Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

JONI JUVONEN

Automatisoitu autopesuri

SÄHKÖ- JA AUTOMAATIOTEKNIIKAN
TUTKINTO-OHJELMA
2022

Tekijä(t)
Juvonen, Joni

Julkaisun laji
Opinnäytetyö, AMK

Päivämäärä
Syyskuu 2022

Sivumäärä
38

Julkaisun kieli
Suomi

Julkaisun nimi
Automatisoitu autopesuri

Tutkinto-ohjelma
Sähkö- ja automaatiotekniikka

Opinnäytetyössä tavoitteena oli luoda automaattisesti toimiva autopesuri mahdollisimman kustannustehokkaasti sekä mahdollistaen osien jälkikäyttö muihin projekteihin. Idea opinnäytetyöhön lähti halusta toteuttaa pieni automaatiojärjestelmä siihen kuuluvin toiminnoin. Tarkoituksena oli myös käyttää mahdollisimman paljon koulussa opittuja taitoja 3D- teknologiassa, sähkötekniikassa sekä ohjelmoinnissa.
Työn tuloksena saatiin aikaiseksi toimiva prototyyppi käyttökelpoisesta autopesurista. Alhaisten kustannusten vuoksi pesuriin jäi parannuskohteita, mutta toiminnaltaan sekä periaatteeltaan se vastasi täysin suunniteltua laitetta. Pesuriin sisällytettiin vaadittavat toiminnot sekä kaksi pesuohjelmaa, joista käyttäjä sai itse valita mieluisan. Pesuriin liitetty automaatiokeskus onnistuttiin myös rakentamaan uusiokäyttöiseksi. Lisäksi rautakehikko säilytti käyttöarvonsa muissa tarkoituksissa.

Työn rakennus toteutettiin kokonaisuudessaan omissa tiloissa, omilla materiaaleilla sekä työkaluilla.

Avainsanat
Arduino, 3D-tulostus, ohjelmointi, koodaus

Author(s) Juvonen, Joni	Type of Publication Bachelor's thesis	Date September 2022
	Number of pages 38	Language of publication: Finnish

Title of publication
Automatic Car Washing Machine

Degree programme
Electrical and Automation Engineering

Purpose of this thesis was to create cost-effective and reusable self-acting car washing machine. Thesis was based on desire to build tiny automatical system with all needed functions. It was important to include abilities from 3D-technology, electrics and programming.
As a result of this thesis, operational prototype from the professional car washing machine was created. Because of the low costs, improvements should be made for better result. Functionality and theory was successful if compared to plan. All needed functionalities were included, for example two washing program. Control unit was reusable with 3D-printed connectors and frame of the machine was usable still after project.

Washing machine was executed in private garage using own materials and tools.

Keywords
Arduino, 3D-printing, programming, coding

ALKUSANAT

Tämä opinnäytetyö suoritettiin suuresta kiinnostuksesta automaatiotekniikkaa kohtaan. Opinnäytetyön päätarkoituksena oli päästä rakentamaan pieni automaatiojärjestelmä sekä tutustumaan käytännön ongelmiin projektin toteutuksen aikana.

Kiitos Joonas Kortelaiselle 3D-tulostimien käytöstä SAMK:n tiloissa. Koulun tulostimien käyttö nopeutti projektin toteutusta huomattavasti. Lisäksi kiitokset kuuluu Hannu Asmalalle opinnäytetyön ohjauksesta sekä avustuksesta projektissa.

Työn aihe tukee tulevaisuutta automaatiouran parissa. Projektin aikana hahmotuskyky kehittyi aivan uudelle tasolle sekä kyky kirjoittaa rivikoodia sekä ratkoa sen tuomia ongelmatilanteita. Projektin aikana mielenkiinto uravalintaa kohtaan kasvoi entisestään.

SISÄLLYS

1 JOHDANTO

Opinnäytetyön tarkoituksena on rakentaa automaattisesti toimiva autopesuri mahdollisimman kustannustehokkaasti sekä käyttäen apuna 3D-tulostusta että kuluttajaystävällisiä sähkökomponentteja. Projektissa luodaan pieni automaatiojärjestelmä, jonka toiminnot tulee olla helposti muokattavissa Arduino IDE- ohjelmiston avulla. Projektiin kuuluva automaatiokeskus tulee olla käyttökelpoinen myös muihin käyttötarkoituksiin projektin jälkeen, jolloin johdotus sekä toiminnallisuus eivät saa olla kiinteitä. Opinnäytetyön aihe valittiin, jotta pystyttäisiin hahmottamaan kykyä suunnitella sekä luoda autojärjestelmä vaadittavin toiminnoin. Aihe on entuudestaan kiinnostava sekä uudelleenkäytettävyyden vuoksi täysin toteutettavissa. Opinnäytetyö toteutetaan itselle oppimistarkoituksessa ja sen toteutuspaikkana toimii omat työtilat, jotka myös mahdollistavat työkalut ja projektin toteuttamisen.
Aion kertoa lyhyesti opinnäytetyön eri osa-alueista, joista perehdyn tarkemmin ohjelmointiin ja koodaukseen. Kerron myös lyhyesti kehikon rakentamisesta sekä suunnittelusta, 3D- tulostuksesta sekä mallien suunnittelusta ja sähkösuunnittelusta.

Projektissa käytettävistä tiedoista ja taidoista automaatioalaan sekä 3D- teknologiaan liittyvät on ansaittu opintojen aikana, kuitenkin täydentäen tietämystä internetistä kun työ on sitä vaatinut. Huomattava määrä projektin tietotaidosta nojautuu kuitenkin sähkö-, automaatio-, sekä koodaustietoihin. Opintojen aikana eniten on käsitelty automaatioon liittyviä asioita, kun taas rivikoodaaminen on jäänyt vähemmälle automaatioon suuntautuneilla. Automaatio-opinnoista on ohjelmoinnin lisäksi hyötyä myös osavalintoihin, jolloin osavalinnat, yhteensopivuudet sekä kokonaishahmottaminen helpottuu opintojen ansiosta.

Projektin rakentamisesta tulee varmasti hyötymään myös tulevaisuudessa sekä alan työelämässä. Projektiin kuuluu laajalti suunnittelua, erilaisten ongelmien ratkaisua, sekä se kehittää yleistä suunnittelukykyä. Siihen sisältyy vastaavanlaisia kokonaisuuksia, kuin muihinkin suurempiin automaatioprojekteihin. Lisäksi projektissa joutuu

soveltamaan sekä kehittämään ratkaisuja, jotka vaikuttavat projektin kustannuksiin ja toteutukseen. Itse projektin toteutus alkoi vuonna 2021 syyskuussa. Aikataulutus ja projektin jäsentely olivat suuressa roolissa, koska ei ollut entuudestaan kokemusta vastaavanlaisesta projektista.

Työn tärkeimmät tiedonlähteet ovat erilaiset ohjekirjat, Arduinon omat nettisivut, keskustelupalstat koodausideoiden etsintään sekä omat testaukset ja niiden raportit. Projektissa suunnittelu on itse toteutettua ja suunnittelun aikana ohjelmistoina ovat toimineet Arduino IDE, SolidWorks, FluidSIM 5 sekä sovittelussa PrusaSlicer. Erilaisista versioista, ideoista, toteutuksista sekä rakennusvaiheista löytyy kuvia liitteistä ja materiaalista. Liitteistä löytyy myös kuvia pesurin toiminnasta, käyttöliittymästä sekä rakennusvaiheen ohjelmoinneista.

2 PROJEKTIN OSA-ALUEET

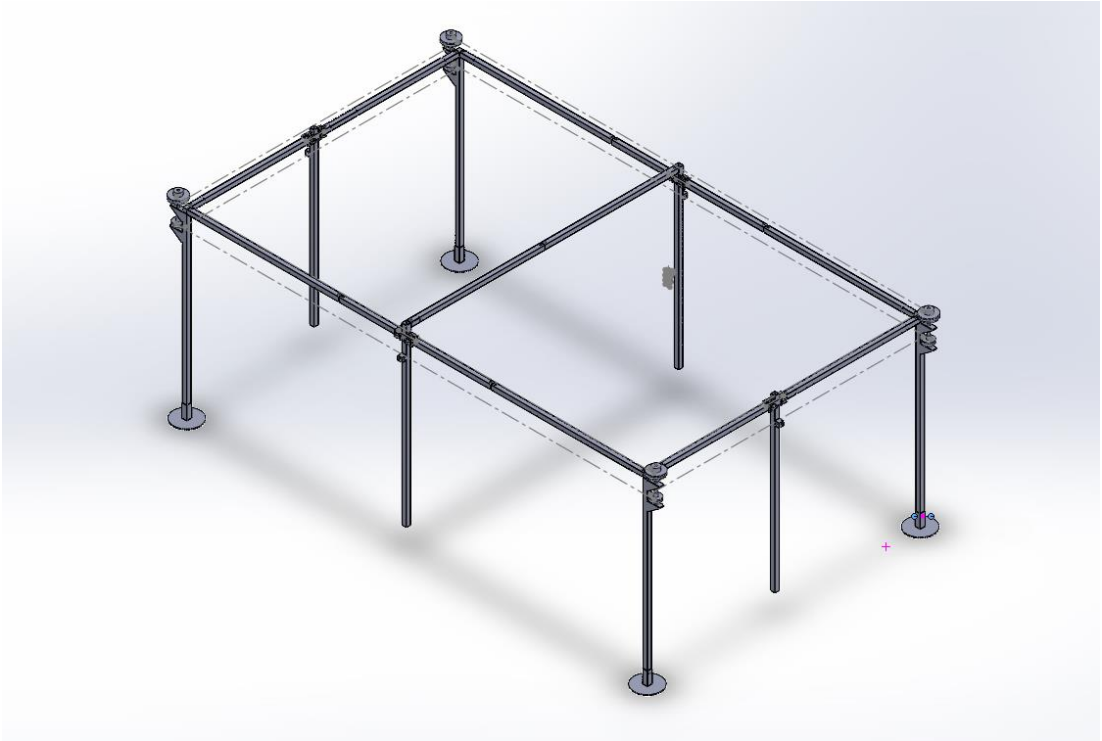
Projektin toteutus alkoi mekaanisella hahmottelulla ja suunnittelulla. Suunnittelun tuloksena syntyi SolidWorks -malli projektissa käytetystä kehikosta. Mallin jälkeen alkoi varsinainen mekaaninen toteutus, mihin kuului itse kehikon lisäksi myös pesukelkat sekä niiden liikuttamiseen vaadittavat komponentit.
Mekaanisen osion jälkeen alkoi itse keskuksen valmistus. Keskukseen tuli 3D -tulostetut 2-napaiset liittimet rajakytkimille sekä moottoreiden ohjaukselle. Lisäksi keskuksen kanteen liitettiin pieni käyttöliittymä kolmella ohjauspainikkeella, joiden avulla sai valittua pesuohjelman, aloitettua sekä keskeytettyä sen. Keskuksen syöttö oli toteutettu 3 vaiheisella kaapelilla, jonka yhdestä vaiheesta otettiin syöttö keskuksen 10A

virtalähteelle. Lopuksi vaiheet menivät kontaktorin kautta painepesurille, joka tuotti paineen vesijärjestelmään.
Keskuksen valmistuttua tuli asentaa kaapelit sekä johtimet pesukelkoille sekä rajakytkimille. Rajakytkimiä tuli vaakasuuntaan vain kaksi, koska vaakasuuntainen liike oli toteutettu yhdellä moottorilla, joihin kaikki kelkat olivat kytköksissä. Jokaisella pystykelkalla oli kuitenkin omat rajakytkimensä. Kaapelit liitettiin keskuksen 2-napaisiin liittimiin.
Viimeisenä vaiheena ennen testausta toteutettiin vesijärjestelmä. Painepesuri jakoi paineen kehikon kahteen kulmaan T-paloille, joista syötettiin vettä itse kelkan suuttimille. Suuttimet sijaitsivat pystykelkoissa, 3D -tulostetussa telineessä. Toiseen pesuohjelmaan sisältyi myös pesuaineen käyttö. Tämä oli toteutettu painepesurissa olevan pesuaine pumpun avulla, jolloin pesuaineen saantia sääteli solenoidiventtiili. Solenoidiventtiilille oli erillinen läpimeno keskukseen syöttökaapeleiden vieressä.

2.1 Mekaniikka

SolidWorks -ohjelmistolla suunnitellun mallin pohjalta aloitettiin itse kehikon rakennus. Kehikon mitat oli määritelty etukäteen niin, että henkilöauto mahtuu sisään vaivatta, eikä autosta ulospääsy ollut hankalaa.
Kehikon ulkomitat:

- Pituus : 5000mm
- Leveys: 2500mm
- Korkeus : 2100mm



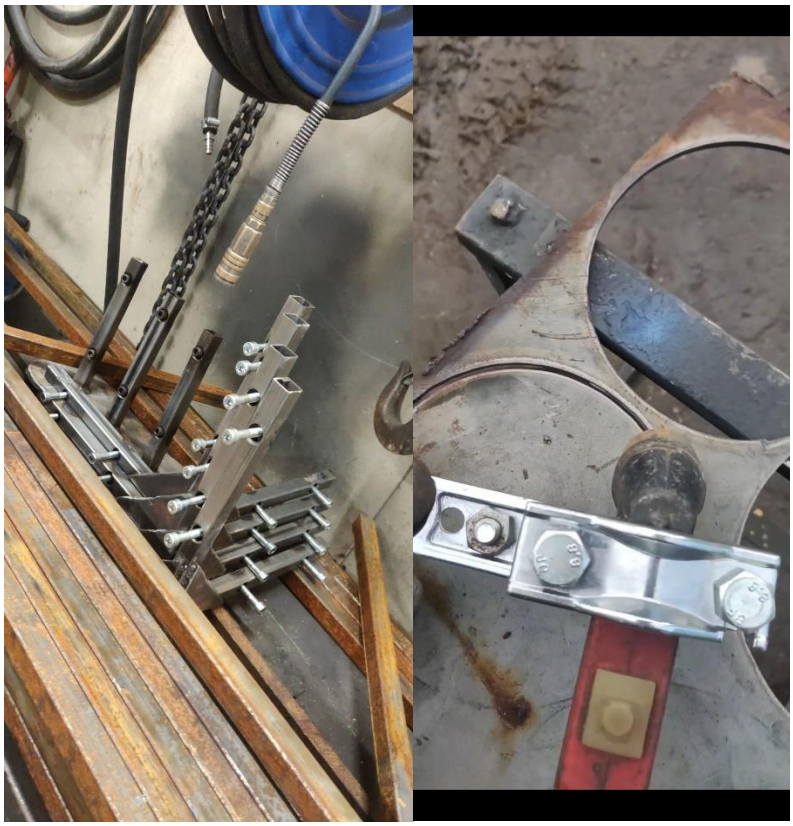
Kuva 1. Mallinnettu SolidWorks -ohjelmistolla. Mallinnus helpotti projektivaiheen suunnittelua sekä kokonaisuuden luomista.

Materiaaliksi kehikolle valittiin 25x25 (mm) rautaprofiili 2.5mm seinämällä. Materiaalivalinta ei ollut kehikolle optimaalisin sen painon sekä pienen ulkomitan vuoksi, mutta se tuki projektin toimintatapaa kustannuksien kannalta. Muussa tilanteessa kehikon materiaaliksi olisi ollut syytä valita suurempi alumiiniprofiili.

Kehikon tuli olla helposti uudelleen pystytettävissä sekä purettavissa mahdollisimman pieneen tilaan. Tavoitteena oli, että kehikko mahtuisi keskimäärin 2 x 1 (m) lattia-alalle. Tällöin kehikon pisimmät komponentit olivat kahden metrin mittaisia. Komponenttien kiinnitys toisiinsa toteutettiin tekemällä niiden sisälle pienempi kiinnityskappale, joka yhdistäisi runkokomponentit toisiinsa. Tärkeää oli huomioida kelkan liikkuminen runkokomponenttien päällä, jolloin kiristyspultit eivät saaneet jäädä kulkupinnan yläpuolelle.

Kuva 2. Pesurin rungon yhdistyspaloista sekä yhdistyksestä: ei kelkan liikettä häiritsevää pultin kantaa.

Lisäksi kehikon kulmiin tuli rakentaa kulmapalat, joihin kiinnitettäisiin myöhemmin myös kelkkoja liikuttavat vaijerit sekä hihnapyörät. Kehikon jaloille valmistettiin myös pienet tassut raudasta plasmaleikkurilla. Tassuihin kiinnittyi myös myöhemmin kehikon alatukiraudat.



Kuva 3. Kehikon kulmapalat valmiina sekä tassujen valmistusta. Raudan leikkaus plasmalla.

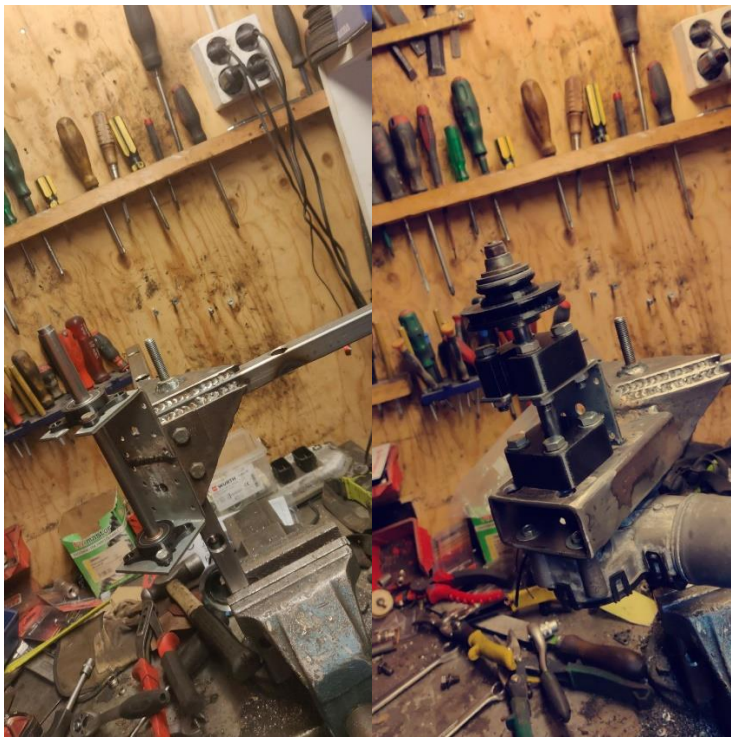
Kuva 4. Kehikko edellisten työvaiheiden jälkeen. Kokonaisuus sekä vaadittavat tuennat vahvistuivat kyseisessä työvaiheessa.

Kehikko kuitenkin ilman risti- sekä alatuentaa oli erittäin epästabiili. Tästä syystä kehikko kiristettiin vaakasuunnassa vaijereilla sekä kehikon jalkoihin tehtiin poikkituet. Saman vaiheen aikana asennettiin vaijerit kiertämään kehikkoa n. 20cm korkeudelle

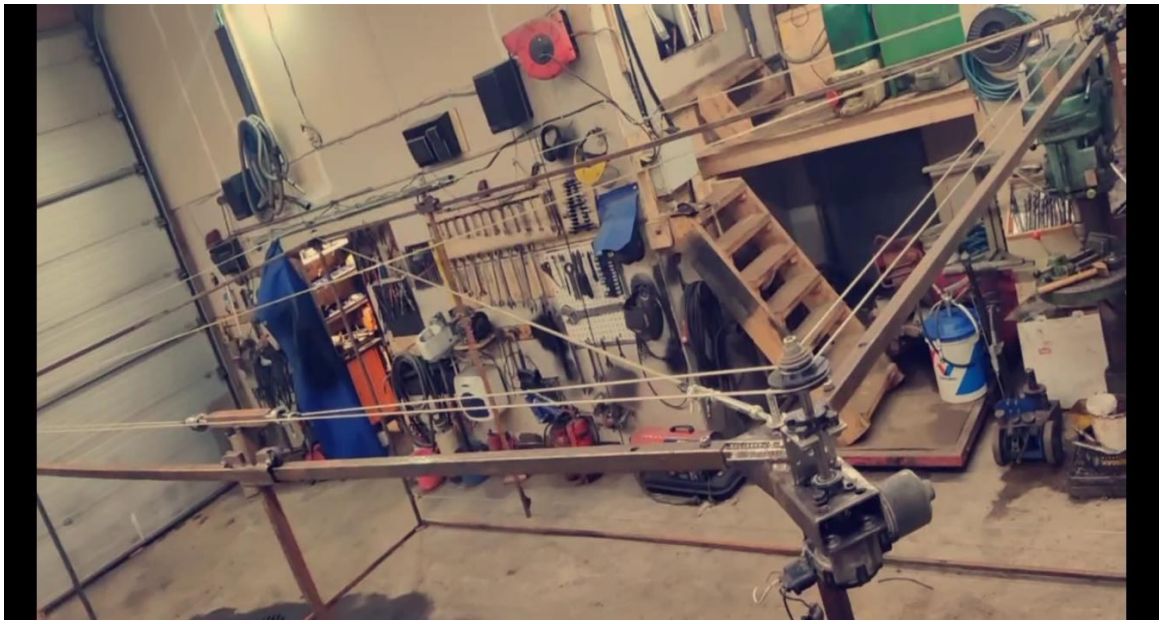
jokaiselle sivulle paitsi sivulle, josta auto ajettaisiin sisään. Vaijereiden tarkoitus oli toimia tukena pesukelkoille, joiden kiinnityspiste oli ainoastaan ylhäällä.

Pesukelkat valmistuivat seuraavaksi. Pesukelkat pujotettiin vaakasuuntaisten ylempien runkoprofiilien ympärille, jonka lisäksi niissä oli pystysuuntainen varsi pystykelkkaa varten sekä kiinnitys yläpuolella vaijerille.

Pesukelkkojen liikkumista varten kulmakappaleisiin valmistui kolme laakeroitua telinettä hihnapyörille. Näiden lisäksi kiinnitettiin yksi hihnapyörä, johon oli asennettu henkilöauton lasinnostajan moottori kelkkojen liikuttamista varten. Kuvissa näkyvät 3D -tulostetut hihnapyörät ovat testivaiheen prototyyppejä.



Kuva 5. Lasinnostajan moottorin kiinnitys päivitettynä. Kovan sivuttaissuuntaisen rasitteen vuoksi laakereiden kiinnityksiä tuli vahvistaa.



Kuva 6. Kehikko liikkuvilla kelkoilla ja voimansiirtomekanismilla.

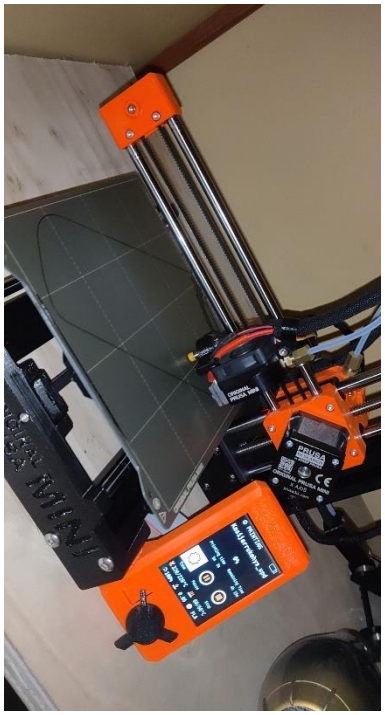
2.2 3D-teknologia

Projektissa suuri osa toiminnallisuuteen liittyvistä osista on 3D -tulostettuja. Tulostetut osat säästivät projektiin kuluvaa aikaa, kun osat voitiin jättää tulostumaan esimerkiksi yön yli ja niiden sopivuus oli usein todella hyvä. Lisäksi useat osat olisivat olleet erittäin vaikeita tuottaa muilla tavoin kuin 3D -tulostuksella, esimerkiksi projektissa käytetyt hihnapyörät. 3D -tulostaminen mahdollisti myös suuria kustannussäästöjä PETG -muovin hinnan ansiosta.
Osien 3D -tulostaminen koostui kolmesta osa-alueesta: Mallinnus, mallin muuttaminen G-koodiksi sekä itse tulostamisesta. Jokainen osa on mallinnettu SolidWorks:lla, jonka jälkeen Prusa:n omalla ohjelmistolla Prusa Slicer:lla mallit on muutettu G-koodiksi, eli tulostimelle luettavaan muotoon. Viimeisenä vaiheena on ollut itse tulostus.

2.2.1 Tulostimet, ohjelmistot sekä käytetty materiaali

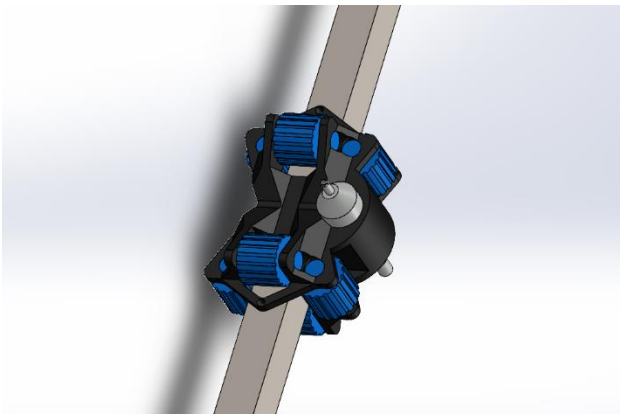
Tulostimena projektin aikana suurimmaksi osaksi toimi Prusa:n Mini. Lisäksi ajan säästämiseksi muutamat osat tulostettiin SAMK:n koulun labrassa olevilla Prusa:n MK3S tulostimilla. Isoimpina projektissa merkitsevinä eroina edellä mainituilla tulostimilla on tulostimien runko sekä tulostusala. Prusa mini:n tulostusala: 180 x 180 x

180mm, kun taas MK3S:n tulostusala 250 x 210 x 210mm. Lisäksi MK3S:n runko on stabiilimpi kahden runkopalkin ansiosta.



Kuva 7. Oma 3D-tulostin. Kyseisellä tulostimella tulostettiin suurin osa projektin osista käyttäen PETG -materiaalia.

Ohjelmistona mallinnuksille sekä hahmotuksille toimi SolidWorks -mallinnusohjelma. SAMK:n lisenssien sekä kurssien ansiosta SolidWorks oli entuudestaan tuttu ja täysin käytettävissä opiskelijana. SolidWorks mahdollisti osien mallinnuksen, mutta myös helpotti hahmottamista osien yhteensopivuudessa. Esimerkkinä prototyyppi pystykelkasta sekä suuttimen kiinnityksestä. Todellisessa toteutuksessa joutui kuitenkin ottamaan huomioon osan tulostettavuuden sekä yritettiin minimoida tukimateriaalin käyttö.



Kuva 8. Pystykelkan prototyyppi. Kuva mallinnettu SolidWorks:lla. Lopullinen toteutus yksinkertaistettu materiaalihukan minimoimiseksi.

Kuva 9. Pystykelkan prototyyppi käytännössä.

ennen varsinaista 3D -tulostusta tarvitsi kuitenkin vielä kääntää mallinnukset G-koodeiksi, eli muotoon jota 3D -tulostin ymmärtää. Prusa Slicer on Prusa:n kehittämä ohjelmisto etenkin Prusa:n tulostimille, josta löytyy valmiit asetukset tulostimille sekä eri filamenteille. Ohjelmistosta pystyy myös muokkaamaan ja hienosäätämään

tulostusasetuksia sekä näkemään tulostimen liikeradat. Kun malli on käännetty G -koodiksi, tiedosto siirrettiin muistitikulle, joka vietiin sitten tulostimelle.

Tulostusmateriaalina toimi PETG -filamentti. PETG sopi tarkoitukseen parhaiten, koska sillä on todella moninaiset ominaisuudet, sitä on helppo tulostaa sekä hinta on kohtuullinen.

PETG:n joitakin hyviä ominaisuuksia:

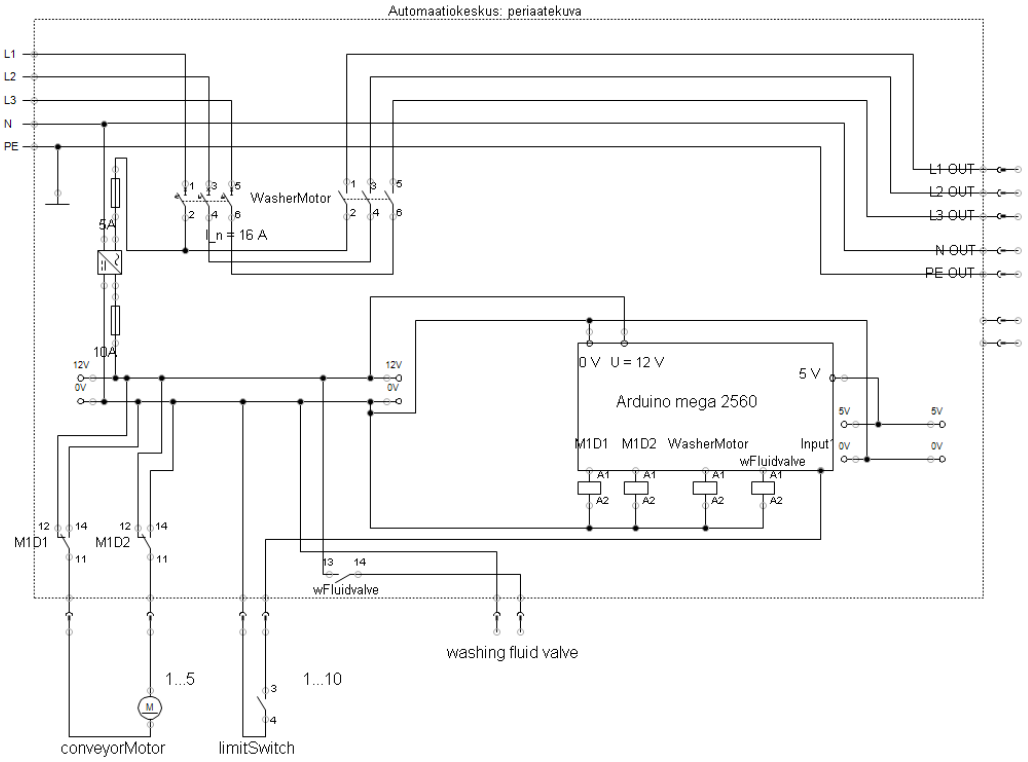
- Korkea lämpötilan kesto
- Helppo tulostettavuus
- Ei kutistu tai väänny huomattavasti jäähtyessä
- Vahvuus ja kesto
- Helppo jälkikäsitellä

PETG:n joitakin huonoja ominaisuuksia:

- Hankala tehdä erittäin pieniä osia
- Tulostettaessa ”seitittää” runsaasti
- Heikko tulostettavuus ilman tukia

2.3 Sähkösuunnittelu

Projektissa sähkösuunnittelukategorian alle lukeutuu eniten keskuksen syöttövirran komponentit sekä virran jakelu keskuksessa. Sulakkeet ovat mitoitettu projektissa käytetylle pesurille. Projektin kuvissa käytetään pesurin ohjaukseen 24VAC vaihtokosketin relettä, joka kuitenkin toimii myös 12VDC ohjauksella, jolla sitä ohjataankin.



Kuva 10. FluidSIM -ohjelmistolla piirretty periaatekuva pesurin keskuksen sähkökytkennästä. Keskuksen kytkentä noudattaa kuvassa esitettyä periaatekuvaa, kuvassa ei näy LCD -näyttöä eikä relekorttien syöttöä tai kytkentää. Relekortit sekä näyttö saivat syöttönsä Arduinolta. LCD -näyttö kytkettiin seuraaviin napoihin:

CS A5

CD A3

RST A4

MOSI 51

MISO 50

SCK 52

LED A0

Keskuksen tietoja:

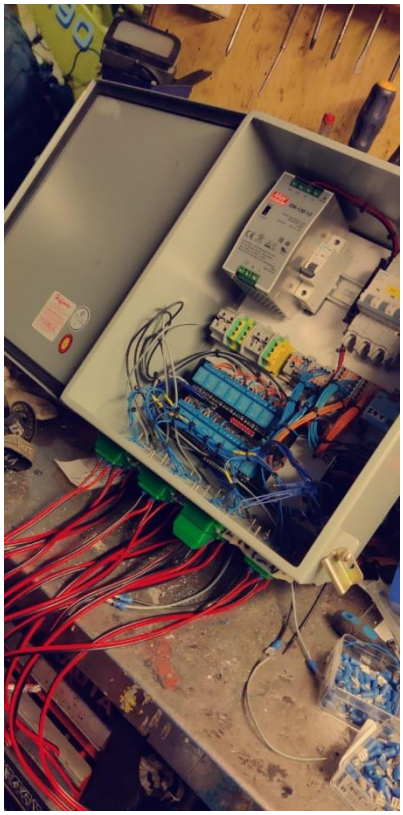
Käyttöjännite: 400VAC

Logiikka- ja automaatiopuolen käyttöjännite: 12VDC tai 5VDC

Virtalähteen käyttöjännite: 230VAC

Mikrokontrolleriin liitetyt laitteet saavat syöttönsä itse mikrokontrollerilta. Laitteita keskuksessa ovat kanteen liitetty LCD- näyttö sekä relekortit. Kontrollerilta tuleva jännite on 5VDC.

Periaatekuvassa conveyorMotor -merkintä viittaa pesurin sivu- sekä pystykelkkoja ohjaaviin moottoreihin, joita on yhteensä viisi. Rajakytkimiä on asennettu kaksi poikittaisliikkeelle, jonka lisäksi jokaisella kelkalla on pystysuuntaiselle liikkeelle omat rajakytkimet. Keskuksen sivuun on asennettu myös pesuaineventtiili, joka annostelee pesuaineen pesurin omalle pumpulle.
Projektissa käytetyt moottorit ovat henkilöauton lasinnostimesta. Alkuperäisestä suunnitelmasta poiketen pesuri ohjaa ainoastaan yhtä moottoria samanaikaisesti, koska projektissa käytetty 120W virtalähde ei riittänyt samanaikaisesti ohjaamaan kaikkia moottoreita.



Kuva 11. Keskuksen pistokkeiden asennusta sekä kytkentöjä.

Kuva 12. Keskuksen kannesta löytyy pääkytkin, pieni 3.2” LCD -näyttö sekä painikkeet käyttöliittymän käyttöön.

2.4 Automaatiosuunnittelu

Projektin automaatiosuunnitteluun lukeutuu koodin arkkitehtuuri ja rakenne, ohjelmien sekvenssit ja kulku sekä tietysti HMI:n (Human – machine Interface) koodaus ja suunnittelu.

Normaaleihin logiikoihin verrattuna projektin ohjelmointi oli hieman haastavampi, koska käytössä oli Arduinon mikrokontrolleri ja koodauskielenä toimi tällöin C++. Toiminnallisuuksia luodessa tuli pitää mielessä, että mikrokontrolleri suorittaa koodia aina rivi kerrallaan, joten tällöin samanaikaisissa toiminnoissa tarvitsi olla tarkkana.

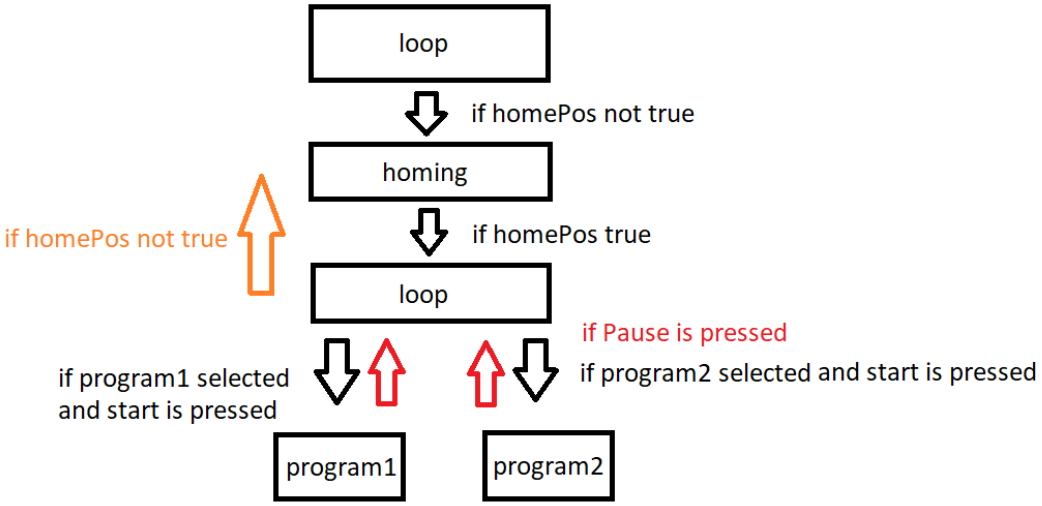
Pesuohjelmien rakenne on projektissa koostettu ns. pesukierroista. Kiertojen määrän sekä ”tarkkuuden” pystyy valitsemaan koodin alusta muuttamalla yhtä muuttujaa. HMI:sta valittavissa on myös pesuohjelma 2, joka sisältää pesuaineen syötön. Tällöin yksi pesukierto ohjelmassa pestään pesuaineventtiili auki, jonka jälkeen ohjelma odottaa koodissa määritellyn ajan, ennen kuin pesee pesuaineen pois vedellä.

HMI:sta pystyy siis valitsemaan kumpaa pesuohjelmaa haluaa käyttää (optiot laajennukselle eli useammalle pesuohjelmalle). Lisäksi HMI:ssa lukee tiedot pesuohjelmasta, arvioitu kesto sekä päivittyvä arvioitu jäljellä oleva aika. HMI:n alapuolella on kommunikointipainikkeet, joilla pystyy HMI:n toimintoja hallitsemaan. Painikkeilla on myös koodausvaiheessa manuaalikäyttö mahdollisuus, jonka voi asettaa päälle koodissa.Tämä helpotti joissain tilanteissa testaamista.

2.5 Rakenne

Ohjelman arkkitehtuuri koostuu yksinkertaistettuna viidestä ohjelmakierrosta.

1. loop - Koodin ensimmäinen ohjelmakierto, sisältää käskyn homing -ohjelmakiertoon siirtymisestä sekä toimii ”odotustilana” muille ennen muita ohjelmakiertoja.
2. homing - Koodin toinen ohjelmakierto, joka suoritetaan aina pesurin käynnistyksessä ilman erillistä käskyä, jos pesuri ei ole kotiasemassa.
3. program1 – Ensimmäinen pesuohjelma. Suoritetaan kun homing on tarvittaessa suoritettu ja käyttäjä on valinnut HMI:sta pesuohjelman 1 ja painanut käynnistyspainiketta. Ohjelmakierrosta on mahdollista poistua painamalla Pause -painiketta, jolloin ohjelma palaa loop -ohjelmakiertoon ja suorittaa homing -ohjelmakierron tarvittaessa.
4. program2 – Toinen pesuohjelma. Suoritetaan kun homing on tarvittaessa suoritettu ja käyttäjä on valinnut HMI:sta pesuohjelman 2 ja painanut käynnistyspainiketta. Ohjelmakierrosta on mahdollista poistua painamalla Pause -painiketta, jolloin ohjelma palaa loop -ohjelmakiertoon ja suorittaa homing -ohjelmakierron tarvittaessa.
5. stopSeq – Ohjelman keskeytykseen käytetty ohjelmakierto. Ohjelmaan sisältyy ainoastaan Pause -painonapin tilan tarkastus. Käytetään, jotta säästettäisiin koodin kokonaispituutta. Mikrokontrolleri suorittaa ohjelmaa rivi kerrallaan, joten ohjelmakutsuja jouduttiin ripottelemaan pesuohjelmiin. Negatiivisena puolena voidaan pitää sitä, että pesuohjelma ei pysähdy välittömästi painonapin painamisen jälkeen, vaan seuraavaan kohtaan missä napin tila tarkastetaan.



Kuva 13. Periaatekuva pesurin ohjelmakierrosta.

Edellä mainittujen ohjelmarakenteiden lisäksi koodiin sisältyy HMI:n ohjelmakoodi, josta saadaan myös signaalit ohjelmavalinnoille sekä käynnistys- ja pysäytyskäskyille. Koodissa on myös manuaalikäyttötila, joka tarvitsee asettaa koodista toimintaan. Tila on olemassa lähinnä testausvaiheita varten, jolloin on helppo todeta moottorien toimivuus ja kelkkojen liikkuminen.

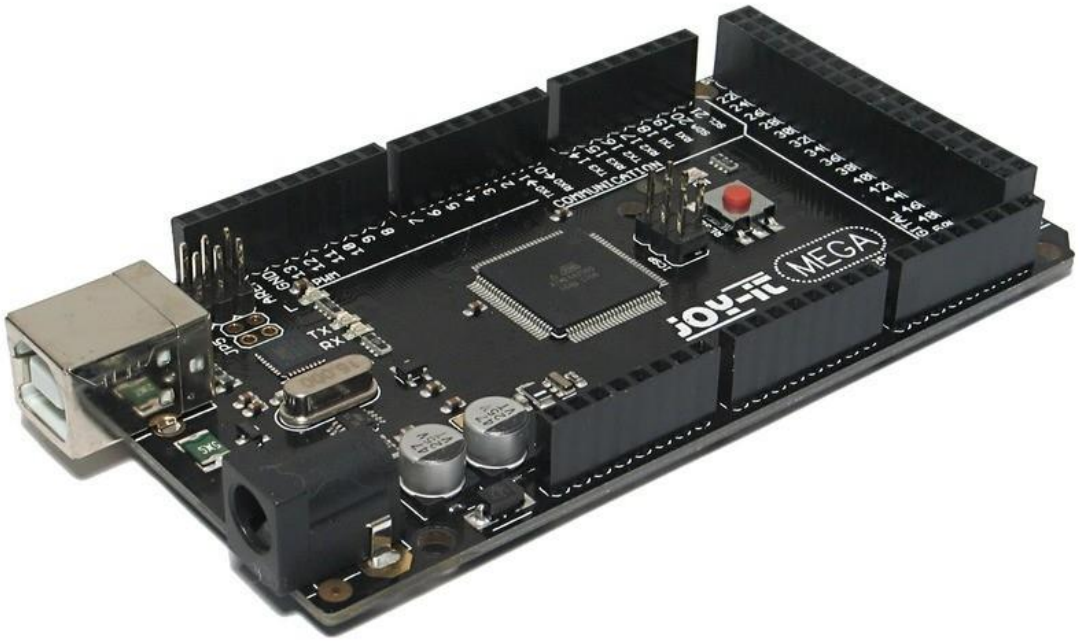
3 OHJELMAN KOODAUS C++

3.1 Arduino mega

Alla olevassa kuvassa on esitetty projektissa käytetty mikrokontrolleri. Kontrolleri valittiin toimittamaan logiikan virkaa projektissa sen monikäyttöisyyden sekä pinnimäärien vuoksi. Arduino mega:ssa on yhteensä 54 digitaalisia tulo- ja lähtöpinnejä. Näistä pinneistä 15 kykenee toimimaan PWM -lähtöinä (PWM = Pulse Width Modulation). PWM -lähdöt toimivat mainiosti esimerkiksi DC -moottoreiden nopeuden säädössä tai valojen himmennyksissä analogiasignaalin tavoin. Arduino kykenee antamaan jokaiselta I/O pinniltä maksimissaan 20mA virran, joka kuitenkin riittää hyvin relekortin kelan ohjaamiseen.

Mikrokontrollerissa on myös EEPROM -muistia käytössä 4KB. Projektissa ei ollut käytössä toimintoja EEPROM -ominaisuuksilla, mutta käytännössä sillä kyetään esimerkiksi tallentamaan muuttujien arvoja muistiin vaikka laitteen virta katkaistaisiin.

Kyseinen mikrokontrolleri valittiin projektin älyksi sen hinnan vuoksi sekä siksi, että kyseisellä alustalla on toteutettu monia erilaisia projekteja, joten informaatiota sekä opastusvideoita löytyy runsaasti. Arduinon mikrokontrolleri on myös sellaisenaan ilman muokkauksia lähellä automaatioprojekteissa käytettäviä I/O -logiikoita, joten sen toiminta on kohtuullisen luotettavaa.



Kuva 14. Projektissa käytetty mikrokontrolleri laitteen älynä. (Partco:n verkkokaupan sivut 2021)

3.2 Ohjelmointiympäristö Arduino IDE

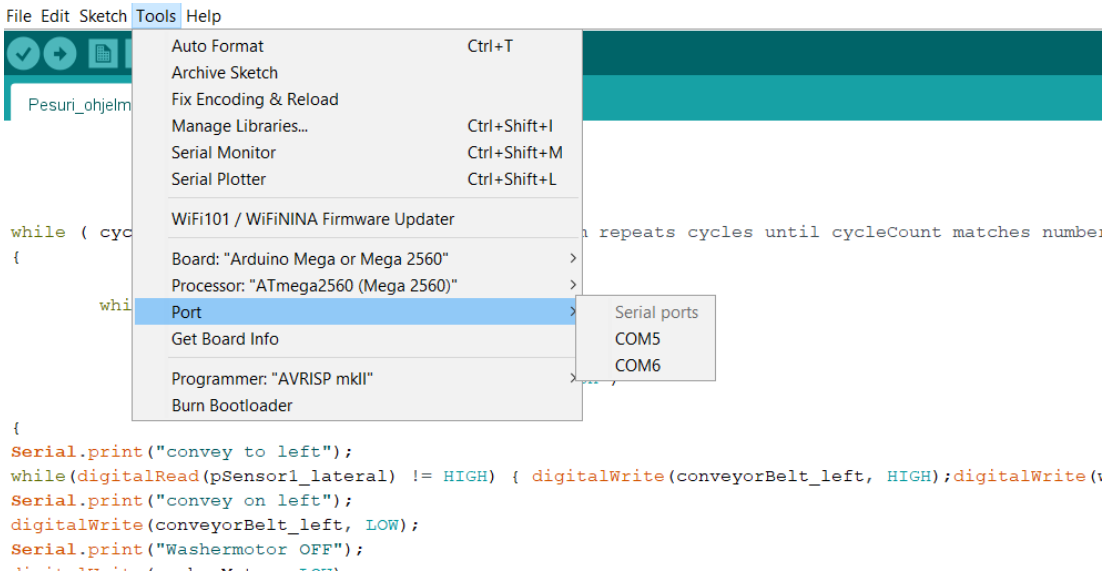
Koodin kirjoitukseen käytettiin Arduino IDE -ohjelmaa. Ohjelma toimii käytännössä kirjoitusohjelmana sekä latauspohjana itse koodille. Ohjelmassa koodi tarkistetaan ennen kuin se ladataan kontrollerille. Ohjelma tulkitsee virheet C++ koodin kirjoituksessa, joita se ei pysty kääntämään. Tällöin koodi on korjattava oikeelliseksi, jonka jälkeen se voidaan ladata kontrollerille.

Alla olevassa kuvassa on kuva pesurin ohjelmakoodista, sekä ohjelmointiympäristöstä. Ohjelmistoympäristössä on mahdollista ladata ja käyttää hyödyksi erilaisia kirjastoja, jotka sisältävät koodirivejä ja täten esimerkiksi lyhentävät, erittelevät tai selkeyttävät varsinaisen koodin kirjoitusta.

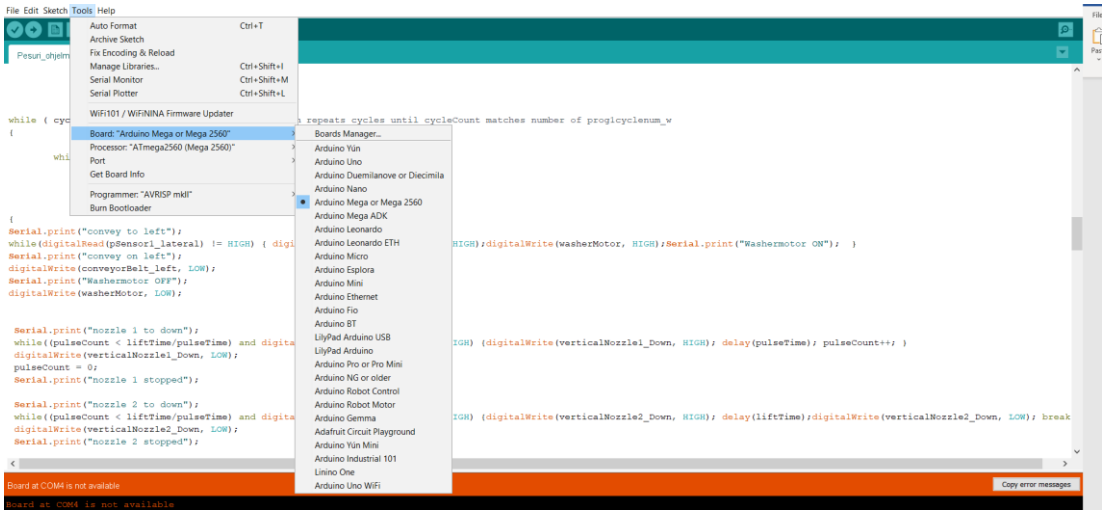
Esimerkiksi kuvassa näkyvä Serial.print -komento on erittäin hyödyllinen toiminto pesurin testausvaiheessa tietokoneen ollessa keskuksessa kiinni. Komento yksinkertaisesti tulostaa annetun tekstin väylämonitoriin sillä hetkellä kun kursori on kyseisen rivin kohdalla. Tällöin pesurin toimintaa sekä mahdollisia virhetilanteita pystyi seuraamaan tietokoneella reaaliaikaisesti. Kyseiset komennot eivät muuta koodin toimintaa millään tavalla. Arduino IDE:ssä väylämonitori on kätevä tapa seurata reaaliaikaisia tuloksia, tilanteita tai signaaleja.

Kuva 15. Kuvat pesurin alkuperäisestä testikoodista Arduino IDE:ssa.

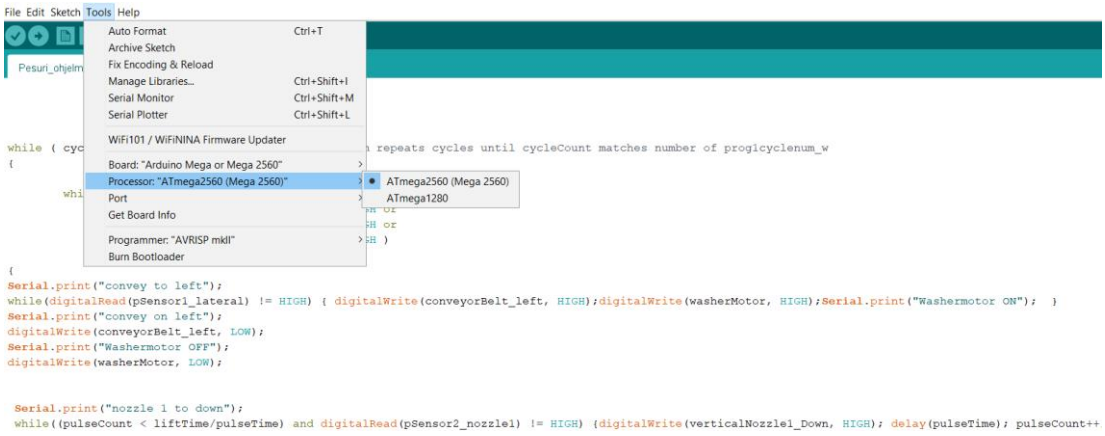
1. Siirrettäessä koodia kontrollerille tulee ensin määritellä käytettävä tietokoneen portti työkalut -välilehdeltä.
2. Tämän jälkeen määritellään kytketty kortti, jos sitä ei ole vielä määritetty.
3. Jos kortilla on useita mahdollisia suorittimia, tulee valita kytketyn kortin käytössä oleva suoritin.



Kuva 16. Arduino IDE portin valinta.



Kuva 17. Oikean kortin valitseminen.



Kuva 18. Kortin prosessorin valinta.

3.3 Ohjelman osat sekä toiminta

3.3.1 Rakenne

Edellisissä osioissa sivuttiin jo ohjelman rakennetta hieman. Tarkastellaan ohjelman rakennetta hieman tarkemmin:

Koodin aluksi määritellään käytetyt muuttujat sekä niiden mahdolliset arvot. Tässä tapauksessa arvoa käytetään yhdisteässä muuttuja kontrollerin pinniin. Pesurin koodissa käytettiin muuttujan tyyppinä integer -tyyppistä muuttujaa boolean -muuttujan sijaan. Erään koodissa käytetyn muuttujan arvo on määritelty koodirivillä seuraavasti:
int pSensor2_nozzle1 = 4; // Controller gets information to stop the movement, vertical, Down
Muuttujan arvoksi määritellään 4, koska anturin paluu on kytketty pinniin 4. Muuttujan arvoa voidaan kuitenkin koodissa muuttaa, koska kyseessä ei ole const int -muuttuja. Tässä tapauksessa toiminnan kannalta ei ole merkitystä kumpaa tietotyyppiä käyttää.

Seuraavaksi koodissa määritetään pinnin käyttötarkoitus, tässä tapauksessa kyse on muuttujan käytöstä tulo- tai lähtötietona. Kyseinen muuttuja: pSensor2_nozzle1 on määritelty koodissa seuraavasti:
pinMode(pSensor2_nozzle1, INPUT_PULLUP);
Tällöin pinni 4 toimii tulokanavana. INPUT_PULLUP tulotietoa käytetään kytkettäessä kyseisessä projektissa rajakytkimen toinen kosketin Ground- eli 0- potentiaaliin ja toinen kosketin kytketään haluttuun pinniin. Tällöin pinnin arvoa luettaessa sen arvo on HIGH kun kosketin on auki (NO-kosketin). Toisaalta tällöin sen tila on myös LOW, kun kosketin on kiinni.

Edellisten vaiheiden jälkeen voidaan aloittaa itse koodin kirjoittaminen. Koodin rakenteessa on käytetty seuraavia kiertoja:

- ScreenControl: Kyseinen loop sisältää näytön sekä siihen liittyviin kommunikaatiopainikkeisiin tarvittavat koodirivit.

- program1: Tämä loop sisältää ensimmäisen määritellyn pesuohjelman koodin ja tiedot. Tähän ohjelmakiertoon ohjataan myöhemmässä vaiheessa.
- program2: Sisältää toisen pesuohjelman ohjelman koodin ja tiedot.
- homing: Sisältää kotiasemaan liittyvät tiedot ja koodin. Kotiasemaan ohjataan aina laitteen käynnistyessä, jos siihen on tarve.
- loop: Varsinainen pääohjelmakierto. Ohjelmakierto on viimeisenä koodissa, koska se sisältää muiden ohjelmakiertojen kutsuja. Ohjelmakierrot on esiteltävä koodissa ennen niiden kutsumista.

3.3.2 Sekvenssi

Pesuohjelmassa toistetaan paljon samaa rakennetta soveltaen if- sekä while- komentoja. Käytännössä pesukelkat lähtevät aina liikkeelle kotiasemasta. Tämän jälkeen kelkat liikkuvat toiseen ääripäähän, laskevat käyttäjän määritellyn ajan (matkan), jonka jälkeen liikkuvat taas toiseen ääripäähän ja toistavat tätä rakennetta. Toisto loppuu kun kaikki kelkat ovat saavuttaneet kelkkojen alaraja-anturin. Tämän jälkeen ohjelmasta riippuen kierto toistuu vastakkaisessa suunnassa, ruiskuttaa pesuaineen tai odottaa määritetyn ajan.

Pesukelkkojen ylä- ja alapäässä sijaitsevat raja-anturit toimivat digitaalisignaalilla eli on/off -tyylillä. Ohjelmoinnin kannalta ongelmaksi koitui kuitenkin kelkkojen laskuaika:

```
Serial.print("nozzle 1 to down");
while((pulseCount < liftTime/pulseTime) and digitalRead(pSensor2_nozzle1) != HIGH) {digitalWrite(verticalNozzle1_Down, HIGH); delay(pulseTime); pulseCount++; }
digitalWrite(verticalNozzle1_Down, LOW);
pulseCount = 0;
Serial.print("nozzle 1 stopped");
```

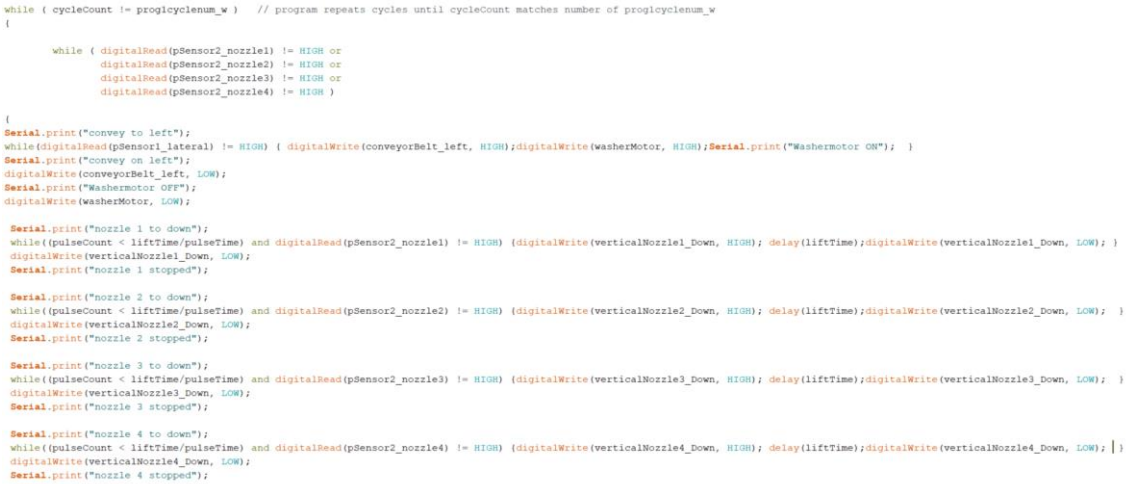
Ylläolevassa koodissa on toteutettu kelkan lasku. Koodin alussa on määritelty muuttujan pulseTime arvo eli aika millisekunteina kuinka usein ohjelma tarkistaa onko kelkka osunut rajakytkimeen. Esimerkiksi:
Kelkan nostoaika: 3000ms
pulseTime eli tarkastusväli: 10ms
pulseCount kasvaa jokaisen while -kierron jälkeen yhdellä, joten:
Ohjelma tarkastaa rajakytkimen tilan 300 kertaa yhden laskun aikana, jolloin kelkan osuessa rajakytkimeen kelkka ajaa päin rajaa maksimissaan 10ms. Tämän jälkeen ohjelmassa pulseCount muuttujan arvoksi muutetaan taas 0, jotta sitä voidaan käyttää seuraavan kelkan ohjauksessa.

verrattuna yksinkertaisempaan tapaan:

```
Serial.print("nozzle 3 to down");
if (digitalRead(pSensor2_nozzle3) != HIGH)
{digitalWrite(verticalNozzle3_Down, HIGH); delay(liftTime);digitalWrite(vertical
Nozzle3_Down, LOW); }
digitalWrite(verticalNozzle3_Down, LOW);
Serial.print("nozzle 3 stopped");
```

Tällöin ongelmaksi koituu tilanne, jossa kursori on edennyt if -funktion sisälle ja liikuttaa kelkkaa alas määrätyn ajan ja tarkistaa raja-anturin tilan ainoastaan aluksi ennen liikuttamista, eikä välitä siitä osuuko kelkka raja-anturiin liikuttamisen aikana.

Pesurin koodissa yksi sekvenssikierto kantaa joko nimeä downSeq tai upSeq, riippuen sekvenssin kulkusuunnasta. Seuraavassa kuvassa on osa pesurin koodia ja tarkemmin kuvattuna downSeq:



```
while ( cycleCount != progIcyclenum_w )   // program repeats cycles until cycleCount matches number of progIcyclenum_w
{

       while ( digitalRead(pSensor2_nozzle1) != HIGH or
               digitalRead(pSensor2_nozzle2) != HIGH or
               digitalRead(pSensor2_nozzle3) != HIGH or
               digitalRead(pSensor2_nozzle4) != HIGH )

{
Serial.print("convey to left");
while(digitalRead(pSensor1_lateral) != HIGH) { digitalWrite(conveyorBelt_left, HIGH);digitalWrite(washerMotor, HIGH);Serial.print("Washermotor ON");  }
Serial.print("convey on left");
digitalWrite(conveyorBelt_left, LOW);
Serial.print("Washermotor OFF");
digitalWrite(washerMotor, LOW);

 Serial.print("nozzle 1 to down");
 while((pulseCount < liftTime/pulseTime) and digitalRead(pSensor2_nozzle1) != HIGH) {digitalWrite(verticalNozzle1_Down, HIGH); delay(liftTime);digitalWrite(verticalNozzle1_Down, LOW); }
 digitalWrite(verticalNozzle1_Down, LOW);
 Serial.print("nozzle 1 stopped");

 Serial.print("nozzle 2 to down");
 while((pulseCount < liftTime/pulseTime) and digitalRead(pSensor2_nozzle2) != HIGH) {digitalWrite(verticalNozzle2_Down, HIGH); delay(liftTime);digitalWrite(verticalNozzle2_Down, LOW);  }
 digitalWrite(verticalNozzle2_Down, LOW);
 Serial.print("nozzle 2 stopped");

 Serial.print("nozzle 3 to down");
 while((pulseCount < liftTime/pulseTime) and digitalRead(pSensor2_nozzle3) != HIGH) {digitalWrite(verticalNozzle3_Down, HIGH); delay(liftTime);digitalWrite(verticalNozzle3_Down, LOW);  }
 digitalWrite(verticalNozzle3_Down, LOW);
 Serial.print("nozzle 3 stopped");

 Serial.print("nozzle 4 to down");
 while((pulseCount < liftTime/pulseTime) and digitalRead(pSensor2_nozzle4) != HIGH) {digitalWrite(verticalNozzle4_Down, HIGH); delay(liftTime);digitalWrite(verticalNozzle4_Down, LOW); }
 digitalWrite(verticalNozzle4_Down, LOW);
 Serial.print("nozzle 4 stopped");
```

Kuva 19. Pesusyklit toteutettu pitkälti while -komentoa käyttäen. Rajakytkimen tarkastus pulseCount -muuttujaa käyttäen.

Kuvassa näkyvä jälkimmäinen while -komento varmistaa, että downSeq suoritetaan niin kauan, kunnes kaikki pystykelkat ovat saavuttaneet alarajan. Käytännössä koko pesuohjelman koodi koostuu näistä elementeistä. Lyhennettynä sekä yksinkertaistettuna pesuohjelman koodi näyttäisi tältä:

```
while ( cycleCount != prog1cyclenum_w )
{
while ( digitalRead(pSensor2_nozzle1) != HIGH or
        digitalRead(pSensor2_nozzle2) != HIGH or
        digitalRead(pSensor2_nozzle3) != HIGH or
        digitalRead(pSensor2_nozzle4) != HIGH )
{ - downSeq - }

while ( digitalRead(pSensor2_nozzle1) != HIGH or
        digitalRead(pSensor2_nozzle2) != HIGH or
        digitalRead(pSensor2_nozzle3) != HIGH or
        digitalRead(pSensor2_nozzle4) != HIGH )
{ - upSeq - }

cycleCount++;
}
```

cycleCount -muuttuja määrittää montako kertaa pesuri tekee pesusekvenssin. cycleCount -muuttujan määrää pystyi muuttamaan koodin alusta sekä sille oli valmisteltu koodi, missä käyttäjä pystyisi päättämään pesuohjelmaa valitessa montako kertaa sekvenssi suoritettaisiin.

3.3.3 Pysäytykset

Automaatiojärjestelmissä tulee olla mahdollisuus pysäyttää laite tai koneisto mahdollisten vaaratilanteiden varalta tai muista syistä. Tässä projektissa oli kahdenlaista pysäytystapaa:

1. Hätäseis – ei ole riippuvainen toimiiko logiikka vai ei, katkaisee virran moottoreiden syötöiltä sekä antaa tiedon Arduinolle katkaisusta.
2. Pause -painike – Koodiin on ripoteltu ohjelmarivejä, joissa tarkastetaan hätäseis- sekä pause- painikkeen tila. Jos hätäseis on painettuna, ohjelma palautuu pääohjelmakiertoon. Jos pause on painettuna, moottorit pysähtyvät, mutta lähtevät uudestaan käyntiin, jos painiketta painetaan uudestaan. Painiketta painettaessa HMI:ssa näkyy vaihtoehto myös pesuohjelmasta poistumiseen.

3.3.4 Näytön koodaus

Näyttönä projektissa toimi 3.2” TFT SPI ILI9341 -näyttö. Syynä tälle ole helppo yhteensopivuus Arduinon kanssa sekä näytön alhaiset kustannukset. Näytössä on myös mahdollisuus käyttää kosketusnäyttöä, mutta projektissa sen käyttöä ei toteutettu liian pienen koon takia.

Opinnäytetyössä LCD -näyttöä ohjattiin suoraan Arduinolla. Tällöin heikkoutena oli näytön hidas päivittyminen. Arduinon toimiessa ohjaimena näyttö päivittyy pikseli kerrallaan ja tällöin esimerkiksi koko näytön värin päivittäminen vei useamman sekunnin. Parempana vaihtoehtona olisi ollut jokin tehokkaampi alusta tai näytön ohjaamiseen tehty kortti. Hitaus ei kuitenkaan haitannut HMI:n toimintaa, koska aluksi näyttöön päivitettiin värit sekä kehykset, mutta tämän jälkeen tulevat päivitykset, esimerkiksi teksti on niin pieni pikselimäärältään, että se päivittyi näytölle nopeasti.

Näytön koodi ohjelmassa perustui pitkälti näytön ohjeiden mukana tulleeseen esimerkkiprojektiin. Esimerkkiprojektissa oli valmiiksi kirjoitetut koodit suorakulmioiden, ympyröiden, tekstien sekä monien muiden geometrioiden luonneille. Näitä valmiita komponentteja käytettiin pesurin ohjelmassa, kuitenkin muokaten näytölle piirretty kuva oikeaksi.

4 RATKAISUMALLIT, JALOSTUS SEKÄ ONGELMAT

Projektissa käytetyt materiaalit sekä komponentit valittiin kyseiseen projektiin kustannussyyt edellä. Parempia vaihtoehtoja sekä toteutustapoja olisi ollut monia, mutta kyseessä oli kuitenkin itse kotona rakennettava sekä kustannettava projekti, jonka komponentteja tuli pystyä hyödyntämään jälkikäteen. Tästä syystä kustannuksia tuli miettiä tarkasti eikä välttämättä pystytty valitsemaan tarkoitukseen parhaimpia komponentteja.

4.1 Nykyinen ratkaisutapa ja sen parannukset

4.1.1 Rungon materiaali

Nykyisen ratkaisutavan suurimmat ongelmat johtuivat lähtökohtaisesti pesurin kehikon materiaalista. Materiaaliksi valittu 25mm rautaprofiili suoritti tarkoituksensa kohtuullisen hyvin, kuitenkin vaatien monta kiristysvaijeria tukemaan rakennetta. Lisäksi kantavat kiskorakenteet olisivat olleet huomattavasti paremmat paksummalla materiaalilla. Kuitenkin raudasta valmistettuna kehikko ei olisi voinut olla hirveästi vahvempi sen keräämän massan vuoksi. Runkomateriaalin heikkoudet olivat pitkälti sen liian pieni koko pituuteen nähden sekä kokoa kasvattaessa painon lisääntyminen. Vaihtoehtoisena sekä huomattavasti parempana materiaalina olisi ollut alumiini. Alumiinin hankkiminen tarvittavissa määrin olisi ollut kuitenkin huomattavasti kalliimpaa, jonka vuoksi materiaalina jouduttiin käyttämään rautaprofiilia.

4.1.2 Pesurin ohjaus

Arduino oli tässä käyttötarkoituksessa kohtuullisen hyvä ratkaisu ohjaamaan pesuria, mutta kuitenkin jalostusvaiheessa esimerkiksi Raspberry Pi olisi voinut olla parempi vaihtoehto. Raspberry:lla esimerkiksi etäohjaukset sekä tiedonsiirron olisi voinut saada helposti toteutettua.

4.1.3 Kelkkojen liikuttaminen

Kelkkoja liikuttavat vaijerit piti alun perin korvata hihnalla, mutta kiilahihna olisi tarvittavissa määrin maksanut huomattavasti enemmän kuin ohut vaijeri. Hihnalla olisi kuitenkin saanut vakaamman sekä pitävämmän linjaston.
Vaijerit kiinnitettiin ainoastaan kelkkojen yläpäähän, jolloin rullien tarvitsi olla sovitukseltaan juuri sopivat, ettei kelkka jäänyt epästabiiliksi liikkeen aikana. Ratkaisutapaa piti jalostaa projektin lopussa lisäämällä toinen kiinnitys ja vetopiste kelkan alapäähän, jolloin veto olisi tullut tasaisesti ylhäältä sekä alhaalta. Kyseinen parannus jäi kuitenkin ajanpuutteen takia lisäämättä.

4.2 Jatkojalostus

Jatkojalostuksen tarkoituksena voisi olla esimerkiksi pesurin idean saattaminen markkinakäyttöön esimerkiksi lisäämällä varmatoimiset mekanismit sekä maksutapa pesuriin. Tällöin tuotteena olisi ns. miehittämätön pesuasema. Kuitenkin ennen tuotantoon saattamista tulisi tehdä huomattavia parannuksia. Esimerkiksi pesurin rungon kiinnittäminen kiinteään konttiin, jonka läpi pystyisi ajamaan autolla. Logiikan tulisi myös olla huomattavasti luotettavampi sekä ammattikäyttöön tehty, koska kyseessä olisi kaupallinen tuote.

5 JOHTOPÄÄTÖKSET JA TULKINTA

Projekti kokonaisuudessaan opetti sekä mahdollisti pienen automaatiojärjestelmän luomista kokonaan itse. Pesurin luomisessa käytettiin monia eri alojen taitoja metallin työstämisestä koodaamiseen. Ohjelmointi, mallintaminen, sähkötekniikka sekä koodaaminen olivat opintojen näkökulmasta opinnäytetyön keskeisimpiä osaamisalueita.

Pesurin toimintaa ei valitettavasti päästy pidemmällä aikavälillä testaamaan aikataulun vuoksi eikä pesujäljestä saatu raportointia. Pesurin testaus onnistui kuitenkin hyvin:

1. Pesuri onnistui tekemään pesuohjelmat virheettä loppuun
2. Käyttöliittymä toimi moitteetta
3. Pysäytykset sekä sekvenssit toimivat hyvin myös mahdollisissa häiriötilanteissa
4. Erilaiset ratkaisut onnistuttiin tekemään kustannustehokkaasti pitäen ratkaisut mahdollisimman luotettavina

Pesurin jokapäiväinen käyttö kuitenkin olisi vaatinut parannuksia runkorakenteisiin, joista mainittiinkin edellisissä kappaleissa. Projektin tavoite kuitenkin onnistui oletettua paremmin: Onnistuttiin luomaan kokonainen automaatiojärjestelmä sekä siihen liittyvät toiminnot kustannustehokkaasti. Käytetyt ohjelmistot olivat osaksi ilmaisia alustoja, mutta esimerkiksi 3D-mallinnusohjelman käytön mahdollisti SAMK:n lisenssit.

Suurin osa työstämisestä sekä projektin vaiheista tapahtui omissa tiloissa omilla välineillä, mutta huomattavan kiitoksen ansaitsee Satakunnan ammattikorkeakoulu. Aikataulun tiuketessa huomattava etu oli päästä käyttämään oman 3D-tulostimen lisäksi myös koulun tulostimia. Tämä nopeutti projektin kulkua huomattavasti.

LÄHTEET

SolidWorks. 2022. https://www.solidworks.com/

Arduino www-sivut. 2022. https://www.arduino.cc/

Arduino. 2022. Digital Input Pull-Up Resistor. https://docs.arduino.cc/tutorials/generic/digital-input-pullup

Prusa3d. 2022. https://www.prusa3d.com/

lcdwiki. 2022. 3.2inch SPI module. http://www.lcdwiki.com/3.2inch_SPI_Module_ILI9341_SKU:MSP3218

Parco verkkokauppa. 2022. Arduino Mega 2560 R3 Klooni. Viitattu 12.7.2022. https://www.partco.fi/fi/arduino/arduino-mallit/19220-ard-mega2560r3.html

SolidWorks. 2022. https://www.solidworks.com/

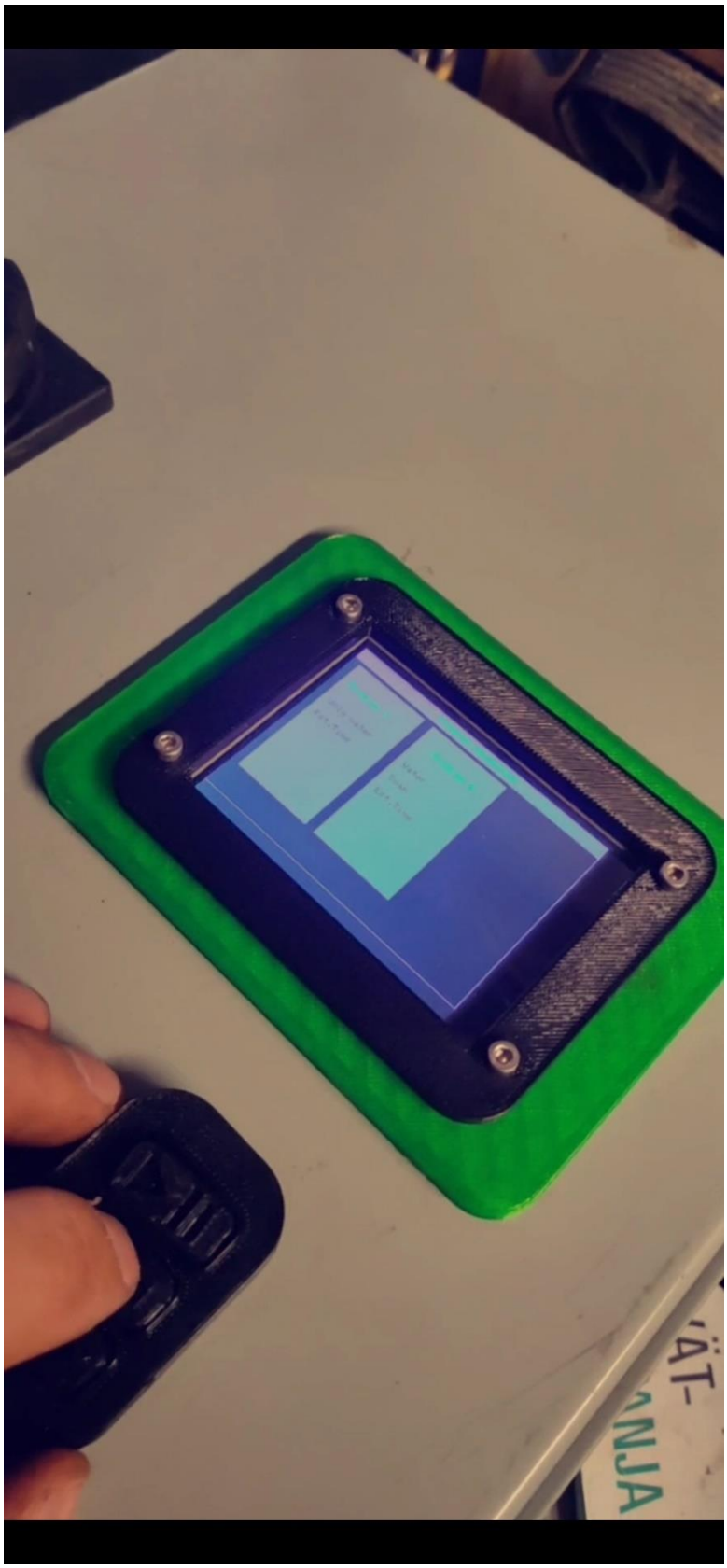
Arduino. 2022. Language Reference. https://www.arduino.cc/reference/en/

Omat dokumentaatiot projektin rakennusvaiheista.

Kuva 20. Ensimmäinen testikerta veden kanssa.

Kuva 21. Kehikon tassujen leikkausta plasmalla.

Kuva 22. Prototyyppikelkan testausta.



Kuva 23. Käyttöliittymä: Käytössä olevat kaksi ohjelmaa, painikkeet oikealle sekä vasemmalle. Ylin painike ohjelman aloitus/ pause.

Kuva 24. Pystykelkkojen alareunassa olevat ohjurit: Auttavat kelkkoja pysymään stabiilina, koska kiinnitys ainoastaan ylhäältä.