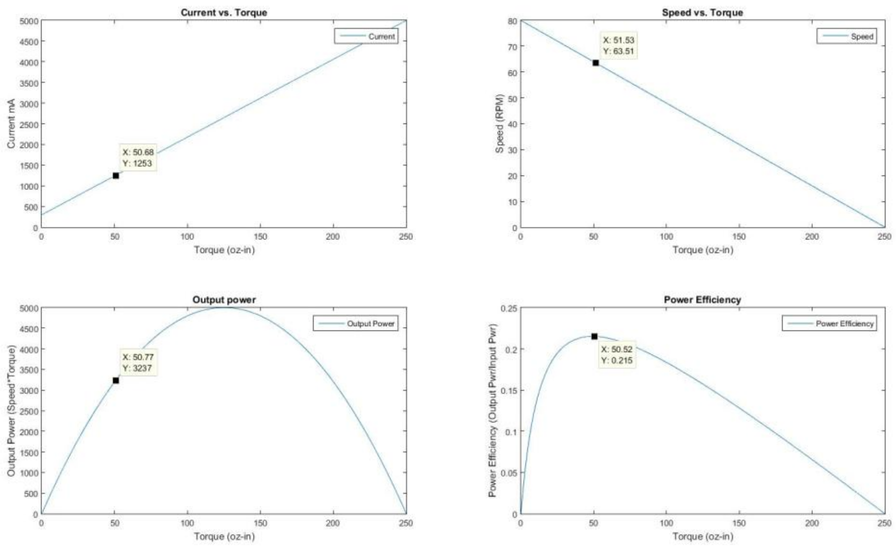


Figure 4.2. Pololu 37DX74L 130:1 performance plot

Based on the MATLAB performance plots, the Pololu 25Dx54L motor with 99:1 gearbox would achieve 0,439 Nm torque and speed of 80 rpm at 1,4 A. This does not meet the torque criteria without additional reduction gearing. If spur gears were used with the efficiency of 90%, the additional reduction gear ratio would have to be at least 7:1 to meet the torque criteria of the horizontal axis. If 7:1 reduction was used, the resulting rotational speed would be 68 deg/s and it would fit the speed criteria.

The Pololu 37Dx73L motor with 100:1 gearbox would reach torque of 0,315 Nm at the speed of 80 rpm. This configuration would need at least 9:1 reduction gear and in that case the reduced rotational speed of 53 deg/s would meet the speed criteria.

The Pololu 37Dx73L motor with 130:1 gearbox would reach torque of 0,395 Nm at the speed of 63 rpm. This configuration would need at least 7,5:1 reduction gear and in that case the reduced rotational speed of 50,4 deg/s would meet the speed criteria.

The smallest and most compact of the second motor type considered was NEMA8 geared stepper motor. The holding torque of this motor is 4 Ncm. It is combined with the 90:1 planetary gearbox with the efficiency of 73% and the output holding torque of the motor is 2,628 Nm. Next in size is NEMA11 geared stepper motor combined with the 51:1 planetary gearbox with 73% efficiency and its output holding torque is 4,467 Nm. The last of the stepper motor combinations is NEMA 11 with the 100:1 gearbox with 73% efficiency and its output holding torque is 8,760 Nm. It appears that all stepper motors can be driven directly without adding any reduction gears. The overview of the pros and cons of the motors are listed below.

Table 4.2. Overview of motor pros and cons

Motor	NEMA8 90:1	NEMA11 51:1	NEMA11 100:1	25DX54L 99:1	37DX73L 100:1	37DX73L 131:1
Pros	Small size	Sufficient torque	Good torque	Small size	Good availability	Good availability
Cons	Low torque			Additional gear and encoder required. Low availability	Additional gear and encoder required. Low torque	Additional gear and encoder required
Suitability	No	Yes	Yes	No	No	Yes

To finalize the motor selection, the motor type is chosen. The choice prefers the stepper motors. This is primarily because steppers can be driven by open loop control and current stepper motors in selection are powerful enough that they could actuate both axis with direct coupling and would not need additional gear reduction. The motor chosen for the current application is NEMA11 size stepper motor (11HS20-0674S-PG51) with the 51:1 planetary gearbox. The current motor combination was preferred over the 100:1 gearbox version because 51:1 delivers a higher amount of rotational speed with sufficient reserve of torque, thus it can be driven by lower pulse ratios and thereby the induction related torque losses and chances to miss steps are smaller compared to the 100:1 gear ratio version.

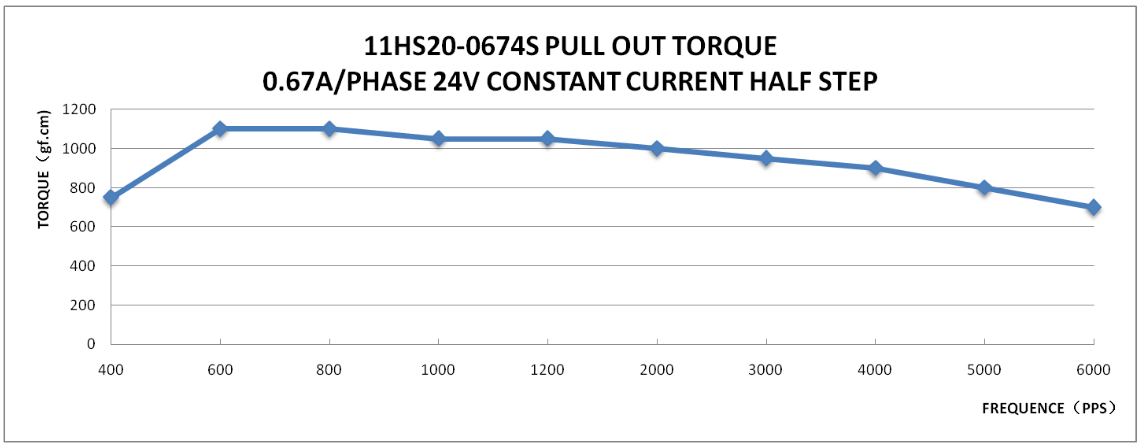


Figure 4.3. Pull-out torque curve of NEMA11 stepper motor

4.2 Choosing the mechanical components

Since the gearbox of the chosen motor has restrictions for axial (5 N) and radial (25 N) load, then it is not possible to mount antennas or the upper housing on the motor shafts directly [13]. Therefore it is necessary to create bearing mounted axis and power transfer mechanisms.

In order to select the shaft diameter for the vertical axis it must be noted that the axis needs to rotate 360˚. Since the upper housing attached to the vertical axis contains the motor and sensors for the horizontal axis, the need for a slipring is evident. The slipring allows routing signal and power cabling through the axis and grants free rotation without winging the wiring. For that the Barlin Times Technology 8mm slipring was chosen. This slipring has a diameter of 8,5 mm and it has 8 circuits with 1 A 240 V AC/DC capability [14]. This sets some constraints for choosing the size of the vertical axis.

The initial diameter chosen for the vertical axis is 15 mm and the axis is with a central bore of 8,5 mm, this yields to the wall thickness of 3,25 mm. To principally check the flexural strength of this axis solution, theoretical calculations were created. It is assumed that the approximate height of the upper housing is about 0,1 m. It is also assumed that the perpendicular force of 490 N (50 kgf) is applied at the top part of the upper housing, creating a bending moment $M_{Bv} = 490{,}5\ N * 100\ \text{mm} = 49050\ \text{Nmm}$. The second moment of area of the axis is:

$W_v = \frac{\pi*(D^4-d^4)}{32D} = \frac{\pi*(12^4-8{,}5^4)}{12*32} = 304{,}532\ \text{mm}^3$, where D is the outer diameter and d is the inner diameter.

The bending stress in the vertical axis is $\sigma_v = \frac{M_{Bv}}{W_v} = 161{,}067\ \text{N/mm}^2$. If the axis material is structural steel S275 with yield tension $\sigma_{lim} = 380\ \text{N/mm}^2$, we can say that the axis will likely not deform due to the 57% reserve in yield tension [15]. The horizontal axis is being sized to 8 mm at its minimum cross sectional dimension. For confirming the suitability of the shaft size mechanically, another theoretical calculation is being made. Assuming that the antenna mount is attached to the axis via two bars with the length of 100 mm and additional load 10 kg is added to the antenna, a torsion moment $M_{t.h} = 11772$ Nmm to the axis is created. The polar moment of inertia of the axis can be obtained as follows: $W_{P.h} = \frac{\pi*d^3}{16} = 100{,}531\ \text{mm}^3$. The torsion stress in the horizontal axis is calculated by using the next equation: $\tau_{t.h} = \frac{M_{t.h}}{W_{P.h}} = \frac{11772}{100{,}531} = 117{,}098\ \text{N/mm}^2$. If the horizontal axis is made of S275 structural steel with yield tension $\tau_{lim} = 160\ \text{N/mm}^2$, then the axis will not deform under added load due to the 26% reserve in yield stress. In that scenario the gearbox of the stepper motor will fail before the horizontal axis if no reducing gears or clutches are being used.

For mounting the vertical axis into the housing, a pair of SKF 7202BEP single row angular contact ball bearings are chosen [16].
For mounting the horizontal axis, the SKF 6000 single row deep groove ball bearings are chosen as suitable [17].
For the power transmission mechanism, timing belts and pulleys are being used.

When considering the options for the timing belt pitch selection, the initial options MXL, XXL, T 2.5 and T5 were observed. Since the imperial trapezoidal pitches MXL and XXL are designed to be the timing applications and have fairly low tooth profile, the choice was to use metric T-pitch belts. The T-pitch belts are commonly used for the conveying applications and therefore have slightly deeper grooves thus offering more reliable meshing [18]. For the pulleys, the SDP/SI A A 6A35M018DF1006, 5 mm (T5) pitch, 18 teeth, 6 mm Bore, 28,8 mm pitch diameter, aluminum alloy timing pulley with a 10 mm wide belt was chosen. This choice was mainly done because of the availability of the pulley that has a 20 mm diameter hub, which makes it suitable for re-boring to fit the vertical axis shaft. There were no other stock size pulleys found that would end up with 1:1 ratio transmission with the motor. The approximately suitable belt to fit the room constraint was chosen: the SDP/SI A 6T35MF033100, 5 mm (T5) pitch, 33 teeth, 10 mm wide single sided polyurethane belt with fiberglass cords. For current pulley pitch diameter and belt combination, the distance between pulley shaft centers is calculated by using the following equation [19]: $C = \frac{p(NB-N1)}{2}$, where C - the center distance of pulleys in mm, p - the pitch of the belt in mm, *NB* - the number of teeth on the belt and *N1* - the number of the teeth on the pulley. The resulting distance for the center of the pulleys is 37,5 mm and it confirms the suitability of the belt to the room constraints. The selected stepper motors have the single step-angle of 1.8 degrees, with the timing belt transmission ratio 1:1, and gearbox reduction ratio 51:1, the resolution of the system output yields 0,035 deg/step or 28.5 steps/deg. This is 2.8 times more resolution that was initially required.

Table 4.3. The bill of materials, mechanical

Description	Item	Provider	Amount
Stepper motor	NEMA11 (11HS20-0674S-PG51)	mc-stepperonline.com	2
Slipring	THR008-08AM	Barlintimes.com	1
Vertical shaft	Custom made		1
Horizontal shaft	Custom made		1
Angular contact ball bearings	SKF 7202BEP	SKF	2
Deep groove ball bearings	SKF 6000	SKF	2
Timing pulley	A 6A35M018DF1006	SDP/SI	4
Timing belt	A 6T35MF036100	SDP/SI	2

4.3 Choosing the electrical components

4.3.1 Driver

The device is actuated by two stepper motors. To run those stepper motors, the motor drivers are needed. According to the datasheets, the motors need 0,67 A per phase and each motor has two phases, so each driver should be able to provide at least 1,34 A. In order to ensure the prolonged lifespan of the drivers, the initial power requirement is multiplied by safety factor 1,5 and due to that the power rating of the drivers has to be 2 A. In order to relieve the microcontroller from directly generating the pulse sequences for each motor coil, the stepper driver with a translator and Pulse/Dir interface is preferred. One of the suitable stepper driver for the task is A4988 DMOS microstepping driver with a translator from Allegro Microsystems Inc [20].

4.3.2 Sensors

In order to actuate the tracker axis, it is necessary to determine the initial positions of the tracker axis. There are two options in consideration how to achieve that. The first option is to use the high resolution absolute encoders on each axis. The benefit of this solution is that at any given moment, the position of the shafts are known. This can reduce time to start up the tracker and provides the possibility to detect missed motor steps. On the downside this solution burdens the microcontroller, makes the control algorithm more complex and increases the cost and mechanical complexity of the unit. The second option considered is implementing end-switches and/or optical sensors on both axis. As the tracker device is powered up, the axis calibration sequence is carried out by actuating the axis until they reach the predetermined endpoints or center point. By reaching the preset point, the zero point is set and the further actuation can be implemented by using the open loop stepper control with counting the steps carried out to determine the current position. The benefits of this solution are the simpler mechanical design and cheaper implementation of the system, but the main drawback is that this solution does not offer any feedback of missed steps. At current point of the design the latter solution is preferred, by using transmissive photo interrupters to limit the travel of the horizontal axis and determine the center point of the vertical axis. As a suitable unit, SHARP GP1A51HRJ00F photointerrupter is considered [21].

4.3.3 Controller

In the consideration of choosing the microcontroller platform, multiple aspects are reckoned with. First: in order to minimize the development time, it is reasonable to use the microcontroller platform with large users community and with ample know-how available. Based on the author's previous experience and skill set, the reasonable choice would be Arduino platform based on the Atmel megaAVR microcontrollers. For the prototyping, the Arduino Uno or Mega 2560 based development boards are going to be used, since they offer simple prototyping options and have a large variety of open source libraries available. In the detailed design, the relevant microcontrollers are going to be embedded into the custom made controller board [22] [23].

4.3.4 Data communication

The system design criteria dictate that the target input signals are sent from GCS via Ethernet connection. To convert the Ethernet communication to some interface that is acceptable by the microcontroller, the signal converter is going to be used. Since the Arduino Mega 2560 board has SPI interface, the ENC28J60 based mini Ethernet module is being used [24].

4.3.5 Power

To determine the required power supply unit for the device, it is necessary to determine the total power consumption of the device. The power requirements of the components are indicated in the table below.

Table 4.4. Approximate power consumption of the components

Item	Power consumption [mA]	Supply voltage [V]	Power demand [W]
Motor driver	2x2000	12	48
Sensors	3x100	5	1,5
Microcontroller	500	5	2,5
Ethernet module	180	3,3	0,6
Total			52,6

Power supply parameters	4380	12	

The table indicates that the total maximum power requirement is approximately 53 watts (4,4 A at 12 V). If adding the overhead margin of 12 %, the rated power of the power supply is 60 W (5 A at 12 V). The device is supplied by 14,4 VDC or 18 VDC from external power supply. Due to the variable input voltages, the power supply should be able to step-down the voltage. Since the tracker system is meant to be used in off-grid conditions and the power is provided mainly by accumulators, then the usage of the linear voltage regulators in this application is not preferred due to the low efficiency and relatively high energy loss during the conversion. Hence the usage of the switching voltage regulators is in order. The suitable model that the company already implements in various applications is Texas Instruments PTN78020WAH integrated switching regulator module [25].

Table 4.5. Electrical components

Description	Item	Provider	Amount
Motor driver	A4988 DMOS microstepping driver	ITT Group	2
Sensors	SHARP GP1A51HRJ00F photointerrupter	Premier Farnell UK	3
Microcontroller	Arduino Mega 2560	ITT Group	1
Ethernet module	ENC28J60 mini Ethernet module	ITT Group	1
Voltage regulator	Texas Instruments PTN78020WAH	ITT Group	1

4.4 Electrical schematics

The electrical setup of the test version of the tracker is built around Arduino Mega 2560 or an equivalent "clone" board. The 12 V is supplied by the Texas Instruments PTN78020WAH switching regulator and 5 V by NCP1117ST50T3G fixed voltage regulator. The components like Ethernet module, motor drivers, motors and sensors are mounted on the breadboard and wired as shown on the figure below.

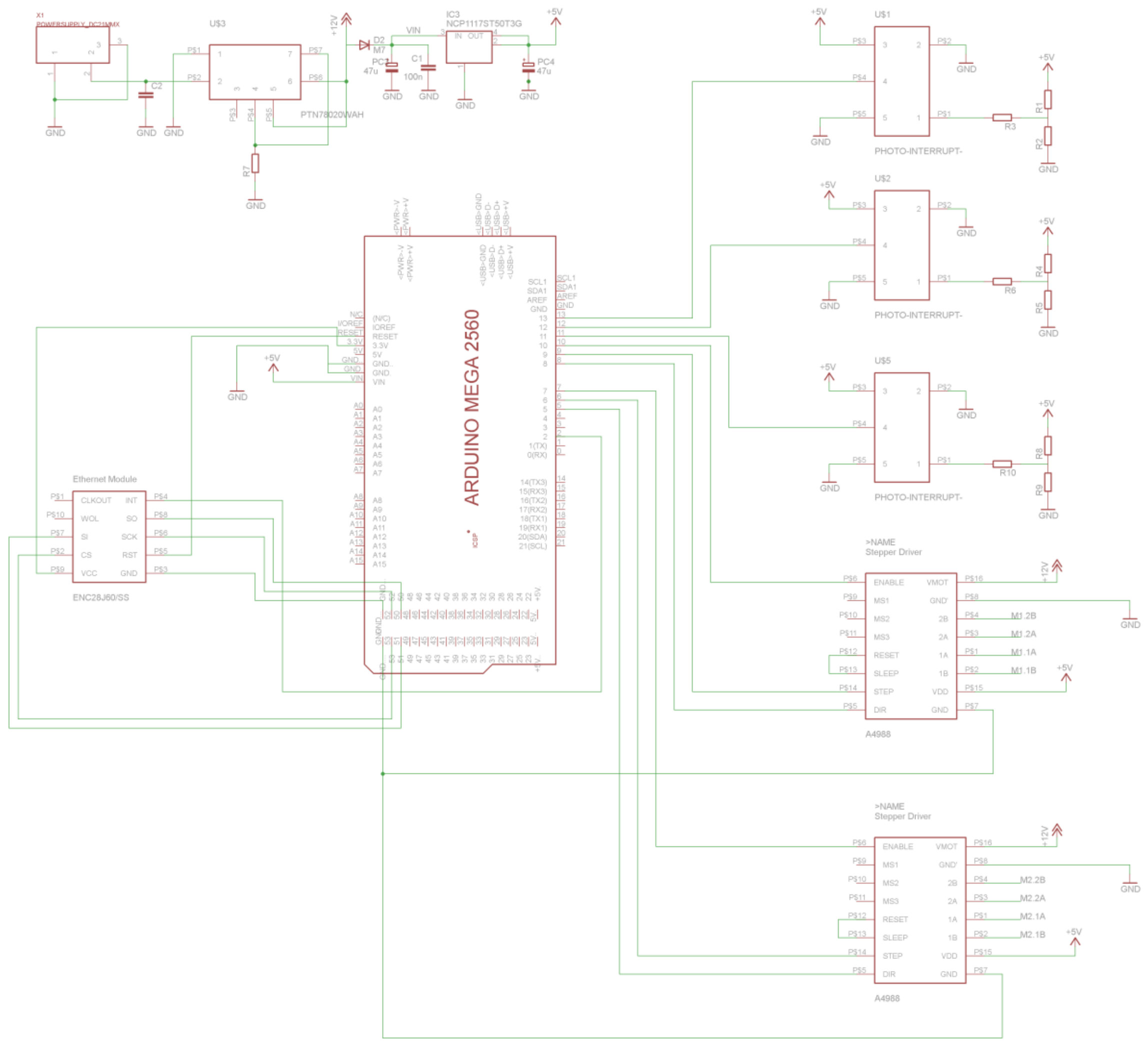


Figure 4.4. Electrical schematics of the test setup

If this configuration proves to be successful, the custom controller board can be manufactured based on this setup. The controller board is built around ATMEGA2560-16AU microcontroller. In addition, LP295-33DBVR based circuit is added to provide 3.3 V supply for the sensor diodes and the Ethernet module. The motor drivers and

Ethernet module are mounted directly on the printed circuit board (PCB) via female pin header sockets thus enabling swift component replacement if needed. Female sockets are mounted on the PCB for connecting the plugs for the motors and sensors. The schematics of the custom controller board is shown at the figure below.

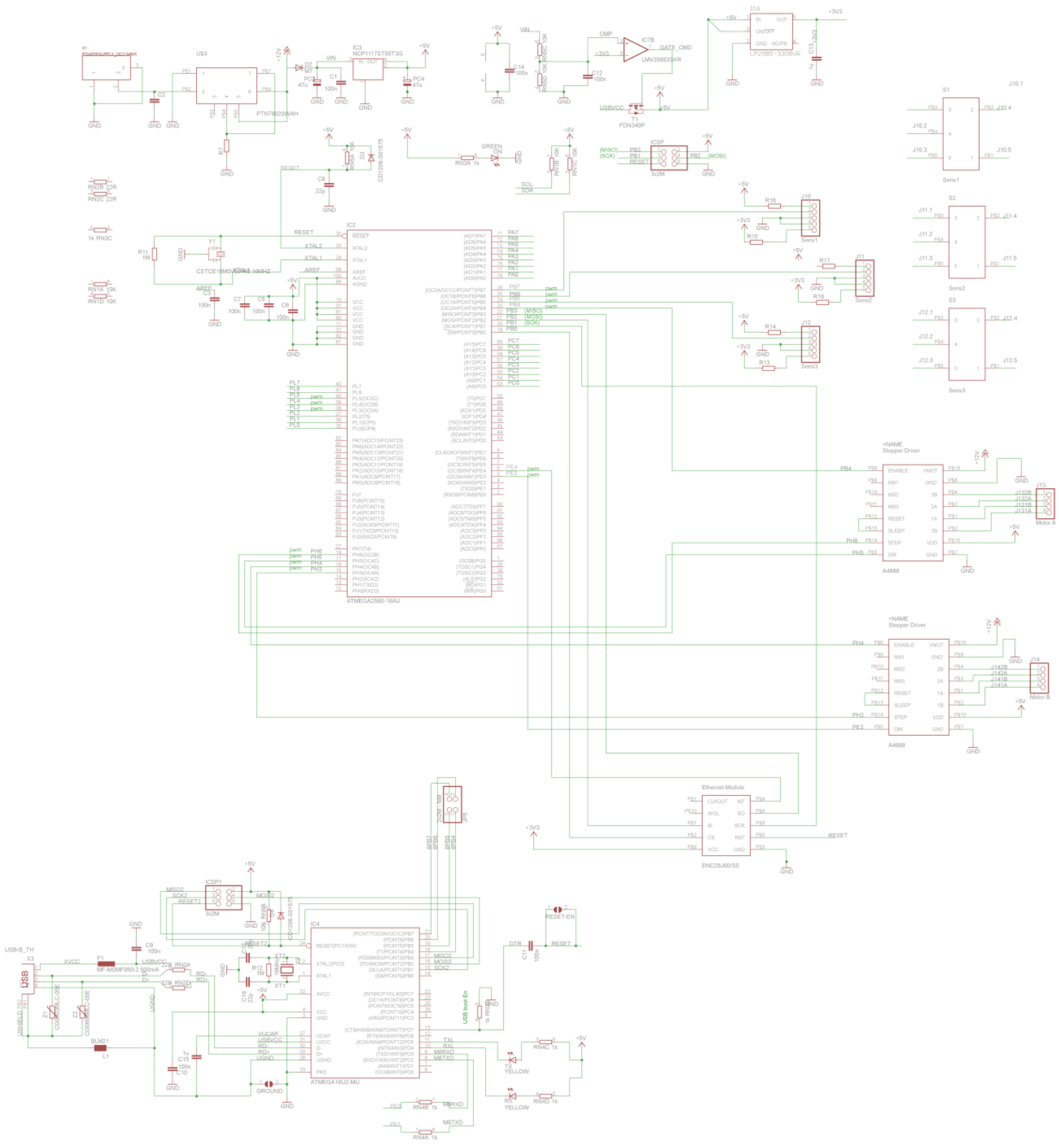


Figure 4.5. Electrical schematics of the custom controller board

4.5 Creating the CAD

Creating and modifying the CAD model(s) is a parallel process to the whole designing procedure and finding the final solution can take many iterations in the design loop. This thesis project was no exception. The first concept based on the conceptual mechanical design described in chapter 3.2, was discarded in early stages due to its flaws in bearing mountings and its general bulkiness. The second concept was somewhat similar to the first one but the upper housing volume was significantly reduced (see figure below). The benefit of this design was that both of the housings were cylindrical so there were no corners that could interfere with the external cables. Another benefit was the ample interior volume for housing the electronics. However, as the design progressed, it became evident that the second design does not meet the size and volume criteria. The minimum circular cross section that could fit the chosen motors and pulley mechanism had a diameter of 140 mm. The approximately 3,7 liter overall volume was too large to ensure the desired compactness and portability. So developing a third concept was in order.

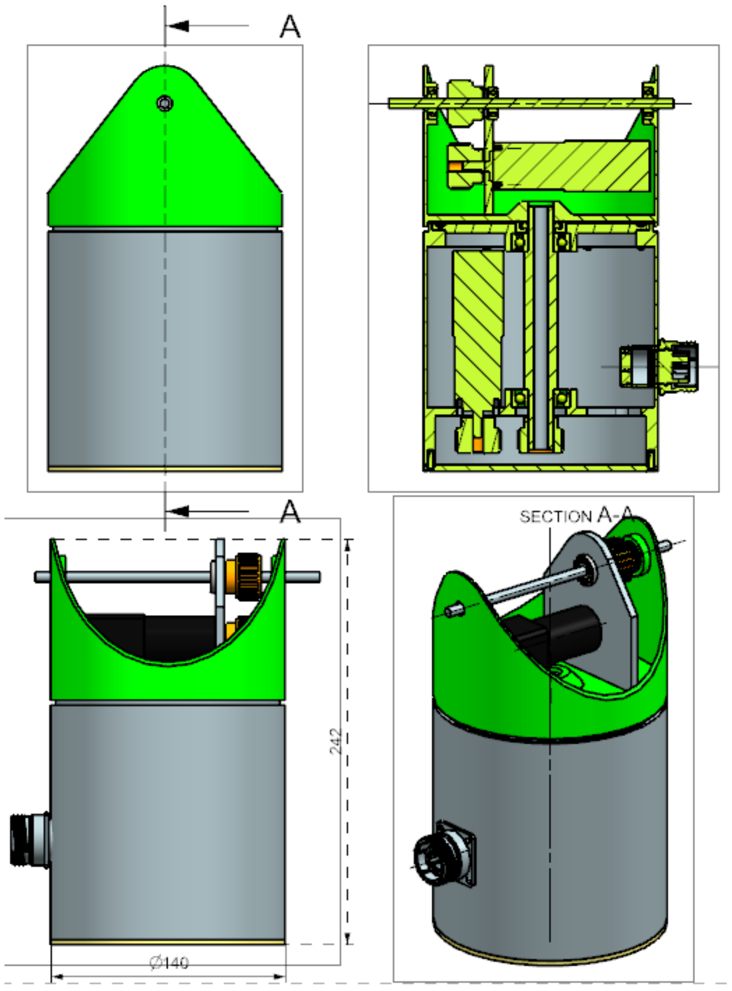


Figure 4.6. The second concept

The third concept differed from the second by the use of a rectangular cross section with rounded corners. By using a rectangular layout, the general volume of the tracker body was reduced by 58 % to 2,1 liters. The smallest dimension obtained was 5 cm (see figure below). The estimated mass of the tracker was 2.55 kg. The tracker was intended to be used with a quick release mechanism that would have allowed easy detachment of the antenna with the mounting flanges. The flanges could be folded parallel with the backside of the batch antenna thus making it a quite compact solution. The lower housing of the tracker consisted of two hollow box-section bodies that were fixed by bolts between three floor surfaces. Two of them, the upper- and mid-floor, incorporated the bearing housings to mount the 7202 BEP ball bearings. Between the bearings, the hollow 15 mm vertical shaft was fixed by the flanges on the shaft. At the bottom end of the vertical shaft, the slipring module was accommodated in the bore of the shaft. On the mid-floor, the stepper motor for actuating the vertical shaft was mounted. The mid-floor also mounted the optointerrupt sensor for determining the vertical shaft position. The lower housing accommodated the belt and pulley mechanism and the electronics for the tracker. The output of the vertical shaft was sealed with SKF15x30x7HMS5_RG shaft seal, to keep moisture out from the lower housing. At the top end of the vertical shaft, upper housing floor was fixed. The joint consisted of the flat end shaft and a matched bore in the floor, the joint was reinforced with two M4 setscrews. Perpendicular to the floor, the bulkhead of the upper housing was attached by two M3 bolts. The bulkhead was used to mount the motor for actuating the horizontal shaft and mounting two optointerrupt sensors for determining the key positions of the horizontal shaft. The horizontal shaft was mounted on the two SKF 6000 ball bearings bedded in the bearing housings at the left-hand and right-hand upper housing covers. The middle section of the shaft was supported by the SKF 6001 ball bearing, housed in the bearing housing at the bulkhead. The openings for the horizontal shaft were sealed with the SKF 10x19x7_HMS5_V1 shaft seals. The horizontal axis had a middle diameter of 12 mm and the diameter at the outer parts was 10 mm. The step in the diameters was implemented to ensure the alignment of the shaft between the 6000 ball bearings. The outmost parts of the shaft were with flat ends and 5 mm threaded bores for attaching the antenna flange mounting discs. The mounts consisted of two discs: the first one fits to the flat end of the horizontal axis and it has a U-shaped elevated section to translate rotation to the antenna mounting flanges; the second disc uses a U-shaped cut out to fit into the first disc and it is used to press the antenna mounting flanges between two discs. The tension for pressing the discs together was provided by the eccentric lever with its stud connected to the 5 mm bore of the horizontal shaft. This solution provides quick release feature to release the antenna and its mounting flanges with minimal effort.

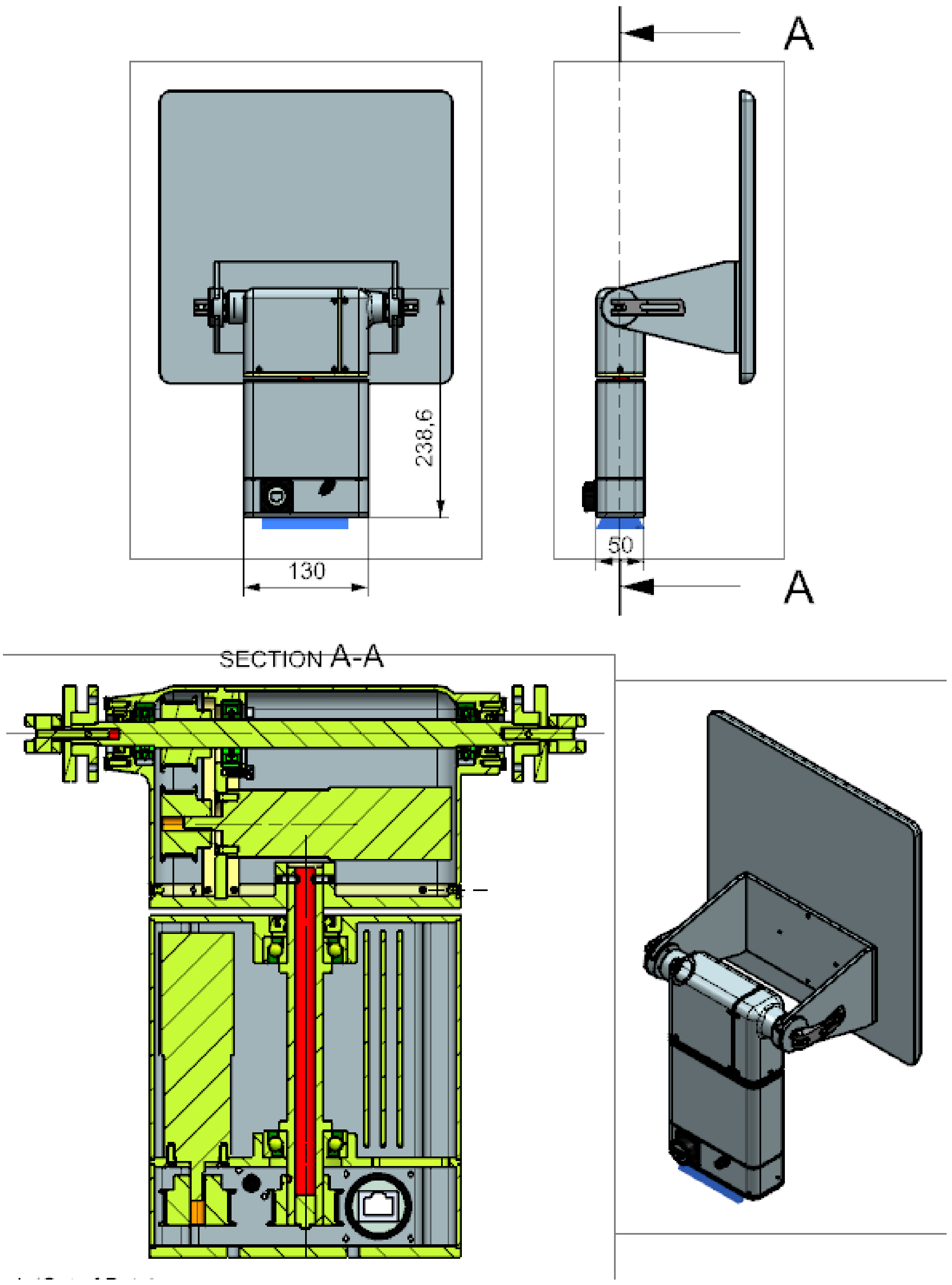


Figure 4.7. Concept 3

The downside of this concept was that the rectangular cross section gave compact dimensions only when the tracker was in the neutral position by the vertical axis. As the tracker rotated, the overall cross section increased up to 130x130 mm square. Due to this aspect, it was necessary to use relatively long antenna mounting flanges to ensure a sufficient negative rotation angle about the horizontal axis and avoid contact between the antenna and the lower housing. However, the most important drawback of this design was the complexity of the manufacturing. Since it was intended to make the components of the tracker out of the aluminum alloy by CNC milling, the rounded corners added complexity and time consumption, thus the price to manufacturing. In order to maintain reasonable manufacturing complexity, a fourth version of the tracker was made. This version is discussed in more detail in chapter 7.

4.6 Data communication

At the prototype level, a personal computer (PC) shall be used as GCS. The PC shall be connected to the router via the Ethernet cable. The router shall route the signals between the ground control station and peripheral devices like the antenna tracker. The data communication between the ground control station and the antenna tracker is accomplished by the utilization of user datagram protocol (UDP). The software in the ground control station shall generate the UDP data package containing the target azimuth and elevation coordinates for the antenna tracker. The UDP package is a string of characters that consists of azimuth coordinates ±123.1234, the semicolon for separating values and the elevation coordinates ±123.1234. The UDP package shall be sent by the ground control station in periodic intervals. At current case, the emitting frequency is 1 Hz. The tracker shall receive the data by Ethernet module and it shall handle the layers 1-4 of the OSI model independently. The controller onboard the antenna tracker uses the EtherCard library to provide the main interface to the ENC28J60 [26]. The EtherCard library uses the ENC28J60 Class to provide the low-level interfacing with the Ethernet module. The main program of the tracker uses the ether.udpServerListenOnPort function to monitor for incoming UDP packages. As the message arrives, the interrupt is fired and udpSerialPrint function is called for parsing the data in to the usable format for other functions in the program.

4.7 The control algorithm

4.7.1 Main

The concept of the main program consists of four parts: axis calibration, monitoring the Ethernet module, interrupt service routines (ISR) and main loop. As the tracker is powered up, the axis calibration process starts to determine the neutral point of the horizontal axis and the limits of the vertical axis. The key points are determined by moving the axis until the sensor detects the points. As the key points are determined, the program waits until the Ethernet module has acquired the polar coordinates for both axis. In the main loop, the current positions of the axis are compared against the target values and the shortest direction of the rotation is chosen. The horizontal (X) axis is actuated by the stepper1.run function and the new position of the axis is stored. Later the limit conditions of the horizontal axis (Y) are checked and the axis actuated by the stepper2.run function. The new position of the axis is stored. As the limits of the Y axis are reached during the movement, the sensor signal triggers ISR and the limit flag is set. This prevents the Y axis reaching its mechanical limits. The block diagram of the main algorithm is shown below.

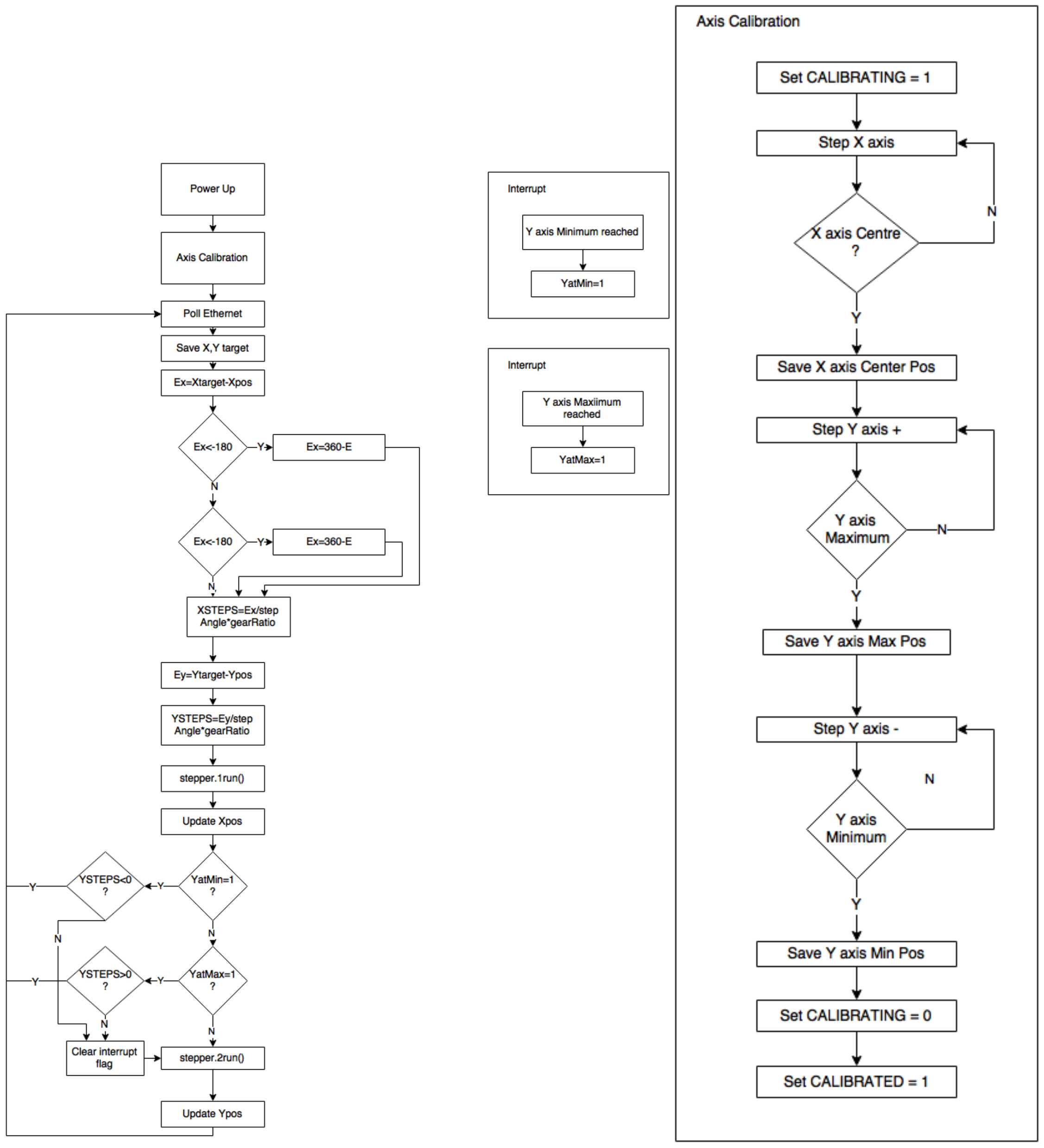


Figure 4.8. Block diagram of the main control algorithm

4.7.2 Motor Control

For controlling the motors, AccelStepper library by Mike McCauley is being implemented [27]. This library fits the algorithm of parametrizing the stepper motor acceleration profiles in real time [28] to be suitable for Arduino platforms. This method uses accurate approximation of the timing for the linear ramp with the constant acceleration and deceleration, using simple fixed-point arithmetic operations and no data tables, thus making it possible to be used on low-end microcontrollers. The target values provided by the EtherCard's udpSerialPrint function are saved as integer values posX and posY. In the each iteration of the main loop, the functions stepper1.moveTo(posX) and stepper2.moveTo(-posY) are called to set the target positions for the motors (the negative sign in front of posY is for inverting the Y coordinate values to ensure the correct direction of rotation, taking the installation orientation of the motor into consideration). Each time those functions are called, the new values are calculated to the acceleration profile. Subsequently the functions stepper1.run() and stepper2.run() are called in the main loop. The run function polls the motor and steps it by one step if the step is due. This function implements the accelerations and deaccelerations to achive the target position. Those two functions must be called as often as possible to achive higher step rates. In order to do that, the main loop must be kept as short as reasonably possible. As mentioned before, the AccelStepper generates the constant ramp acceleration speed profile for each target point set. At the test configuration the stepping speed of 4000 pps and acceleration of 2000 pp/s^2 is being used. The example of 16000 steps movement is illustrated below.

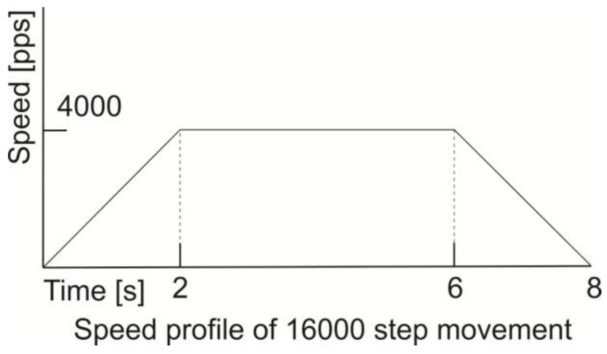


Figure 4.9. Stepper speed profile

As the new setpoint should be set while the last task is still carried out and the new setpoint is nearer than the previous, the new speed profile is generated. If the set point is closer than the point where the motor could be stopped with current speed and acceleration settings, the motor will stop as quickly as possible, using the current speed and acceleration parameters and then corrects the overshoot automatically, using the same parameters. The overshoot correction is illustrated in the figure below.

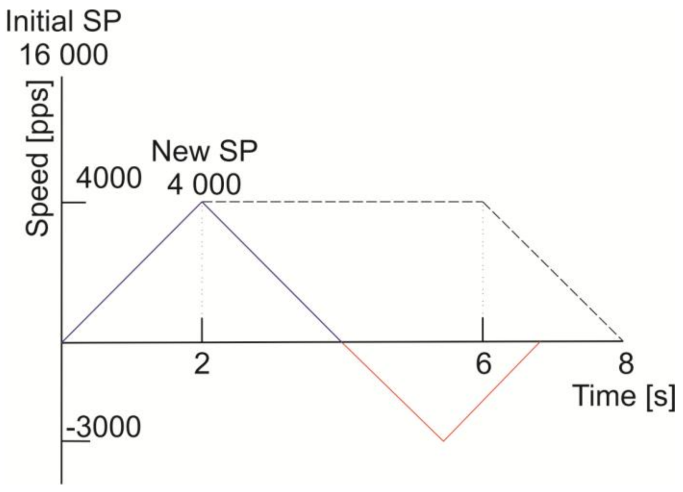


Figure 4.10. New setpoint overshoot correction

The stepper resolution is 200 steps per revolution and the excact gear ratio of the steppers gearbox is $51\frac{4397}{4913}$ (51,895). This yields to the combined stepper resolution of 10349 steps per revolution and 0,035 degrees per step. To ensure the smoothest running, the different stepping modes like half-step and quarter-step are considered. The stepping mode shall be decided during the motor test trials. If the stepping mode is altered, the resolution will be affected accordingly.

5 TESTING THE CONCEPT

5.1 Suitability test for the control devices

As the test setup of the control devices was assembled according to chapter 4.4 and initial program written, general tests of functionality were conducted. The functionality tests that were carried out: running motor(s) with different speeds and accelerations; running the single motor, while the targets stored to the microcontroller; running two motors with targets stored in the microcontroller; running two motors while receiving target values from Ethernet. During the functionality tests, the issue of using the ATMEGA 2560 microprocessor in the combination with Ethernet module while running two steppers came forth. Namely it became evident that the task of receiving and processing the target data was taking considerable amount of time for the microcontroller. While the controller was occupied with obtaining and processing new target values, the loop for controlling two stepper motors was interrupted. This resulted with a little jerk in the actuation of the motors each time new target data was received. This was not an issue when the target was far away from the tracker and the increments in the positions were small enough that the full movement could be completed in one data receive cycle. However, as the actuation distances increased over the point that the movement had to last longer than the receiving cycle, the jerks became evident. They were most disturbing while the motors were running at maximum speed. While the motor controlling halted, the counting of steps halted as well, but the motors had gained some inertia. Thereby it cannot be out ruled that the motor shafts did not continue moving while the step counting was paused and the actual position was disturbed. To overcome that problem, the concept of controlling the device was amended and the second microcontroller was introduced to the system. The initial controller remained receiving the data from the Ethernet and processing it to the format acceptable for controlling the motors. Then the data (converted to the format of steps) was sent via the I2C serial bus to the second microcontroller that was controlling the two stepper motors (at current case the ATMega328 based Arduino Nano). By doing so the controller responsible for controlling the motors was released of time consuming tasks of managing the Ethernet connection, parsing and processing the data and could concentrate entirely to the controlling of the motors. Implementing the new design fully eliminated the jerky movement. The amended design will be discussed in more detail in chapter 6.2.

5.2 Suitability test for the actuators

In order to confirm the suitability of the chosen stepper motors, the test was carried out on the actual motors. If the antenna tracker is being used in longer missions, the possibility that one of the steppers misses some steps increases. The slipping steps can be caused by the wrong combination of the load and stepping speed. All stepper motors start slipping at some speed - load range. At lower speeds, when stepping period is considerably larger than the electrical time constant, the time taken to build up the phase current is insignificant when compared to the step time. Then the waveform of the phase current can be assumed to be rectangular. At higher speeds, the phase current may not be able to build up during the step time due to the increased induction effects [29]. The result is decreased torque and when combined with improper load, the motor starts to slip through and miss steps. Thus the testing the motor and load combination is necessary in order to determine if the motors are properly tuned for the current loads. The practical testing of the motors serves other purposes: determining the suitability of the motor driver and controller combination, and the overall performance (actuation smoothness, speed and accuracy).

The mission duration of the UAS, that the antenna tracker is primarily meant to be used with, is approximately two hours. To simulate the UAS mission, the MATLAB algorithm was created to obtain the array of coordinates for the flight trajectory of the UAS. The trajectory consisted of 6105 points that imitate the position of the UAS moving with the speed of 16 m/s. The flight path consists of eight legs forming two number 8 shaped figures placed at two different horizontal planes and fit into the approximately 10 km sphere. See the figures below.

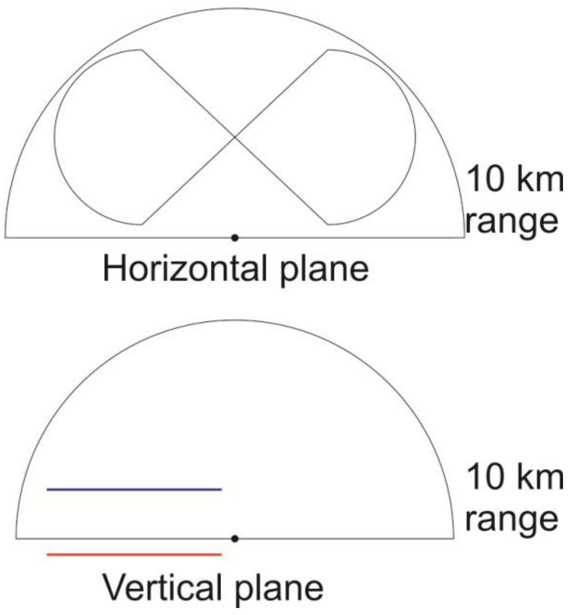


Figure 5.1. Flight path in 10 km range

The number 8 figures are located at two different altitudes regarded to the position of the antenna tracker. The position of the tracker was chosen to be at the zero point of the coordinate system and the lower flight level was chosen to be 730 m below the tracker altitude to simulate the long range flights in mountainous areas. The higher level was chosen to be 2250 m above the tracker level in Z axis. The centers of the curved parts of the number 8 figures were chosen to be at 4250 m offset from both sides of the tracker location by the X axis. The distances of the centers from the zero line by the Y axis are 4600 m. The radiuses of the curved parts of the number 8 figures are 4000 m. There are two legs in the flight path that changed the altitude from one flight level to another and interconnecting two loops. This trajectory yields to the operation envelope from -82 degrees to +82 degrees in azimuth and +33 degrees to -12 degrees in elevation. See the three dimensional representation of the flight path in the illustration below.

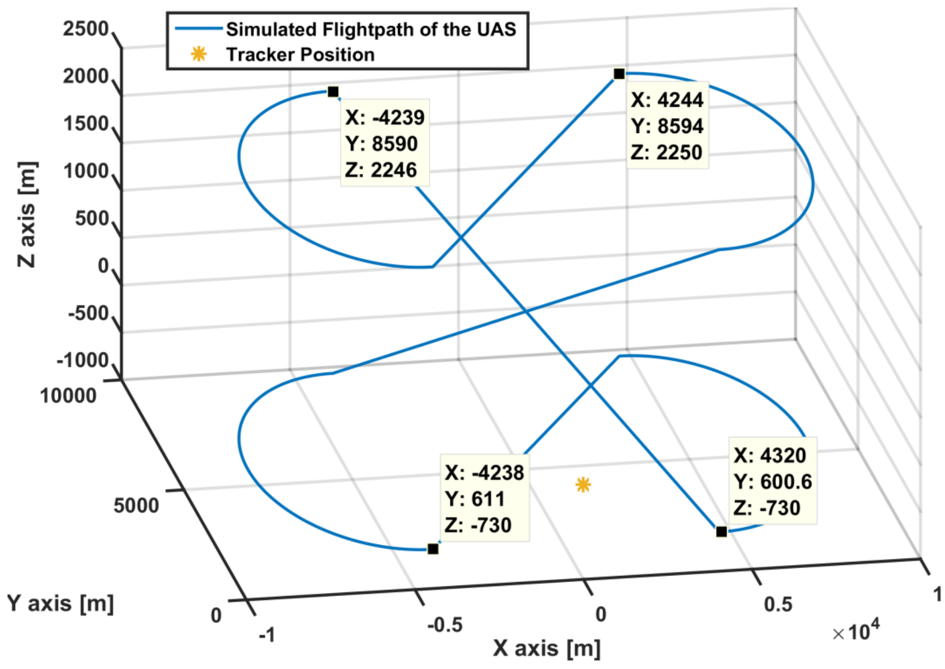


Figure 5.2. Simulated flightpath of the UAS

For transmitting those coordinates, another MATLAB script was written. The script compiled a UPD data package and sent the coordinates with a one second interval to the tracker's microcontroller via the Ethernet link. The scripts and flight path data are available at the appendix 8.3.

For testing the microcontroller, the motor drivers and the motors themselves, the test rig was constructed. Like the tracker, the rig consists of two parts: the upper housing and the lower housing. The lower housing is built from two CNC cut sheets of plywood and is fit together by four threaded rods at the corners. The plywood sheets have openings to mount the motors and accommodate the countersunk bearing wells, that are used to fix 8x22x7 mm bearings. The 8 mm vertical shaft is installed between the bearings and to the shaft, T2,5 timing pulley is fitted. The motor is fitted with matching pulley and the transmission is carried out by the 158x6 mm timing belt. The plywood-made upper housing is fit to the vertical shaft and it accommodates the second motor. The glass fiber payload carrying flange with the lever arm of 120 mm is attached directly to the horizontal motor shaft. Since the planned main payload, the MA-WA24-2X batch antenna, weighs 860 g, the same weight was attached to the payload flange.

To determine the orientation of the payload of the tracker, the laser module was added to the mockup payload. The test rig is pictured on the figure below.

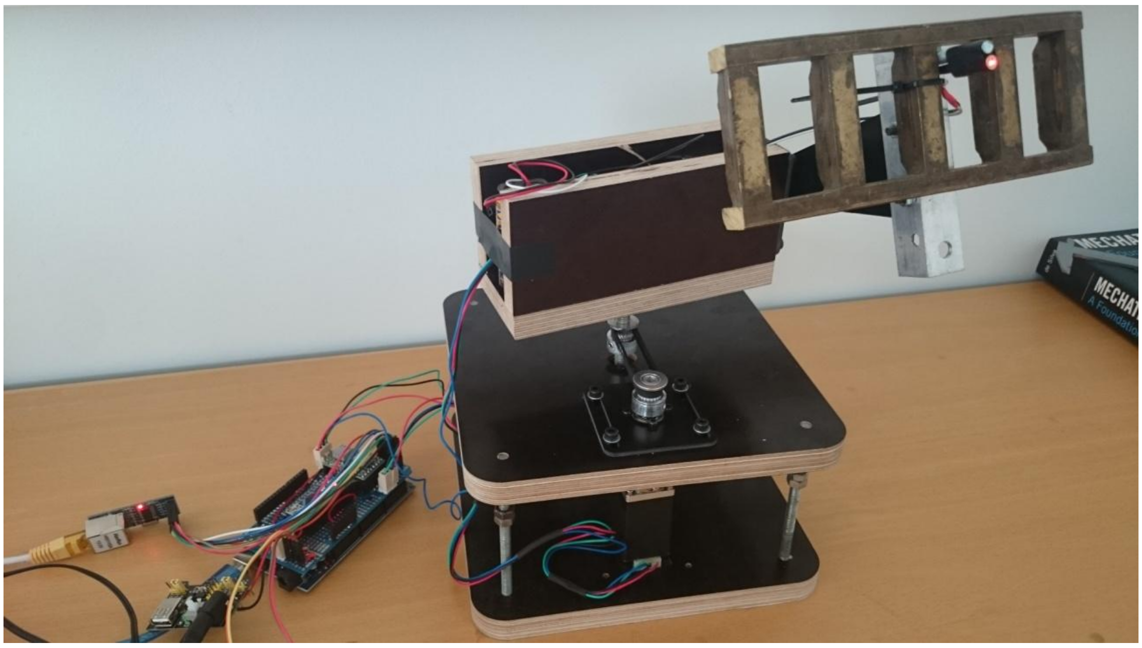


Figure 5.3. Test setup of the antenna tracker

To carry out the test, a 20 m distance was measured and the zero point was set and marked at that distance. The MATLAB algorithm was started to imitate the mission of the UAS and the actuations were carried out with real speed (no acceleration of simulation was used). The test lasted for 6105 seconds and during the test, the tracker moved to the corresponding number of different positions. At the end of the test the tracker was moved back to the zero position and the offset of the position, indicated by laser, was measured. See figure below.

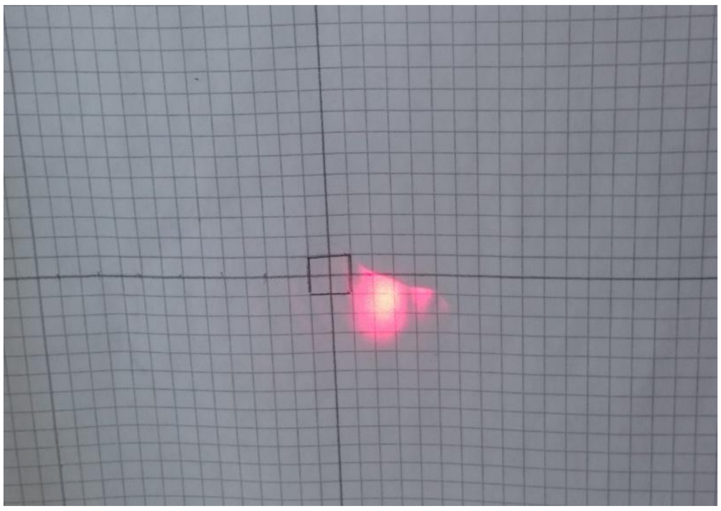


Figure 5.4. Test result

The offset in azimuth was approximately +1,5 cm and in elevation -1 cm at 20 m distance. The corresponding angular offsets are 0,043 and 0,028 degrees. If to observe the datasheet of the motors, it can be noted that no load backlash for the gearbox is smaller or equal to 1 degree. Based on that, it can be said that the current offsets are rather caused by the gearbox or transmission backlash than the motors missing steps. Note that the tests were carried out in quarter-step mode to ensure optimal resolution with smooth running and trading off some rotational speed. The theoretical maximum angular speed of the motors with current settings is 36,685 degrees per second. The full rotation of the test rig's vertical axis took 13 seconds including the acceleration and deceleration, and it translates to the average rotational speed of 27,7 degrees per second. This is slightly less than specified in the requirements but still superior than the maximum rotation speeds of TAS-5000, Seatite Wave Relay Tracking system and OS-PT-25 antenna tracker [30]. Based on this test it can be said that the motors can handle the current load and speed configuration fairly well and even if the test rig is constructed with inferior quality compared to the actual tracker, the test rig fits the accuracy and resolution requirements for the tracker.

6 DETAILED DESIGN

6.1 Mechanical

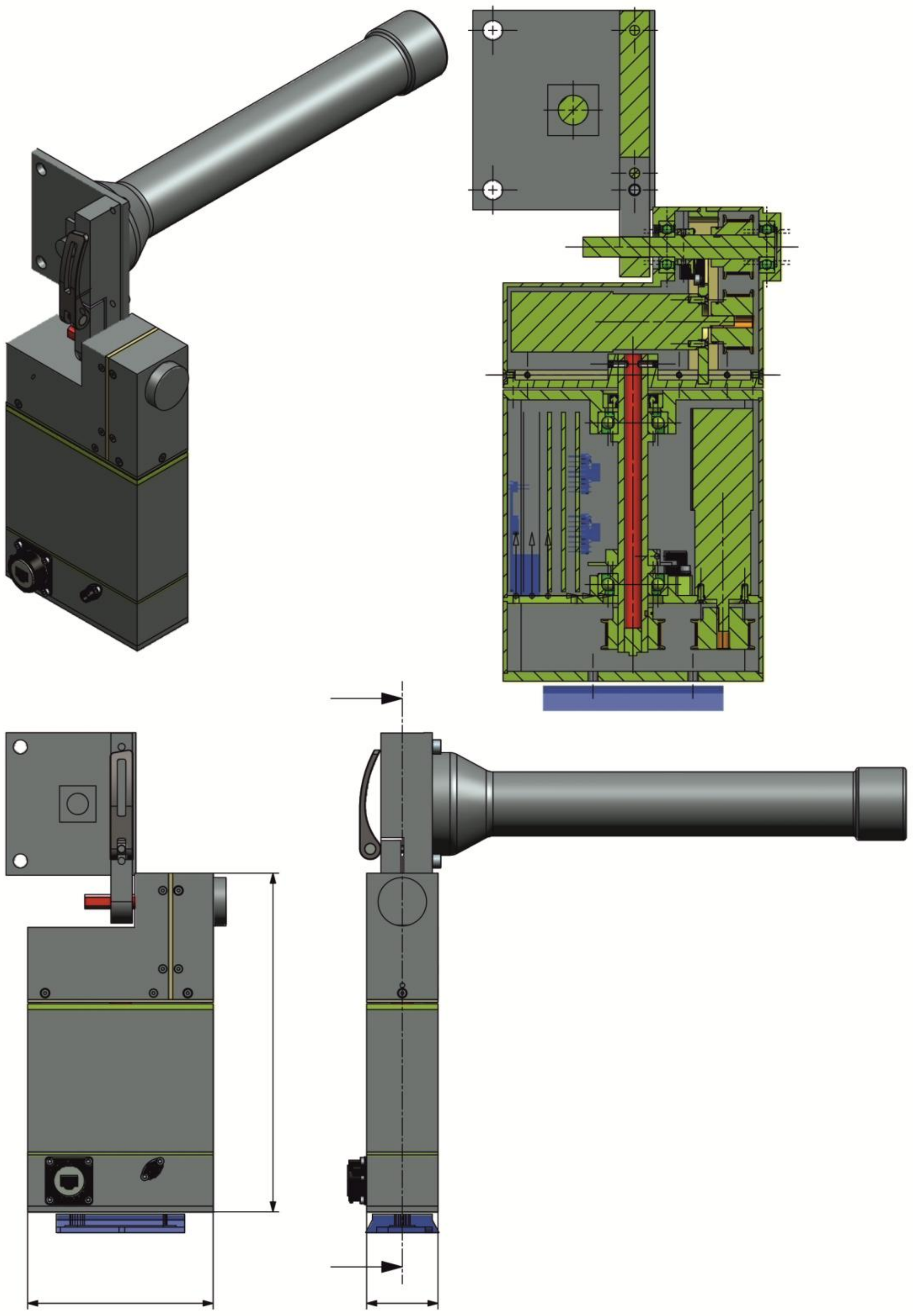


Figure 6.1. Concept 4

The fourth and the last version of the antenna tracker is based mainly on the third version (see Figure Figure 6.1. above). There are a number of modifications made. First, all the rounded corners of the trackers outer edges were removed to simplify the production. The second development was introduced by the change in the main payload. If previous concepts were designed mainly for carrying the MA-WA24-2X batch antenna, then during the development the main payload changed due to the company's implementation of the new main antenna type. The new antenna that the tracker has to carry is STAR ANTENNA's SCH-2412 helical antenna [31]. Since the new antenna is lighter (535 g) than the previous antenna (860 g), it was not necessary to reassess the design criteria. The overall dimensions of the new antenna are 100 x 90 x 395 mm, this leads to the possibility to use different mounting methods. So the previous tracker design with two horizontal shaft outputs was altered. The right-hand (RH) side of the shaft output was neglected and the RH bearing housing was enclosed. The shape of the left-hand (LH) upper housing was changed. The bearing housing was brought closer to the bulkhead, leaving some space to fit the horizontal shaft limit disc. The middle ball bearing mounted on the bulkhead was neglected, making the room available for the two optointerrupt sensors. The LH end of the horizontal shaft is exposed by 35 mm out from the LH upper housing, providing the clearance to attach the antenna mounting arm to the flattened end of the shaft. The mounting arm has a bore with a shape matched to the flat end of the horizontal shaft. The section of the mounting arm is cut to create the slot opening leading to the bore of the mounting arm. This slot creates the possibility to change the diameter of the bore by tightening the eccentric lever attached to the mounting arm. This creates the quick release mechanism for the antenna mounting arm. The arm can be easily removed by pulling it off the horizontal axis. Since the antenna mounting arm is oriented vertically at the neutral position, the antenna can be actuated around the horizontal axis in a larger range than it was possible before. This is achieved mainly by avoiding long vertical antenna surfaces, offset from the rotation axis, that can be interfering with the lower housing of the tracker. The antenna can be mechanically actuated in the range of +90 to -90 degrees. Since the negative elevation angles in such magnitude are not needed, the rotation below horizon is limited to -20 degrees by using the horizontal shaft limit disc and limiting in the control algorithm. The limit disc is mounted on the horizontal shaft and it is the 250 degree section of the disc from -20 to +90 degrees, rest of the disc is cut away. The disc interacts with two optointerrupt sensors placed at a 90 degree angle relative to each other. If the first sensor (left in the image below) senses the state change while the second (right sensor in the image below) is not interrupted, the current position refers to the minimal position of the elevation. If the first sensor is not interrupted and the second sensor senses the state change, the current position refers to the zero position of the elevation. And if the second sensor is interrupted and the first sensor changes the state, the current shaft position is at maximum allowed elevation.

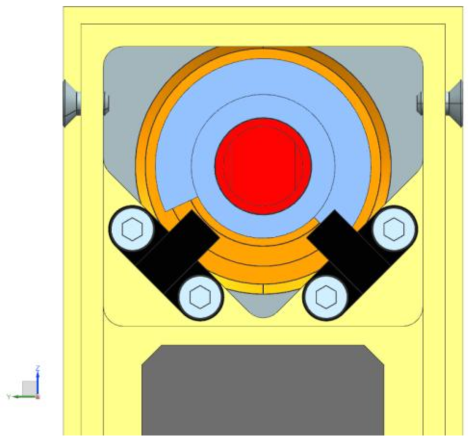


Figure 6.2. Optointerrupt placement

The neutral position of the vertical axis is determined by the similar limit disc (with 180 degree section removed) and one optointerrupt sensor that registers the state change. The tracker can be mounted in the commercially available camera mounting systems like Manfrotto 501 Head mount by two 6 mm bolts at the bottom floor. The estimated weight of the tracker without the antenna and mounting arm is 2.7 kg, the general outer dimensions are 130 x 50 x 238 mm and overall volume approximately 2 liters.

Next the assembly process of the antenna tracker is described. The assembly of the tracker starts by fitting the 7202 bearings into the bearing housings situated on the middle and top floors (see Figure Figure 6.3. below). Subsequently the top floor is screwed in place at the top surface of the middle housing. After that, the stepper motor and the optointerrupt sensor are attached to the middle floor. Then the limit disc is fitted to the vertical shaft. Following, the vertical shaft is fitted into the bearing situated in the top floor. After that, the middle floor with the assembled components is fit to place at the bottom surface of the middle housing and is temporarily bolted. This fixes the vertical shaft between the ball bearings. Next the timing pulleys and the timing belt are attached to the vertical shaft and stepper motor. After the pulleys are installed, the slipring capsule is inserted in the hollow vertical shaft and the cables are driven through the shaft. After that, the power and Ethernet bayonet sockets are installed to the bottom housing. The temporary bolts are removed and the bottom housing with the bottom floor are aligned with the assembly and bolted together. After that, the shaft seal is inserted to the cavity in the top floor. By this the lower housing of the antenna tracker is assembled.

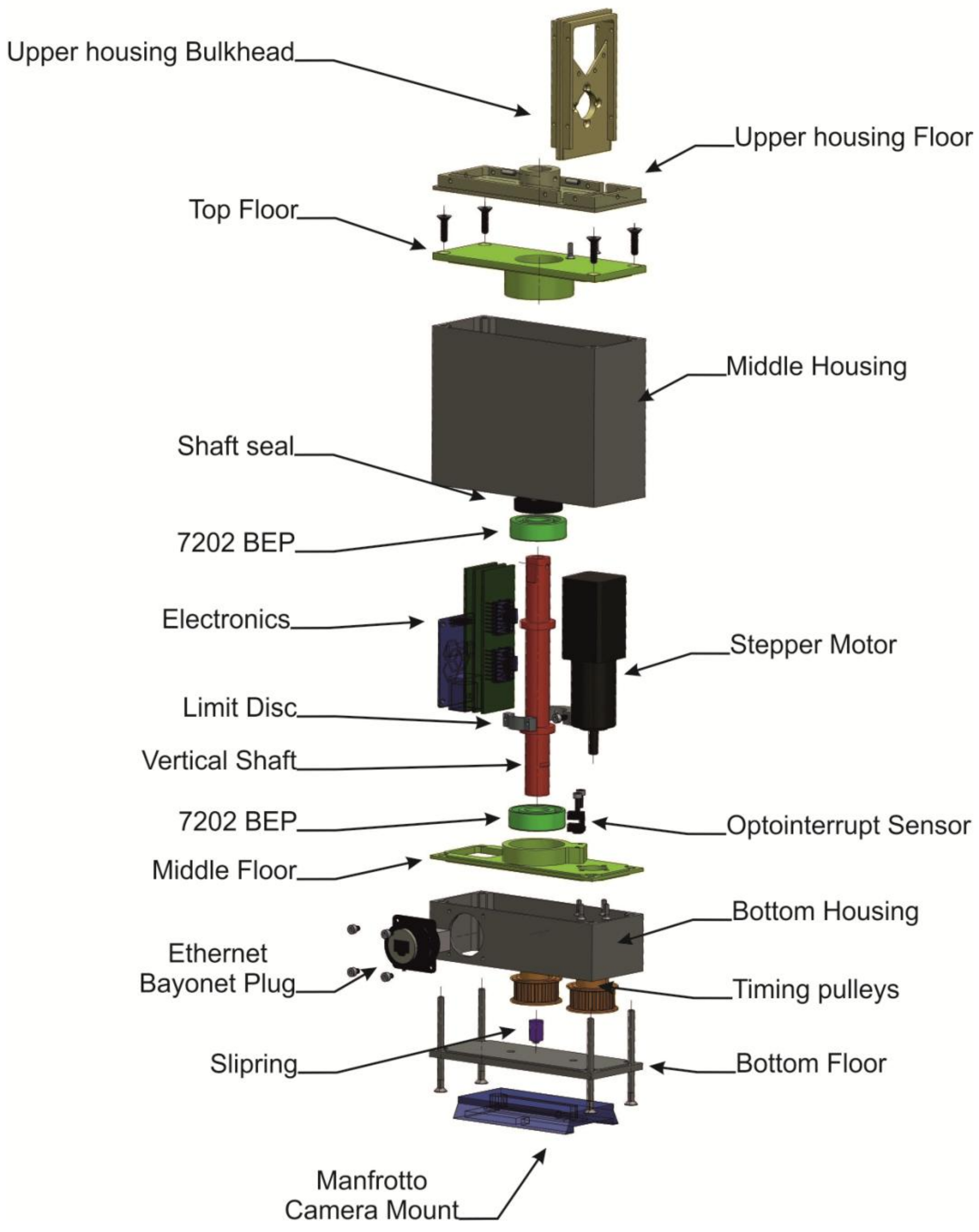


Figure 6.3. Exploded view 1

While using some types of the antennas, only the actuation of the vertical axis is required. In that case the lower housing of the tracker can be easily converted for that purpose by attaching the antenna directly to the vertical shaft. In order to actuate both axis, the upper housing is assembled and added to the lower housing.

The assembly of the upper housing is pictured in the Figure Figure 6.4. below. To assemble the upper housing, the upper housing bulkhead is bolted to the upper housing floor and attached to the exposed end of the vertical shaft. The joint is reinforced with two set screws. Subsequently the second stepper motor and the optointerrupt sensors are fixed to the bulkhead, followed by attaching the limit disc, the ball bearings and the timing pulley to the horizontal shaft. During this process the shaft is settled through the cavity in the bulkhead. Then the timing belt is being installed. Next the right-hand upper housing is installed and the horizontal shaft with its right-hand bearing is fitted into the bearing well in the right-hand upper housing. Next the left-hand upper housing is fixed to its place and the left-hand bearing of the horizontal shaft is being settled in the bearing well at the left-hand upper housing. This fixes the horizontal shaft between the two bearing wells situated at the upper housings. At last the antenna is bolted to the antenna mounting arm and the arm is slided to the suitable position at the horizontal shaft and the eccentric lever is used to tighten the fit around the horizontal shaft.

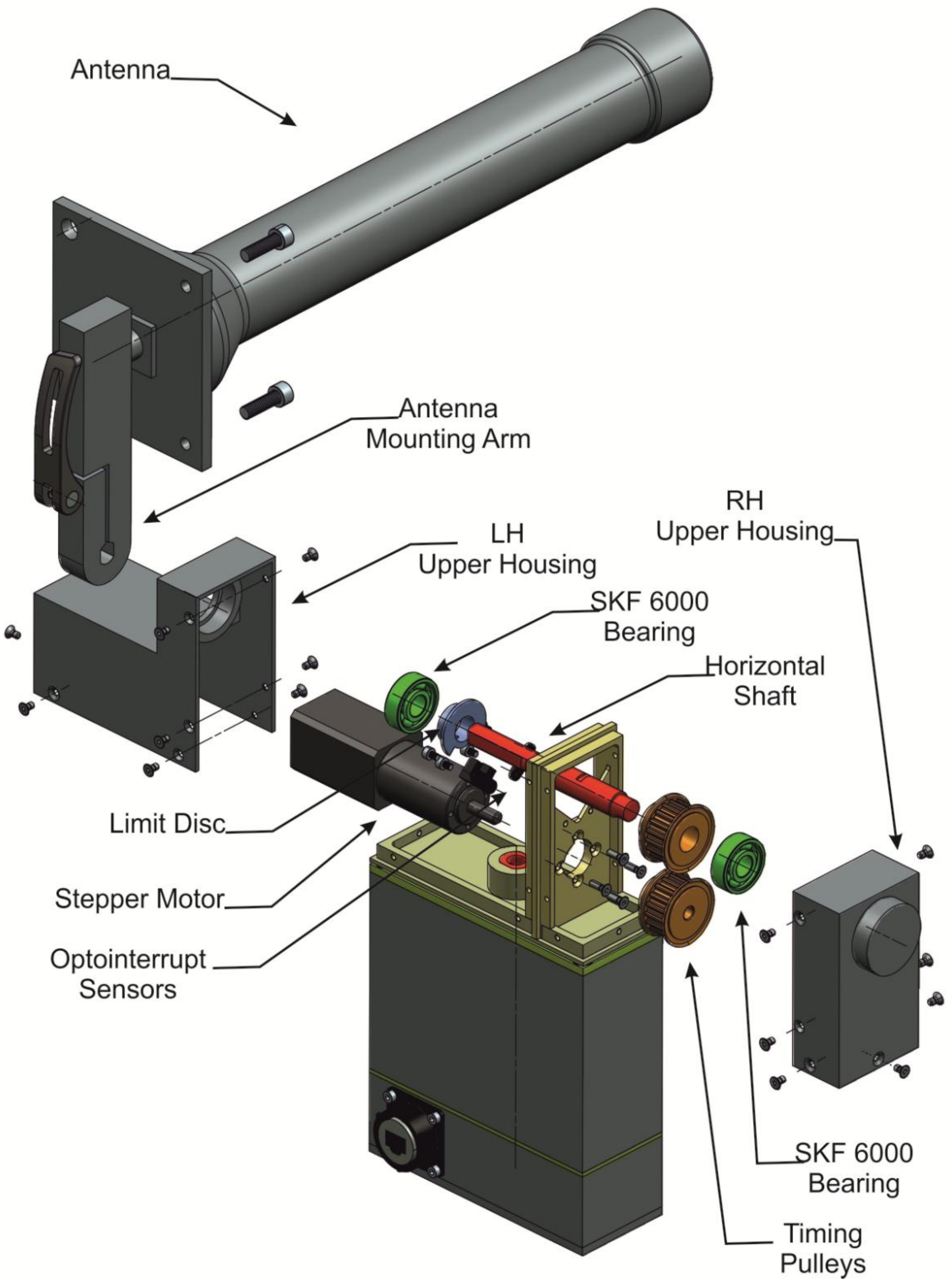


Figure 6.4. Exploded view 2

6.2 Electrical

As it was pointed out in chapter 5.1, the electrical design of the tracker had to be altered to ensure the smooth operation of the tracker. The final design of the tracker controller consists of two microcontrollers that work in conjunction with the Ethernet card, motor drivers and the optointerrupt sensors. The master controller, ATMega 2560 based Arduino MEGA was used as basis of the controller block. The prototyping board based on the Arduino Mega was used to create the shield type second layer for mounting the rest of the items to the master controller. On to that shield the slave controller, ATMega328 based Arduino Nano, the motor drivers and the connectors for the power, motors and sensors were fitted. For the test version, Arduino Nano was chosen to be the slave controller mainly because of its compact size that made it possible to be mounted on the shield. The assembled test controller is shown in the figure below.

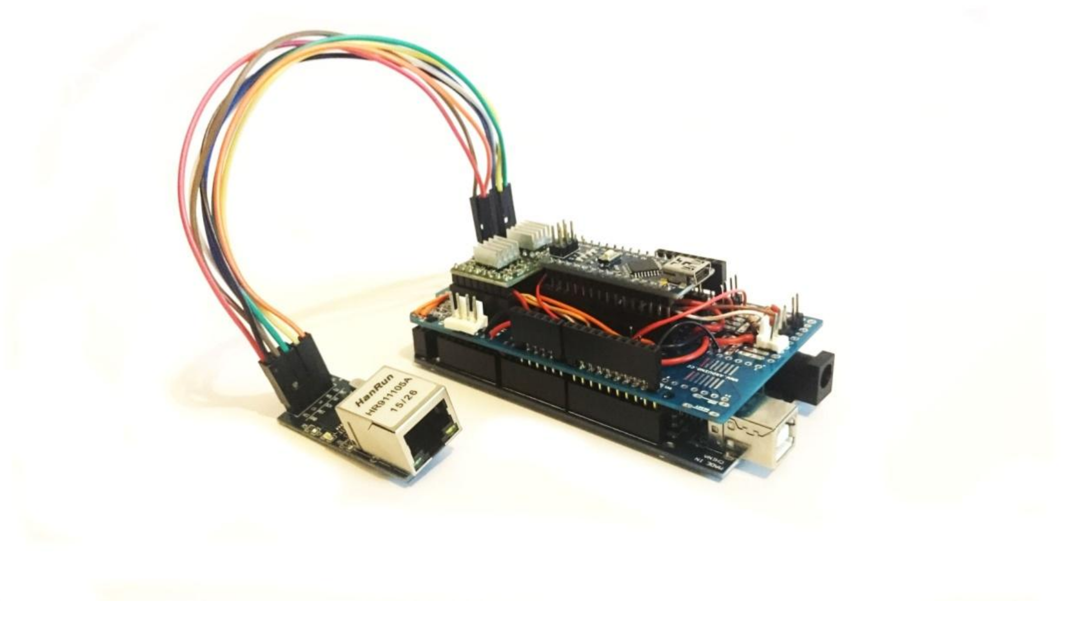


Figure 6.5. Improved control block

The Ethernet module is powered by the master controller and it communicates with the master controller by using the Serial Peripheral Interface (SPI) bus. The Ethernet module is wired to the digital IO pins 53 (CS), 51 (SI), 50 (SO), 52 (SCK), RESET (RESET), D2 (INT), 3V3 (VCC) and GND (GND). The master controller is responsible of receiving the data from the Ethernet and processing the data to the suitable format for the motor controlling library in the slave controller. The processed data will be sent to the slave controller via I2C serial bus. The I2C connection is established via SCL (A5) and SDA (A4) pins. The schematics of the connections of the controllers and Ethernet module as shown on the Figure Figure 6.6. below.

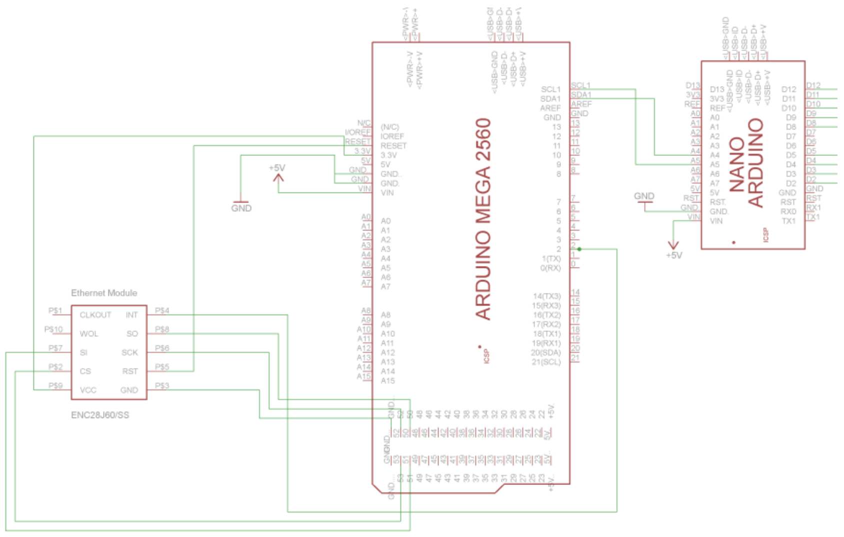


Figure 6.6. Dual microcontroller layout (controllers and Ethernet module)

The slave controller is responsible for managing the peripheral devices like the A4988 motor drivers and the optointerrupt sensors. All the devices on the prototype shield are mounted by using the female pin headers on the prototype board and the matching spiked pin headers on the devices to gain the ability for easy mounting and demounting the devices for quick troubleshooting. The slave controller uses the digital IO pins 11 and 12 to enable the current flow in the X and Y axis motors via the motor drivers. The digital IO pins 4 and 5 are used to feed the step impulses and the direction of rotation for the X axis motor driver, and the digital IO pins 3 and 4 are used to feed the step impulses and the direction of rotation for the Y axis motor driver. The stepper drivers are hard-wired to run in the quarter-step mode, by applying +5 V to the MS2 pins, to gain the smooth motion of the steppers and maximum resolution. The I2C interface at the slave controller is wired to the analog pins 4 (SDA) and 5 (SCL). The optointerrupt sensors use the voltage divider with 220 and 427 ohm resistors to provide the 3,3 V supply voltage for the diode. The connections between the slave controller and the peripheral devices are shown in the Figure Figure 6.7. below.

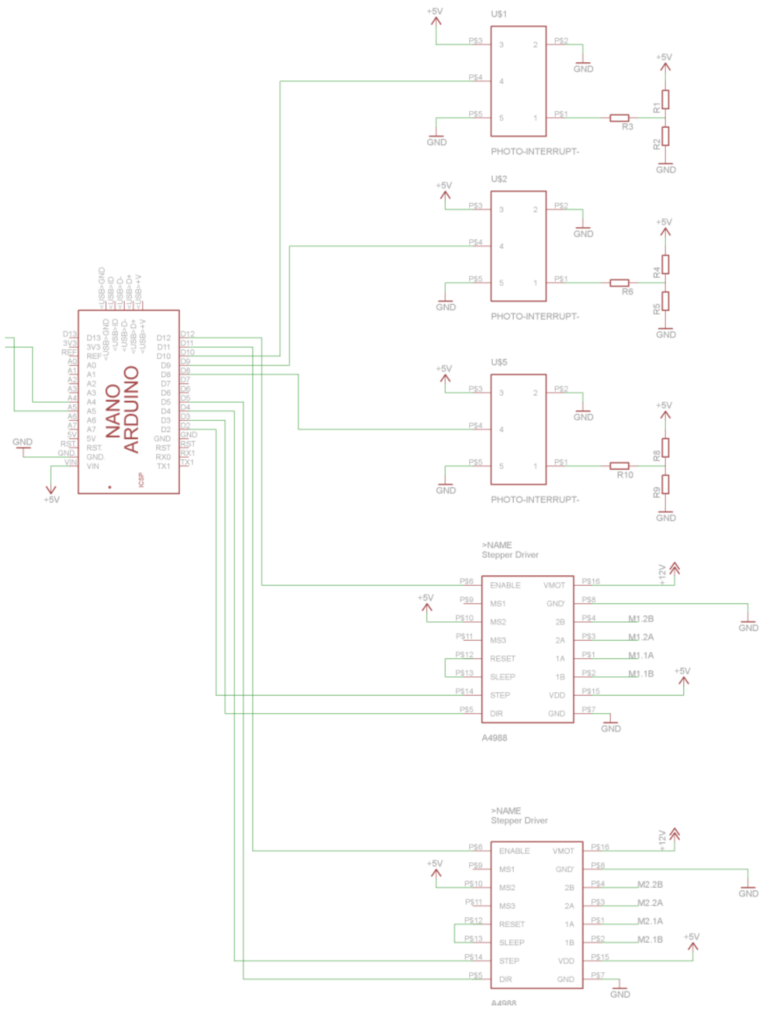


Figure 6.7. Slave controller and peripherials

The test version of the control device is powered by external power supply. The test version of the tracker controller is used for proving the concept and testing the operability of the concept. As it comes to the prototype for the actual tracker, the custom made stackable circuit boards must be made based on the design in the thesis. The circuit boards embed the actual microcontrollers with their peripheral components and mountings for the stepper drivers. The full schematics of the test version are available at the appendix 8.2 on page 70.

6.3 Control

Since the improved concept of the electric design utilizes two microcontrollers, the control code of the tracker system is split in two as well. First we observe the operation of the master controller code. The controller code utilizes the previously described EtherCard library for managing the Ethernet module and the Easy Transfer Library developed by Bill Porter for I2C communication [32]. As the controller powers up, the setup sequence begins. In the setup, first the libraries, the global variables and the IO pins are defined. Subsequently the I2C parameters like the slave address (9) and the data structure (int absoluteX;int absoluteY;) are defined. The definition of the Ethernet parameters like MAC address, IP address, default gateway and ports follow. After the definitions, the I2C and Ethernet connections are initialized. If the Ethernet fails to initialize, the software reset is carried out to restart the controller and reattempt to establish the Ethernet connection. The controller continues this loop until the Ethernet connection is established. As this happens, the setup is finished and the main loop begins. The main loop is fairly simple, it consists of polling the Ethernet card. As the Ethernet card has received the UPD message, the udpSerialPrint function is called. This function saves the UDP message as the string value and parses the string into two separate temporary values (tempX, tempY) using the colon sign as the separator. The separate values are converted to float values (targetX, targetY) that are suitable for calculations. The target values (absX, absY) for the stepper.run functions in the slave controllers are obtained through the following calculations.

$$absX = 4 * targetX * 28{,}8305;\ absY = 4 * targetY * 28{,}8305.$$

The multiplier 4 in the formula comes in because the stepper drivers are using the quarter-step mode and thus need four times more impulses to move the steppers through the same distance as in the single step mode. The coefficient 28,8305 is obtained by simplifying the following equation into one resulting coefficient for saving the time of execution.

$$\frac{SPR*GR}{360} = \frac{200*51{,}895}{360} = 28{,}8305 \text{ steps/deg},$$

where SPR - number of motor steps in one shaft revolution, GR - the gearbox reduction ratio. As the coordinates are converted to the step values, the obtained values are sent to the slave controller via I2C by ET.sendData function.

The flowchart of the master controller code is shown at the figure below. To see the full code in more detail, refer to the appendix 8.3.3 on page 74.

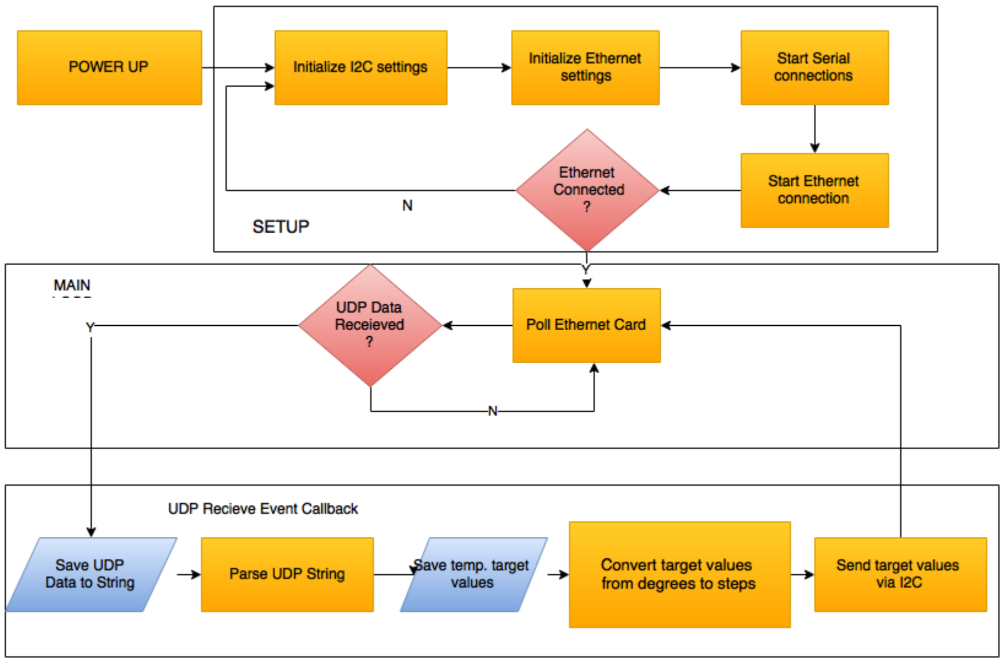


Figure 6.8. Master controller flowchart

The slave controller's code consists of two parts: the setup and the main loop (see figure below). In the setup section, first the global variables and libraries are defined, then the I2C parameters (address and data structure) are defined. Subsequently the motor parameters are set. The current motor configurations are: maximum speed 4000 pps and acceleration of 2000 p/s^2 for both of the motors. Next the I2C connection is established with the master device and the axis calibration process begins.

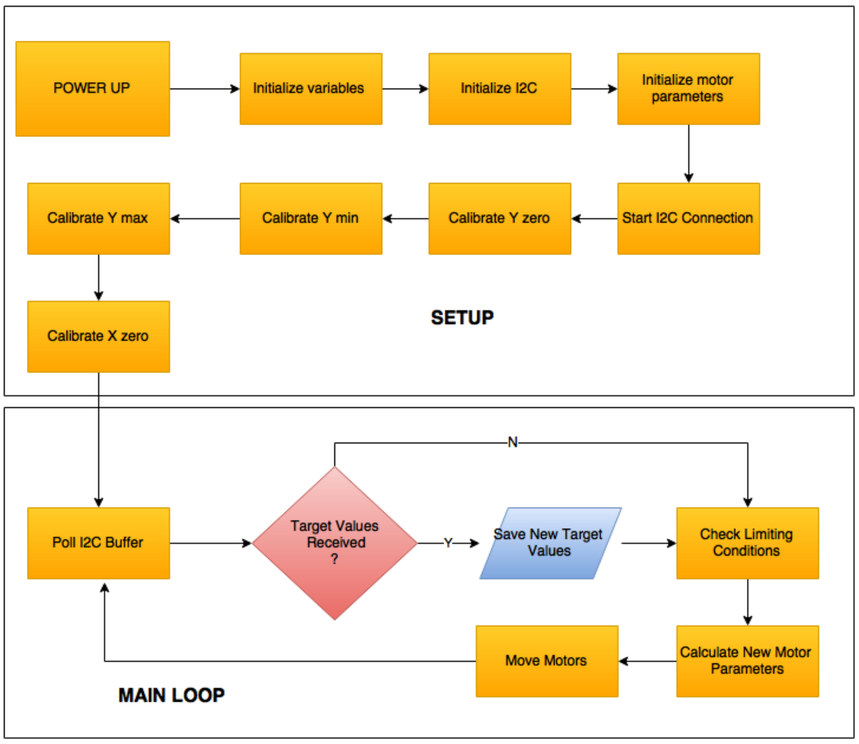


Figure 6.9. Slave controller code flowchart

First the Y axis calibration takes place. The Y axis is first because the antenna must be oriented safely before the movement of the vertical axis can be done, otherwise there is a possibility to damage the antenna. The axis calibration is carried out by reading the values from the optointerrupt sensors and moving the axis, based on the calibration algorithm. First the Y axis zero point, where the antenna is oriented to the horizon, is found. For finding the zero point, the program enters a loop and stays there while the flag Yz, indicating that the zero point is reached, is risen. In the loop, first the sensor values (s1 and s2) are read, if the sensor values differ from the last read, the state change indicators (L1 and L2) for sensor values are toggled. By following the algorithm described on the flowchart below, the loop increases or decreases the set point for the motor controllers and actuates the motors by one step until the Y axis reaches the zero point. In the algorithm, two levels of checking conditions are done. First the states of the sensor values are tested to determine the region of the position and second, the state change indicators are tested to determine the specific conditions like if the edge of the optical disc has just passed the sensor during the last step, causing the sensor state to change or the last step has been carried out in the middle of the constant state.

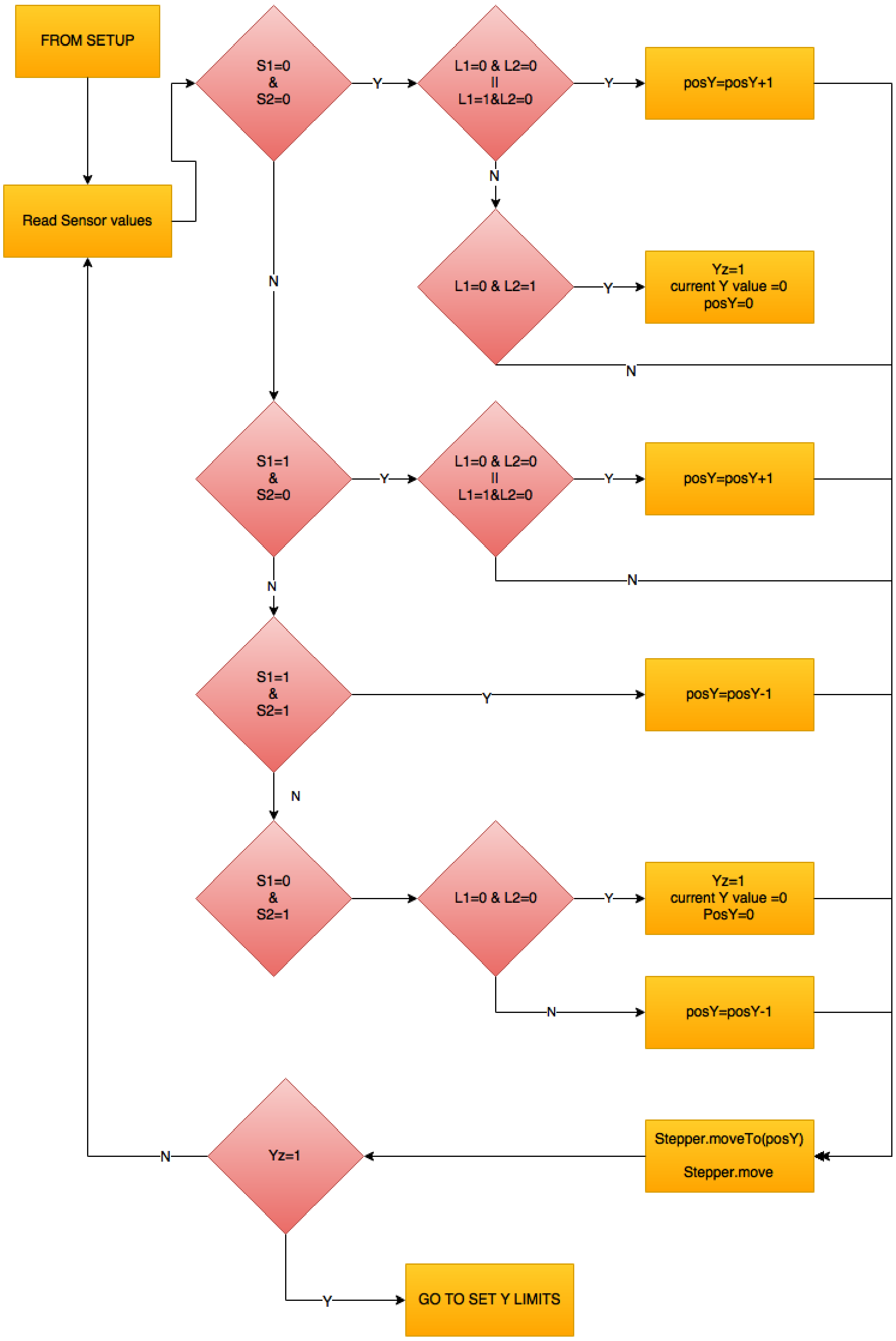


Figure 6.10. Y axis zero point calibration flowchart

After the zero point is reached, new loop begins to determine the minimum allowed position. For that the position of the horizontal shaft is lowered by one step by each iteration of the loop and is stopped when the optic disc reaches the sensor. The step value for that position is stored. Subsequently similar loop is being entered to move the horizontal shaft to maximum position and store its step value. By replacing the optic disc with a disc with different geometry or by adding or subtracting a certain coefficient to the limit variables, the maximum positions of the axis can be altered. As the Y axis maximum point is determined, the Y axis shaft is moved to the neutral position and the X axis zero calibration begins. The flowchart for the X axis zero point determination is illustrated in the figure below.

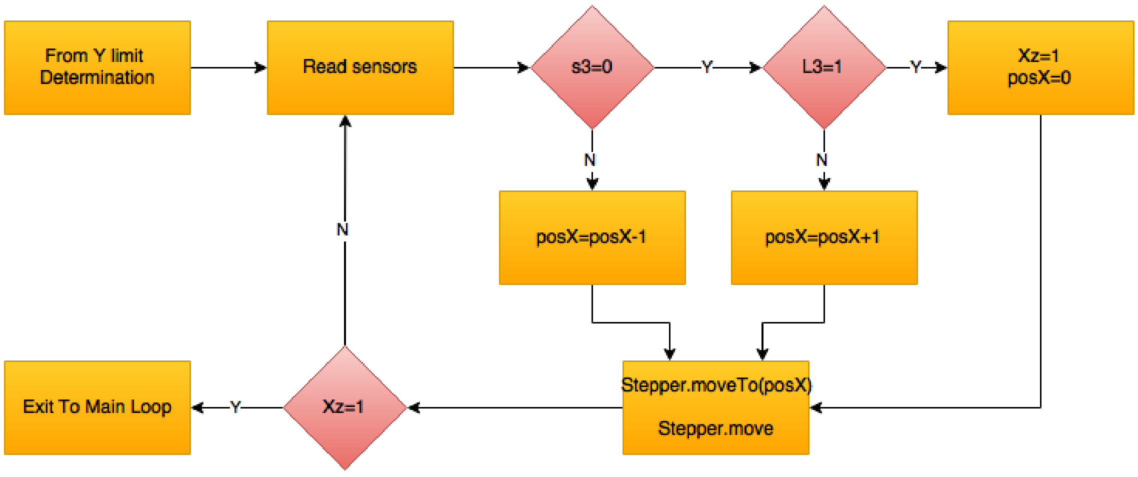


Figure 6.11. X axis zero calibration flowchart

Similarly to the Y axis calibration, the X axis zero point determination algorithm uses the current state and last state values of the optical sensor (s3 and L3). The algorithm searches for one specific combination of the values that appears when one edge of the optical disc moves over the optical sensor. As this happens, the current position of the steps is regarded as a new neutral point and the target value is zeroed. Until that happens, the target value of the stepper controller is ether increased or decreased in each iteration of the loop and the movement is carried out by the motors. As the axis are calibrated, the setup ends and the main loop starts. The main loop starts with checking if there has been new set of coordinates received via I2C. If so, then the new values are saved, if not, the old ones are still regarded as valid. Next the coordinate values are tested against the Y axis minimum and maximum allowed values. If the target values

exceed the limit values, the target values are overwritten by the maximum allowed values. This is regarded as the software limit switch, which does not allow the motors to exceed some predefined point. Subsequently the stepper.moveTo functions are called to calculate new speed profiles for current set points. Next the functions stepper.run are called to actuate the steps. The main loop is kept as short as possible to ensure the most frequent calling of the motor control functions stepper.moveTo and stepper.run. As the calling frequency of those functions increases, the maximum speed of driving the steppers increases as well.
The complete slave controller code is available at the appendix 8.3.4 on page 77.

7 SUMMARY

In this chapter the conclusions for the work carried out are presented. In addition, the future tasks and actions that should be performed, are pointed out.

7.1 Conclusions

The design process of the antenna tracker device that was carried out, followed the common stages of the engineering design process. In the two first chapters the definition of the design criteria and investigation of the existing solutions and methods was carried out. The aim was to create the antenna tracker design that would be compact and suitable to be carried by one man, while meeting the certain performance criteria. The thesis can be regarded successful if the prototype of the antenna tracker could be built, based on the work. In the third chapter, the conceptual design for the tracker was created. Next, the design loop was entered for creating the preliminary design. The design process of some of the preliminary design versions were halted as it became evident that some of the design criteria were not met. The design in hand was altered or a new version was created. The preliminary design evolved by each iteration of the design loop. The fourth chapter describes the process of creating the preliminary design. The theoretical requirements for the key components were determined and the suitable components selected. Based on the defined parameters and selected components, the detailed CAD assembly model was created. The preliminary version of the code for the microcontroller was compiled. As the preliminary design was composed, the integration of the key electrical components and the suitability of the actuators were tested in the fifth chapter. The test results were positive and they also revealed some of the imperfections of the preliminary design. Considering the imperfections, the adjustments were made in the design. Based on the amended design, the detailed design was derived. The detailed design is described in the sixth chapter. The detailed design consists of the electric solution, the code of the microcontrollers, detailed three dimensional CAD assembly model of the tracker device and the two dimensional schematics for the parts that are needed to be manufactured.

Relying on the operational tests carried out with the test rig and the resulting detailed design, the performance parameters of the antenna tracker device are estimated to be the following: mass 2,7 kg, volume 2 dm^3, pan range 360 degrees, tilt range 110 degrees (from -20 to +90 degrees), maximum theoretical rotation speed 36,6 degrees per second, resolution 115 steps per degree, actuation accuracy 0,04 degrees. All the parameters listed above fit to the design criteria set in the specification of the design requirements. Also the final design is compact enough to be carried within the single person's gear, and thus meeting the most important criteria.

In author's opinion; based on the work carried out in this thesis, the first prototype of the antenna tracker can be built, although further development work is needed to develop the market-ready product out of the prototype design. The suggestions of the future actions are discussed in the discussion section below.

7.2 Discussion

Relying on the results of the tests of the actuators and the integration of the electric components, it can be stated that the current combination of the selected components is suitable for building the first prototype of the antenna tracker device. The first prototype would be a subject to field tests. The tests should assess the performance of the mechanical design and point out possible imperfections. During the field tests the fine tuning of the control algorithms should be carried out. Also the integration of the data communication between the ground control station software and the microcontrollers should be carried out. The device drivers for the ground control station's operating system should be developed. As the mechanical design proves to be suitable, the test version of the tracker control hardware should be fitted into the suitable size for housing it inside the tracker housing. For that, the custom made PCB design should be carried out.

KOKKUVÕTE

Antud töö käigus läbi viidud antenni suunamisseadme konstrueerimise protsess järgis üldlevinud konstrueerimise samme. Eesmärgiks oli konstrueerida antenni suunamise seade mis oleks piisavalt kompaktne ja portatiivne, et seda saaks ühe inimese varustuses transportida. Samas peaks antennisuunaja ka vastama teatud soorituslikele nõuetele. Töö loetakse kordaläinuks kui selle baasil saab valmistada antennisuunaja esimese prototüübi. Esimeses kahes peatükis määrati konstrueerimise aluskriteeriumid ning uuriti olemasolevaid lahendusi ning meetode. Selle baasil loodi kolmandas peatükis kontseptuaalne kavand, mida hakati edasi arendama esialgseks lahenduseks. Esialgse lahenduse loomiseks läbiti konstrueerimise protsessi mitmeid kordi. Mõningate tööversioonide arendamine katkestati, kuna tuli ilmsiks, et mõni tööversiooni omadustest ei vasta konstrueerimiskriteeriumitele. Kui nii juhtus, siis konstruktsiooni muudeti ning selle põhjal arendati välja uus tööversioon antenni suunamisseadmest. Esmane konstruktsioon päädis versiooniga, mis vastas vajalikele kriteeriumitele. Esmase konstruktsiooni loomiseks tehti neljandas peatükis teoreetilised arvutused, määramaks võtmesõlmede parameetrid ning antud arvutuste põhjal valiti sobivad komponendid. Sobilike komponentide ja detailide baasil loodi kolmemõõtmeline antennisuunamise seadme koostu mudel. Samuti koostati elektriskeemi kuvand ning visandati esialgne versioon juhtprogrammist. Viiendas peatükis koostati esialgse versiooni põhjal testseade, katsetamaks süsteemide integratsiooni ja komponentide sobivust. Katsetused osutusid edukaks ning nende tulemuste põhjal korrigeeriti esialgset konstruktsiooni ja selle baasil töötati välja lõplik versioon antenni suunamise seadmest. Kuues peatükk kirjeldab seadme lõplikku versiooni. Lõplikust versioonist koostati kolmemõõtmeline koostumudel, tööjoonised toodetavatele detailidele, funktsioneeriv juhtprogramm ning elektriline lahendus.

Tuginedes katsetustele ning lõpliku versiooni mudelile saab välja tuua alljärgnevad antennisuunamise seadme lõpliku versiooni parameetrid: mass 2,7 kg, ruumala 2 dm^3, aktuatsiooni diapasoon vertikaal teljel 360 kraadi, horisontaalsel teljel 110 kraadi (alates -20 kuni +90 kraadi), maksimaalne teoreetiline liikumiskiirus 36,6 kraadi sekundis, resolutsioon 115 sammu kraadi kohta, täpsus 0,04 kraadi. Kõik eelpool mainitud parameetrid vastavad püstitatud eesmärkidele ning lõplik lahendus on piisavalt kompktne, et seda saaks kanda seljakotis.

Seega on autori hinnangul töö tulemuseks on antenni suunaja lahendus, millele tuginedes on võimalik toota esimene prototüüp. Selleks, et antud töös esitatud lahendusest saaks müügikõlbulik toode, on tarvis veel mõningast tootearendust. Alljärgnevas lõigus esitab autor soovitusi ja ettepanekuid millele võiks seadme edasi arendamisel tähelepanu pöörata.

Arutelu

Tuginedes seadmete integratsiooni- ja aktuaatorite sobivuskatsete tulemustele saab väita, et antud töö käigus valitud komponendid on sobilikud esmase prototüübi valmistamiseks. Selleks et arendada antenni suunajast müügikõlbulik toode tuleks esmase prototüübiga läbi viia välikatsetused mille käigus hinnataks mehhaanilise konstruktsiooni töökindlust reaalses töökeskkonas ja leitaks võimalikud kitsaskohad lahenduses. Katsetuste käigus saaks optimeerida ja peenhäälestada juhtprogrammi ning integreerida andmeside maapealse juhtkeskuse tarkvaraga. Kui katsetulemused osutuvad edukaks, tuleks väljatöötatud kontrolleri riistvara koondada sobivas formaadis trükkplaatidele ning integreerida need olemas oleva antennisuunaja korpusega.

REFERENCES

[1] L. Engineering., "tracking antenna," 04 2016. [Online]. Available: https://latitudeengineering.com/products/antenna/,. [Accessed 15 04 2016].

[2] "OptimumSolutions," OptimumSolutions, 2007. [Online]. Available: http://www.optimumsolution.com/item19.html. [Accessed 15 04 2016].

[3] G. Woan, in *The Cambridge Handbook of Physics Formulas*, Cambridge, ambridge University Press, 2000, p. 75.

[4] G. Woan, in *The Cambridge Handbook of Physics Formulas*, Cambridge, Cambridge University Press, 2000, 2000, p. 75.

[5] C. W. d. Silva, in *Mechatronics: A Foundation Course.*, Boca Ranton, CRC Press, 2010, p. 476.

[6] P. Corporation, "25D mm Gearmotors," 2016. [Online]. Available: https://www.pololu.com/category/115/25d-mm-gearmotors. [Accessed 07 04 2016].

[7] P. Corporation, "31:1 Metal Gearmotor 37Dx73L mm with 64 CPR Encoder,," 2016. [Online]. Available: https://www.pololu.com/product/2827. [Accessed 07 04 2016].

[8] S. Online, "Gear Ratio 51:1 Planetary Gearbox With Nema 11 Stepper Motor 11HS20-0674S-PG51," 2016. [Online]. Available: http://www.omc-stepperonline.com/gear-ratio-511-planetary-gearbox-with-nema-11-stepper-motor-11hs200674spg51-p-135.html. [Accessed 07 04 2016].

[9] S. Online, "Gear Ratio 100:1 Planetary Gearbox With Nema 11 Stepper Motor 11HS20-0674S-PG100," 2016. [Online]. Available: http://www.omc-stepperonline.com/gear-ratio-1001-planetary-gearbox-with-nema-11-stepper-motor-11hs200674spg100-p-36.html. [Accessed 07 04 2016].

[10 S. Online, "Gear Ratio 90:1 Planetary Gearbox With Nema 8 Stepper Motor

] 8HS15-0604S-PG90," 2016. [Online]. Available: http://www.omc-stepperonline.com/gear-ratio-901-planetary-gearbox-with-nema-8-stepper-motor-8hs150604spg90-p-132.html. [Accessed 07 04 2016].

[11] P. Corporation, "31:1 Metal Gearmotor 37Dx73L mm with 64 CPR Encoder, resources.," 2016. [Online]. Available: https://www.pololu.com/product/2827/resources. [Accessed 07 04 2016].

[12] P. Corporation, "31:1 Metal Gearmotor 37Dx73L mm with 64 CPR Encoder, description," 2016. [Online]. Available: https://www.pololu.com/product/2827. [Accessed 07 04 2016].

[13] S. Online, "Nema 11 Stepper Motor 11HS20-0674S-PG51 Dtatasheet," 2016. [Online]. Available: http://www.omc-stepperonline.com/download/pdf/11HS20-0674S-PG51.pdf. [Accessed 08 04 2016].

[14] Barlintimes, "Capsule Slipring 8mm, Datasheet," 2016. [Online]. Available: http://www.barlintimes.com/pdf/THR008-08AM.pdf. [Accessed 08 04 2016].

[15] R. G. M. H. .. Ulrich Fischer, in *Mehaanikainseneri käsiraamat*, Tallinn, TTÜ kirjastus, 2015, pp. 45-49.

[16] SKF, "7202 BEP," 2016. [Online]. Available: BEP.http://www.skf.com/group/products/bearings-units-housings/ball-bearings/angular-contact-ball-bearings/single-row-angular-contact-ball-bearings/single-row/index.html?designation=7202%20BEP. [Accessed 08 04 2016].

[17] SKF, "6000," 2016. [Online]. Available: http://www.skf.com/group/products/bearings-units-housings/ball-bearings/deep-groove-ball-bearings/single-row-deep-groove-ball-bearings/single-row/index.html?designation=6000. [Accessed 08 04 2016].

[18] G. M. Inc, "Tooth Pitch Selection.," 2016. [Online]. Available: http://www.gatesmectrol.com/mectrol/brochure.cfm?brochure=3143. [Accessed 09

04 2016].

[19] S. Instrument, "Stock Drive Products," 23016. [Online]. Available: http://sdp-si.com/D265/HTML/D265T079.html,. [Accessed 09 04 2016].

[20] I. Allegro MicroSystems, "A4988 DMOS Microstepping Driver with Translator," 2016. [Online]. Available: http://www.elecrow.com/download/a4988_DMOS_microstepping_driver_with_translator.pdf. [Accessed 12 04 2016].

[21] S. Corporation, "GP1A51HRJ00F datasheet.," 2005. [Online]. Available: http://www.farnell.com/datasheets/1671196.pdf. [Accessed 12 04 2016].

[22] Arduino, "Arduino UNO & Genuino UNO," 2016. [Online]. Available: https://www.arduino.cc/en/main/arduinoBoardUno. [Accessed 12 04 2016].

[23] Arduino, "Arduino MEGA 2560 & Genuino MEGA 2560," 2016. [Online]. Available: https://www.arduino.cc/en/Main/arduinoBoardMega2560. [Accessed 12 04 2016].

[24] 2. Ltd., "ENC28J60-H development board Users Manual," 2016. [Online]. Available: http://ee.farnell.com/olimex/enc28j60-h/enc28j60-ethernet-spi-dev-board/dp/1701543. [Accessed 13 04 2016].

[25] T. Instruments, "PTN78020WAH datasheet," 2011. [Online]. Available: http://www.farnell.com/datasheets/1874637.pdf. [Accessed 13 04 2016].

[26] jeelabs, "Ardino interface library for the ENC28J60 Ethernet controller chip (GPL)," 2015. [Online]. Available: http://jeelabs.org/pub/docs/ethercard/md_README.html. [Accessed 05 05 2016].

[27] M. McCauley, "AccelStepper library for Arduino," 2016. [Online]. Available: http://www.airspayce.com/mikem/arduino/AccelStepper/index.html. [Accessed 15 04 2016].

[28 D. Austin, "Generate stepper-motor speed profiles in real time," 2004. [Online].

] Available: time.http://www.embedded.com/design/mcus-processors-and-socs/4006438/Generate-stepper-motor-speed-profiles-in-real-time. [Accessed 15 04 2016].

[29] C. W. d. Silva, in *Mechatronics: A Foundation Course*, Boca Ranton,, CRC Press, 2010, p. 475.

[30] S. R. Systems, "Wave Relay Tracker System," 2013. [Online]. Available: systems.com/products/radio-communications/tracker-system.html. [Accessed 14 05 2016].

[31] C. ELECTRONIC, "HELICAL RHCP / LHCP ANTENNA.," 2016. [Online]. Available: http://www.starantenna.com/model-detail.asp?RecID=60. [Accessed 15 05 2016].

[32] B. Porter, "Easy Transfer Arduino Library," 2011. [Online]. Available: http://www.billporter.info/2011/05/30/easytransfer-arduino-library/. [Accessed 14 05 2016].

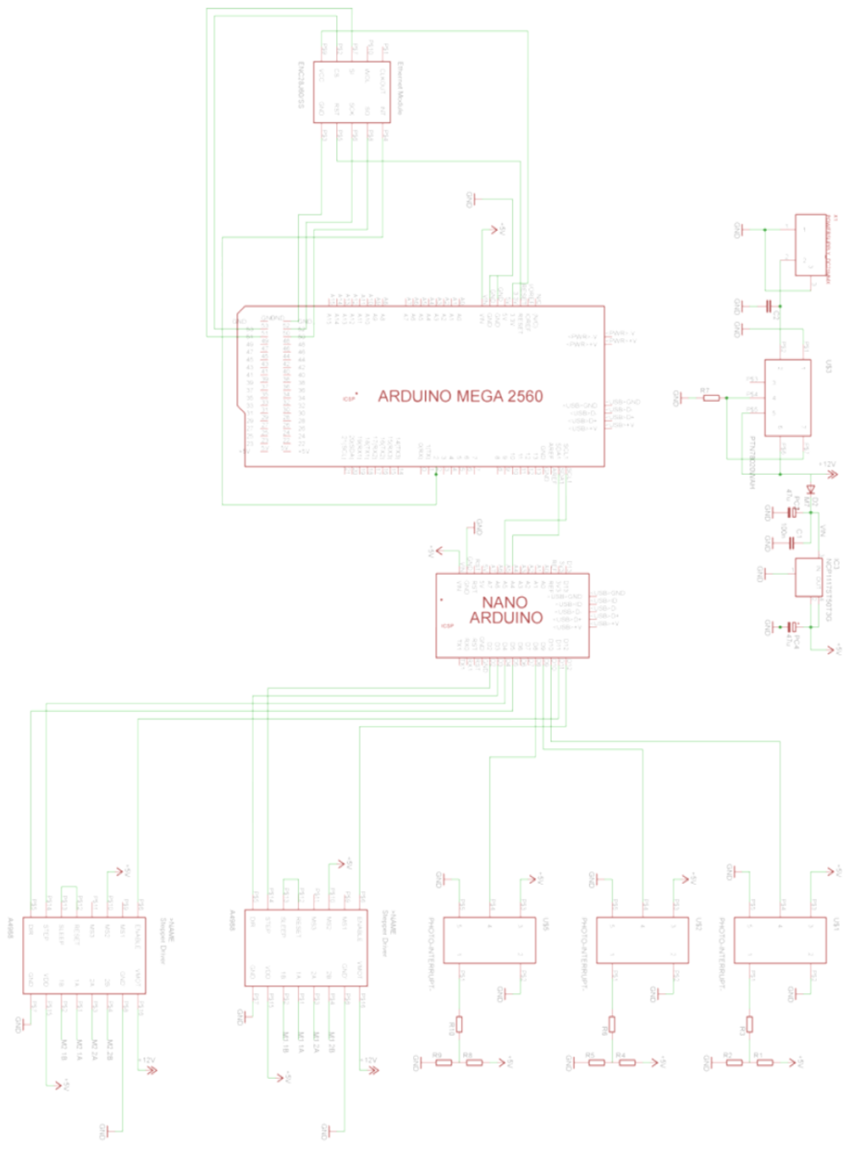
[33] L. S. Sterling, The Art of Agent-Oriented Modeling, London: The MIT Press, 2009.

8 APPENDIXES

8.1 Mechanical CAD drawings

The mechanical CAD drawings are appended to the end of the thesis. Pages 1 to 4 illustrate the general views, cut section and exploded views of the assembled antenna tracker. The pages 5 to 19 illustrate the schematics of the components to be manufactured or to be altered.

8.2 Electrical schematics of the test rig



8.3 Programs

8.3.1 MATLAB flight path generation code

```
clear
%Define speed
v=16;


% Defining Points

%LEG1
p1=[-4250 8600 -730];
p2=[-4250 600 -730];
%LEG2
p3=[4250 8600 -730];
%LEG3
p4=[4250, 600, -730];
%LEG4
p5=[-4250, 8600, 2250];
%LEG5
p6=[-4250 600 2250];
%LEG6
p7=[4250 8600 2250];
%LEG7
p8=[4250, 600, 2250];
%LEG8
%p8--->p1
p9=[0, 0, 0];

%--------------------------------------LEG1
 % Circle x²+y²+z²=Constant
 %=> z=sqrt(Constant-x²-y²-);
 Center2=((p2+p1))./2;
 xc1=Center2(1);
 yc1=Center2(2);
 zc1=Center2(3);
 % Radius
 R1=norm((p2-p1))/2;

 %Min=min(min(p1,p2));
 %Max=max(max(p1,p2));
 %x=linspace(Min,Max,20);
 %y=linspace(Min,Max,20);

 %x=linspace(p1(1),p2(1),200);
% y=linspace(p1(2),p2(2),200);
 %z=linspace(p1(3),p2(3),200);
 %y=yc1+sqrt((R1^2)-((x-xc1).^2)-((z-zc1).^2));
 for c1= 1:(pi*R1/v)
  x(c1,1) = R1*sin(-c1*v/R1)+xc1;
  y(c1,1) = R1*cos(-c1*v/R1)+yc1;
  z(c1,1) = R1*0+zc1;

 end
 %------------------------------------LEG2
 L1=norm(p3-p2);
 xk1=(p3(1)-p2(1))/L1;
 yk1=(p3(2)-p2(2))/L1;
 zk1=(p3(3)-p2(3))/L1;

 for c2=1:(L1/v)
   x(end+1,1) = p2(1)+(xk1*c2*v);
   y(end+1,1) = p2(2)+(yk1*c2*v);
   z(end+1,1) = p2(3);
 end
%---------------------------------------LEG3
 % Circle x²+y²+z²=Constant
 %=> z=sqrt(Constant-x²-y²-);
 Center3=((p4+p3))./2;
 xc3=Center3(1);
 yc3=Center3(2);
 zc3=Center3(3);
 % Radius
 R3=norm((p4-p3))/2;


 for c3= 1:(pi*R3/v)
  x(end+1,1)  = R3*sin(c3*v/R3)+xc3;
  y(end+1,1)  = R3*cos(c3*v/R3)+yc3;
  z(end+1,1)  = R3*0+zc3;

 end

 %--------------------------------------------LEG4

 L4=norm(p5-p4);

 xk4=(p5(1)-p4(1))/L4;
 yk4=(p5(2)-p4(2))/L4;
 zk4=(p5(3)-p4(3))/L4;


 for c4=0:(L4/v)
   x(end+1,1)  = p4(1)+(xk4*c4*v);
   y(end+1,1)  = p4(2)+(yk4*c4*v);
   z(end+1,1)  = p4(3)+(zk4*c4*v);
 end



 %--------------------------------------------LEG5


 Center5=((p6+p5))./2;
 xc5=Center5(1);
 yc5=Center5(2);
 zc5=Center5(3);
 % Radius
 R5=norm((p6-p5))/2;


 for c5= 1:(pi*R5/v)
  x(end+1,1)  = R5*sin(-c5*v/R5)+xc5;
  y(end+1,1)  = R5*cos(-c5*v/R5)+yc5;
  z(end+1,1)  = R5*0+zc5;

 end

%-----------------------------------LEG6


 L6=norm(p7-p6);
 xk6=(p7(1)-p6(1))/L6;
 yk6=(p7(2)-p6(2))/L6;
 zk6=(p7(3)-p6(3))/L6;

 for c6=0:(L6/v)
   x(end+1,1)  = p6(1)+(xk6*c6*v);
   y(end+1,1)  = p6(2)+(yk6*c6*v);
   z(end+1,1)  = p6(3);
 end
```

```
%-----------------------------------LEG7

 % Circle x²+y²+z²=Constant
 %=> z=sqrt(Constant-x²-y²-);
 Center7=((p8+p7))./2;
 xc7=Center7(1);
 yc7=Center7(2);
 zc7=Center7(3);
 % Radius
 R7=norm((p8-p7))/2;


 for c7= 1:(pi*R7/v)
  x(end+1,1)  = R7*sin(c7*v/R7)+xc7;
  y(end+1,1)  = R7*cos(c7*v/R7)+yc7;
  z(end+1,1)  = R7*0+zc7;

 end


 %----------------------------------------LEG8

 L8=norm(p1-p8);

 xk8=(p1(1)-p8(1))/L8;
 yk8=(p1(2)-p8(2))/L8;
 zk8=(p1(3)-p8(3))/L8;



 for c8=0:(L8/v)
   x(end+1,1)  = p8(1)+(xk8*c8*v);
   y(end+1,1)  = p8(2)+(yk8*c8*v);
   z(end+1,1)  = p8(3)+(zk8*c8*v);
 end


 %--------Cartesian coordinates to Azimuth, Elevation and distance


 [az,el,r] = cart2sph(x,y,z)

 %radians to deg
 az = rad2deg(az)
 for i=1:size(az)
  az(i)=az(i)-90;
 end

 el = rad2deg(el)
 r = rad2deg(r)
 time=[1:size(az)],
 time=transpose(time);
 %AZ=[time(:,1) az(:,1)];

 [m,n]=size(az);

 minAz=min(az);
 maxAz=max(az);
 minEl=min(el);
 maxEl=max(el);

 %compose output matrix
 dataOut=[time,x,y,z,az,el,r];


 %save output data
 filename = 'Flightpath.xlsx';
```

```
xlswrite(filename,dataOut)


 figure, plot3(x,y,z), grid on, hold on

 plot3(p1(1),p1(2),p1(3),'*','MarkerSize',10),
 plot3(p2(1),p2(2),p2(3),'*','MarkerSize',10),
 plot3(p3(1),p3(2),p3(3),'*','MarkerSize',10),
 plot3(p4(1),p4(2),p4(3),'*','MarkerSize',10),
 plot3(p5(1),p5(2),p5(3),'*','MarkerSize',10),
 plot3(p6(1),p6(2),p6(3),'*','MarkerSize',10),
 plot3(p7(1),p7(2),p7(3),'*','MarkerSize',10),
 plot3(p8(1),p8(2),p8(3),'*','MarkerSize',10),
 plot3(p9(1),p9(2),p9(3),'*','MarkerSize',10),
```

8.3.2 Matlab code for transmitting coordinates via UPD protocol

```
rep=0;
 u = udp('192.168.1.65', 1337, 'LocalPort', 5666);
 figure, plot3(x,y,z), grid on, hold on
while 1


  rep=rep+50;
  if rep>= size(az)-1;
    rep=1;
  end


  AzOut=az(rep);
  AzOut=num2str(AzOut,4);
  ElOut=el(rep);
  ElOut=num2str(ElOut,4);
  fopen(u);
  colon=':';

  fprintf(u, strcat(AzOut,colon,ElOut));
  %fprintf(u, AzOut);
  fclose(u);
  xo=x(rep);
  yo=y(rep);
  zo=z(rep);
  %plot3(xo,yo,zo), grid on, hold on
  plot3(xo,yo,zo,'*','MarkerSize',5);
  %plot3m(x,y,z,'r.-', 'MarkerSize', 20);
  view(3);
  pause(1);

end
```

8.3.3 Master controller code

```
#include <EtherCard.h>
#include <IPAddress.h>
#include <Wire.h>
#include <EasyTransferI2C.h>

//create object
EasyTransferI2C ET;

#define STATIC 0  // set to 1 to disable DHCP (adjust myip/gwip values below)
```

```
#if STATIC
// Ethernet interface ip address
static byte myip[] = { 192,168,1,65 };
// gateway ip address
static byte gwip[] = { 10,0,0,138  };
#endif

// Ethernet mac address - must be unique on your network
static byte mymac[] = { 0x70,0x69,0x69,0x2D,0x30,0x31 };

byte Ethernet::buffer[500]; // tcp/ip send and receive buffer

//DEFINIGN GLOBAL VARIABLES
int absX=0;
int absY=0;

float targetX=0;
float targetY=0;

//DEFINIGN I2C DATA CONTENTS
struct SEND_DATA_STRUCTURE{
  //put your variable definitions here for the data you want to send
  //THIS MUST BE EXACTLY THE SAME ON THE OTHER ARDUINO
  int absoluteX;
  int absoluteY;
};

//give a name to the group of data
SEND_DATA_STRUCTURE mydata;

//define slave i2c address
#define I2C_SLAVE_ADDRESS 9


//callback that activates on the event of reciving the UPD data
void udpSerialPrint(uint16_t dest_port, uint8_t src_ip[4], uint16_t src_port, const char *data, uint16_t len){
  IPAddress src(src_ip[0],src_ip[1],src_ip[2],src_ip[3]);

  //----------FOR DEBUGING INCOMING UDP
 Serial.println("DataIn");
  Serial.println(data);

  //Data to temp string
  String tempstring=data;

 // look for colon
// parse string to X and Y values
  String tempX=tempstring.substring(0,tempstring.indexOf(':'));
  String tempY=tempstring.substring(tempstring.indexOf(':')+1,tempstring.length()+1);

 //For deguging
  Serial.println("string values");
    Serial.println(tempstring);
    Serial.println(tempX);
    Serial.println(tempY);

//temp string to float

 targetY = atof(tempY.c_str());
 targetX = atof(tempX.c_str());

 //Gearbox and step angle correction
    absX=4*targetX/0.03468544;
    absY=4*targetY/0.03468544;

  mydata.absoluteX = absX;
  mydata.absoluteY = absY;

 //For deguging
 Serial.println("target values");
    Serial.println(targetX);
    Serial.println(targetY);
    Serial.println("Absolute values");
    Serial.println(absX);
    Serial.println(absY);
    Serial.println("I2C values");
    Serial.println(mydata.absoluteX);
    Serial.println(mydata.absoluteY);

  //send the data via I2C
  ET.sendData(I2C_SLAVE_ADDRESS);
}


//BEGIN SETP
void setup(){
  Serial.begin(9600);

  Wire.begin();
  //start the library, pass in the data details and the name of the serial port. Can be Serial, Serial1, Serial2, etc.
  ET.begin(details(mydata), &Wire);

  // Wire.begin(); // join i2c bus (address optional for master)
  Serial.println(F("\n[backSoon]"));

  if (ether.begin(sizeof Ethernet::buffer, mymac,53) == 0){
    Serial.println(F("Failed to access Ethernet controller"));
    delay(5000);
     Serial.println("Resetting....");
     delay(1000);
    software_Reset() ;}
#if STATIC
  ether.staticSetup(myip, gwip);
#else
  if (!ether.dhcpSetup()){
    Serial.println(F("DHCP failed"));
    delay(5000);
    Serial.println("Resetting....");
    delay(1000);
    software_Reset() ;
  }
#endif

  ether.printIp("IP:  ", ether.myip);
  ether.printIp("GW:  ", ether.gwip);
  ether.printIp("DNS: ", ether.dnsip);

  //register udpSerialPrint() to port 1337
  ether.udpServerListenOnPort(&udpSerialPrint, 1337);

  //register udpSerialPrint() to port 42.
  ether.udpServerListenOnPort(&udpSerialPrint, 42);
}


void loop(){
  //this must be called for ethercard functions to work.
  ether.packetLoop(ether.packetReceive());
}


void software_Reset() // Restarts program from beginning but does not reset the peripherals and registers
{
```

```
asm volatile ("  jmp 0");
}
```

8.3.4 Slave controller code

```
#include <Wire.h>
#include <EasyTransferI2C.h>
#include <AccelStepper.h>

AccelStepper stepper1(1, 4, 5); // Defaults to AccelStepper::FULL4WIRE (4 pins) on 2, 3, 4, 5
const int enable1 = 11;

AccelStepper stepper2(1, 2, 3); // Defaults to AccelStepper::FULL4WIRE (4 pins) on 2, 3, 4, 5
const int enable2 = 12;


// DEFINING GLOBAL VARIABLES
long posX = 0;
long posY = 0;

const int sens1 = 8;
const int sens2 = 9;
int s1 = 0;
int s2 = 0;
int s3 = 0;
int L1 = 0;
int L2 = 0;
int L3 = 0;
int Ycalib = 0;
int Xbalib = 0;
int Yz = 0;
int Ymax = 0;
int Ymin = 0;
int Xz = 0;

long lastDebounceTime = 0;  // the last time the output pin was toggled
long debounceDelay = 50;    // the debounce time; increase if the output flickers

//create I2C object
EasyTransferI2C ET;

struct RECEIVE_DATA_STRUCTURE {
  //put your variable definitions here for the data you want to receive
  //THIS MUST BE EXACTLY THE SAME ON THE OTHER ARDUINO
  int absoluteX;
  int absoluteY;
};

//give a name to the group of data
RECEIVE_DATA_STRUCTURE mydata;

//define slave i2c address
#define I2C_SLAVE_ADDRESS 9


//------------------------READ SENSORS
void readSensors() {

  L1 = s1;
  L2 = s2;

  int reading1 = digitalRead(sens1);
  int reading2 = digitalRead(sens2);
  if (reading1 != s1) {
    s1 = reading1;
  }

  if (reading2 != s2) {
    s2 = reading2;
  }

}
//--------------------------------START SETUP
void setup() {
  pinMode(enable1, OUTPUT);
  // Change these to suit your stepper if you want
  stepper1.setMaxSpeed(4000);
  stepper1.setAcceleration(2000);
  stepper1.moveTo(posX);

  pinMode(enable2, OUTPUT);
  stepper2.setMaxSpeed(2000);
  stepper2.setAcceleration(2000);
  stepper2.moveTo(posY);


  digitalWrite(enable1, LOW);
  digitalWrite(enable2, LOW);

  Serial.begin(9600);
  Wire.begin(I2C_SLAVE_ADDRESS);

  //start the library, pass in the data details and the name of the serial port. Can be Serial, Serial1, Serial2, etc.
  ET.begin(details(mydata), &Wire);
  //define handler function on receiving data
  Wire.onReceive(receive);


  pinMode(sens1, INPUT);
  pinMode(sens2, INPUT);

  //------------------------------------------START Y ZERO CALIB

  readSensors();
  do {
    readSensors();

    /* //FOR DEBUGING SENSOR VALUES
      Serial.println("Sensor Values");
      Serial.print(s1);
      Serial.print(":");
      Serial.println(s2);
      Serial.println("Sensor last states");
      Serial.print(L1);
      Serial.print(":");
      Serial.println(L2);
      Serial.println(" ");
      Serial.println(posY);*/

    if (s1 == 0 && s2 == 0) {
      if ((L1 == 0 && L2 == 0) || (L1 == 1 && L2 == 0)) {
        posY = posY + 1;
        //Serial.println("Case1.1 ");
      }



      if (L1 == 0 && L2 == 1) {
        stepper2.setCurrentPosition(0);
        posY = 0;
        Yz = 1;
        //  Serial.println("Case1.3 EXIT CASE ");
      }

    }

    if (s1 == 1 && s2 == 0) {
      if ((L1 == 0 && L2 == 0) || (L1 == 1 && L2 == 0)) {
        posY = posY + 1;
        //   Serial.println("Case2 ");
      }

    }

    if (s1 == 1 && s2 == 1) {
      posY = posY - 1;
      //  Serial.println("Case3 ");
    }


    if (s1 == 0 && s2 == 1) {
      if (L1 == 0 && L2 == 0) {
        stepper2.setCurrentPosition(0);
        Yz = 1;
        posY = 0;

        //  Serial.println("Case4.1 EXIT CASE ");
      }

      else {
        posY = posY - 1;
        // Serial.println("Case4 ");
      }

    }


    stepper2.moveTo(-posY);
    stepper2.run();

    //delay(1000);


  } while (Yz == 0);
  Serial.println("Y Zero calibration complete!");

  //----------------------------------------END Y ZERO CALIBRATION

  //----------------------------------------START Y MIN CALIB

  do {
    readSensors();
    posY = posY - 1;
    stepper2.moveTo(-posY);
    stepper2.runSpeedToPosition();

  } while (s1 == 0);
  Ymin = stepper2.currentPosition();
  posY = 0;
  Serial.print("Y min=");
  Serial.println(Ymin);
  Serial.print("Y min calibration complete!");

  //----------------------------------------END Y  MIN CALIBRATION

  //----------------------------------------START Y MAX CALIB
  int i = 0;
```

```
  do {
    readSensors();
    posY = posY + 1;
    stepper2.moveTo(-posY);
    stepper2.runSpeedToPosition();
    i = i + 1;
    if (i == 100) {
      Serial.println(posY);
      i = 0;
    }


  } while (!((s1 == 1 && s2 == 1) && (L1 == 0 && L2 == 1)));
  Ymax = stepper2.currentPosition();
  posY = 0;
  Serial.print("Y max=");
  Serial.println(Ymax);
  Serial.print("Y max calibration complete!");
  //----------------------------------------END Y  MAX CALIBRATION
  //----------------------------------------START X ZERO CALIB
    Serial.print("Calibratting X ZERO");
    do{
       readSensors();
    if(s3==0){

    if(L3==1){
    stepper1.setCurrentPosition(0);
        Xz =1;
        posX=0;
    }
    else {
      posX=posX+1;
    }

    }

    else{
      posX=posX-1;
      }

        stepper1.moveTo(-posX);
         stepper1.run();

      }while(Xz==0);
         Serial.print("X zero=");
    Serial.println(posX);
    Serial.print("X zero calibration complete!");
  //----------------------------------------END X  ZERO CALIBRATION

}
//--------------------------------------------------END SETUP


//----------------------------MAIN LOOP
void loop() {
  //check and see if a data packet has come in.
  if (ET.receiveData()) {
    posX = mydata.absoluteX;
    posY = mydata.absoluteY;
    //Serial.println(posX);
    //Serial.println(posY);

  }

  if (-posY >= Ymax) {
    posY = Ymax;
  }
  if (-posY <= Ymin) {
    posY = Ymin;
  }
  stepper1.moveTo(posX);
  stepper2.moveTo(-posY);
  stepper1.run();
  stepper2.run();
}
void receive(int numBytes) {}
```