CHALMERS

Bygge av autonom solcellsdriven båt

Design- & konstruktionsprocess samt kartläggning av framtida utmaningar

Examensarbete inom högskoleingenjörsprogrammet Mekatronik

NIELS BOARDMAN JONSSON
JOSEF VERNERSSON

EXAMENSARBETE INOM MEKATRONIK 2019

Bygge av autonom solcellsdriven båt

Design- & konstruktionsprocess samt kartläggning av framtida utmaningar

NIELS BOARDMAN JONSSON

JOSEF VERNERSSON

CHALMERS

Institutionen för Elektroteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige 2019

Bygge av autonom solcellsdriven båt
Design- & konstruktionsprocess samt kartläggning av framtida utmaningar
NIELS BOARDMAN JONSSON
JOSEF VERNERSSON

Handledare: Pierre Ekvall, Fredrik Lundgren, Anneli Pettersson, Kasper Westman, Olle Norelius, Nils Gangby, Infotiv AB

Examinator: Bertil Thomas, Institutionen för elektroteknik

Institutionen för Elektroteknik
Chalmers tekniska högskola
SE-412 96 Göteborg
Sverige
Telefon: +46 (0)31-772 1000

Förstasida: Bild tagen under test 3, Göteborg 14-05-20

Sammanfattning

"Vi vet mer om månens yta och om Mars än vad vi gör om [jordens] havsbottnen, trots att vi ännu inte har extraherat ett gram mat, ett andetag av syre eller en droppe vatten från dessa himlakroppar."
- Paul Snelgrove, marinbiolog

Långvarig datainsamling på öppet vatten är en svår utmaning än idag. Härifrån kommer idéen om en ny typ av ”smart mikrobåt”. En mindre båt med förmågan att under långvariga perioder befinna sig på öppet vatten och självständigt samla in data. Denna rapport beskriver ett ex-jobb där en prototyp utvecklades för att undersöka möjligheter och utmaningar med denna typ av mikrobåt.

Större delen av utvecklingsarbetet genomfördes på Infotiv AB i Göteborg. Infotiv AB finansierade projektet och agerade intressenter samt projektbeställare. Testerna som utfördes på vatten skedde mellan mars 2019 och juni 2019 i Göteborgs hamnar och skärgård.

En komplett prototyp med navigering, solpanel och energihantering samt datainsamling utvecklades och konstruerades under projektets gång. Olika byggblock köptes in, såsom skrov och elektronikkretsar, för att sedan byggas ihop till den färdiga prototypen. Data från elsystemstest visade på att under soliga dagar gav solcellerna tillräckligt med energi för att driva båtens alla system och samtidigt ladda upp batteriet, resultatet är viktigt för långvarig drift. Detta då nettoenergin över dagen måste vara stor nog för att ladda batterier inför natten och mulna dagar.

Under testning av navigationssystemet gavs, trots elektronikhårdvara på under 1000kr, en mycket hög precision. Detta inkluderar geografisk position, båtens orientering, riktning och hastighet. För vidareutveckling samt masstillverkning är detta resultat avgörande för att hålla nere kostnaderna. Projektet resulterade i en fullt fungerande prototyp vilket påvisar att denna nya typ av ”smart mikrobåt” har kapaciteten att under längre perioder samla in data på öppet vatten. Dock krävs vidare testning och ett nytt skrov för att optimera solcellsytan samt för att klara tuffare väder.

Projektets avgränsningar innefattade en budget på 20 000 kr varav 15 000 användes. Inget skrov designades, inga tester på över 10h samt inga testr under extrema förhållanden genomfördes, detta var på grund av tidsbrist.

Keywords: Automation, smartbåt, solceller, långtidsexpeditioner, smart, mikrobåt

Abstract

"We know more about the surface of the Moon and about Mars than we do about [the deep-sea floor], despite the fact that we have yet to extract a gram of food, a breath of oxygen or a drop of water from those bodies." - Paul Snelgrove, marine biologist

Today, long-term data collection on open waters is a difficult challenge. From that challenge comes the idea for a new type of "smart micro-boat". A small boat with the ability to operate on open waters for longer periods of time while independently collecting data. This report describes a bachelors thesis project where a prototype was developed to investigate opportunities and challenges with this type of smart micro-boat.

Most of the developmental work was carried out at Infotiv AB in Gothenburg. Infotiv AB financed the project and acted as a stakeholder as well as a client. The tests carried out on open water took place between March - June 2019 in Gothenburg's ports and archipelago.

A complete prototype with navigation, solar panels, energy management and data collection were developed and constructed during the project. Various building blocks were purchased, including electronic circuits and a hull, then engineered into the now finished prototype. Data from the electrical system tests showed that during sunny days the solar cells gave enough energy to power the boat's entire system and at the same time charge the battery. These results are important for long-term operations, as the net power throughout the day must be enough to charge the batteries for the night as well as for cloudy days.

During testing of the navigation system, despite electronics hardware for less than 1000 SEK, it provided a very high precision. These include geographical position, boat orientation, direction and speed. For further development and mass production, these results are crucial for keeping costs down. The project resulted in a fully functional prototype which showed that this new type of "smart micro-boat" has the capacity to collect data on open water for a longer period of time. However, further testing and a new hull are required to optimize the solar cell surface area and to cope with harsher weather.

The project's limitations included a budget of SEK 20 000, of which SEK 15 000 was used. Due to time constraints, the authors did not design the hull nor perform tests over 10h or under extreme conditions

Keywords: Automation, smart boat, solar power, long-term expeditions, micro boat

Förord

Denna rapport är ett examensarbete på Chalmers tekniska högskola hos institutionen för elektroteknik. Arbetet utfördes på företaget Infotiv AB under avdelningen Embeded Systems i Göteborg. Examensarbetet omfattar 15 högskolepoäng och utfördes på 20 veckor. Projektet innefattar alla huvuddelar av mekatronik, där med programmering, elektronik samt maskinteknik men även fysik och projektplanering.

Vi vill först tacka Infotiv AB, primärt Pierre Ekvall och Fredrik Lundgren, för att ni trodde på oss och vår idé. Inte minst för ert engagemang och hjälp under hela processen. Några andra vi vill tacka på Infotiv är Anneli Pettersson, Kasper Westman, Olle Norelius och Nils Gangby samt många mer på IES-avdelningen.

Niels vill speciellt tacka sin fästmö Embla Stenström som stöttat honom under hela utbildningen, i vått och torrt. För att hon tolererat de långa dagarna, sena kvällarna och förstår hans passion för teknik. Han vill även tacka sin mamma Cheryl Boardman och sin pappa Tomas Jonsson som varit stora inspirationskällor. Till sist vill han även tacka Josef för det exemplariska samarbetet, utan honom hade detta projekt aldrig kunnat ros i land.

Josef vill tacka sin familj och släkt för att de har lånat ut honom under tre år till Göteborg. Han vill även passa på att tacka alla nya och även några gamla vänskaper under tre riktigt bra år i Göteborg. Sist men inte minst så vill han tacka Niels för många lyckade samarbeten under tre år där detta exjobb med rapport varit pricken över i:et .

Niels Boardman Jonsson och Josef Vernersson, Göteborg 2019

Terminologi

- Arduino - Ett mikrokontrollerkort samt en elektronikplattform baserad på lättanvänd hårdvara och mjukvara.
- PLA - Vanligt förekommande typ av plast vilket används bland annat för 3D-utskrifter
- CAD - Computer-aided design
- SOC - ”State of charge”, hur mycket potentiell energi det finns kvar i ett batteri
- Hysteres – I samband med batterier är hysteres spänningsskillnaden vilket uppstår mellan uppladdning och urladdning oberoende av dess SOC
- Pitch - Vinkel på propellerblad
- Li-ion - Förkortning för batteri av typen Litium jon
- C - En konstant som beskriver laddning- och urladdning hastigheter vilket är ett mått på den hastighet med vilken ett batteri laddas eller urladdas i förhållande till dess kapacitet.
- s - Förkortning för antalet battericeller i serie, exempelvis 4 st. i serie kan skrivas 4s.
- p - Förkortning för antalet battericeller parallellt exempel 4 st. parallellt kan skrivas 4p.
- Cell - En elektrokemisk cell är en anordning som kan generera elektrisk energi från kemiska reaktioner, flera celler bygger upp ett komplett batteri
- Kattegatt - Havsområde mellan Västsverige och nordöstra Danmark
- PCB - Mönsterkort, även känt som ett kretskort utan pålödd hårdvara
- RC - Radiokommunikation, används för att sända kommandon via radiovågor i luften
- Termisk rusning - När temperaturen stiger till den grad att batterier skadas permanent.
- ESC - Electric speed controller, hastighetregulator för elmotorer.
- CPPM - Kommunikationsprotokoll som ofta används i samband med radiostyrda produkter.
- MPPT - Maximum power point tracking, en modul som används för att maximera energin producerad av solpaneler

Innehåll

1 Introduktion
I detta kapitel presenteras projektets bakgrund, syfte, avgränsningar, begränsningar samt vilka mål och frågeställningar som rapporten behandlar.

1.1 Bakgrund
Smarta marinfordon är ett högaktuellt ämne där automation och elektrifiering ofta står i centrum. Detta då det finns potential att kunna ge effektivare och därmed billigare övervakning- och transport-lösningar. Utvecklingen har påskyndats de senaste åren då battericeller har blivit både billigare och effektivare, något Logan Goldie-Scot beskriver i sin artikel [1]. Parallellt med detta har intresset för autonoma drönare och dess tillämpning inom transport och rekognoscering ökat, något som Magnus Vasells artikel påvisar [2]. En av de största nackdelarna med drönare är den korta batteritiden och därmed arbetsintervallet. Detta leder till att i dagsläget är flygningar över längre avstånd eller över öppna hav svårt.

Härifrån kommer idéen om att utveckla en autonom soldriven båt som kan utföra förutbestämda uppgifter.

Genom att bygga en prototyp, testa och utvärdera denna hoppas vi kunna utvärdera möjligheterna för autonoma mikrobåtar i framtiden. Att bygga små självkörande båtar skulle också spara på resurser i form av bränsle och mantimmar som annars skulle krävas för att göra samma uppgift. Denna typ av mikrobåt skulle kunna användas för forskningsändamål genom att autonomt samla data till på öppet vatten under en längre tid. Det kan också finnas militära användningsområden såsom långsiktig marinövervakning.

Projektet genomförs primärt hos Infotiv AB Göteborg som agerar som projektbeställare och intressent.

1.2 Syfte
Att undersöka och testa marin automation i kombination med utnyttjandet av solenergi på öppna vatten samt utvärdera konceptet med smarta mikrobåtar. Projektet är också ämnat för att kartlägga framtida risker och utmaningar inom området för att underlätta för framtida besläktade projekt. Detta görs genom att bygga en mindre prototyprigg i form av en mindre båt som är utrustad för att samla in relevant mätdata.

1.3 Avgränsningar
Budget från Infotiv AB på 20 000 kr.
Inget optimalt skrov kommer konstrueras under projektet utan ett färdigt skrov kommer införskaffas. Detta innebär begränsad solcellsarea jämfört med ett mer optimerat skrov med plats för mer solceller och där av mer in-energi. Båten kommer inte heller kunna hantera extrema förhållanden. Därmed kommer inga långvariga tester eller tester under dåliga väderförhållanden att kunna utföras.
Båtens komplexitet i form av optimalt roder, propeller, djupgång och andra marintekniska element kommer begränsas.

1.4 Precisering av mål och frågeställning

Projektets delmål:

- Konstruera en testplattform för att mäta potentiell solenergi och energilagring samt samla in grundläggande navigationsdata lämplig för en båt
- Konstruera en sjöduglig båtprototyp där testplattformen kan appliceras och användas i marina miljöer
- På öppet vatten autonomt korsa en längre distans, såsom Kattegatt, genom att följa en förutbestämd rutt samtidigt som datainsamling sker

Båtens målspecifikation:

- En större RC båt som kan styras av RC-kontroller
- Konstruerad för bra väder
- Konstruerad för hastigheter på 5 km / h - 15 km / h
- Globalt positioneringssystem
- GSM-kommunikation från båt, SMSa GPS koordinater
- Autonom navigation- och styrsystem för att tillförlitligt nå mål och delmål
- En 60 W solpanel för laddning av Li-Ion-batterier
- Batteri och BMS skalad för 24h användning med bra väder

Frågeställningar:

- Kan solenergi effektivt tillämpas på öppet vatten med båtar?
- Har smarta mikrobåtar kapaciteten att tillförlitligt ta sig fram samt samla in relevant data?
- Vad är möjliga närliggande applikationer av en smart mikrobåt?
- Vad är de större problemområdena inför framtida arbete med smarta mikrobåtar?

2 Teknisk bakgrund

I detta kapitel presenteras den tekniska bakgrunden till de begrepp och komponenter som används i detta projekt.

2.1 DC-Motor

DC står för ”direct current” eller på svenska likspänning. Det finns några olika kategoriseringar av DC-motorer men en av de vanligaste uppdelningarna är ”brushless” eller ”brushed”, vilket på svenska betyder med eller utan kol. Kolen ligger emot den roterande delen av motorn och slits långsamt vilket gör att DC-motorer med kol har en bestämd livslängd, medan motorer utan kol inte har någon bestämd livslängd. De flesta motorer som kallas för DC-motorer styrs med antingen PWM eller trefasväxelspänning för att kunna variera varvtalet, men det finns undantag som endast styrs med likspänning.

2.2 Servomotor

Servomotorer känns oftast igen från bilar där de underlättar manövreringen genom att tillföra ett extra moment på styraxeln. Det är också vanligt inom RC hobbyvärlden, där små servomotorer används för att kontrollera styrstången på exempelvis modellflygplan eller modellfartyg. Servomotorer för RC hobbyn har en mycket exakt lägesåterkoppling som gör det möjligt att styra eller rotera väldigt konsekvent när det gäller både hastighet och position.

2.3 Monocrystalline Solar Cells

Solceller finns i många former och varianter. En av de vanligaste och effektivaste i förhållande till pris är den så kallade monokristallina silikonvarianten. De innehåller en enda stor kiselkristall som omvandlar solljuset till elektricitet. Varje cell som är ungefär 100x100 mm stora kan leverera upp till 25 % effektivitet, det är vanligare att effektiviteten är runt 20 %. När en ensam cell utsätts för solljus kan de leverera uppåt 7 A ström och 0,6 V spänning vilket kan resultera i en effekt strax över 3,5 W.

2.4 Li-Ion Batteri

Ett Li-ion-batteri, eller litiumjonbatteri är en typ av laddningsbart batteri. Med sin höga energidensitet används dessa ofta till bärbar elektronik och elfordon, detta då vikt och utrymme ofta är prioriterade. En annan fördel med Li-ion är hur långsamt de självurladdar. Några negativa aspekter är kravet på skyddskretsar samt hur batterierna åldras efter ett visst antal cykler.

2.5 Skyddskrets för Li-Ion batteri

Denna modul skyddar batteriet mot termiska rusningar samt förlänger batteriets livslängd. Under laddning skyddar modulen batteriet genom att övervaka spänningen hos samtliga celler och vid obalans balanseras cellerna. Under urladdning övervakas spänningen hos alla celler samt totalspänning och utströmmen. Om batteriet överskrider någon av de förinställda gränserna såsom att batteriet laddar ur för långsamt eller för snabbt, kommer modulen bryta kretsen och på så sätt skydda batteriet från skador.

2.6 9-Axels rörelsesensor

Med 9 axlar menas en elektrisk krets som kombinerar ett 3-axels gyroskop, 3-axels accelerometer och 3-axels magnetometer. Detta används för att mäta objekts orientering genom att mäta rotation, acceleration samt magnetfält. Den kombinerade informationen kan sedan användas för att exempelvis läsa ut en exakt kompassriktning oavsett läge.

2.7 Datalogger
En datalogger samlar in en mängd data av en eller flera variabler som kan utläsas vid ett senare tillfälle. Ett utmärkt exempel på en datalogger är en så kallad black box i flygplan som samlar data som kan underlätta vid utredningen av ett eventuellt haveri. Datan kan sedan sparas på en mängd olika typer av hårdvara såsom minneskort och hårddiskar.

2.8 Fuktsensor
En fuktsensor är en sensor som mäter någon typ av fuktighet, det kan handla om fuktighet i luften, fuktighet i olika material eller exempelvis fuktighet i jorden för att reglera ett automatiskt bevattningssystem. Detta fungerar genom att konduktiviteten mäts mellan två elektroder. Om elektroderna är nedsänkta i vatten kommer spänningsfallet vara lägre än om det bara var luft emellan.

2.9 Buck converter
”Buck converter” är en typ av spänningsomvandlare som omvandlar en hög spänning ner till en lägre spänning. Motsatsen är en ”boost converter” som omvandlar från låg till högre spänning. Det finns en rad olika sorters ”buck converters” men det de har gemensamt är att de omvandlar från likspänning till likspänning. Spänningen är alltid lägre på utsidan och strömmen alltid högre. Det finns inga spänningsomvandlare som är 100% effektiva även om vissa kommer nära.

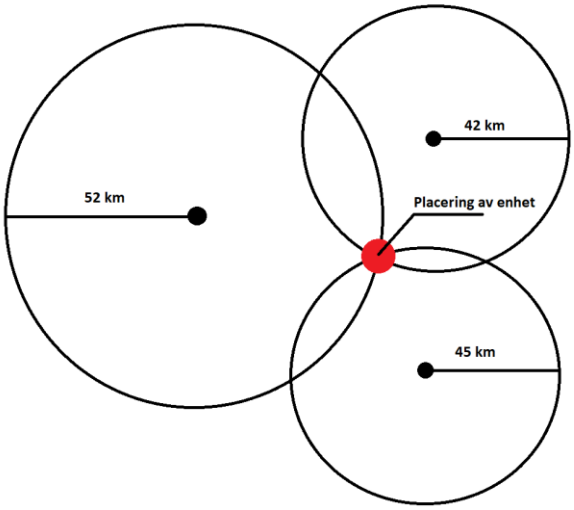
2.10 MPPT
MPPT, eller Maximum power point tracking, är en ”buck converter” som automatiskt anpassar och optimerar spänning och ström från solpanel till batteri samt belastning. Optimeringen sker genom att kretsen belastar solcellerna så pass mycket att spänningen sjunker men att den aldrig blir lägre än att det går att ladda batteriet från solcellerna, på det viset ges maximal ström och spänning.

2.11 Strömmätare
I detta projekt används en linjär Hall-sensorkrets för att mäta strömmen. Magnetfältets styrka är proportionell mot strömmen som går genom ledningen och detta används för att räkna ut strömmen. En annan metod är att använda ett motstånd med en känd liten resistans, ett så kallat shuntmotstånd, där det uppmätta spänningsfallet och ohms lag används för att beräkna strömmen genom motståndet. Att använda ett enkelt shuntmotstånd är ett simplare lösning men den drar mer effekt under användning än Hall-sensorn.

2.12 Motorkontroller
En motorkontroller styr hög spänning och ström från batteriet till motorn med hjälp av PWM styrning. En motorstyrenhet är nödvändig eftersom en mikrokontroller normalt kan ge ungefär 100 mA ampere men de flesta ställdon (likströmsmotorer, likströmsmotorer, servomotorer och mer) kräver mer ström för att fungera.

2.13 GPS

GPS är en positioneringsmetod som med hjälp av atomur och satelliter i omloppsbana kring jorden, skickar tiden till en mottagare nere på jorden. När mottagarenheten har tiden från tre eller fler atomur samtidigt kan denna beräkna sin position i förhållande till atomuren med hjälp av triangulering enligt figur 1.



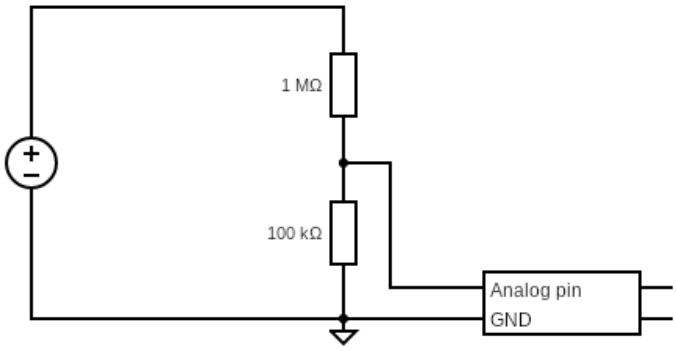
Figur 1, triangulering

2.14 GSM

GSM är ett globalt system för kommunikation via mobilsamtal och sms, det stödjer också mobilnätverk i låga hastigheter. GSM är också känt som 2G, vilket är den mest grunläggande metoden för kommunikation via mobilen idag men också den som har bäst täckning.

2.15 Spänningsdelare

Enkortsdatorer såsom Arduino kan ha flera analoga portar som kan mäta DC spänningar, och i vissa fall AC, mellan 0 och +5 volt. För att öka mätintervallet kan en spänningsdelare användas genom att använda två resistorer, se figur 2. Detta sänker den uppmätta spänningen så att den passar in i intervallet för de analoga ingångarna. Sedan beräknas värdet om i mikrodatorn för att få det korrekta mätvärden enligt formeln: V.batteri = V.analog * (R2/(R1+R2))



Figur 2, spänningsdelare lämpad för 50 V

3 Metod

Tillsammans med Infotiv sattes det upp 4 grindar, gates på engelska, för att få en strukturerad arbetsgång för projektet, se bilaga 1. De fyra grindarna användes för att bryta ner projektets mål i delmål med en kravspecifikation. Följande grindar sattes upp:

- Gate 1, förvärva hårdvara och testa grundläggande funktionalitet
  - Ekonomi
  - Navigation
  - RC båt
  - Solcellsladdning
- Gate 2, testning av block
  - Navigation – öppet hav
  - RC båt - öppet vatten
  - Solcellsladdning - på land
- Gate 3, sjötester
  - Navigation - autonomt
  - Power System - öppet vatten
- Gate 4, sluttest och dokumentation

Delmålen sattes upp för att kunna verifiera varje funktion och block parallellt med varandra, detta för att tidigt kunna upptäcka problem.

Koden som skrevs för projektet byggdes med samma princip, små byggblock på grundläggande funktionsnivå för att sedan appliceras i större kodblock.

Experimenten gjordes i en miljö som var så lik den slutgiltiga användningsmiljön som möjligt, alltså antingen nära vatten eller ute på öppet vatten. Båten testades i en realistisk miljö för att tidigt kunna minimera risk inför senare tester. Data som GPS och kompass jämfördes med smartphones när riktmärken inte fanns. En datalogger implementerades i projektet för att spara all data båten samlade in under testning.

Även en viktad riskanalys gjordes för att i god tid kunna planera för potentiella risker och problemområden.

4 Genomförande

Detta kapitel beskriver hur projektet genomfördes för att uppnå projektets syfte och mål. Arbetet utfördes mellan 7e januari och 2e juni 2019.

4.1 Riskanalys

Tidigt i projektet gjordes en riskanalys, se den kompletta analysen i bilaga 2. Riskerna med största potentiella negativa påverkning för projektet efter viktning blev vattenskador mot kritisk elektronik, såsom batteri och processor. Detta skulle kunna leda till förlorad navigation samt kommunikation vilket skulle resultera i att båten skulle kunna förloras.
För att undvika förlusten av båten evaluerades en reservdator med satellitkommunikation och en separat batterikälla innesluten i en separat kapsel i båten. Denna lösning ströks då priset och tidsåtgången för projektet snabbt skulle öka. Istället övervakas vattenintrång noggrant i projektet och flera lösningar planeras för såsom dubbla tätningslister och en svamp för att suga upp vatten inne i skrovet.

4.2 Lagar och regler

En risk för projektet var vilka lagar och regler som gäller för autonoma båtar, både på nationellt och internationellt vatten. 13 mars, 2019, utfördes en intervju med Olle Lindmark, tekniklektor vid Chalmers inom mekanik och maritima vetenskaper. Han berättar att lagar inom autonoma båtar inte följer med teknikens framfart. Vilket leder till att på internationellt vatten finns det inga lagar som behandlar projektbåten och dess användning. Samma gäller enligt Olle även på svenskt vatten. Det finns dock länder som redan idag hanterat frågan, där bland Norge. Vi fick rådet att kontakta Transportstyrelsen för att få ett ställningstagande om projektet, detta gjordes via e-post och telefon. Enligt Martin Bloom 28 feb, 2019, fartygsinspektör hos Transportstyrelsen i Stockholm, finns inga föreskrifter eller regler som hanterar obemannade båtar på nationellt vatten. Martin Bloom fortsätter med att säga, skulle olyckan vara framme skulle tolkningen av lagen vara subjektiv angående vem som är ansvarig. Så länge någon befinner sig inom synhåll av projektbåten och har möjlighet av att ta kontroll över båten anser Martin att, en så pass liten autonom båt inte tillför några större risker för andra båtar och är därmed godkänt för nationellt vatten.

4.3 Tidsplan

Utifrån metoden och de valda grindarna skrevs en tidsplan. Utöver grindarna planerades även tid för förstudie, efterforskning samt dokumentation, se figur 3 över GANNT schema.

		Jan		Feb				Mars				April					Maj				Juni			
Week		4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
	Estimated time (h)																							
Education	**24**																							
Communication CAN	**8**							x																
Lidar	**8**								x															
Visual	**8**									x														
Research	**60**																							
Hardware acquisition	**20**	x	x																					
Power System	**16**		x	x																				
Navigation	**16**			x	x																			
RC boat	**8**				x																			
Developement	**104**																							
- Build Design	**8**					x																		
- Power system	**16**					x																		
- Framework Foundation	**20**												x											
- Debugging and Optimization	**60**											x							x		x			
GATE: Acquisition and Basic Functionality	**64**										x													
- GPS, GSM, Compass	**24**						x																	
- Motor, servo, flotation	**24**							x																
- Solar cells, MPPT and batteries	**16**								x	x														
GATE: Module Testing	**104**																		x					
- RC Boat	**24**													x										
- Solar Charging	**40**														x	x								
- Stena Test	**40**																x	x	x					
GATE: Lake Test	**56**																				x			
- Navigation	**32**																			x				
- Power system	**24**																			x				
Final Test: Ocean	**32**																						x	
- Complete system test	**24**																					x		
- Final analysis	**8**																						x	
Documentation	**70**																							
Product description	**4**		x																					
Finalize Time Plan	**2**		x																					
Education	**20**																						x	x
Documentation (2h/week)	**44**	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Figur 3, GANNT schema.

4.4 Elsystem

Arduino väljs som utvecklingsplattform då marknaden för Arduino-kompatibel hårdvara är bred samt att många kodexempel finns tillgängliga. Utifrån kravspecifikationen, riskanalysen och med budgeten i beaktning valdes alla komponenter ut.

4.4.1 Batteripack

Det första som valdes när batteripacket skulle designas var arbetsspänningen, en lämplig spänning att använda är 12 V. Detta eftersom att en stor del av de elektriska kretsar som går att köpa på nätet är byggda för ett 12 V system, specifikt motorer till mindre båtar. Genom att undvika en omvandling av spänningen upp eller neråt så undviker man också mycket förluster som alltid är en följd av omvandlingar. På grund av sin väldigt höga energidensitet valdes Li-ion batterier. Det förhållandevis tunga men lilla batteriet göra att tyngdpunkten kommer ner lågt i båten och gör båten mer stabil. Li-ion batterier är vanligt i många moderna konstruktioner vilket gör att det är lätt att ta fram information om dem. I vårt projekt så underlättade den enklare informationssökningen att hitta färdiga laddningskretsar som balanserar och skyddar batteriet mot höga strömmar samt höga och låga spänningar.

12V i samband med Li-ion batterier brukar i själva verket vara en spänning som varierar mellan 12,6V när batterierna är fulladdade och ner till 9V när de är helt urladdade. Eftersom att varje batteri har en nominell spänning på 3,7V så krävs det att man seriekopplar tre batterier för att komma upp i 12V, att seriekoppla tre batterier kan också kallas 3s.

För att öka kapaciteten och räckvidden på båten så kopplas flera batterier parallellt. För att beräkna lämplig kapacitet så är det inte enbart hur mycket energi batteriet ska innehålla utan också hur mycket ström man ska kunna ta ut och ta emot utan att påfresta cellerna. Urladdnings och laddningsström mäts i C, C är ett mått på strömmen i förhållande till batteriets kapacitet. Exempelvis har ett batteri med kapaciteten 5 Ah och med C konstanten 0,5 en laddningsström på 2,5 A, detta räknas ut med kapaciteten multiplicerat med C konstanten. Solcellerna kan enligt leverantören, se bilaga 3, leverera

max 6 A under de bästa förhållanden och de batterier som valts ut har en kapacitet på 3400 mAh och kan laddas med maximalt 0,5 C. Det betyder att laddströmmen inte får gå över 1,7 A, enligt 3,4*0,5 = 1,7 A, för att inte riskera att skada cellerna. Alltså måste batterierna parallellkopplas för att laddströmmen inte ska bli för hög för varje individuell cell. Laddströmmen delat med vad en cell kan laddas med ger hur många celler som krävs parallellt 6/1,7 = 3,5, alltså behöver parallellkopplingen bestå av minst 4st batterier för att inte riskera att skada eller förstöra cellerna.

Samma sak gäller också när det plockas ut ström från batteriet, utifrån tidigare beräkningar och bilaga 4 kommer utströmmen att vara max 6.5 A och att en cell kan laddas ur med maximalt 2 C kontinuerligt. Detta gör att vår cell på 3400 mAh kan laddas ur med 2*3,4 = 6,8 A, alltså skulle det räcka med ett batteri för att tillgodose vårt system med ström från batterierna under en kortare period.

En annan parameter vid val av batteri är spänningsfallet vid belastning, ett stort spänningsfall gör det svårt att beräkna hur mycket kapacitet batterierna har kvar då metoden för att mäta kapaciteten bygger på att mäta batterispänningen. Ett spänningsfall gör det också svårare för batteriladdaren och BMS:en att styra hur batterierna ska laddas och balanseras vilket kan leda till ett snabbare åldrande av cellerna. Spänningsfall kan beräknas enligt $U=R_{in}*I$ där R_{in} är batteriets inre resistans.

Batterierna valdes också ut utan en intern säkerhetskrets för att förebygga att en cell som sitter parallellt med en eller flera andra celler inte ska kopplas bort i händelse av överhettning eller kortslutning. Detta kan leda till att när batteriet automatiskt kopplas tillbaka igen så kan stora mängder ström flöda mellan cellerna för att balansera upp potentialskillnaden som uppstått. Istället så förlitar sig systemet på en skyddskrets som bryter hela systemet vid fel som exempelvis hög temperatur eller kortslutning.

Med tidigare beräkningar så krävs det alltså ett batteripaket med tre batterier i serie och minst fyra batterier parallellt också kallat 3s4p. För att nå ett mål som att korsa Kattegatt, 100 km bort, med vår lägsta hastighet, 5 km/h så kommer det att ta ca 20 timmar. Tidiga beräkningar visade på att båten drar 3,5A i snitt, under 20 timmar kommer båten med andra ord att förbruka ca 70 Ah. Solen kan tillföra upp till 40 Ah en solig dag enlig Global Solar Atlas [3] med den solpanel som båten har. Detta betyder att batterierna måste ha en kapacitet på mist 30Ah. Detta ger 30/3,4 = 8,8, alltså krävs 8 celler parallellt för att tillföra den extra effekten som krävs för att färdas 100 km. Men för att verkligen vara på den säkra sidan och att inte riskera att förlora kontakt med båten eller att påfresta batterierna allt för mycket så dubblas kapaciteten till totalt 12 celler parallellt och tre celler i serie till alltså 36 celler totalt.

4.4.2 Solpanel

På båten används 27 st. monocrystallina solceller, dessa är kopplade i serie för att bygga på spänningen då spänningen över varje solcell bara är 0,6 V. Vi valde att använda solceller från SunPower av typen C60 Bin H, det finns ett stort sortiment med solceller på marknaden men dessa valdes på grund av dess höga effektivitet, de är relativt enkla att montera och de har ett bra pris. Varje cell levererar mitt på en solig sommardag 3.5 W enligt datablad, den maximala effekten som skulle gå att plocka ut blir med andra ord ca 95 W.

Solcellerna måste hanteras med försiktighet då de är väldigt känsliga för smuts och fett, därför används plasthandskar innan de är monterade se figur 4. I monteringen så monteras de i båten med klar epoxi vilket både skyddar solcellerna och håller dem på plats.
solcellerna levererar 0,575 V och 5,85 A, den totala spänningen med 27 st solceller i serie kommer alltså att komma upp i maximalt 15,5 V. Det är viktigt att använda kablar som klarar av 5,85 A kontinuerligt vid montering av solcellerna.
Det är svårt att montera solcellerna på ytor som inte är helt platta, därför köptes det in en plastskiva som gör monteringen smidig och ger en viss styvhet till solcellerna. Att montera solcellerna på en plastskiva underlättar också att komma åt batterier och elektronik som sitter på insidan skrovet.

Figur 4, solceller monteras med plasthandskar

4.4.3 Datalogger
I hårdvaran finns det installerat en datalogger. Det underlättar utvecklingen samt möjligör analys av kördatan samt soldatan. En gång i sekunden sparas mätvärden såsom GPS position, tid, hastighet, riktning, spänning, strömmar och SOC.

4.4.4 Fuktmätare
I båten finns två fuktsensorer som i ett tidigt skede känner om det kommer in vatten, en kommer placeras långt fram längst ner i botten och en längst bak. Om någon av dessa skulle komma i kontakt med vatten så kommer båten gå in i ett säkerhetsläge och genast skicka ut ett varningsmeddelande till användaren.

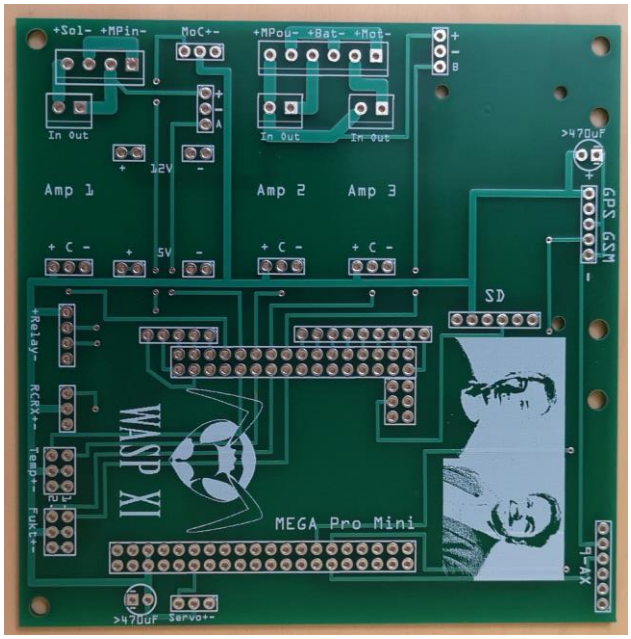
4.4.5 Spänningsmätare
Spänningen kommer att mätas på två punkter, spänningen som kommer in från solcellerna och spänningen över batterierna. Spänningen som mäts över batterierna kommer att användas för att beräkna batteriernas SOC. Eftersom att det är förhållandevis små strömmar så kommer hysteresen bli så pass liten att spänningen är nästan linjär emot hur mycket kapacitet som finns kvar i batteriet.

Spänningsmätaren är tillverkad av en vanlig spänningsdelare med två resistorer som sitter i serie där man läser ut ett analogt värde mellan de två resistorerna.

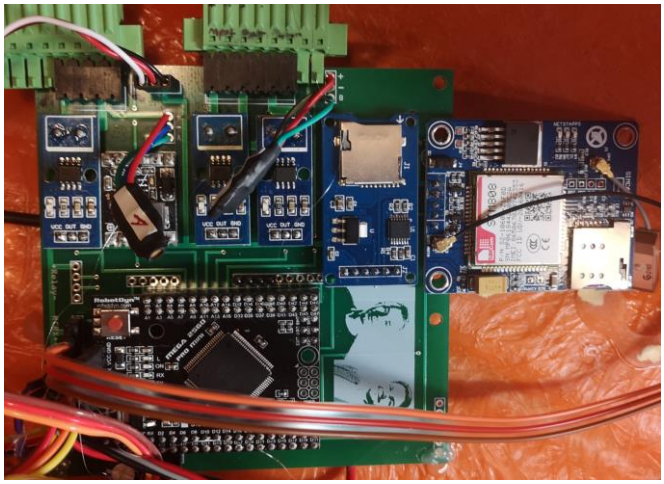
4.4.6 Strömmätare
Tre strömsensorer av modellen ASC712 som klarar att mäta upp till 20A finns i båten, de mäter även vilken riktning strömmen har. De tre sensorerna sitter placerade på följande sätt: en sitter mellan solceller och MPPT för att mäta den totala strömmen in. En sitter vid batteriet för att mäta hur mycket ström som går in eller ut ur batteriet, den sista sitter vid motorn för att mäta hur mycket ström som förbrukas av motorn. Övrig strömförbrukning mäts genom att ta skillnaden på hur mycket som kommer in, går till eller från batterierna och ut till motorn.

4.4.7 PCB

För att minimera kabeltrassel samt optimera översikten av elsystemet beslutades det att designa en egen PCB. För att underlätta utvecklingen så monterades en arduino mega pro på upphöjda monteringslister. Några kopplingsplintar används för att koppla solceller, MPPT, batteri och motor. Kondensatorer användes för att ge stabilitet vid strömspikar. Några extra plintar drogs ut från mikrodatorn för eventuella framtida behov. På figur 5 och 6 kan man se hur kretskortet ser ut innan modulerna är fastsatta och efter att de är monterade. Se bilaga 5 för komplett komplingschema.



Figur 5, kretskort utan moduler, logga designad av Embla Stenström



Figur 6, kretskort med moduler

4.5 Skrov

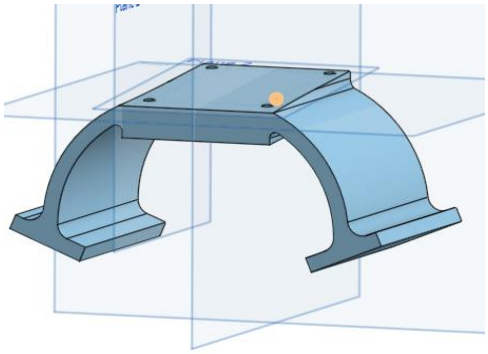
När skrovet valdes fanns det ett antal kriterier för att få en bra grund att bygga på, dessa kriterier var:

- Skrovet ska vara stort nog för att klara öppet hav under bra väderförhållanden.
- Det ska finnas plats för så många solceller som möjligt på ovansidan.
- I skrovet ska all elektronik få plats, bland annat motor och batteri.
- Det får inte vara för stort, max 2m, då det inte finns möjlighet till ett längre skrov i de lokaler som Infotiv tillgodoser med. Detta håller även nere priset på komponenter då det behövs mer solceller, större motor och större batteri till en större båt.

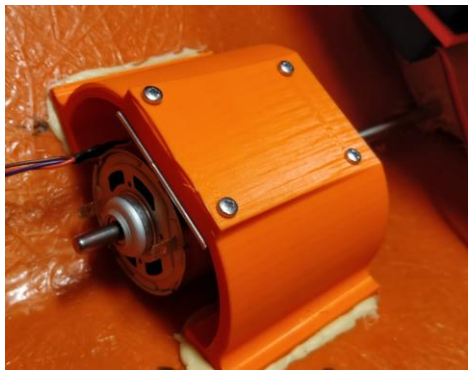
Kategorin på skrov som verkade mest relevanta för projektet var skrov för storskaliga radiostyrda båtar och som oftast är gjorda för att köras i lite oroligare vatten eller på hav. En tillverkare i England som tillverkar storskaliga radiostyrda båtar hade ett skrov som höll specifikationerna. Tillverkaren tillverkade både enkel och dubbelköliga skrov, efter konsultation från tillverkaren så valdes det enkelköliga skrovet på grund av att det skulle hantera vågor bättre samt att det dubbelköliga är byggd för två motorer. Planen med projektet var också att göra en energisnål båt som kunde förlita sig på solceller. Därför valdes också den med en köl och en motor istället för två, vilket skulle kräva mer energiåtgång. Tillverkaren hjälpte oss också att ta fram en toppkåpa som var helt platt för att underlätta montering av solceller senare i projektet. Längden på skrovet är 145 cm och bredden är 38cm. måtten på skrovet är utmärkt för att montera tre solceller på bredden då varje solcell är 12,5cm. Skrovet är tillverkat i glasfiber och är lackad för att lätt flyta genom vattnet.

4.5.1 Motormontering

Motorn levereras med en infästningsplatta av metall, den första tanken var att montera motorn direkt i skrovet med infästningsplattan och epoxilim. Efter att ha utvärderat detta med olika CAD modeller, se bilaga 6, så observerades det att vinkeln på axeln för att hålla propellern under vattnet skulle ha blivit allt för stor. Detta skulle kunna leda till stora energiförluster. För att lösa detta så monterades istället motorn upp och ner så att motorns rundning passar in i båtens naturliga bottenrundning, se figur 7 och 8. Med detta knep kunde lutningen på axeln reduceras till bara 5 grader istället för 15 grader, vilket minskade energiåtgången avsevärt. Se bilaga 9 ritningar till motorfästet och andra egentillverkade komponenter.



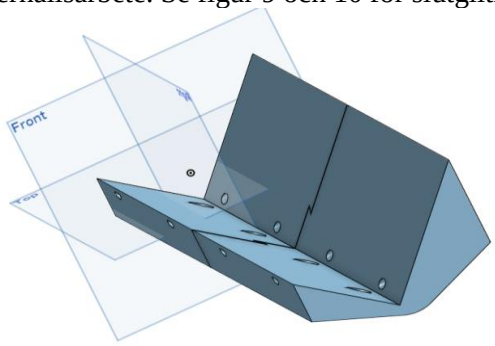
Figur 7, CAD modell av motorfäste



Figur 8, 3D-utskrivet motorfäste

4.5.2 Batterimontering

Batteriet är det absolut tyngsta i båten förutom själva skrovet, därför är det viktigt att det ligger lågt och sitter fast hårt. Skulle batteriet lossna så skulle det kunna riskera hela båten. Det är också en fördel att kunna flytta på batteriet som en ballast för att kunna optimera så att båten får så lite motstånd som möjligt i vattnet samt att propellern sitter på ett lämpligt djup. Med dessa aspekter formades en batterihållare där batteriet står på högkant för att ligga så lågt som möjligt i v-skrovet. Batteriet fästs i hållaren med buntband som både är säkert och hållbart med som lätt går att byta ut om man skulle vilja ta ut batteriet för att exempelvis göra mätningar eller underhållsarbete. Se figur 9 och 10 för slutgiltig design.



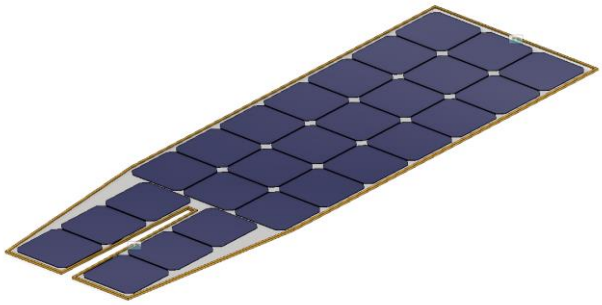
Figur 9, CAD modell batterihållare

Figur 10, 3D-utskriven batterihållare

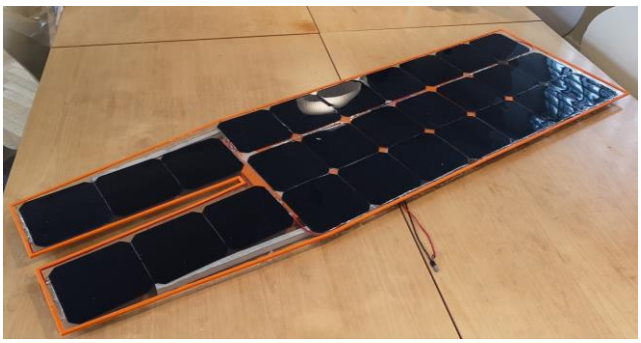
4.5.3 Solcellsmontering

Solcellerna var det svåraste att montera både i hänsyn till tid och komplexitet, därför valdes det att ta det sent och tänka igenom det noggrant innan. Istället för att montera solcellerna direkt på båten som skulle fått vissa komplikationer som exempelvis att luckan in inte skulle gå att öppna så monterades solcellerna på en plexiglasskiva som går att skruva av och på båten se figur 11 och 12 hur solcellerna monterades på en skiva.

Cellerna löddes ihop med de specialtillverkade monteringsdon, såkallade tabbing wire, som är till för denna typ av solcell. När samtliga celler var lödda placerades de ut på plexiglasskivan som var utskuren för att passa båtens form och ett tunt lager av epoxi appliceras ovanpå solcellerna för att skydda dem från smuts, fett och vatten. Några L-profiler i aluminium limmas även fast för att ytterliggare stärka upp panelens styvhet.



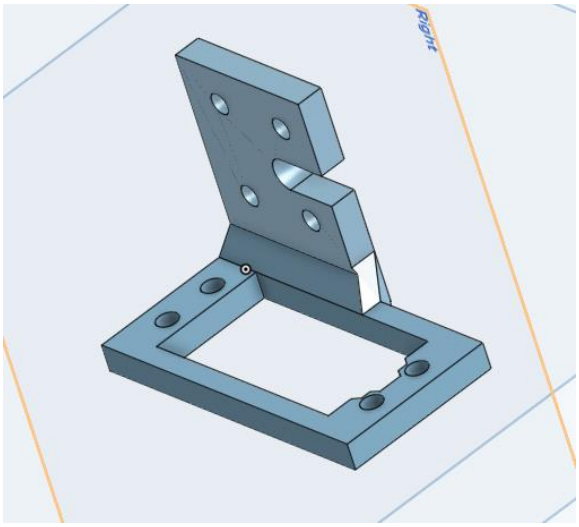
Figur 11, solcellsyta i CAD



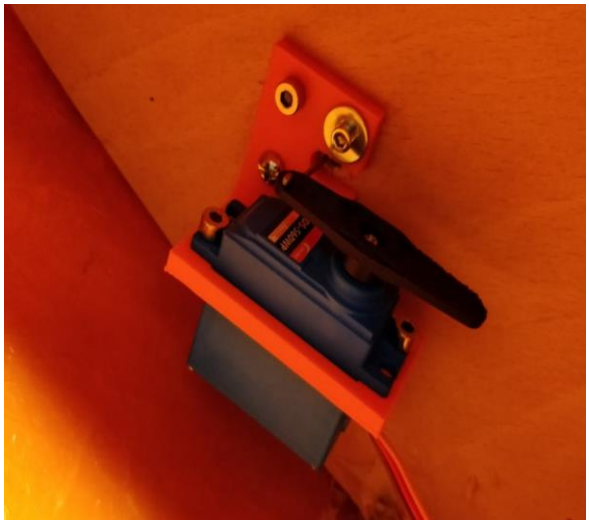
Figur 12, solcells yta färdig

4.5.4 Roder

Rodret är av standardtyp för radiostyrda båtar av större modell, vilket innebär att det är ungefär ett 1,2 dm långt aluminiumroder. Rodret monteras med fyra genomgående skruvar som sedan dras fast på insidan av akterspegeln. Mellan de fyra skruvarna går också ett styrstag som på insidan av båten styrs av en servomotor. Servermotorn är monterad med en 3D utskriven plastbit, se figur 13 och 14, som hålls på plats med samma skruvar som också håller själva rodret på plats. Efter de första testerna med båten i vatten framgick det tydligt att rodret var underdimensionerat vad gäller storlek i förhållande till de låga hastigheter det kommer att användas till. Därför uppgraderades rodret med ett bredare blad som gav utmärkta testresultat.



Figur 13, CAD modell av servo fäste



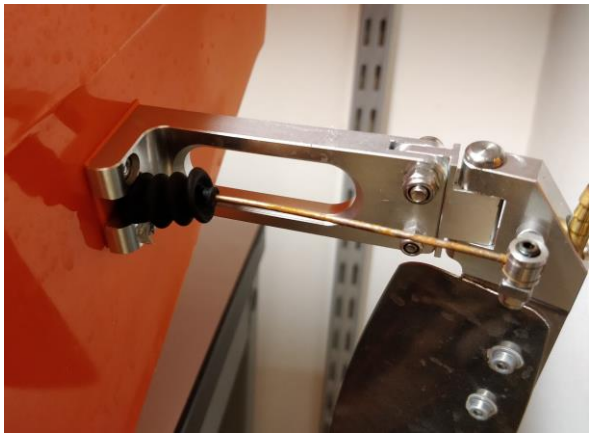
Figur 14, implementerat servo fäste

4.5.5 Tätning

Båten är tätad på ett antal olika ställen och med ett antal olika tekniker. Runt om locket och mellan undersidan av solcellsglaset sitter en tätlist av gummi som tätar från stänk och översköljande vatten. Tätningslisten sitter på ett sådant sätt att när skruvarna dras åt för att stänga locket så sluts också de båda listerna.

Propelleraxeln som består av en invändig axel och ett utvändigt rör är tätad med vattentålig epoxi där det yttre röret går igenom akterspegeln. Den inre axeln är tätad med fett, två kullager och en bricka som förhindrar vatten att tränga in i axelgenomföringen. Alla skruvar som går genom skrovet för att hålla solcellerna på plats är tätade med samma epoxi som axelröret. Styrstaget som går mellan servomotor och roder är tätad med en gummidamask som är väldigt mjuk och följer därför stagets rörelser väl.

För att täta mellan hål och styrstag sitter en damask som är klämd med en annan plastbit som skrivits ut, se figur 15.



Figur 15, infästning roder

4.6 Navigation och övervakning

För att uppnå kravspecifikationen krävdes positionsdata, färdriktning, kommunikation, framdrift och styrning. Detta löstest på följande sätt.

4.6.1 Framdrivning och styrning

Framdrivning sker med en motor av typen ”brushed” eller på svenska med kol, motorn är styrd via en motorkontroller eller så kallad ESC, electric speed controller. Motorkontrollen styr varvtalet på motorn via en PWM-styrning som i sin tur får sina signaler från arduinon. Motorn är lätt överdimensionerad för att på blåsiga dagar eller i kraftig ström ändå kunna ta sig framåt. Motorn får sin ström direkt från batteriet med en spänning på 12V, dock så finns det ingen möjlighet att byta polerna på motorn för att backa. Det köptes in ett par relä för att implementera en back men detta färdigställdes aldrig. Arduinon styr också själva rodret på båten som via ett servo kontrolleras med PWM signaler precis som motorn.

4.6.2 Radiostyrning

För att kunna testa båten och utföra svårare manövrar i hamn så utrustas båten med ett radiosystem. Detta radiosystem räcker inte mer än max ett par kilometer så det är inget som går att använda när man befinner sig långt ifrån båten. Infotiv hade sedan ett tidigare exjobb en radiosändare till en drönare av typen Spectrum DX6 som vi också fick använda i vårt projekt för att spara pengar. Med radiokommunikationen kan det sändas upp till 6 kanaler men bara fyra av dessa används, de är: hastighet, styrning, aktivering och självkörande eller manuellt.

Mottagaren beställdes först som en kopia från Kina som skulle fungera med mottagaren, men det visade sig att sändningsprotokollet, DSM2 som kina-mottagaren jobbade med inte var lagligt i Europa. Därför så beställdes istället en från Europa av typen Orange Rc R615X som sänder över protokollet DCMX istället för DSM2. Mottagaren har inte bara 6st PWM kanaler utan också en kanal med CPPM som gör det möjligt att bara använda en digital port på Arduinon istället för 6st, vilket sparar både pinnar och minskar antalet lösa kablar i båten.

4.6.3 GPS & GSM

Båten är utrustad med både GPS och GSM. GPS:en används för att navigera genom att den ger kurs, hastighet, position, höjd och tid. Med GSM-modulen kan båten skicka sms, ringa samtal och även koppla upp oss på nätet, men i detta projekt kommer bara funktionen som skickar sms användas.

GSM-modulen är den enda kommunikationskanalen med båten när den ej är under uppsikt. Den skickar varje halvtimme ett meddelande med alla intressanta parametrar såsom hastighet, position och batteriladdning. Det är också möjligt att läsa ut meddelanden som skickas till båten för att kunna avbryta uppdrag, skicka nya koordinater eller kalla tillbaka båten om ett problem skulle uppstå. För att inte förbruka för mycket ström kommer meddelanden bara skickas ett par gånger i timmen. Att skicka ett sms kan dra upp till 2A under en kort period.

Både GPS- och GSM-modulen sitter i samma kretskort som heter SIM808, det är ett relativt vanligt kretskort som används för utveckling i ett tidigt skede. Kommunikationen mellan kretskortet och Arduino sker med seriellt protokoll och kommandona för att skicka och ta emot data med SIM808 kallas för AT-kommandon.

4.6.4 Kompass

För att kunna reglera båtens färdriktning var behovet för en kompass uppenbart. En 9-Axels rörelsesensor valdes som en elektronisk kompass. Magnetometern är huvudkomponenten med accelerometern som tillägg för att kunna säkerställa planheten av kompassmätningen. Enheten som användes var av modellen GY91 som består av magnetometer, accelerometer samt gyro. Implementering baserades på Michael J. Caruso artikel "Applications of Magnetoresistive Sensors in Navigation Systems" [4]. Beräkningsmetoden som används för att beräkna kompassens riktning kommer primärt från detta dokument.

I flera tester där mikrodatorns kompass jämfördes med kompassen i en smartphone, i detta fallet en Huawei P20 Pro, var resultaten dåliga. Efter vidare läsning i Michale J. Carusos rapport gavs två felkällor, den första felkällan var kalibrering av magnetometern. För att kalibrera in denna skrevs en kortare kod som läste ut max och min på samtliga axlar samtidigt som modulen roterades, se bilaga 7. Min och maxgränserna användes sedan i kompasskoden för att kompensera X, Y och Z bias. Även en skalning gjordes med kalibreringen för att ge fina sinuskurvor med amplituden ett och bias noll. En observation som gjordes var att amplituden var 20% mindre vid mätningar ute på Stenpiren Göteborg jämfört med kalibrering gjorda i Infotivs labb. Senare i projektet kalibreras kompassen igen fast denna gången under ett test på Stena Line. Resultaten från denna kalibrering var att amplituden var 52% mindre än kalibrering i Infotivs labb. Detta beror troligen på elektromagnetiska störningar från radiosignaler, motorer och annan elektronik.

Den andra felkällan var att kompassen måste vara horisontell mot marken för att få ut relevant mätdata. Med hjälp av accelerometern kan man bestämma båtens lutning och kompensera för denna.

4.7 Mjukvara

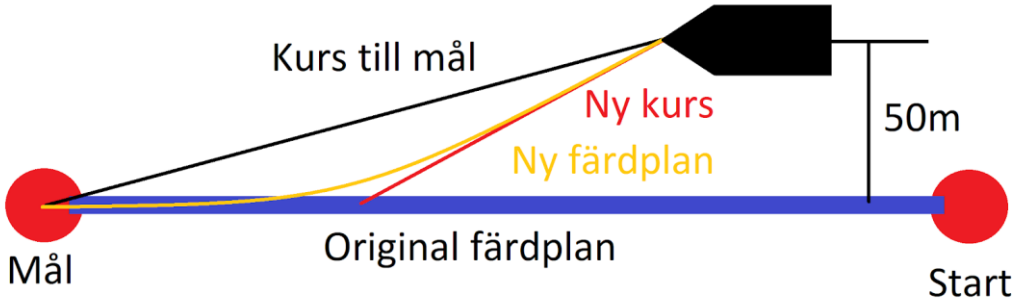
Allt är kodat i små block som först testades och verifierades enskilt innan de sattes ihop till ett komplett program. Ett problem som uppstod när de olika programmen sattes ihop var att biblioteken: servo.h och cppm.h båda använder interrupt timer1. Detta lösts genom att installera det alternativa servo biblioteket servotimer2.h som istället använder interrupt timer0. En annan möjlig lösning hade varit att själv gå in i biblioteksfilen och justera så att biblioteken använder olika interrupt, men i vårt fall så sparade vi mycket tid på att endast byta biblioteket. Hela koden kan studeras i bilaga 10.

Koden har tre olika lägen, dessa är: avstängd, manuell och auto. Se bilaga 8 för att få en överblick hur de tre lägena är implementerade i koden. Föraren kan när som helst när denne är inom räckhåll byta mellan de olika lägen via en switch på radiodosan.
I avstängt läge så är alla styrmekanismer avstängda och båten hämtar endast data från sina sensorer och sparar datan i en datalogger. I det avstängda läget så är fortfarande solcellerna inkopplade och laddar batteriet.

Läget där föraren styr båten manuellt kan också kallas radiostyrt, föraren har full kontroll över både roder och propeller vi radiokontrollen. Räckvidden för radiokommunikationen ska vara upp till 2 km men det är alltid bra att hålla båten inom synhåll.
I auto kör autopiloten båten utifrån fördefinierade delmål där en algoritm räknar ut kurs. I detta läget finns det ingen gräns för hur långt bort båten kan framföras men ute på öppet vatten försvinner snabbt GSM-täckningen och med den också möjligheten att veta båtens position och status.

4.7.1 Börvärdesreglering

Då båten svänger relativt lätt och snabbt läggs fokus på att reglera båtens börvärde. Börvärdesregleringen utgår först från kursen rakt mot målet, sedan används båtens avvikelse från originella färdplanen som reglering. Avvikelsen multipliceras med en vald faktor och adderas till börvärdet, se figur 16. Börvärdet kan aldrig bli större eller mindre än 90 grader från orginella färdplanen, detta för att undvika att båten åker i motsatt riktning från målet. Detta ger att ju mer båten driver av kurs desto större kommer den försöka ta sig tillbaka till den planerade rutten. Kommer den utanför en maxgräns kommer även motorn köra på max för att akut komma tillbaka innanför maxgränsen. På öppet hav kan denna gräns samt avvikelsefaktorn vara snällare men ska båten söka av ett specifikt område eller navigera inomskärs måste dessa gränser vara snävare. När börvärdet reglerats regleras rodervinkeln för att svänga in båten till börsvärdets kurs.

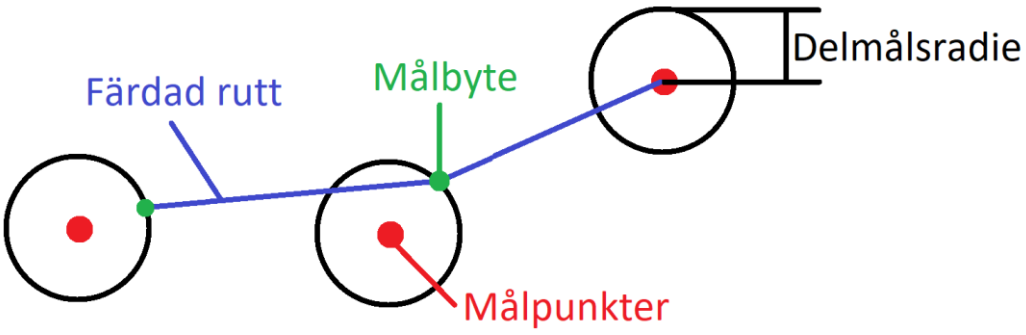


Figur 16, börvärdesreglering då båt är 50m av kurs

4.7.2 Delmålsnavigering

För att ge en mer tillförlitlig navigation konstruerades ett delmålssystem för navigering. Båten skall alltså nå varje delmål innan den får lägga i kurs mot nästa delmål. Att båten skulle nå en exakt GPS punkt är svårt då väder och vågor konstant ändrar båtens position och kurs. För att undvika att båten aldrig når GPS punkten sätts en radie kring varje punkt. Detta gör att båten siktar mot punkten men byter till nästa mål när den är inom radien, se figur 17. När sista delmålet är nått kommer båten hålla sig inom radien tills en människa byter läge på båten. Skulle båten av någon anledning missa ett delmål och vara betydligt närmare nästa delmål byts nuvarande mål till det närmare.

Innan båten sätts i autonomt läge måste en rutt planeras in som sedan programmeras in. Under arbetets gång kommer inte några direkta krav fram på hur många delmål som ska finnas per längdenhet. Maxantalet begränsas bara av enhetsminnet.



Figur 17, exempel på delmålsnavigering

4.8 Testning

Under design och konstruktionsprocessen testas komponenter och delsystem flytande men inför de större grindarna i projektet planerades större och mer specifika tester. Detta för att testa plattformens olika funktioner men även att få in relevant data. Dessa tester sker på öppet vatten och inte i labbet för att simulera en så verklig miljö som möjligt. Testerna sker mellan april och början på juni 2019 i Göteborgs kanal, på Lindholmen, i Eriksberg och på Kattegatt. Efter varje test diskuterades båtens prestanda, framtida risker och möjliga designförändringar.

4.8.1 Jungfrufärd, testkörning #1

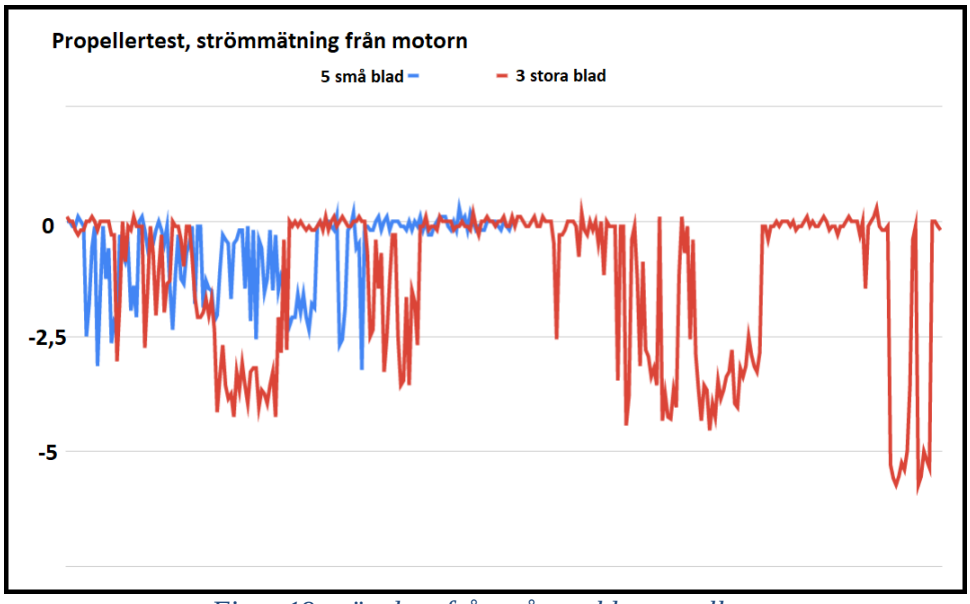
I första provkörningen så testades de grundläggande funktionerna och strömmar uppmättes i Sannegårdshamnen, se figur 18. Funktionerna som testades var framdrivning, styrning, radiomottagare och strömsensorer. Båten styrdes helt manuellt från radiosändaren, som senare kommer att användas som en extra livlina ifall den autonoma delen slutar fungera eller i trånga passager där man behöver styra manuellt. Avsikten med den första provkörningen var att mäta upp maxhastighet, minsta hastighet med styrfart och strömförbrukningen i olika hastigheter. Styrningen utvärderades så att den fungerade som förväntat och att båten ligger bra i vattnet och är stadig. En annan viktig punkt som utvärderades är om propellern ligger tillräckligt långt ner i vattnet för att det inte ska bildas virvlar som drar ner luft och minskar effektiviteten. För att testa lite olika tyngdpunkter på båten så flyttades batteriet mellan några olika lägen, slutsatsen blev att batteriet skulle ligga så långt bak som möjligt för att minska luftsuget i propeller.

Efter första provkörningen kunde ett antal slutsatser om båtens egenskaper dras. Efter en halvtimmes körning så var båten helt torr på insidan det bevisar att alla tätningar har varit täta under hela körningen och därför behöver inga mer åtgärder vidtas för att täta roder och propelleraxel. Det konstaterades också att styrningen var inverterad, men det är en lätt sak att åtgärda med mjukvara. Ett par andra problem som framkom var att rodret var för litet vilket försämrade styrförmågan vid de låga hastigheterna samt att vid höga varvtal drar proppen ner luft från ytan. Problemet med rodret kommer att åtgärdas genom att skruva på ett bredare roder på det redan befintliga rodret. Problemet att propellern drar luft kunde varit svårare att lösa. En åtgärd är att sänka ner propellern genom att flytta bak batteriet och på så sätt tyngdpunkten på båten. Men med lite tester visade det sig att detta inte räcker, istället kommer propellern byggas in i ett rör som kanske kan komma att motverka malströmmar som söker sig ner till propellern och som också eventuellt kan skydda propellern från skräp. En annan åtgärd som kommer att göras inför testkörning 2 är att köpa in en större propeller med större pitch, med den är förhoppningen att kunna minska varvtalet och minska risken för malströmmar.

Under testet mättes också strömförbrukning, se figur 19. I diagrammet kan man se att det skiljer mycket mellan de olika propellrarna, men slutsatsen blev att en stor propeller kommer behövas för att kunna hålla uppe en fart som kan klara motvind och eventuella strömmar. Men samtidigt gäller det att hålla nere strömförbrukningen för att lyckas ta sig över längre distanser. I diagrammet syns också den större strömförbrukningen när båten trycktes ner längre i vattnet för att testa hur mycket respektive propeller drar när den inte suger ner luft. Slutsatsen blev att strömförbrukningen är rimlig och det kommer med stor säkerhet kunna ta sig över en sträcka som Sverige till Danmark med den energi som finns i batterierna och den ström båten kan fångar upp av solen.

En annan del av testet var att mäta upp hastigheten med de olika propellrarna, det uppmättes en hastighet av ca 5 km/h. Detta gjordes på ett mycket primitivt sätt genom att mäta upp en sträcka och att sedan ta tiden för båten att köra sträckan. Mätmetoden utvecklas vidare i senare tester då det uppstod två problem med mätningen: det var svårt att hålla båten i rak linje och det blåste så pass mycket under testet att det påverkade båten. Slutsatsen av hastighetsuppmätnigen blev att det krävs högre hastigheter för att kunna parera starkare strömmar och vindar ut till havs, men inför test 2 köptes en större propeller, malströmmsskydd testas och ett större roder implementeras för att öka hastighet och styrförmåga.

Figur 18, testning utan solceller i Sannegårdshamnen



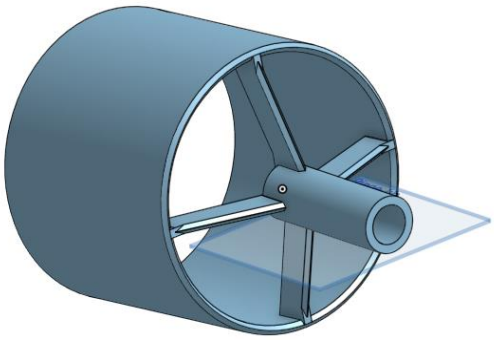
Figur 19, stömdata från två utvalda propellrar

4.8.2 Testkörning #2

I testning nummer 2 utvärderas åtgärder för problem som upptäcktes under test 1 i lite oroligare vatten utanför Eriksberg se figur 20. De problem som gjorts åtgärder emot och utvärderas är att båten var seg att styra och svängde åt fel håll, delvis för att den sög luft.

Problemet med att den var svår att styra åtgärdas genom att bredda rodret betydligt till ett nästan tre gånger så brett roder, även koden programmerades om för att åtgärda felet med inverterad styrning. Problemet med att propellern orsakade malströmmar som sög ner luft åtgärdades med två lösningar, en lösning var att köpa en större propeller med mer propellerblad och högre pitch. Denna propeller gör att båten kan flytta mer vatten än tidigare och borde på så sett kunna hålla varvtalet lägre och förhoppningsvis undvika malströmmar. Den större propellern möjliggör även att båten skulle kunna öka farten vid behov. Lösning nummer två är ett skydd runt propellern som ska störa ut eventuella malströmmar se figur 21.

Figur 20, provkörning Eriksberg

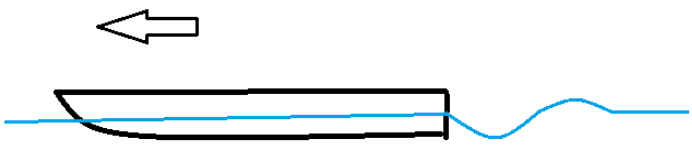


Figur 21, malströmmsskydd

Slutsatserna från test nummer två är att skyddet runt propellern inte hade den önskade funktionen, istället för att skydda från luftsug så kom det in mer luft och hastigheten blev ännu sämre, även styrningen blev betydligt sämre.

Utan skyddet så gick båten mycket bättre än första provkörningen, detta till följd av den större propeller som sög mindre luft och det bredare rodret som fick båten att bli avsevärt lättare att manövrera. Därför kommer inte malströmsskyddet efter test nummer två inte användas mer.

Inför ett senare test kommer en förlängning på propelleraxeln att testas, detta för att sänka ner propellern längre ner i vattnet. Detta sker både genom att propelleraxeln har en svag lutning men förhoppningen är även att kunna placera propellern bakom den lokala sänkningen av vattennivån som blir där båten trängt undan vattnet. Istället för att propellern ligger i sänkan så hoppas vi kunna placera propellern där vattnet istället är något högre än normalnivån, se figur 22.



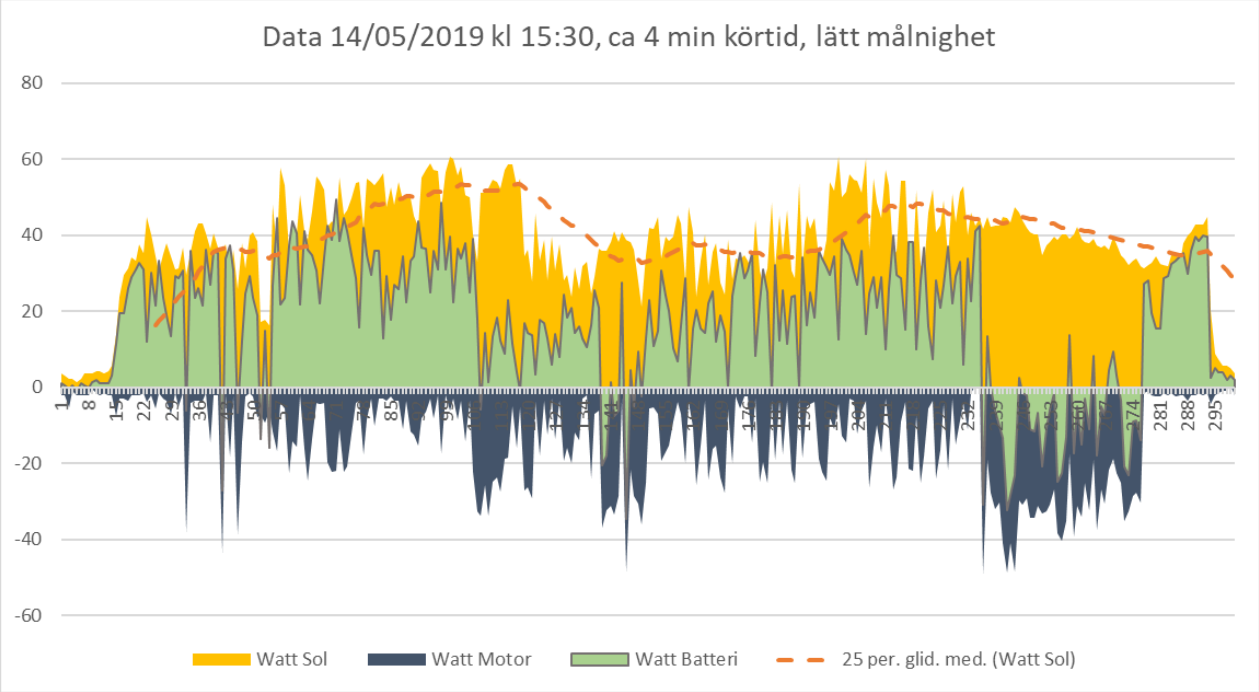
Figur 22, vattensänka

4.8.3 Testkörning #3

Under testkörning 3 testades all slutgiltig hårdvara tillsammans förutom kompassen, de punkter som testades är:

- huruvida GPS, GSM störs av att solcellerna ligger ovanför.
- tyngdpunkt och stabilitet med solceller installerade på båten.
- strömförbrukning.
- strömförsörjning från solcellerna.
- MPPT:ens verkningsgrad och funktion.

Slutsatser som kunde dras från provkörning tre var att GPS och GSM inte störtes av solcellerna. Däremot så märktes det en klar skillnad när solcellerna placerades på toppen av båten att båten blev ostadigare än tidigare, men inga exakta mätningar gjordes på detta.
Den inkommande strömmen in från solcellerna på eftermiddagen var förvånansvärt hög och batterierna kunde laddas samtidigt som motorn drev framåt, se figur 23 över strömmen in och ut från MPPT samt till motor. MPPT:en fungerade över förväntningarna och levererade en effektivitet på 95% under provkörningen. Under provkörningen så fortsatte problemet med att propellern gärna suger luft och därför så kommer den förlängda propelleraxeln tillämpas under nästa provkörning, se figur 21 på föregående sida.



Figur 23, graf över strömmar i båten

4.8.4 Stena Line test

Ett större test för navigation och kommunikation gjordes den 10 maj 2019. För att testa hårdvaran i så lämplig miljö som möjligt utan några risker för båten och projektet valdes en överdagenresa på Stena Line Danica till Fredrikshamn från Göteborg. Under resan gjordes flera tester ute på däcket för att verifiera i en så verklighetstrogen miljö som möjligt. Rutten som körs passar dock inte smarta mikrobåtar, de bör inte köra på trafikerade farleder mer än nödvändigt utan transponder.

Följande funktionalitet med krav testades:

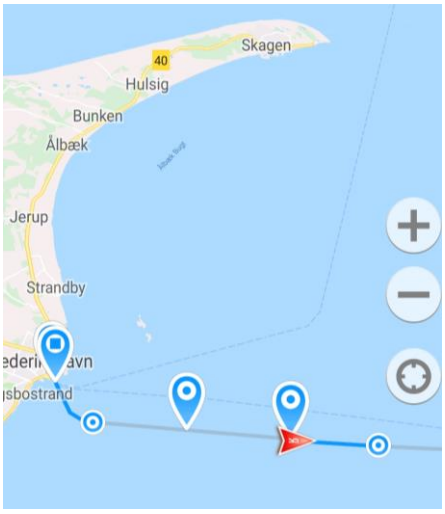
- Att ta emot GPS-koordinater under hela resan med noggrannhet på under 10 m
- Använda GSM nätverk för att skicka koordinater när båten har GSM-täckning
- Mäta kompassdata som är inom +- 3 grader
- Validera delmålssystemet med varierande mål och avstånd

Vädret under testet, enligt SMHI, var molnigt och med en vindhastighet på ca 5 m/s. Vågorna som observerades var från 0,1 m till 1,5 m. Mätningarna skedde på toppdäck, alltså ca 10m ovanför vattenytan. Kompassen behövde kalibreras om under testet då kalibreringen som gjorts i Göteborg gav felvärde på ca 20 grader. Nya kalibreringsvärden gav en maxamplitud på halva Göteborgkalibreringen samt en förskjuten baslinje.

Resultat av Stena Line test var följande:

- GPS-täckning gav ett noggrannhetsvärde på 0,7–0,9 HDOP (mätvärde på GPS nogranhet) vilket motsvarar 5–6,5 meters noggrannhet.
- GSM täckningen gick tyvärr inte testa då Danica hade en egen sändare, men GSM-modulen fungerade lika bra i det danska nätverket som det svenska.
- Kompassen ger +- 2,5 grader kontinuerligt jämfört med GPS och smartphone, se figur 25.
- Avståndsberäkningar samt kursberäkningar funkade som förväntat och där av bytte delmålssystemet mål alltefter resans gång se figur 24.

Slutsats av Stena Line test:
Navigationssystemets datainsamling fungerar bättre än kravspecifikationen. SMS kommunikation fungerar på både danskt och svenskt vatten. Tekniken är verifierad och godkänd. Dock ifrågasätts projektbåtens förmåga att klara sig på öppet hav på grund av våghöjden, trots relativt lugnt väder.



Figur 24, navigationstest

Figur 25, Josef testar GPS-kurs på Danica

4.8.5 Sluttest

Sluttestet utfördes i sjön Aspen intill Lerum på en någorlunda blåsig dag, 2a juni 2019. Flera test utfördes på den förlängda propelleraxeln, båtens sjöduglighet i lite oroligare vatten och framförallt autopiloten.

Det visade sig att den förlängda axeln fungerade bra, både den kortare förlängning på ungefär 5 cm och en längre variant på 10 cm som i sin tur fungerade bättre än den korta. Fördelarna märktes framförallt genom en högra hastighet och vid ett högre varvtal. Vid analys av strömförbrukningen framgick det också att strömförbrukningen gick ner i högre hastigheter i förhållande till tidigare tester. Nackdelen är att den långa axeln blir mer utsatt, speciellt under transport på land där axen riskerar att böjas.

Båten visade sig klara av de större vågorna mycket bra, dock så kördes hela testet utan solcellerna. Istället satt ett skyddslock som är lättare och bidrar till en lägre tyngdpunkt i förhållande till solcellslocket. När testerna var färdiga så visade det sig att lite vatten kommit in i springorna på locket på grund av att det locket inte har en lika bra tätning som tätningen för solcellerna. Vattnet skvätte upp på locket på grund av de höga vågorna som vi inte testat i innan men ingen åtgärd kommer göras då detta locket inte kommer användas vid riktiga uppdrag.

Autopiloten testades också framgångsrikt, till en början upplevdes ett par problem: Båten körde gärna i en ring efter att ha kört en kortare sträcka för att sedan fortsätta på den designerade rutten. Det andra problemet som framkom var att båten hade en kraftig självsvängning där rodret slog fram och tillbaka. Självsvängningarna orsakar ett par problem när det gäller strömförbrukningen, det första är att roderservot drar förhållandevis mycket ström under kontinuerlig användning. Det krävs även energi för båten att ändra riktning vid självsvängningar då den förlorar mycket hastighet vid varje sväng. Båda problemen löstes genom mjukvara, problemet med att båten körde i ringar visade sig vara en bugg med kompassen och självsvängningarna kunde hävas genom att justera parametrarna för autopiloten.

5 Resultat

Projektet resulterade i en fungerande prototyp, se figur 26, som nådde majoriteten av projektmålen. En större RC båt konstruerad för lätt väder och låga hastigheter. Navigationssystemet innehåller fungerande GPS, GSM och ett autonomt navigationssystem, se figur 27 för komplett elsystem. Solpanelen blev även större än specifikationen, 2 till solceller, detsamma gällande batteriet. Enligt beräkning skulle båten fungera för 24h körningoch under goda förhållanden väsentligt längre. På grund av tidsbrist och väder kunde aldrig det längre testet, att korsa Kattegatt, genomföras.

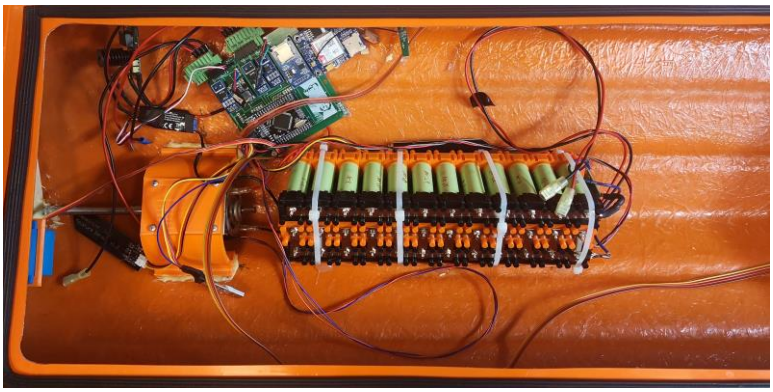
Testerna som utfördes under projektets gång visade en god förmåga att samla in data, vilket var testriggens uppgift. Under testningen framgick det även att testriggen inte skulle klara av hårdare väder. Specifikt så skulle större vågor eller extrema vindar utgöra en stor risk föra att välta båten då den inte har någon djup köl som balanserar.

Båten gav mätdata som enligt testkörning #3 visade att batteriet laddades mer än vad som drogs ut. Båten kan alltså på ren solenergi ta sig fram på öppet vatten samtidigt som datainsamling sker. Skulle ett flertal dagar med tunga moln inträffa under körning så kommer båten troligen få slut på energi.

Kommunikation med hjälp av GPS och GSM fungerade väldigt bra inomskärs och i sjöar. Dock har inte GSM nätet täckning mer än några mil utanför fastlandet. Skulle tester genomföras utanför täckningszonen måste detta ske med en följebåt eller satellitsändare, både för kommunikationens skull samt på grund av lagkrav.

Sluttestet genomfördes framgångsrikt på sjön Aspen, istället för Kattegatt, där autopilotens funktion testades framgångsrikt.

Figur 26, färdig testplattform med solceller



Figur 27, det kompletta elsystemet

6 Diskussion och slutsats

I introduktionskapitlet 1.4 preciserades projektets frågeställningar.

- Kan solenergi effektivt tillämpas på öppet vatten?

Under de rätta omständigheterna är detta fullt möjligt. Båten som byggdes i detta projekt designades med en låg vikt i relation till solcellsytan. Vid användning till större fartyg skulle en väsentligt större solcellsyta behövas. Troligen finns inte tillräckligt med yta för att solen ska kunna driva motorer i storleksklassen som ett containerfartyg. Användning av solceller skulle dock vara lämpligt för att driva datorsystem och andra sekundära system för större skepp samt agera som reservströmkälla. Däremot så skulle solenergi effektivt kunna utnyttjas på lättare båtar, mikrobåtar samt bojar för användning inom datainsamling samt framfart för båtar med en smart skrovdesign.

- Har smarta mikrobåtar förmågan att tillförlitligt ta sig fram samt samla in relevant data?

I sjöar och inne i skärgård skulle detta kunna ske under bra förhållanden utan större modifikationer av den befintliga båten. En bättre skrovdesign skulle vara viktig för att utföra uppgifter på öppet hav och i tuffare väder. Data samlades in tillförlitligt under alla testerna, vilken slags data som skulle kunna samlas in har stora utvecklingsmöjligheter.

- Vad är möjliga närliggande applikationer av en smart mikrobåt?

Besvaras i kapitel: 6.2 Förslag på vidareutveckling

- Vad är de större problemområdena inför framtida arbete med smarta mikrobåtar?

Besvaras i kapitel: 6.2 Förslag på vidareutveckling

6.1 Etik och moral

Att använda solenergi inom sjöfart skulle ha positiva effekter på miljön genom att minska användning av fossila bränslen. Detta skulle då också minska risken för oljespill och skadan på marina djur. Ett dilemma är dock behovet av batterier, enheter som kräver mängder med naturresurser att tillverka, men i dagens båtar så finns med en hög säkerhet redan en mängd av dessa ämnen.

Det uppstår även en frågeställning gällande automation som även varit aktuell inom bilindustrin. Om ingen kör, vem är då ansvarig? Idag finns det inga lagar som hanterar denna fråga men det kommer behövas inom snar framtid. Det skulle kunna vara så att båtar som kan orsaka större skador alltid måste övervakas av en person som är ytterst ansvarig. För smarta mikrobåtar skulle detta kanske inte krävas då deras storlek och vikt inte äventyrar egendom eller liv. Det skulle även kunna argumenteras att, de små riskerna som mikrobåtarna skulle kunna ge upphov till är acceptabla i jämförelse med den tjänst de gör både för miljön och de mantimmar de sparar.

6.2 Förslag på vidareutveckling

Bygget av båten har från start inneburit att framtidssäkra samtliga funktioner för framtida exjobb och fortsatta bygge. Att framtidssäkra båten har gjorts på ett antal punkter till att börja med så har så lite ingrepp i båten som möjligt gjorts. Exempelvis så har inte solcellerna monteras direkt på skrovet utan istället installerats på en plexiglasskiva. Också elektroniken har tagits fram för framtiden genom att exempelvis välja en motorkontroller med mycket fler portar än vad som behövs för projektet. Detta

för att i framtiden ha möjlighet att koppla in mer sensorer eller motorer till prototypen. Motor, roder och motorstyrning är överdimensionerade så att båten i kommande projekt ska kunna användas där högre fart och aggressivare styrning krävs.

Det finns många möjligheter att jobba vidare med för att vidareutveckla båten i framtiden. Båten kan anslutas till ett stort antal sensorer som kan användas till många olika sorters uppdrag till sjöss. En intressant sensor att arbeta vidare med är sonar. Detta är en sensor som används för att bestämma djup, bottenvegetation och att se föremål i mellanvattnet exempelvis fiskar. Med hjälp utav en eller flera autonoma små båtar så kan stora ytor på sjö eller hav kartläggas. Men också andra uppgifter kan tänkas utföras med ekolod som att hjälpa polisen med att hitta föremål, söka efter arkeologiska föremål eller rent militäriska ändamål som att leta efter fientliga ubåtar. Förutom ekolod så kan båten utrustas med alla tänkbara sensorer både relaterade till vattnet men också till luften. några användningsområden skulle kunna vara:

- bestämning av fiskbestånd
- kartläggning av bottnar
- hitta vrak
- lyssna på ljud under vattnen
- analysera radiotrafik
- samla in mätvärden i vatten som: salthalt, temperatur, CO_2, pH med flera
- samla in mätvärden i luften som: syre, CO_2 med flera

För att hantera tuffare väder samt längre turer skulle ett nytt skrov behöva designas. Det nya skrovet skulle kunna vara självrättande, ha en större solcellsyta och en bättre vattentätning. Även en lampa och flagga skulle kunna användas för att minska risken att bli påkörd av andra båtar. Till sist borde någon typ av ”black box”, så som flygplan har, implementerats. Denna skulle vara förseglad, använda ett eget batteri och kunna skicka ut GPS data om resten av båten inte längre fungerar som tänkt.

Regulatorn som styr autopiloten skulle kunna optimeras då den nuvarande är väldigt rudimentär. Speciellt om ett nytt skrov skulle användas så krävs en ny regulator som funkar med det nya skrovets egenskaper.

6.3 Slutsats

Projektets syfte har uppnåtts vilket var att undersöka utmaningarna med marin automation i kombination med solenergi på öppna vatten genom att bygga och testa en prototyprigg. Det finns fortfarande många obesvarade frågor inom området, så som testriggens långvariga funktion, som skulle kräva vidare testning. Vidare testning skulle också krävas för att vidare verifiera resultaten som tagits upp i denna rapport.

Den färdiga båten behålles av Infotiv AB som en utbildningsplattform och för uppvisning på mässor. Förhoppningsvis så kommer framtida anställda eller exjobbare att arbeta vidare och utveckla funktioner för kartläggning med hjälp av olika sensorer.

Referenser

[1] L. Goldie-Scot, "A Behind the Scenes Take on Lithium-ion Battery Prices," BloombergNEF, 5 maj. 2019, [Online]. Tillgänglig: https://about.bnef.com/blog/behind-scenes-take-lithium-ion-battery-prices/, hämtad: 2019-05-15.

[2] M. Vasell, " Drönare först på olycksplats - unikt flygtillstånd för Göteborgsföretag," Göteborgsposten, 3 dec. 2018. [Online]. Tillgänglig: https://www.gp.se/ekonomi/drönare-först-på-olycksplats-unikt-flygtillstånd-för-göteborgsföretag-1.11411544, hämtad: 2019-05-15.

[3] The World Bank Group, "Global Solar Atlas," 2019. [Online]. Tillgänglig: https://globalsolaratlas.info/, hämtad: 2019-02-12.

[4] M. Caruso, " Applications of Magnetoresistive Sensors in Navigation Systems," Honeywell Inc, Plymouth MN, USA, [Online]. Tillgänglig: https://aerospace.honeywell.com/en/~/media/aerospace/files/technical-articles/applicationsofmagnetoresistivesensorsinnavigationsystems_ta.pdf, hämtad: 2019-03-24.

Bilagor

Bilaga 1, Gate specifications

Gate specifications

Gate 1 - Acquisition and Basic Functionality

General

- Checking acquisition list, all components should have arrived
- Budget, under 12k used

Navigation

- GPS tested outside in plastic container. General positioning with 10m> accuracy
- GSM tested with two-way communication, sending and receiving SMS and saving appropriate info
- Compass accuracy tested with proximity to 70W or more circuit within 10 cm, v

RC Boat

- Testing sufficient propulsion, using motor in a floating container with weights in a body of water, v11
- Servo power tested vs expected external forces together with rudder, also maximum/minimum rudder angle.
- Hull verification via testing flotation, water level with weights and water leakage, v11

Solar Charging

- MPPT outputs 12,6v continuously with varying input voltage and current
- Batteries are according to specs
- Solar cells have appropriate efficiency and therefor appropriate output power

(Gate 2,3 & 4 on next page)

Gate 2 – Module Testing

Navigation – Stena Test

- Bringing the navigation equipment along on a trip to Denmark to test functionality on open water.
    - Receiving appropriate GPS coordinates during entire trip, 10m> accuracy
    - Using GSM network to send coordinates when closer to land
    - Get compass bearings that are within 5 degrees
    - Testing checkpoint system with varying target distances
    - Preferably with a 60W circuit within 20 cm
- Test data compared with smartphones

RC Boat

- Boat with fully functional "RC boat" functionality while having estimated final weight load in choppy waters
    - Steering, reaching target locations with ease
    - Propulsion, Speed > 5 km/h
    - RC controlled
    - No water intrusion

Solar Charging

- Complete solar panel with 24-26 cells in series
    - Soldered
    - Epoxy coated
    - Water protected
- Solar panel with MPPT can charge the 36-cell battery pack on a sunny day
- Producing on average 50 W during sun hours (minus dusk and dawn) when horizontal near large body of water

Gate 3 – Lake test

Navigation

- Vessel autonomously reaching given GPS position with 10 m > accuracy
- Vessel autonomously reaching series of GPS positions
- Sending SMS when close to and when final destination is reached
- Vessel not following positions to the extreme but working within a set window

Power System

- Boat running only on solar power
- Boat running only on batteries
- Boat running on only solar power and charging batteries simultaneously

Gate 4 – Final presentation of Project and Education

- Vessel autonomously crossing Kattegat, finished all deliverables
- Educational material finished for estimated 4h lecture including labs
- Present education setup guide

Bilaga 2, Riskanalys

Simplified risk assessment

Project name:	Autonomus Solar Driven Boat	Consequesnce	3	Afloat but potential consequences
Last updated:	04-02-2019	1 Sub optimal	4	Dead in the water
		2 Slower travel	5	Loss of boat

Risk #	Potential Risk	Consequences	Risk Rating	Consequence	Planned Action
1	Motor Overheated	Motor will no longer work and will lose propulsion	1	4	Current sensor by motor. Evaluate Satellite module
2	Rudder damage	Loss of direction control	1	3	Controlsystem with feedback
3	Propeller stuck in biomass	Overheating motor or loss of speed	2	4	Current sensor by motor, reverse rotation
4	Water damage motor	Loss of propulsion	4	4	Current sensor by motor. Evaluate Satellite module
5	Water damage CPU	Loss of control and boat will be dead in water with no comunication	4	5	Evaluate Satellite module
6	Water damage batteries	Shorts circucite and potential fire or catastrophic failure	4	5	Evaluate enclosure, more?
7	GSM no signal	Hard to find final position and retreive boat	2	3	Evaluate Satellite module
8	GPS no signal	No navigation, driving blind	1	3	Evaluate Satellite module
9	Navigation system failure, Poor design	Stuck at sea and potential loss of the boat	3	4	Danmark,sbåten
10	Solar panels, fysical damage	Breaking the curcuit or lowering power output	1	3	Create pre depature checklist
11	Solar panels, electrical	Systen failur, lower or no power output	1	3	Create pre depature checklist
12	Solar panels, shaded	Lower power output	4	2	Glare spots on boat to soare brids
13	MPPT not functional	Loss of charge function, Fried system	1	5	Testing in workshop
14	Batteries, thermal Event	Catastrophic failure	1	5	BMS and MPPT
15	Batteries, BMS not functional	Systems power failure	1	5	Testing in workshop
16	Batteries, poor electrical design	Poor safty design resulting in fire	2	5	Testing in workshop and consultation
17	Run over by other boats	Smashed to pieces	3	5	Lamp, glare, orange and well planned route. More?
18	Weather	Water damage or flipped boat	2	5	well planned route and weather
19	Budget	All goals might not be achived	1	2	Critical component choice
20	Deliveries	Higher cost or smaler scope	2	3	Checklist of what has arrived
21	Laws	Could be illigal and boat confiscated	2	5	Consultation with professor
22	Time	Smaler scope and goals not achived	3	3	Detailed plan and a high work morale
23	Magnetic interference	Inaccurate navigation	3	3	Meassuring interference with maximum magnetic field
24	To high center of gravity	Boat could flip and not be self correcting	2	5	Make sure that parts are placed low in the boat and test stability with all components onboard

Bilaga 3, Solcellsdatablad

Electrical Characteristics of Typical Cell at Standard Test Conditions (STC)

STC: 1000W/m², AM 1.5g and cell temp 25°C

Bin	Pmpp (Wp)	Eff. (%)	Vmpp (V)	Impp (A)	Voc (V)	Isc (A)
G	3.34	21.8	0.574	5.83	0.682	6.24
H	3.38	22.1	0.577	5.87	0.684	6.26
I	3.40	22.3	0.581	5.90	0.686	6.27
J	3.42	22.5	0.582	5.93	0.687	6.28

All Electrical Characteristics parameters are nominal
Unlaminated Cell Temperature Coefficients
Voltage: -1.8 mV / °C Power: -0.32% / °C

Datablad från sunpower

Bilaga 4, Energiberäkningar

Battericeller (36st tot)		Paket (3s12p)	
3,4	Ah	40,8	Ah
3,7	V	11,1	V
13	Wh	453	Wh

Förväntad snittförbrukning (Watt)	
Dator och sensorer	1,8
Styrning	2
Belysning	3
Motor	30
Komunication	1,5
Förluster	5%
Totalt	40

Förväntad burstförbrukning (Watt, Ah)	
Dator och sensorer	2,0
Styrning	3
Belysning	3
Motor	45
Komunication	15
Förluster	5%
Totalt	71

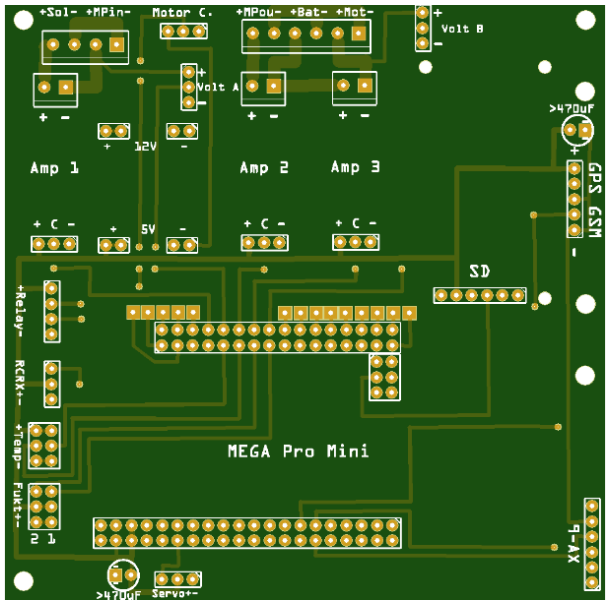
Förbukning / 12 h	483	W

Förbukning / 12 h med burst	514	W

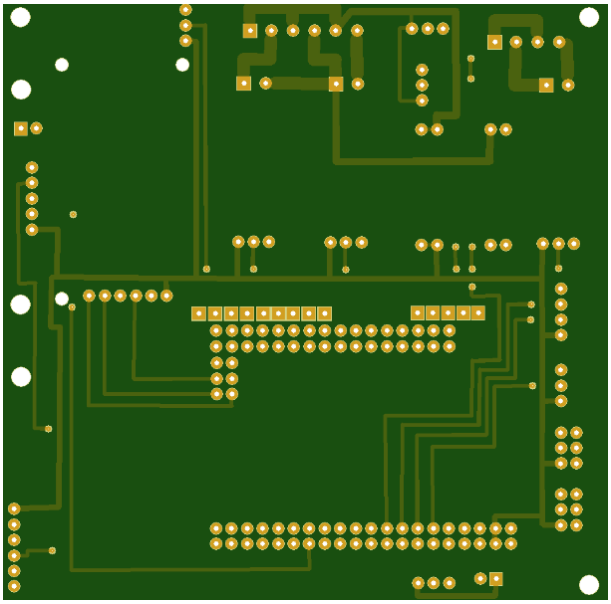
Båt körtid		
Batteri vid start	453	Wh
Laddning på en bra dag	460	Wh
Total körtid	21	h
Körtid med marginal	19	h
Maximal räckvid	126	Km

Tidiga energiberäkningar

Bilaga 5, Kretskort fram och baksida

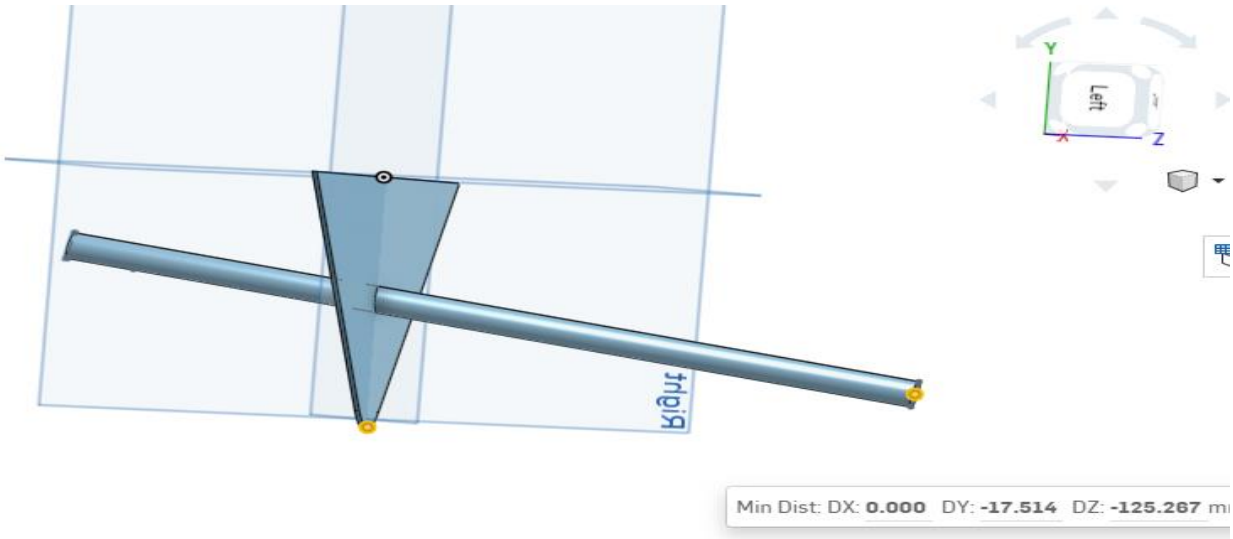


PCB framsida

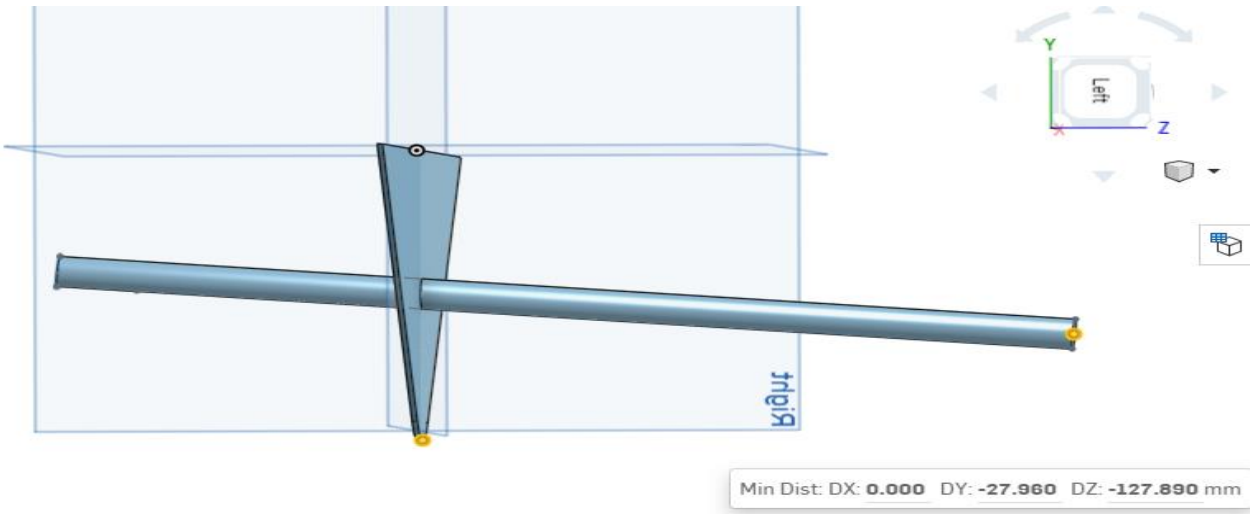


PCB baksida

Bilaga 6, Vinklar på motoraxel



Axellutning 15 grader

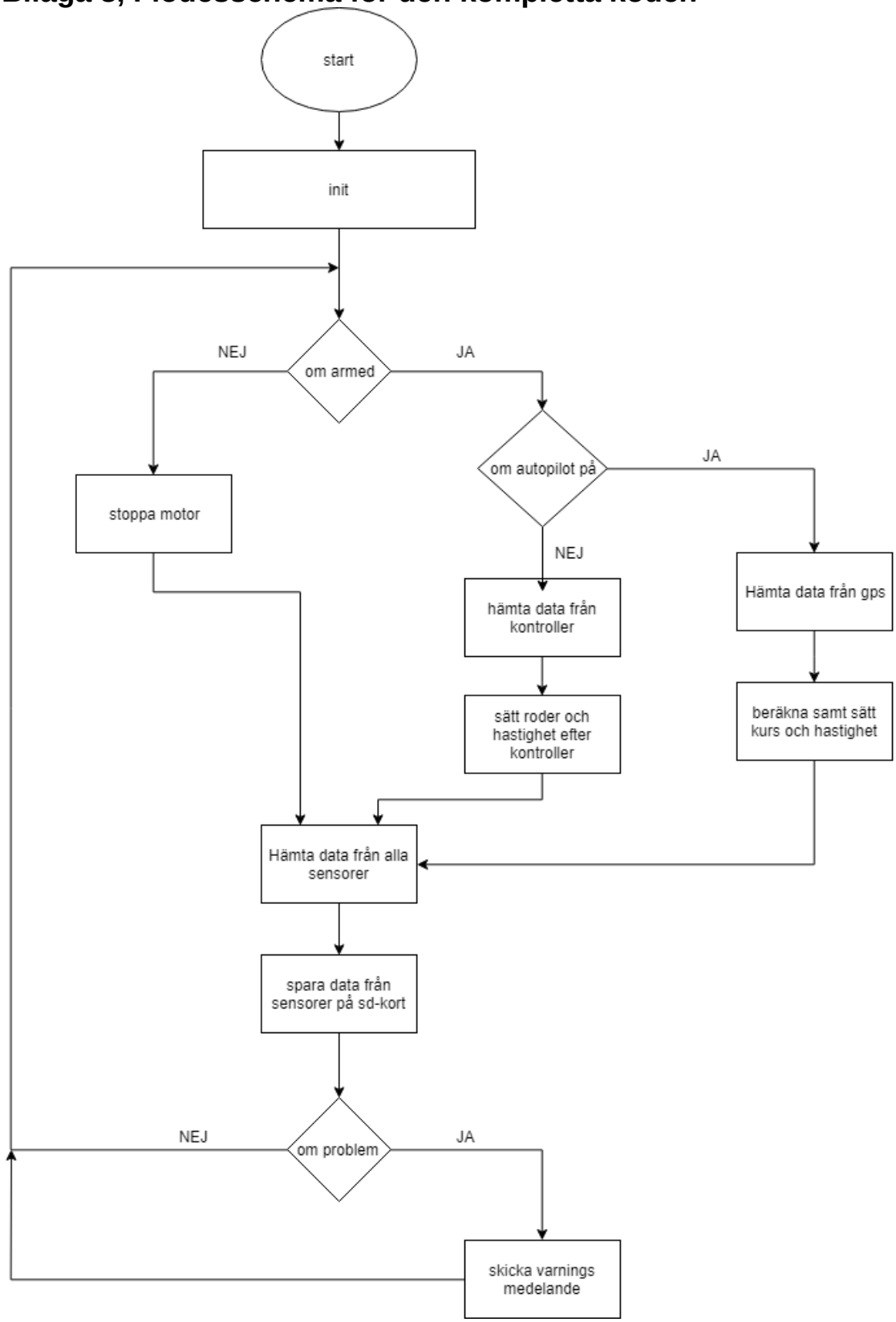


Axellutning 5 grader

Bilaga 7, Magnometer kalibreringskod

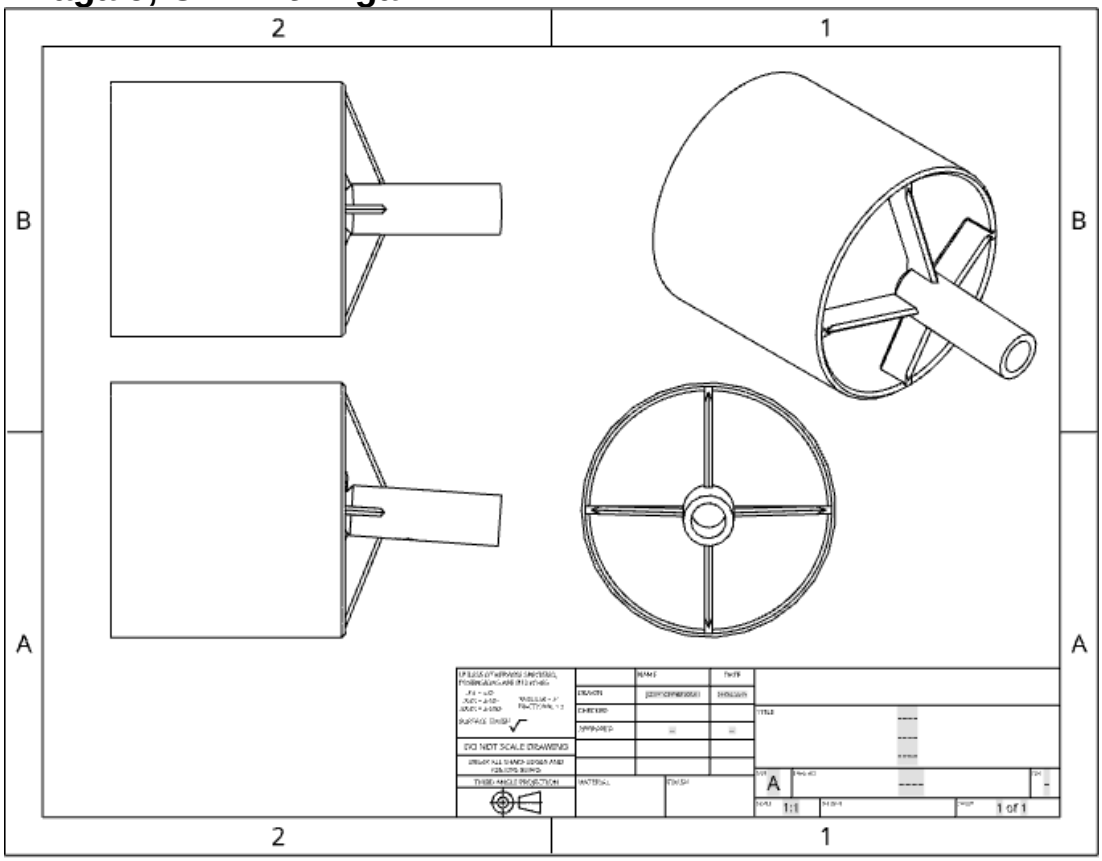
```
#include "MPU9250.h"
MPU9250 mpu = MPU9250();

float mx_max = 0, mx_min = 0;
float my_max = 0, my_min = 0;
float mz_max = 0, mz_min = 0;

float offset_x, offset_y, offset_z;
float scale_x, scale_y, scale_z;
void setup(void) {
  Serial.begin(9600);
  uint8_t temp = mpu.begin();
}

void loop() {
  //Mag
  mpu.set_mag_scale(SCALE_14_BITS);
  mpu.set_mag_speed(MAG_8_Hz);
  if(!mpu.get_mag()){
    mpu.get_mag_t();

    if (mpu.mx_t > mx_max) mx_max = mpu.mx_t;
    if (mpu.my_t > my_max) my_max = mpu.my_t;
    if (mpu.mz_t > mz_max) mz_max = mpu.mz_t;

    if (mpu.mx_t < mx_min) mx_min = mpu.mx_t;
    if (mpu.my_t < my_min) my_min = mpu.my_t;
    if (mpu.mz_t < mz_min) mz_min = mpu.mz_t;

  offset_x = (mx_max + mx_min)/2;
  offset_y = (my_max + my_min)/2;
  offset_z = (mz_max + mz_min)/2;

  scale_x = (mx_max - mx_min)/2;
  scale_y = (my_max - my_min)/2;
  scale_z = (mz_max - mz_min)/2;

    Serial.print("Offset X"); Serial.print(offset_x);
    Serial.print("Offset Y"); Serial.print(offset_y);
    Serial.print("Offset Z"); Serial.println(offset_z);

    Serial.print(" Scale X "); Serial.print(scale_x);
    Serial.print(" Scale Y"); Serial.print(scale_y);
    Serial.print(" Scale Z"); Serial.println(scale_z);

  }
}
```

Bilaga 8, Flödesschema för den kompletta koden

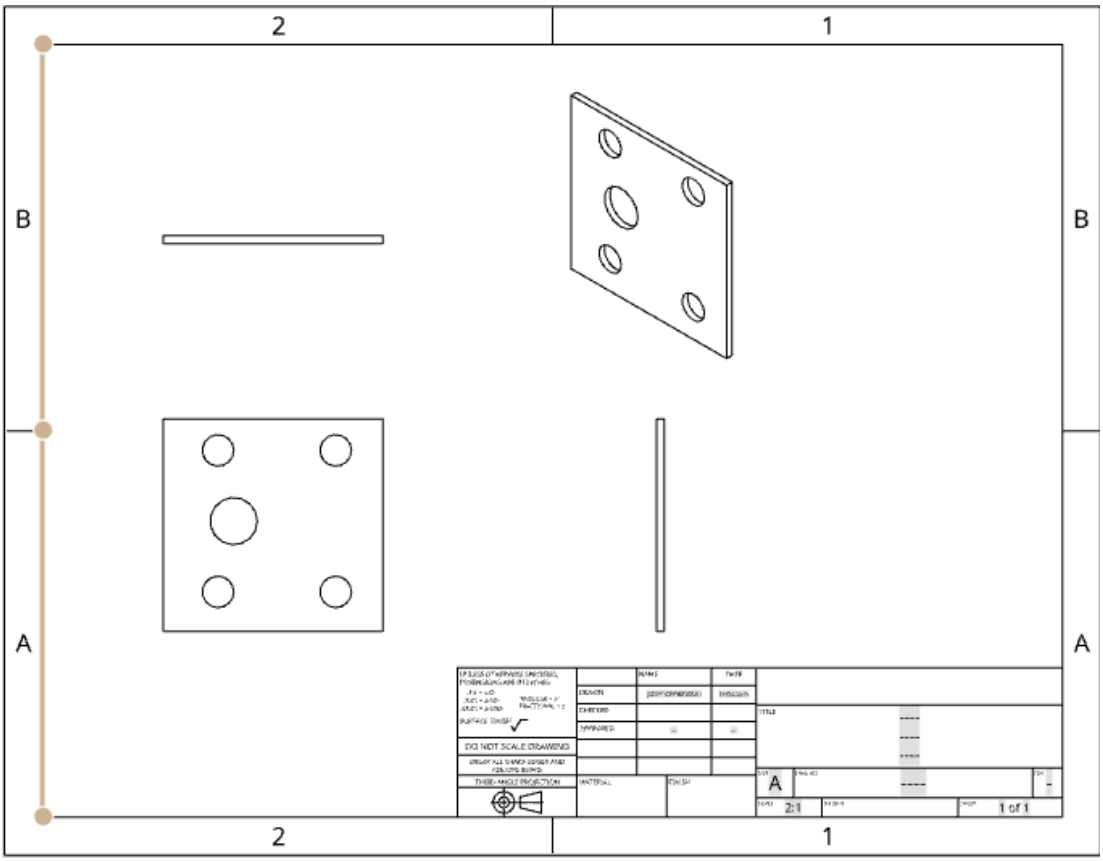


Komplett flödesschema

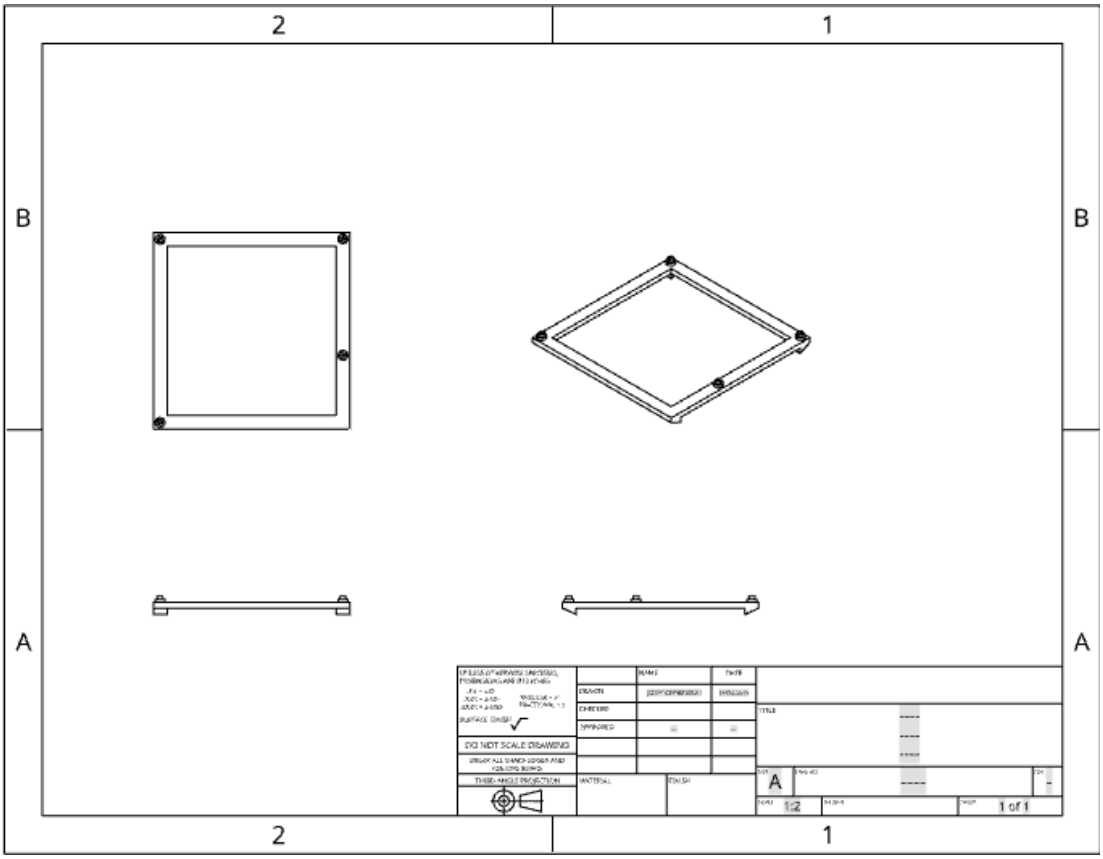
Bilaga 9, CAD-ritningar



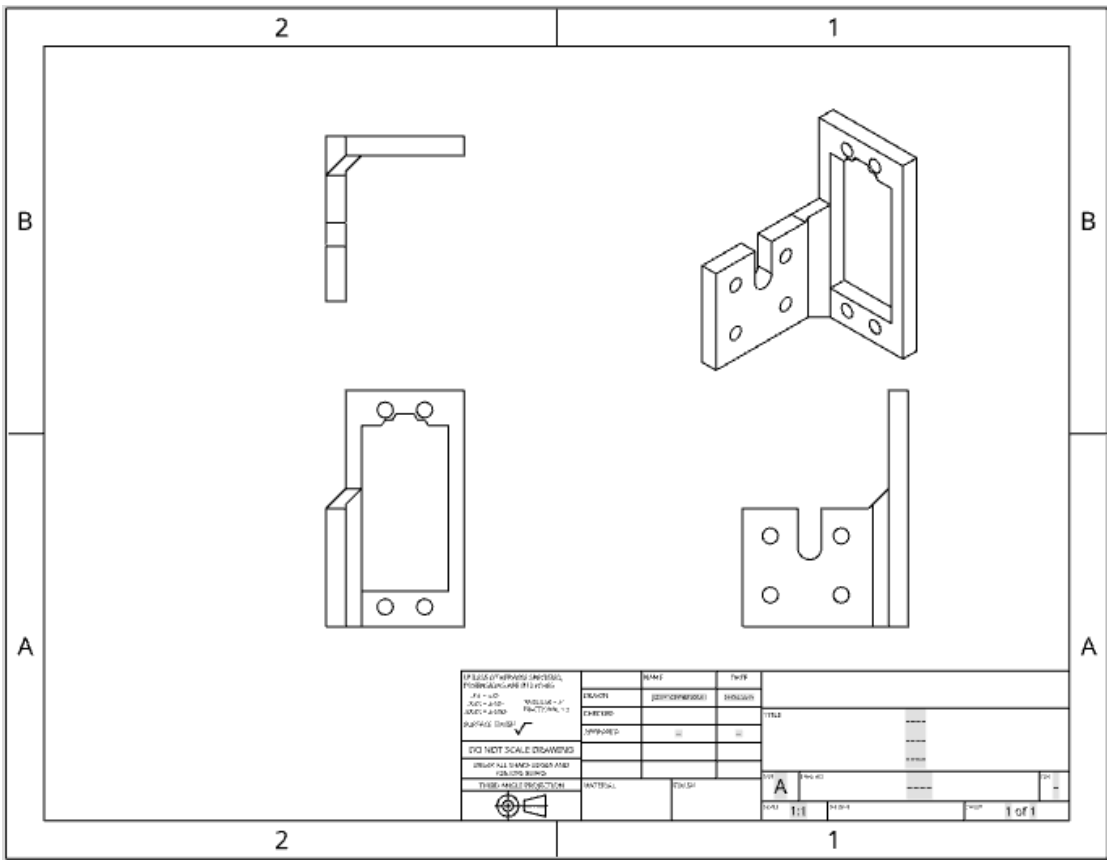
CAD-modell: Malströmmskydd



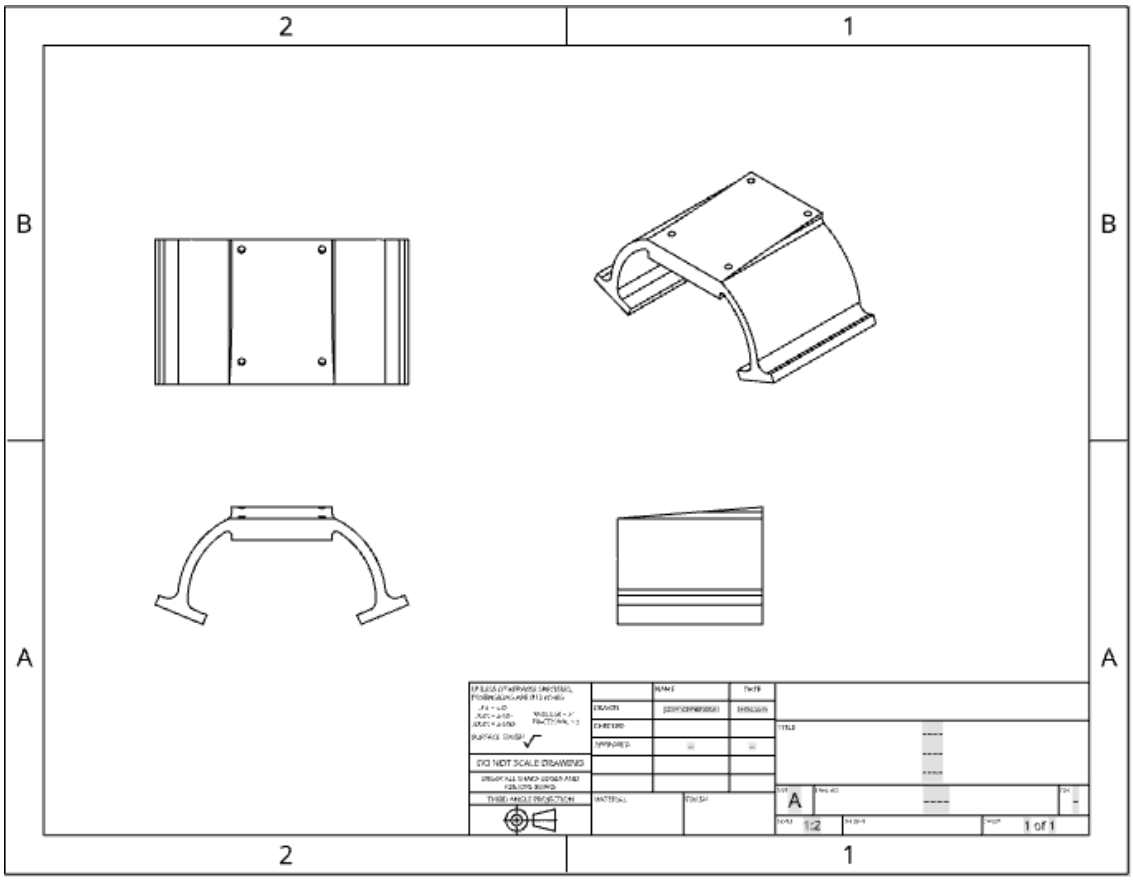
CAD-modell: Damaskhållare



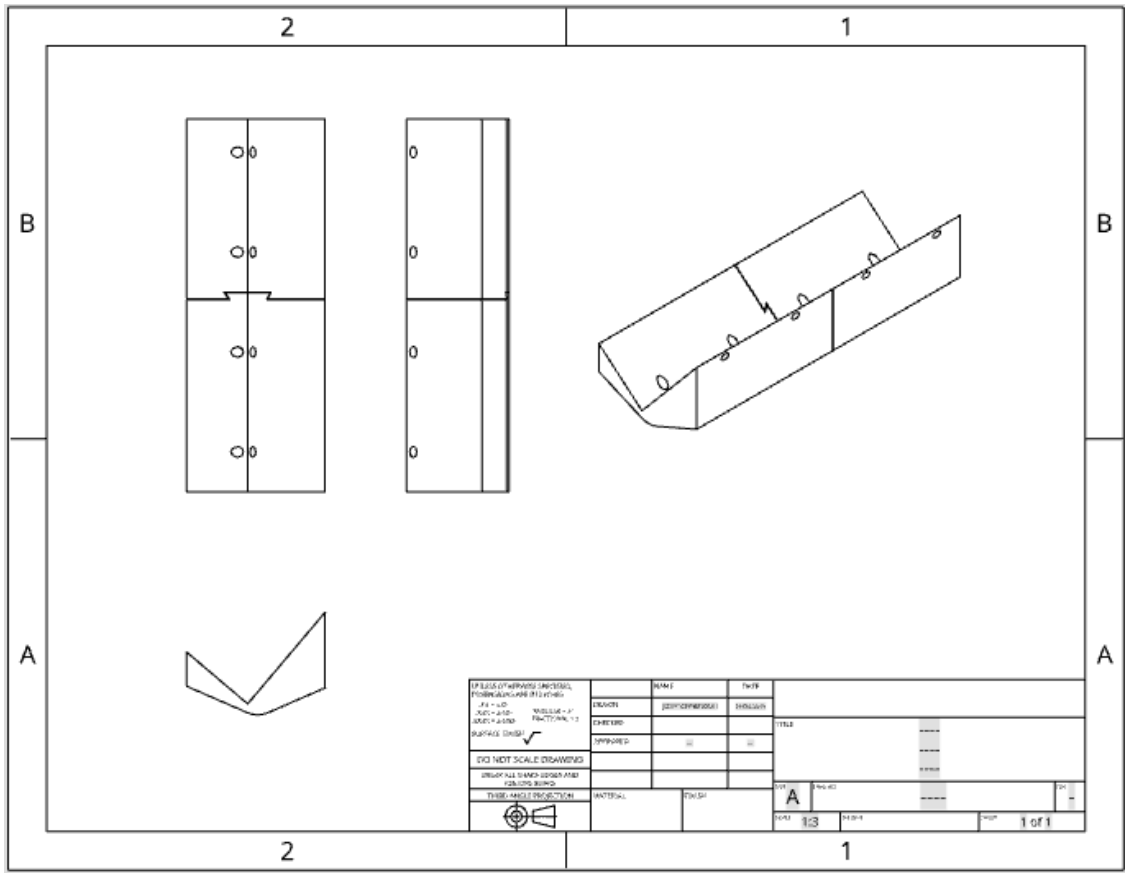
CAD-modell: Kretskorthållare



CAD-modell: Servofäste



CAD-modell: Motormontering



CAD-modell: Batterihållare

Bilaga 10, Komplett kod

```
#include <ServoTimer2.h> // används istället för servo.h för att undvika att både servo.h och cppm använder interrupt timer 1
#include "CPPMRX.h"
#include <SPI.h> //standard biblotek
#include <SD.h> //standard
#include <SoftwareSerial.h> //för gps och gsm
#include "defines.h"//egna start up filer
#include "MPU9250.h"

MPU9250 mpu = MPU9250();

ServoTimer2 motor;  // create servo object to control a servo
ServoTimer2 servo;

int aile = 0;   //signaler för rc dosa
int elev = 0;
int thro = 0;
int rudd = 0;
int gear = 0;
int aux1 = 0;

String ar = "";
String manad = "";
String dag = "";
String timmar = "";
String minuter = "";
String sekunder = "";


static const uint8_t RXPIN = 2;
static const uint8_t NCHAN  = 6;
static CPPMRX rx(RXPIN, NCHAN);

double Amps1 = 0; //signaler för strömmätare A6
double Amps2 = 0; //A7
double Amps3 = 0;

float voltageA = 0;   //spänning sesnor
float voltageB = 0;
int fuktsensor1 = 0;  //fuktsesnor
int fuktsensor2 = 0;
float temperatur1 = 0;
float temperatur2 = 0;
char flag = '0';     //varningsflagga 0 = normalt, 1 = fukt, 2 = temp batteri, 3 = temp motor, 4 = lågt batteri, 5 = hög ström

const int current1 = A6;  //registrering av alla portar
const int current2 = A10;
const int current3 = A12;
const int chipSelect = 53;
const byte rxPin = 10; // för gsm/gps
const byte txPin = 11; //  för gsm/gps
const byte float_pin0 = 18; // kopplad till D10 och måste lämnas flytade så att den inte påverkar
const byte float_pin1 = 19; //kopplad till D11
const int temp1 = A11;
const int temp2 = A9;
const int V_A = A8;
const int V_B = A14;
const int Rela1 = 6;
const int Rela2 = 4;
const int fukt1 = A15;
const int fukt2 = A13;

int A_key = 5;  //signaler för gsm/gps
String inData;
String number = "+46725660082";
String quality;
```

```
int A_key = 5;  //signaler för gsm/gps
String inData;
String number = "+46725660082";
String quality;

struct GSM_data gsm;
struct GPS_data GPS;  //slut gps/gsm signaler
struct GSM_data_float gsm_float;

SoftwareSerial SIM900(rxPin, txPin); // initerar komunikationen till sim

////////////////////////////////////////////////////////här börjar niels variabler//////////////////////////////////////////////////////


float  temp_brng;
int checkpoint_nr = 1;

// Alla kordinater som båten ska åka till i rätt ordning
                    // 0          1          2          3          4
float target_lat[] =  { 57.766653, 57.766687 ,57.767576, 57.767582, 57.766653}; //All GPS cordinates in order for longitude including start
float target_long[] = { 12.250370, 12.248649 ,12.248571, 12.250320, 12.250370}; //All GPS cordinates in order for latitude
#define total_checkpoints 4

// Längst mer kusten, norr och sedan söder
                  //        0          1          2          3          4          5          6          7
//float target_lat[] =  { 57.766658, 57.767428 ,57.768510, 57.768510, 57.766653, 57.768120, 57.767244, 57.766658}; //All GPS cordinates in order for longitude including start
//float target_long[] = { 12.250373, 12.250294 ,12.249340, 12.249340, 12.250370, 12.247668, 12.248872, 12.250373}; //All GPS cordinates in order for latitude
// #define total_checkpoints 7
////////////////////////////////////////////////////////////////////////////////

float SoC = 0.5;  //State of Charge, max = 1 min = 0 half = 0.5

int speed_case = 0;


int motor_speed = -100; // Motor speed
int rudder_angle = 0; // Rudders angle

float setpoint = 0; //Börvärde
float boat_brng = 0; //Årvärde

float brng_boat_point = 180;
float distance_boat_point = 1000;


float brng_true = 180,distance_point_point = 1000;
int dist_to_true_vector = 0; //också känd som h



////////////////////////////////////////////////////här slutar niels variabler/////////////////////////////////////////////////////////

void setup() {

//////////////här börjar niels setup////////////////////////

  uint8_t temp = mpu.begin();
  mpu.set_accel_range(RANGE_4G);
  mpu.set_mag_scale(SCALE_14_BITS);
  mpu.set_mag_speed(MAG_8_Hz);

//////////////här slutar niels setup////////////////////////
```

```
  pinMode(float_pin0, INPUT); //lälmnar pinnen flytande
  pinMode(float_pin1, INPUT);


  motor.attach(8);  // initsiering av motor
  servo.attach(3);
  motor.write(0);
  rx.begin();
  Serial.begin(9600);

  sim_setup();

  delay(3000); //behövs för att starta upp ESC

  Serial.print("Initializing SD card...");    //Uppstart av data logger
  if (!SD.begin(chipSelect)) {
    Serial.println("Card failed, or not present");
  }
  Serial.println("card initialized.");
  get_gps();
  SD_logg_setup();
  //send_info(); //skickar ett sms med gps info vid uppstart
}

void loop() {
  cppm_cycle(); //hämtar färska värden
  fetch();

  if (aux1 >= 80) {  //Armed

    if (gear >= 80)  { //autonom del påslagen
            if( get_compass() > 0){
                temp_brng = get_compass();
                if (temp_brng>=0)boat_brng = temp_brng;
                get_data ();
                set_rudder ();
                set_speed ();
                Serial.println (brng_boat_point);
                Serial.println (setpoint);
                Serial.println (boat_brng);

          }
    }
    else {
      set_motor_speed(thro);  //styrning med kontroller
      set_servo_roder(aile);
    }
  }
  else {
    set_motor_speed(-100); //viktigt att motorn stängar av sig med -100 när båten är avstängd
    set_servo_roder(0);
  }
}
void fetch() {
  if (timer(1000) == 1) { //uppdatering av alla variabler
    currentSensor();
    voltage_sensor();
    fukt_sensor();
    temp();
    get_gps();
    SD_logg();
  }
}
```

Main()

```
// Samlar in all data från:
// GPS_Struct, AMP_1/2/3, V_A/B, comp_brng
// Beräkna hastighet (medel senaste min), avstånd till nuvarande mål, avstånd till bör vector
// Kolla om båten är vid nästa mål, uppdatera nuvarande mål
// Vatten i båten
// Temp


void get_data () {
  SoC = (V_B - 9.6)/3;



  //Calcs line and angle from boat to next point
  brng_boat_point = brng_p2p (gsm_float.Latitude, gsm_float.Longitude, target_lat[checkpoint_nr], target_long[checkpoint_nr]);
  distance_boat_point = dist_p2p(gsm_float.Latitude, gsm_float.Longitude, target_lat[checkpoint_nr], target_long[checkpoint_nr]);

  //if //GOAL IS REACHED UPDATE TO THE NEXT GOAL AND CALC DISTANS AND ANGLE
  //Calcs line and angle from last goalpoint to next goal
  brng_true = brng_p2p (target_lat[checkpoint_nr-1] ,target_long[checkpoint_nr-1], target_lat[checkpoint_nr], target_long[checkpoint_nr]);
  distance_point_point = dist_p2p (target_lat[checkpoint_nr-1] ,target_long[checkpoint_nr-1], target_lat[checkpoint_nr],target_long[checkpoint_nr]);

   dist_to_true_vector = distance_boat_point * sin((brng_true - brng_boat_point)*PI/180);
}
```

Get_data()

```cpp
#include "MPU9250.h"

uint8_t MPU9250::begin() {
  Wire.begin();
  Wire.setClock(400000L);

  if (__read_byte(MPU9250_ADDRESS, WHO_AM_I) != WHO_AM_I_RESP)
  {
    while(1){
      Serial.print("MPU ID is: 0x");
      Serial.print(WHO_AM_I_RESP, HEX);
      Serial.print(", but the correct should be: 0x");
      Serial.println(__read_byte(MPU9250_ADDRESS, WHO_AM_I), HEX);
      delay(1000);
    }
  }

  // Reset the internal registers and restores the default settings.
  __write_byte(MPU9250_ADDRESS, PWR_MGMT_1, 0b10000000);

  delay(100);

  // Activate all sensors
  __write_byte(MPU9250_ADDRESS, PWR_MGMT_2, 0x00);


  //Mag startup
  // Activate bypass to access Magnetometer
  __write_byte(MPU9250_ADDRESS, INT_PIN_CFG, 0x02);

  // Verify ID
  if (__read_byte(AK8963_ADDRESS, WHO_AM_I_AK8963) != WHO_AM_I_RESP_MAG)
  {
    while(1){
      Serial.print("MAG ID is: 0x");
      Serial.print(WHO_AM_I_RESP_MAG, HEX);
      Serial.print(", but the correct should be: 0x");
      Serial.println(__read_byte(AK8963_ADDRESS, WHO_AM_I_AK8963), HEX);
      delay(1000);
    }
  }

  // Read factory ajustment data
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0x00);
  delay(10);
  // Enter Fuse ROM access mode
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0x0F);
  delay(10);

  // Read the x-, y-, and z-axis calibration values
  __read_bytes(AK8963_ADDRESS, AK8963_ASAX, 3);

  // Return x-axis sensitivity adjustment values, etc.
  MagAdjustment[0] =  (float)(Wire.read() - 128)/256. + 1.;
  MagAdjustment[1] =  (float)(Wire.read() - 128)/256. + 1.;
  MagAdjustment[2] =  (float)(Wire.read() - 128)/256. + 1.;
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0x00); // Power down magnetometer
  delay(10);

  // Set magnetometer data resolution and sample ODR
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0b00010110);
  delay(10);

  return 0;
}
```

```
/****************************************
Accel
****************************************/

void MPU9250::get_accel()
{
  __read_bytes(MPU9250_ADDRESS, ACCEL_XOUT_H, 6);

  x = (Wire.read()<<8)|(Wire.read());
  y = (Wire.read()<<8)|(Wire.read());
  z = (Wire.read()<<8)|(Wire.read());
}

void MPU9250::get_accel_g()
{
  get_accel();

  uint8_t range = get_accel_range();
  uint16_t divider;
  if (range == RANGE_16G) divider = 2048;
  if (range == RANGE_8G) divider = 4096;
  if (range == RANGE_4G) divider = 8192;
  if (range == RANGE_2G) divider = 16384;

  x_g = (float)x / divider;
  y_g = (float)y / divider;
  z_g = (float)z / divider;
}

void MPU9250::set_accel_range(accel_range range)
{
  uint8_t reg = __read_byte(MPU9250_ADDRESS, ACCEL_CONFIG);
  __write_byte(MPU9250_ADDRESS, ACCEL_CONFIG, ((reg & 0b11100111)|(range<<3)));
}

accel_range MPU9250::get_accel_range()
{
  /* Read the data format register to preserve bits */
  return (accel_range)((__read_byte(MPU9250_ADDRESS, ACCEL_CONFIG) & 0b00011000)>>3);
}

/****************************************
Gyro
****************************************/

void  MPU9250::get_gyro()
{
  while(!(__read_byte(MPU9250_ADDRESS, INT_STATUS)&0b1));

  __read_bytes(MPU9250_ADDRESS, GYRO_XOUT_H, 6);

  gx = (Wire.read()<<8)|(Wire.read());
  gy = (Wire.read()<<8)|(Wire.read());
  gz = (Wire.read()<<8)|(Wire.read());
}

void MPU9250::get_gyro_d()
{
  get_gyro();

  uint8_t range = get_gyro_range();
  float divider;
  if (range == RANGE_GYRO_2000) divider = 16.4;
  if (range == RANGE_GYRO_1000) divider = 32.8;
  if (range == RANGE_GYRO_500) divider = 65.5;
  if (range == RANGE_GYRO_250) divider = 131.0;
```

```
  uint8_t range = get_gyro_range();
  float divider;
  if (range == RANGE_GYRO_2000) divider = 16.4;
  if (range == RANGE_GYRO_1000) divider = 32.8;
  if (range == RANGE_GYRO_500) divider = 65.5;
  if (range == RANGE_GYRO_250) divider = 131.0;

  gx_d = (float)gx / divider;
  gy_d = (float)gy / divider;
  gz_d = (float)gz / divider;
}

void MPU9250::set_gyro_range(gyro_range range)
{
  uint8_t reg = __read_byte(MPU9250_ADDRESS, GYRO_CONFIG);
  __write_byte(MPU9250_ADDRESS, GYRO_CONFIG, ((reg & 0b11100111)|(range<<3)));
}

gyro_range MPU9250::get_gyro_range()
{
  return (gyro_range)((__read_byte(MPU9250_ADDRESS, ACCEL_CONFIG) & 0b00011000)>>3);
}

/*****************************************
Mag
*****************************************/

int MPU9250::get_mag()
{
  uint8_t temp[7];

  while(!(__read_byte(AK8963_ADDRESS, AK8963_ST1)&0x01));

  __read_bytes(AK8963_ADDRESS, AK8963_XOUT_L, 7);

  for(int i=0; i<=6; i++)
    temp[i] = Wire.read();

  if(!(temp[6] & 0x08)){
    //Data stored as little endian
    mx = (temp[0])|(temp[1]<<8);
    my = (temp[2])|(temp[3]<<8);
    mz = (temp[4])|(temp[5]<<8);
  }
  else
  {
    return 1;
  }
  return 0;
}

int MPU9250::get_mag_t()
{
  if(get_mag()){
    return 1;
  }

  uint8_t range = get_mag_scale();
  float divider;
  if (range == SCALE_14_BITS) divider = 8190.0;
  if (range == SCALE_16_BITS) divider = 32760.0;

  mx_t = (float)mx*4912.0*MagAdjustment[0] / divider;
  my_t = (float)my*4912.0*MagAdjustment[1] / divider;
  mz_t = (float)mz*4912.0*MagAdjustment[2] / divider;

  return 0;
}
```

```
void MPU9250::set_mag_scale(mag_scale scale)
{
  // Save mag configs
  uint8_t temp = __read_byte(AK8963_ADDRESS, AK8963_CNTL) & 0b00000110;
  // disable mag
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0);
  delay(10);
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, (scale << 4) | temp);
}

mag_scale MPU9250::get_mag_scale()
{
  return((__read_byte(AK8963_ADDRESS, AK8963_CNTL)>>4)&0x01);
}

void MPU9250::set_mag_speed(mag_speed mspeed)
{
  // Save mag configs
  uint8_t temp = __read_byte(AK8963_ADDRESS, AK8963_CNTL) & 0b00010000;
  // disable mag
  __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0);
  delay(10);
  if(mspeed==MAG_8_Hz)
    __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0b0010| temp);
  else
    __write_byte(AK8963_ADDRESS, AK8963_CNTL, 0b0110| temp);
}

mag_speed MPU9250::get_mag_speed()
{
  uint8_t temp = __read_byte(AK8963_ADDRESS, AK8963_CNTL)&0x0F;
  if (temp==0b0010)
    return MAG_8_Hz;
  else
    return MAG_100_Hz;
}

/****************************************
Temperature
****************************************/

int16_t MPU9250::get_temp(){
  __read_bytes(MPU9250_ADDRESS, TEMP_OUT_H, 2);

  return ((Wire.read()<<8)|Wire.read());
}

/****************************************
Functions to read and write in I2C
****************************************/
```

```
void MPU9250::__write_byte(uint8_t adress, uint8_t reg, uint8_t value)
{
  Wire.beginTransmission(adress);
  Wire.write(reg);
  Wire.write(value);
  Wire.endTransmission();
}

uint8_t MPU9250::__read_byte(uint8_t adress, uint8_t reg)
{
    #if defined(__SAM3X8E__)
      // For Arduino Due bug
      Wire.requestFrom(adress, 1, reg, 1, true);
    #else
      Wire.beginTransmission(adress);
      Wire.write(reg);
      Wire.endTransmission(false);

      Wire.requestFrom(adress, 1);
    #endif

    if (!Wire.available()){
      return 0;
    }
    return Wire.read();
}

inline void MPU9250::__read_bytes(uint8_t adress, uint8_t reg, int qty){
  #if defined(__SAM3X8E__)
    // For Arduino Due bug
    Wire.requestFrom(adress, qty, reg, 1, true);
  #else
    Wire.beginTransmission(adress);
    Wire.write(reg);
    Wire.endTransmission(false);

    Wire.requestFrom(adress, qty);
  #endif
}
```

MPU9250.cpp

```
#ifndef MPU9250_h
#define MPU9250_h

#include <Wire.h>
#include <arduino.h>

#define MPU9250_ADDRESS  0x68 // Change to 0x69 if AD0 is in high state
#define WHO_AM_I_RESP    0x71// MPU9250 is 0x71 and MPU9255 is 0x73

#define AK8963_ADDRESS   0x0C // MPU9250 must bypass to access the AK8963 in INT_PIN_CFG
#define WHO_AM_I_RESP_MAG 0x48

//Accel and Gyro registers
#define SMPLRT_DIV       0x19
#define CONFIG           0x1A
#define GYRO_CONFIG      0x1B
#define ACCEL_CONFIG     0x1C
#define ACCEL_CONFIG2    0x1D
#define INT_PIN_CFG      0x37
#define INT_ENABLE       0x38
#define INT_STATUS       0x3A
#define ACCEL_XOUT_H     0x3B
#define TEMP_OUT_H       0x41
#define GYRO_XOUT_H      0x43
#define PWR_MGMT_1       0x6B
#define PWR_MGMT_2       0x6C
#define WHO_AM_I         0x75

//Magnetometer Registers
#define WHO_AM_I_AK8963  0x00 // should return 0x48
#define AK8963_ST1       0x02  // data status
#define AK8963_XOUT_L    0x03  // data
#define AK8963_CNTL      0x0A  // Power down (0000), Continuous measurement mode 1 (0010), CMM2 (0110), Fuse ROM (1111) on bits 3:0
#define AK8963_ASAX      0x10  // Fuse ROM x-axis sensitivity adjustment value
#define AK8963_ASAY      0x11  // Fuse ROM y-axis sensitivity adjustment value
#define AK8963_ASAZ      0x12  // Fuse ROM z-axis sensitivity adjustment value

typedef enum
{
  RANGE_16G          = 0b11,
  RANGE_8G           = 0b10,
  RANGE_4G           = 0b01,
  RANGE_2G           = 0b00
} accel_range;

typedef enum
{
  RANGE_GYRO_2000    = 0b11,
  RANGE_GYRO_1000    = 0b10,
  RANGE_GYRO_500     = 0b01,
  RANGE_GYRO_250     = 0b00
} gyro_range;

typedef enum
{
  SCALE_14_BITS      = 0,
  SCALE_16_BITS      = 1
} mag_scale;

typedef enum
{
  MAG_8_Hz           = 0,
  MAG_100_Hz         = 1
} mag_speed;
```

```
class MPU9250
{
  public:
    uint8_t begin();

    //Accel
    void get_accel();
    void get_accel_g();
    int16_t x, y, z;
    float x_g, y_g, z_g;
    void set_accel_range(accel_range range);
    accel_range get_accel_range();

    //Gyro
    void get_gyro();
    void get_gyro_d();
    int16_t gx, gy, gz;
    float gx_d, gy_d, gz_d;
    void set_gyro_range(gyro_range range);
    gyro_range get_gyro_range();

    //Mag
    int get_mag();
    int get_mag_t();
    int16_t mx, my, mz;
    float mx_t, my_t, mz_t;
    void set_mag_scale(mag_scale scale);
    mag_scale get_mag_scale();
    void set_mag_speed(mag_speed mspeed);
    mag_speed get_mag_speed();
    float MagAdjustment[3] = {0, 0, 0};

    //Temp
    int16_t get_temp();

  private:
    void __write_byte(uint8_t adress, uint8_t reg, uint8_t value);
    uint8_t __read_byte(uint8_t adress, uint8_t reg);
    inline void __read_bytes(uint8_t adress, uint8_t reg, int qty);
};


#endif
```

MPU9250.h

```
void SD_logg() {

  String dataString = "";

  dataString += String(millis() / 1000); //här skrivs data in, lägg till tid, hastighet, riktning, position, spänning, SOC, etc.
  dataString += "\t";//gör en toggle mellan varje värde
  dataString += String(timmar);
  dataString += ':';
  dataString += String(minuter);
  dataString += ':';
  dataString += String(sekunder);
  dataString += "\t";
  dataString += String(flag);
  dataString += "\t";
  dataString += String(Amps1);
  dataString += "\t";
  dataString += String(Amps2);
  dataString += "\t";
  dataString += String(Amps3);
  dataString += "\t";
  dataString += String(voltageA);
  dataString += "\t";
  dataString += String(voltageB);
  dataString += "\t";
  dataString += String(temperatur1);
  dataString += "\t";
  dataString += String(temperatur2);
  dataString += "\t";
  dataString += String(fuktsensor1);
  dataString += "\t";
  dataString += String(fuktsensor2);
  dataString += "\t";
  dataString += String(gsm.On);
  dataString += "\t";
  dataString += String(gsm.Fix);
  dataString += "\t";
  dataString += String(gsm.satt); //satteliter som syns
  dataString += "\t";
  dataString += String(quality);  //i procent 0 är max
  dataString += "\t";
  dataString += String(gsm.Altitude);
  dataString += "\t";
  dataString += String(gsm.Speed);
  dataString += "\t";
  dataString += String(gsm.Course);
  dataString += "\t";
  dataString += String(gsm.HDOP); //multipliceras med 7 för att få nogrannhet
  dataString += "\t";
  dataString += String(gsm.Latitude);
  dataString += "\t";
  dataString += String(gsm.Longitude);







  File dataFile = SD.open("datalog.txt", FILE_WRITE);
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    Serial.println(dataString);  //skriv ut datan i monitor
  }
  else {
    //Serial.println("error opening datalog.txt"); //om komunikationen slutar
    Serial.println(dataString);  //skriv ut datan i monitor

  }
}
```

SD_logg()

```
void SD_logg_setup() {
  String dataString = "";

  dataString += String("tss"); //tid sedan start
  dataString += "\t";//gör en toggle mellan varje värde
  dataString += String("tid");
  dataString += "\t";
  dataString += "\t";
  dataString += String("flag");
  dataString += "\t";
  dataString += String("ström1");
  dataString += "\t";
  dataString += String("ström2");
  dataString += "\t";
  dataString += String("ström3");
  dataString += "\t";
  dataString += String("spännA");
  dataString += "\t";
  dataString += String("spännB");
  dataString += "\t";
  dataString += String("temp1");
  dataString += "\t";
  dataString += String("temp2");
  dataString += "\t";
  dataString += String("fukt1");
  dataString += "\t";
  dataString += String("fukt2");
  dataString += "\t";
  dataString += String("On");
  dataString += "\t";
  dataString += String("Fix");
  dataString += "\t";
  dataString += String("sattel");
  dataString += "\t";
  dataString += String("gsm");
  dataString += "\t";
  dataString += String("Alt");
  dataString += "\t";
  dataString += String("Speed");
  dataString += "\t";
  dataString += String("Course");
  dataString += "\t";
  dataString += String("HDOP");
  dataString += "\t";
  dataString += String("Lat");
  dataString += "\t";
  dataString += "\t";
  dataString += String("Lon");
  dataString += "\t";
  dataString += "\t";
```

```
  File dataFile = SD.open("datalog.txt", FILE_WRITE);
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    Serial.println(dataString);  //skriv ut datan i monitor
  }
  else {
    Serial.println("error opening datalog.txt"); //om komunikationen slutar
    Serial.println(dataString);  //skriv ut datan i monitor
  }


    dataString = " ";           // nollar stringen och skriver ut datum vid uppstart

      dataString += String(ar);
      dataString += '-';
      dataString += String(manad);
      dataString += '-';
      dataString += String(dag);

      dataFile = SD.open("datalog.txt", FILE_WRITE);
      if (dataFile) {
        dataFile.println(dataString);
        dataFile.close();
        Serial.println(dataString);
      }
      else {
        Serial.println("error opening datalog.txt"); //om komunikationen slutar
  }
}
```

SD_setup()

```
//Roll Pitch variables
double fXg = 0, fYg = 0, fZg = 0;
double pitch, roll, Xg, Yg, Zg;
const float alpha = 0.5;

//Compass vars
double m_x, m_y, m_z;
double comp_raw;
int filter_var = 0;
double filter_buffer = 0;
int lvl = 0;
double compass = 1;
float short_buffer;

float get_compass() {

    mpu.get_accel_g();

    //Low Pass Filter
    fXg = mpu.x_g * alpha + (fXg * (1.0 - alpha));
    fYg = mpu.y_g * alpha + (fYg * (1.0 - alpha));
    fZg = mpu.z_g * alpha + (fZg * (1.0 - alpha));

    //Roll & Pitch Equations
    roll  = (atan2(-fYg, fZg)*180.0)/M_PI;
    pitch = (atan2(fXg, sqrt(fYg*fYg + fZg*fZg))*180.0)/M_PI;


if (( -lvl_max < pitch && pitch < lvl_max) && ( -lvl_max < roll && roll < lvl_max )){
    lvl = 0;
    filter_buffer = 0;
    filter_var = 0;
    short_buffer = 0;
    while(filter_var < l_pass_loop){
      mpu.get_mag_t();

      m_x = (mpu.mx_t - offset_x) / scale_x;
      m_y = (mpu.my_t - offset_y) / scale_y;
      m_z = (mpu.mz_t - offset_z) / scale_z;

      comp_raw = geo_comp + 180 + (180/PI) * atan2(m_y, -m_x);
      if (comp_raw>360) comp_raw -= 180;
      if (comp_raw<0) comp_raw += 180;
      if (filter_var>0){
      if ((short_buffer-comp_raw) > 350 || (short_buffer-comp_raw) < - 350) return 0.5;
      }
      short_buffer = comp_raw;
      filter_buffer = filter_buffer + (360 + comp_raw);

      filter_var ++;
    }

    if (filter_var>0){
    compass = (filter_buffer/filter_var) - 360;
    }
  return compass;
  }
  else return -1;

}
```

Compass()

```
void cppm_cycle(void) //hämtar data från radiomottagaren
{
  if (rx.gotNewFrame()) {

    uint16_t rcData[NCHAN];

    rx.computeRC(rcData);

    aile = map(rcData[0],1050,1900,-100,100);
    elev = map(rcData[1],1050,1900,-100,100);
    thro = map(rcData[2],1050,1900,-100,100);
    rudd = map(rcData[3],1050,1900,-100,100);
    gear = map(rcData[4],1050,1900,-100,100);
    aux1 = map(rcData[5],1050,1900,-100,100);

  //  Serial.println(aux1);

  }
}
```

Cppm cycle()

```
void currentSensor(){

double mVperAmp = -98; // verkar vara lite olinjär, men är justering för invidell sesnor
double RawValue= 0;
double ACSoffset = 2500;
double Voltage = 0;

 RawValue = analogRead(current1);
 Voltage = (RawValue / 1024.0) * 5000; //läser ut ström 1
 Amps1 = ((Voltage - ACSoffset) / mVperAmp);

 RawValue = analogRead(current2);
 Voltage = (RawValue / 1024.0) * 5000; //läser ut ström 2
 Amps2 = ((Voltage - ACSoffset) / mVperAmp);

 RawValue = analogRead(current3);
 Voltage = (RawValue / 1024.0) * 5000; //läser ut ström 3
 Amps3 = ((Voltage - ACSoffset) / (mVperAmp*-1));

 if(Amps2 > 10){
  flag = '5';
  //send_info();
  }


// Serial.print("Raw Value = " ); // shows pre-scaled value
// Serial.print(RawValue);
// Serial.print("\t mV = "); // shows the voltage measured
// Serial.print(Voltage,3); // the '3' after voltage allows you to display 3 digits after decimal point
// Serial.print("\t Amps = "); // shows the voltage measured
// Serial.print(Amps1,3); // the '3' after voltage allows you to display 3 digits after decimal point
// Serial.println(Amps2,3); // the '3' after voltage allows you to display 3 digits after decimal point
// delay(1000);

  }
```

currentSesnor()

```
#define max_speed 0
#define norm_speed -10
#define min_speed -50
#define max_motor_temp 90 //högst 90 grader på motorn
#define max_bat_temp 90 // högst 80 grader på batteriet
#define drift_max 0.5 // om nästa mål/2 och närmare än nuvarande
#define MAX_DIST 15 // Hur nära båten måste vara nära målet för att gå till nästa i m

#define s_left -40
#define l_left -80
#define s_right 40
#define l_right 80


//Kompassdefines
#define geo_comp 4.0 // för att finjustera vinkeln, bör egentligen vara på +4
#define lvl_max 5.0 // Max/min degrees roll & pitch för att få bra compassdata
#define l_pass_loop 4 // hur många läsningar compassen gör, tar sedan medel

////Mag calibration för göteborg
//#define offset_x -2.87
//#define offset_y -18.10
//#define offset_z 5.24
//
//#define scale_x 47.32
//#define scale_y 46.33
//#define scale_z 47.82

//Mag calibration för stenaline
#define offset_x -0.72
#define offset_y -17.74
#define offset_z 5.93

#define scale_x 26.53
#define scale_y 27.87
#define scale_z 27.93



struct GPS_data {
  String On;  //gps påslagen
  String Fix; //position låst
  String Time;
  String Latitude;
  String Longitude;
  String Altitude;
  String Speed;
  String Course;
  String HDOP;  //precition
  String satt;
};
```

```
struct GSM_data {
  char On[2];
  char Fix[2];
  char Time[19];
  char Latitude[11];
  char Longitude[12];
  char Altitude[9];
  char Speed[7];
  char Course[7];
  char HDOP[5];
  char satt[2];
};

struct GSM_data_float {
  float On;
  float Fix;
  float Time;
  float Latitude;
  float Longitude;
  float Altitude;
  float Speed;
  float Course;
  float HDOP;
  float satt;
};
```

Defines.h

```
void fukt_sensor() {

  fuktsensor1 = analogRead(fukt1);
  fuktsensor2 = analogRead(fukt2);
  if (fukt1 < 500 || fukt2 < 500)
    flag = '1';
    //send_info();
  else
    flag = 0;
}
```

Fukt_sesnor()

```
void get_gps() {

  SIM900.println("AT+CGNSINF");
  delay(30);
  hoppa_over_rader(2);


  GPS.On = inData.substring(10, 11);    //läser ut On rått itsället då det följer med massa skit annars
  GPS.Fix = getValue(inData, ',', 1);
  GPS.Time = getValue(inData, ',', 2);
  GPS.Latitude = getValue(inData, ',', 3);
  GPS.Longitude = getValue(inData, ',', 4);
  GPS.Altitude = getValue(inData, ',', 5);
  GPS.Speed = getValue(inData, ',', 6);
  GPS.Course = getValue(inData, ',', 7);
  GPS.HDOP = getValue(inData, ',', 10);
  GPS.satt = getValue(inData, ',', 15);


  GPS.On.toCharArray(gsm.On, 5);  //konverterar string till char för att kunna skicka sms
  GPS.Fix.toCharArray(gsm.Fix, 2);
  GPS.Time.toCharArray(gsm.Time, 19);
  GPS.Latitude.toCharArray(gsm.Latitude, 11);
  GPS.Longitude.toCharArray(gsm.Longitude, 12);
  GPS.Altitude.toCharArray(gsm.Altitude, 9);
  GPS.Speed.toCharArray(gsm.Speed, 7);
  GPS.Course.toCharArray(gsm.Course, 7);
  GPS.HDOP.toCharArray(gsm.HDOP, 5);
  GPS.satt.toCharArray(gsm.satt, 3);

  gsm_float.On = atof(gsm.On);  //konveterar till float så att niels är nöjd
  gsm_float.Fix = atof(gsm.Fix);
  gsm_float.Time = atof(gsm.Time);
  gsm_float.Latitude = atof(gsm.Latitude);
  gsm_float.Longitude = atof(gsm.Longitude);
  gsm_float.Altitude = atof(gsm.Altitude);
  gsm_float.Speed = atof(gsm.Speed);
  gsm_float.Course = atof(gsm.Course);
  gsm_float.HDOP = atof(gsm.HDOP);
  gsm_float.satt = atof(gsm.satt);

  hoppa_over_rader(2);

  //signal_quality(); //hämtar signal kvaliteten --tar en massa tid
  tid_splitt(); // delar upp time i dess enklaste delar

}
```

```
// https://stackoverflow.com/questions/9072320/split-string-into-string-array
String getValue(String data, char separator, int index) //plockar ut stings mellan ',' tecken
{
  int found = 0;
  int strIndex[] = {0, -1};
  int maxIndex = data.length() - 1;

  for (int i = 0; i <= maxIndex && found <= index; i++) {
    if (data.charAt(i) == separator || i == maxIndex) {
      found++;
      strIndex[0] = strIndex[1] + 1;
      strIndex[1] = (i == maxIndex) ? i + 1 : i;
    }
  }
  return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}

void tid_splitt() {

  ar = GPS.Time.substring(0, 4);
  manad = GPS.Time.substring(4, 6);
  dag = GPS.Time.substring(6, 8);
  timmar = GPS.Time.substring(8, 10);
  minuter = GPS.Time.substring(10, 12);
  sekunder = GPS.Time.substring(12, 14);


}
```

Get_gps()

```
void send_info() {
//  if (timer2(0) > 18000)  { //inte testat men ser till att inte skcika sms mer än var 5:e minut
  static char buff[80];

  SIM900.println("AT+CMGF=1");
  delay(30);
  SIM900.println("AT+CMGS=\"" + number + "\"\r"); //AT kommandon för att skcia ett sms
  delay(30);
  sprintf(buff, "%s%s%s%s%s%s%s%s%s", "https://www.google.com.hk/maps?q=N", gsm.Latitude, ",E", gsm.Longitude, "  hastighet:", gsm.Speed, "km/h"," felkod: ",flag);
  SIM900.println(buff);
  delay(30);
  SIM900.println((char)26);// ASCII code of CTRL+Z
  delay(30);
  hoppa_over_rader(3);
  Serial.println("sms skickat");
  timer2(1);
//  }
}

void signal_quality() { //tar ungefär 1s --använd bara om absolut nödvändigt!

  SIM900.println("AT+CSQ");
  delay(30);
  hoppa_over_rader(1);
  quality = SIM900.readStringUntil('\n');
  quality = quality.substring(6,10);
  hoppa_over_rader(3);


}
```

Send_info()

```
void set_motor_speed( int throttle){  //motor hastighet -100->100

   throttle = map(throttle,-100,100,750,2250); //de två sista värderna justeras mellan 750 och 2250 för hastighet på motor
   motor.write(throttle);

  }
```

Set_motor_speed()

```
static float R = 6371e3; // metres
float h_abs;
float h_factor;
float corrected_setpoint;

//Tar in två gps kordinater och ger riktningen från punk 1 till 2 i grader satt från Nord
float brng_p2p (float lat_1, float lon_1, float lat_2, float lon_2){
  float fi_1 = lat_1 * PI / 180;
  float fi_2 = lat_2 * PI / 180;
  float landa_1 = lon_1* PI / 180;
  float landa_2 = lon_2* PI / 180;

  float y = sin(landa_2-landa_1) * cos(fi_2);
  float x = cos(fi_1) * sin(fi_2) -
            sin(fi_1) * cos(fi_2) * cos(landa_2-landa_1);
  double brng = atan2(y,x) * 180 / PI;
  brng = fmod(brng+360,360);

  return brng;
}

//Tar in två gps kordinater och ger längden från punk 1 till 2 i km, kurvatur av jord inräknad
float dist_p2p (float lat_1, float lon_1, float lat_2, float lon_2){
  float fi_1 = lat_1 * PI / 180;
  float fi_2 = lat_2 * PI / 180;
  float delta_fi = (lat_2-lat_1)* PI / 180;
  float delta_landa = (lon_2-lon_1)* PI / 180;

  float a = sin(delta_fi/2) * sin(delta_fi/2) +
        cos(fi_1) * cos(fi_2) *
        sin(delta_landa/2) * sin(delta_landa/2);
  float c = 2 * atan2(sqrt(a), sqrt(1-a));

  float d = R * c;
  return d;
}

float get_setpoint(float h, float brng_boat_point, float brng_true){
  corrected_setpoint = brng_boat_point;

  h_abs = abs(h);
  h_factor = h_abs*900;

  if (h>=0){
    corrected_setpoint = brng_boat_point - h_factor;
    if (abs(corrected_setpoint) > (90-abs(brng_boat_point))) corrected_setpoint =  (brng_true - 90) ;
  }

  else{
    corrected_setpoint = brng_boat_point + h_factor;
    if (abs(corrected_setpoint)> (90-abs(brng_boat_point))) corrected_setpoint =  (brng_true + 90) ;
  }

 if (corrected_setpoint>360) corrected_setpoint = corrected_setpoint - 360;
 if (corrected_setpoint<0) corrected_setpoint = corrected_setpoint + 360;

 return corrected_setpoint;

}
```

Nav_funcs

```
void set_servo_roder(int input){

  input = map(input,-100,100,1600,900); //de två sista värderna justeras mellan 750 och 2250 för ändlägesvinkel på servo
  servo.write(input);
  //Serial.println(input);
  delay(15); //kanske  behövs om servo skakar?


  }
```

Set_servo_speed()

```
//kollar vilken kurs båten borde ha och kompenserar för avvikelse från färd linje
// Räknar ut vilken vinkel rodret
//
void set_rudder () {

  setpoint = get_setpoint (dist_to_true_vector, brng_boat_point, brng_true);

  rudder_angle = rudder_reg (); // Reglersystemet för insvänging av båten

  set_servo_roder(rudder_angle);
}



//reglersystemet för insvängning
int rudder_reg (){

float temp = 0;

temp = setpoint - boat_brng;

temp = map(temp,-360,360,-100,100);

return temp;

}
```

Set_rudder()

```
// Implementera josefs funktion som hetter motor_speed, döp om till set_motor_speed
// Setspeed kollar på batteriets laddning för att sedan välja hastighet.
// När batteriet är i mitten av laddningen kollas medelhastighet och avvikelse till position.
// Om temp når max tvingas motorn gå på lågvarv
void set_speed () {
  if (SoC>0.75) speed_case = 1;
  else if (0.05 <= SoC < 0.15) speed_case = 2;
  else if (SoC < 0.05) speed_case = 4;
  else speed_case = 3;

  if (temperatur1 > max_motor_temp) speed_case = 2;

  switch (speed_case){
    case 1:
        motor_speed = max_speed;
        break;

    case 2:
        motor_speed = min_speed;
        break;

    case 3:
        motor_speed = norm_speed;

//        motor_speed =  norm_speed + (abs(dist_to_true_vector));
//
//        if (motor_speed>100) motor_speed = 100;
        break;

    case 4:
        motor_speed = -100;
        break;
  }
  set_motor_speed (motor_speed);
}
```

Set_speed()

```
void sim_setup(){
  pinMode(A_key, OUTPUT);
  digitalWrite(A_key,HIGH);
  delay(1000);
  digitalWrite(A_key,LOW);

  SIM900.begin(9600); // sätt bundrate på sim till 9600
  delay(1000); // låt modulen starta
  SIM900.println("AT+IPR=9600");  //skriver till modulen att den ska jobba på 9600
  delay(30);
  hoppa_over_rader(2);
  SIM900.println("AT+CGNSPWR=1"); // slår på GPSen
  delay(30);
  hoppa_over_rader(2);


//    String Latitude = "57.84528"; //testvärldedn
//    String Longitude = "12.96771";
//    String velocity = "13.37";
//
//    Latitude.toCharArray(gsm.Latitude,11);
//    Longitude.toCharArray(gsm.Longitude,12);
//    velocity.toCharArray(gsm.Speed,12);

  }

  void hoppa_over_rader(int antal_rader) {
  for (int i = antal_rader; i > 0; i--) {
    inData = SIM900.readStringUntil('\n');
  }
}
```

Sim_setup

```
float temp() {

   temperatur1 = analogRead(temp1); //läser ut temp, dock verkar en analog pinne på arduinon vara sönder
   temperatur2 = analogRead(temp2);

  //Serial.println(temperatur1);

  temperatur1 = ( temperatur1 / 1024.0) * 5000;
  temperatur1 = temperatur1 / 10;

  temperatur2 = ( temperatur2 / 1024.0) * 5000;
  temperatur2 = temperatur2 / 10;

   if(temperatur1 > 50 || temperatur2 > 50){
  flag = '2';
  //send_info();
  }




}
```

Temp()

```
boolean timer(int interval){  //kan användas för att blinka led utan delay

static int currentTime = 1; //static för att komma ihåg värdet utanför funktionen
static int previousTime = 0;
currentTime = millis();
  if(currentTime - previousTime >= interval){
    previousTime = currentTime;
    return(1);
  }
  else
    return 0;
}
```

Timer()

```
//är målet inom 100m? isf uppdateras målet
// Kollar om det är närmare till nästa mål än nuvarande
// retunerar 1 om nytt mål hittat
// 0 Om samma mål är aktivt
int update_goal (){
if (checkpoint_nr < total_checkpoints){
   //if ((distance_boat_point < MAX_DIST) || (distance_boat_point > (drift_max * dist_p2p(testLatitude, testLongitude, target_lat[checkpoint_nr+1], target_long[checkpoint_nr+1]))))
   if ((distance_boat_point < MAX_DIST) )
    {
      checkpoint_nr++;
      return 1;
    }

    else {
      return 0;
    }

    }
if (checkpoint_nr == total_checkpoints && (distance_boat_point < MAX_DIST / 2)) return 2;
}
```

Update_goal()

```
void voltage_sensor() {
  voltageA = analogRead(V_A);
  voltageA = voltageA/59.8; //scaler som kalibreras
  voltageB = analogRead(V_B);
  voltageB = voltageB/60.6; //scaler som kalibreras

  if(voltageA < 9.3){
  flag = '4';
  //send_info();
  }
}
```

Voltage_sesnor()