TALLINNA TEHNIKAÜLIKOOL

INSENERITEADUSKOND

Elektroenergeetika ja mehhatroonika instituut

RASPBERRY PI PLATVORMIL PÕHINEVALE LIIKUVROBOTILE SOODSAMA ALTERNATIIVI LEIDMINE, ÜMBEREHITAMINE TEISELE PLATVORMILE JA LAHENDUSE AVALIKUSTAMINE

CONSIDERATION, IMPLEMENTATION AND PUBLICATION OF A CHEAPER ALTERNATIVE FOR AN EXISTING RASPBERRY PI- BASED ROBOT VEHICLE

BAKALAUREUSETÖÖ

Üliõpilane: Mattias Allpere

Üliõpilaskood: 206102 EAAB

Juhendaja: Anton Rassõlkin, kaasprofessor tenuuris

Kaasjuhendaja: Diana Belolipetskaja

Tallinn 2024

(Tiitellehe pöördel)

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.
Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

13.mai 2024

Autor: Mattias Allpere

Töö vastab bakalaureusetöö/magistritööle esitatud nõuetele

"....." ..... 20…..

Juhendaja: .....
/ allkiri /

Kaitsmisele lubatud

"....."..... 20… .

Kaitsmiskomisjoni esimees .....
/ nimi ja allkiri /

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Mattias Allpere

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose RASPBERRY PI PLATVORMIL PÕHINEVALE LIIKUVROBOTILE SOODSAMA ALTERNATIIVI LEIDMINE, ÜMBEREHITAMINE TEISELE PLATVORMILE JA LAHENDUSE AVALIKUSTAMINE,

mille juhendajad on Anton Rassõlkin, kaasprofessor tenuuris ja Diana Belolipetskaja,

1.1 reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2 üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

---

13. mai 2024

[1] *Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.*

Elektroenergeetika ja mehhatroonika instituut

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Mattias Allpere, 206102 EAAB
Õppekava: EAAB23, elektroenergeetika ja mehhatroonika
Peaeriala: mehhatroonika
Juhendaja(d): Anton Rassõlkin, kaasprofessor tenuuris ja Diana Belolipetskaja

Lõputöö teema:
(eesti keeles) Raspberry Pi platvormil põhinevale liikuvrobotile soodsama alternatiivi leidmine, ümberehitamine teisele platvormile ja lahenduse avalikustamine

(inglise keeles) Consideration, Implementation and Publication of a Cheaper Alternative for an Existing Raspberry Pi Based Robot Vehicle

Lõputöö põhieesmärgid:

1. Luua odavam, töötav variant olemasolevast robotplatvormist.
2. Luua eeldused robotplatvormi tulevaseks kasutamiseks hariduslikul eesmärgil.
3. Saada kogemust mehhatroonikateadmiste praktilises rakendamises.

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Koguda taustainfot	01.03.24
2.	Kirjuta valmis töö teoreetiline osa	01.04.24
3.	Saa valmis töö praktiline osa	13.05.24

Töö keel: eesti keel

Lõputöö esitamise tähtaeg: 13.05.2024
Üliõpilane: Mattias Allpere "...." 13.05.2024
Juhendaja(d): kaasprofessor tenuuris Anton Rassõlkin, Diana Belolipetskaja "....", "....." 13.05.2024
Programmijuht: Marek Tull "...." 13.05.2024

Kinnise kaitsmise ja/või lõputöö avalikustamise piirangu tingimused formuleeritakse pöördel

SISUKORD

Lühendite ja tähiste loetelu

1. HDMI (ingl k *High-Definition Multimedia Interface*)
2. BGR (ingl k *Blue-Green-Red*)
3. RGB (ingl k *Red-Green-Blue*)
4. LAN (ingl k *Local Area Network*)
5. PoE (ingl k *Power over Ethernet*)
6. MIPI DSI (inglise keeles *Mobile Industry Processor Interface Display Serial Interface*)
7. MIPI CSI (ingl k *Mobile Industry Processor Interface Camera Serial Interface*)
8. TRRS (ingl k *Tip-Ring-Ring-Sleeve*)
9. eMMC (ingl k *Embedded MultiMediaCard*)
10. microSD (ingl k *Micro Secure Digital*)
11. GPIO (ingl k *General Purpose Input/Output*)
12. EEPROM (ingl k *Electrically Erasable Programmable Read-Only Memory*)
13. USART (ingl k *Universal Synchronous/Asynchronous Receiver/Transmitter*)
14. I2C (ingl k *Inter-Integrated Circuit*)
15. SPI (ingl k *Serial Peripheral Interface*)
16. PWM (ingl k *Pulse Width Modulation*)
17. USB (ingl k *Universal Serial Bus*)
18. RAM (ingl k *Random Access Memory*)
19. ROM (ingl k *Read-Only Memory*)
20. WiFi (ingl k *Wireless Fidelity*)
21. SRAM (ingl k *Static Random Access Memory*)
22. ToF (ingl k *Time of Flight*)
23. WAP (ingl k *Wireless Access Point*)
24. PC (ingl k *Personal Computer*)
25. WLAN (ingl k *Wireless Local Area Network*)

SISSEJUHATUS

Autonoomsetel sõidukitel on ühiskonnas pidevalt suurenev roll. Nende laiapõhjalise kasutuselevõtuga kaasneksid mitmed eelised. Täiustades tehnoloogiat, mis isesõitvaid sõidukeid juhib, on võimalik teha liiklus ohutumaks läbi mõneti ettearvamatu inimkomponendi rolli järk-järgulise minimeerimise ja sõidukitevahelise suhtluse võimaluste arendamise. Kui panustada isesõitvate sõidukite ja sellega kaasneva toetava infrastruktuuri arendamisesse piisavalt ressurssi, võiks kaugem tulevik olla liiklusõnnetustest vaba ning meie teedel ei peaks enam kunagi kedagi hukkuma. Samuti kaasneb autonoomse transpordiga potentsiaal ajakulude ja saaste vähendamiseks. Hästi toimima saadud tuleviku liiklussüsteem võiks olla piisavalt optimeeritud, et liiklusummikud jäävad minevikku ning linnaliikluses pidevalt kohapealt kiirendavad autod ei saasta enam nii palju meie Maa niigi saastatud atmosfääri. Selleks et ühel päeval selliste kasudeni jõuda, tuleb teha palju arendustööd ja tuleb kaaluda mitmeid võimalusi iseliikuvate sõidukite realiseerimiseks. Käesoleva lõputööga üritab autor luua veel ühe lisavõimaluse tuleviku insenerides valdkonnahuvi äratamiseks ja esimese robootikakogemuse tekitamiseks läbi soodsa ja kättesaadava isesõitva robotplatvormi, mida oleks võimalik tulevikus erinevates kooliastmetes õppetöö eesmärgil kasutada.

Käesoleva bakalaureusetöö sooritamiseks andis esialgse idee professor Anton Rassõlkini pakutud teema, mille raames oli vaja koostada ja avalikustada olemasoleva isesõitva roboti kohta juhendmaterjalid, et neid saaks edaspidi õppetöös kasutada. Arutelu käigus tekkis idee, et käesoleva bakalaureusetöö raames saaks otsida ka võimalusi roboti edasiarenduseks. Kuna soov ise robotit ehitada oli suur, siis edasiarenduse võimaluste otsimise käigus tekkis idee kasutada sarnase funktsionaalsusega roboti ehitamiseks mõnda muud arvutiplatvormi ja panna kokku alternatiivne versioon isesõitvast robotist, saades selleks abi olemasolevatest juhendmaterjalidest, mille kasutamist võimaldas kaasjuhendaja Diana Belolipetskaja. Et õigustada alternatiivse lahenduse ehitamist, pidi uuel variandil olema olemasoleva ees mingi märgatav eelis. Lõputöö kirjutamisele eelnev ja selle käigus toimuv taustauuring ja erinevaid monoplaatarvuteid ja mikrokontrollereid tutvustav teoreetiline osa tekitavad vajaliku arusaama erinevate platvormide võimalustest ja täpsustavad eesmärke, mida on võimalik käesoleva bakalaureusetöö praktilise osa raames saavutada. Teoreetilises osas on välja toodud võrdluse kriteeriumid, kirjeldatud olemasolevat Raspberry Pi-l põhinevat robotplatvormi ja võrreldud erinevaid monoplaatarvuteid ja mikrokontrollereid. Lõputöö praktilises osas on kirjeldatud projekti üldist ülesehitust, teostuse realiseerimiseks vajaminevaid katsetusi ning ületatud raskuspunkte ning samuti on antud ülevaade saavutatud lõpptulemusest.

1. ERINEVATE MONOPLAATARVUTITE JA MIKROKONTROLLERITE VÕRDLUS, OPTIMEERITAVATE PARAMEETRITE LEIDMINE

1.1 Lõputöö eesmärk, protsessi kirjeldus

Käesoleva bakalaureusetöö lõppeesmärk on olemasoleva isenavigeeruva Raspberry Pi monoplaatarvutit kasutava robotplatvormi põhjal ehitada soodsam ja sarnase võimekusega robotplatvorm, mis on võimeline tuvastama pildi/videotöötlustarkvaraga sobivat liikumistrajektoori ning seda iseseisvalt läbima. Olemasolev robot on kokku pandud Diana Belolipetskaja ja Daniil Valme poolt. Antud robotplatvormi kasutusala oleks haridusvaldkonnas, tutvustamaks põhikooli- ja gümnaasiumiastme õpilastele robootikat läbi praktilise näite, lastes neil instruktsioonide põhjal standardiseeritud robotit ise kokku panna, programmeerida ja käitada. Töö tulemuseks olev robotplatvorm peab olema lihtsalt kättesaadav ja roboti kokkupanek peab olema loodavate õppematerjalide poolt arusaadavaks tehtud. Selle ülesande teostamiseks on vajalik erinevate robootikas kasutatavate monoplaatarvutite ja mikrokontrollerite võrdlus, et kaardistada võimalusi lõputöö eesmärgi saavutamiseks. Lisaks on tarvis põhjalikult analüüsida olemasolevat robotit, mis töötab Raspberry Pi Model 4B monoplaatarvuti baasil. Võrdluse käigus kirjeldatakse tehnilisi parameetreid, leitakse näiteid kasutusest sarnastes robotplatvormides ja tuuakse välja tugevused ja nõrkused käesoleva lõputöö kontekstis.

Koostatava võrdluse eesmärgiks on identifitseerida olemasoleva roboti nõrkusi ning alternatiivsete lahenduste tugevusi, et ehitada töö teises osas robot, mis on olemasolevast lahendusest mingite parameetrite järgi parem. Samuti pole valikust väljas olemasoleva roboti osaline ümberehitamine, et paremini sihtparameetreid täita. Võrdluse tulemuste põhjal formuleeritakse alternatiivse roboti poolt parandatavate parameetrite nimistu ja seatakse nõuded selle koostule. Alternatiivse robotplatvormi poolt parandatavate parameetrite valik peab sobima seejuures antud roboti lõpliku kasutusala konteksti. Näitena ei ole mõtet ehitada väga kallist robotit, mis on kordades võimsam olemasolevast lahendusest, sest liiga kallis robot oleks ebapraktiline hariduslikus kasutuses ja piiraks selle kättesaadavust koolidele ja robootikahuvilistele õpilastele.

2. EDASIARENDUSE PÕHJAKS OLEV ROBOTPLATVORM

2.1 Üldine kirjeldus ja võrdluskriteeriumid

Edasiarenduse põhjaks olev robotplatvorm ehitati Tallinna Tehnikaülikooli aine EEX5030 Kaasaegne transport raames, eesmärgiga anda üliõpilastele baasteadmised tulevikus võistluste raames isesõitvate robotite konstrueerimiseks. Peale esialgse eesmärgi täitmist, tekkis idee jagada robotiga seonduvaid materjale ja koostejuhiseid teistele koolidele hariduslikul eesmärgil ning see on ka käesoleva lõputöö väljundiks. Diana Belolipetskaja ja Daniil Valme poolt kokku pandud robot on põhjaks autoripoolsele edasiarendusele, mille käigus soovib autor arendada alternatiivse, odavama robotplatvormi, mis ei jää võimekuselt oluliselt alla olemasolevale robotlahendusele. Eesmärgiks on soosida robotplatvormi kasutamist laiema audientsi poolt, kelle jaoks võib Raspberry Pi-l põhineva robotlahenduse relatiivselt kõrge hind takistada kasutuselevõttu. Autoripoolse robotivariandi funktsionaalsus on suuresti inspireeritud eelmainitud tudengite poolt kokku pandud roboti funktsionaalsusest. Käesoleva lõputöö väljund peab saavutama mikrokontrolleriga teostatava roboti puhul olemasolevale robotile võimalikult lähedase võimekuse oma põhiülesannete täitmisel. Autor on võimekuse kriteeriumid defineerinud primaarseteks ja nende saavutamiseks vajalikeks sekundaarseteks kriteeriumiteks:

- Aeg, mis kulub etteantud trajektoori läbimiseks kontrollitud tingimustes.
    - Juhtimisprogrammi tuvastamisvõimekus (ilmselt kvalitatiivne hinnang).
    - Videopildi kaadrisagedus.
    - Andmeedastuse kiirus töötlevale seadmele.
    - Töötleva seadme töötlemise kiirus.
    - Andmeedastuse kiirus töötlevalt seadmelt.
    - Roboti liikumiskiirus.
    - Roboti keeramiskiirus.
    - Erinevate valgustingimuste mõju juhtimisprogrammi tööle.

- Roboti maksimaalne tööaeg.
    - Sobiva akusüsteemi valik.
    - Energiatarbe juhtimine
- Efektiivne andmesidestuse distants.
    - Roboti andmesidestuse maht ajaühikus.
    - Roboti andmesidestuse töösagedus ja lainekarakteristikud.
    - Töötleva seadme andmesidestuse maht ajaühikus.

- Töötleva seadme andmesidestuse töösagedus ja lainekarakteristikud.

2.2 Komponentide nimistu ja spetsifikatsioonid

Lõputöö baasiks oleva roboti nö "ajuks" on Raspberry Pi Model 4B. Raspberry Pi Model 4B on põhjalikult kirjeldatud käesoleva töö esimeses peatükis. Raspberry monoplaatarvutile annab võimu 10 000 mAh mahtuvusega akupank AlphaQ 10. Antud roboti kaamerafunktsiooni täidab Raspberry Pi poolt toodetud kaameramoodul SC0023, mis töötab resolutsioonidel 1080 x 720 ja 640 x 480, kasutab liidestumisel CSI protokolli ja toodab videomaterjali RAW formaadis. Roboti kere koosneb kahest osast. Üks on akupanka sisaldav 3D prinditud karp, mis on disainitud Solidworks tarkvara abil. Karbi peale paigutub Raspberry Pi monoplaatarvuti ja kahe liigendühendusega struktuur, mille abil on võimalik kaamerat liigutada. Karp kinnitub nelja jalaga akrüülplastikust plaadile. Plaadis on mitmeid avausi, et roboti komponente juhtmetega edukalt ühendada ning et saaks kinnitada roboti erinevaid vajalikke osi. Plaadi küljes on roboti kolm ratast, üks neist on ees, nina küljes ja aitab roboti juhtimisele kaasa ja kaks neist on ühendatud kahe 50 g-se DFRobot DC-mootoriga, mille küljes on ka SJ-01 enkoodrid. Plaadi küljes on ka patareihoidja, milles on jadamisi ühendatud 4 AA patareid, mis annavad tööks vajalikku elektrit mootorite tööd juhtivale L298N konfiguratsioonil põhinevale mootori draiverile. Draiver on 25 W võimsusega, ja suudab taluda sisendvoolu vahemikus 0-46 VDC.

Robotit saab edukalt programmeerida, kuna Raspberry Pi-ga on ühendatud mälukaart, millele on installitud Raspbiani operatsioonisüsteem. Ühendades Raspberry Pi monitoriga (HDMI kaabel monitori külge, HDMI/micro-HDMI adapter teise otsa ja micro-HDMI Raspberry Pi külge), on võimalik robotit sel viisil programmeerida. Selleks oli vaja ühendada Raspberry Pi-ga ka klaviatuur ja hiir, mis on võimalik tänu USB 3.0 portidele. Edasiarenduse põhjaks oleva roboti kood pärineb Diana Belolipetskaja materjalidest, mis on kokku pandud üliõpilaste juhendamiseks ja töö autorile lõputöö sooritamisel abimaterjaliks saadetud. Robot kasutab enda ülesannete täitmiseks erinevaid teeke, suur roll on OpenCV-l ja NumPy-l, et teostada esemete tuvastamist ja selleks vajalikku tööd andmemassiividega. OpenCV on avatud lähtekoodiga C++ keele teek, mida saab kasutada nii C++, Python kui ka Java programmeerimiskeeltega, samuti liidestub teek MATLAB programmiga. OpenCV-d iseloomustab mitmekesisus, teek sisaldab üle 2500 algoritmi, mis on valdavalt suunitletud masinnägemisülesannete täitmiseks. OpenCV kodulehel kirjeldatakse algoritmide funktsionaalsust järgnevalt (ingl. k): "Järgmisi algoritme saab kasutada nägude tuvastamiseks ja äratundmiseks, objektide identifitseerimiseks, inimtegevuste klassifitseerimiseks videotes, kaameraliikumiste jälgimiseks, liikuvate objektide jälgimiseks, objektide 3D-mudelite eraldamiseks,

3D-punktipilvede loomiseks stereokaameratest, piltide ühendamiseks kõrge resolutsiooniga stseeni üldpildi loomiseks, sarnaste piltide leidmiseks pildibaasist, punaste silmade eemaldamiseks välguga tehtud piltidelt, silmade liikumise jälgimiseks, maastiku äratundmiseks ja markeerimiseks, et see üle katta täiendatud reaalsusega, jne.” Selle teegi kasuks räägib muuhulgas madal võimsustarve, suhteline kasutamislihtsus ja võimekus teatud ülesandeid paralleelis teostada, kiirendades pildi/videomaterjali töötlemist.

OpenCV tööd teeb efektiivsemaks NumPy teegi kasutus. NumPy on mitmekülgselt võimekas matemaatiliste operatsioonide teostaja, pakkudes kasutajale laia funktsioonide valikut ja suurt kiirust. OpenCVs on levinud praktika pilte kujutada pikslitest koosneva NumPy andmemassiividena ning NumPy ndarray andmestruktuur lubab neid massiive mugavalt töödelda. Taoline töödeldavus on kasulik piltide eeltöötluseks ja piltidelt erinevate elementide kättesaamiseks. NumPy on OpenCVga tugevalt integreeritud, kasutades NumPy massiive algoritmide sisendite ja väljunditena. NumPy matemaatiline võimekus loob eelduse piltide parendamiseks, filtrite kasutamiseks, piltide omaduste muutmiseks ja erinevate tunnuste pildilt kättesaamiseks. Lisateegid nagu Matplotlib toimivad hästi koostöös OpenCV ja NumPy-ga andmete visualiseerimiseks.

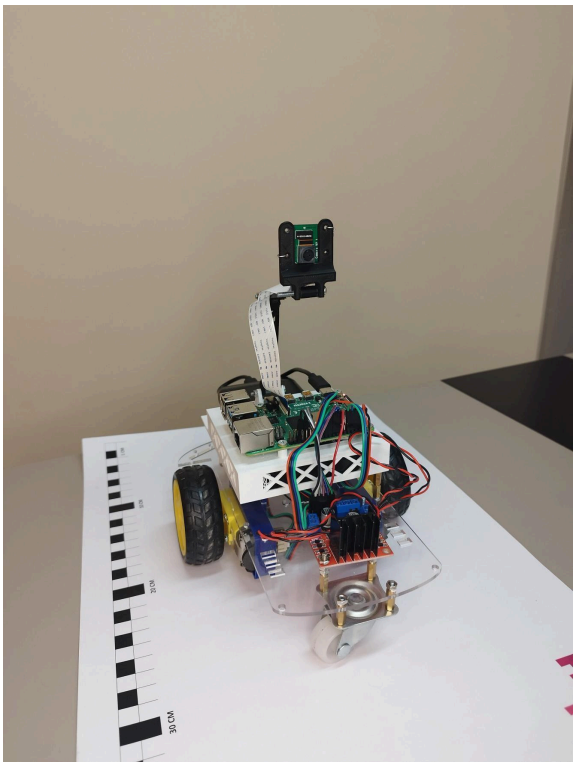
Pilditöötluseks kasutatakse näitekoodis erinevaid OpenCV funktsioone. OpenCVs on pildi värviformaat esindatud kolme kanaliga: sinine, roheline ja punane, mille väärtused vahemikus 0-255 annavad kokku iga piksli värvi. Värviformaat on vaikimisi BGR, aga funktsioon cvtColor() lubab muuta pildi värviskeemi, näiteks BGR - RGB, BGR - Gray jms värviskeemid. Olemasolevas koodis kasutatakse kolme filterfunktsiooni. Filtrid aitavad esile tuua pildi teatud karakteristikuid või neid muuta. Esimene neist on Gaussi hägufilter, mille eesmärgiks on vähendada pildimüra ja muuta pilti siledamaks. Gaussi hägufilter aitab järgnevatel filtritel tulemuslikum olla, nimelt toetab see äärtetuvastamisalgoritmide tööd. Antud filter võtab vastu X ja Y argumenti, mis määravad hägufiltri tugevuse. Teiseks kasutatakse lävendamise funktsiooni, mis teeb hallist pildist binaarpildi. Gray formaadis pildil on musta ja valge kanalid väärtustega 0-255, aga binaarpildil on 0 valge piksel ja 1 must piksel. Selline pildi lihtsustamine säilitab mõistlikul määral pildi sisu, aga vähendab edasiste pilditöötlusoperatsioonide keerukust olulisel määral.

Töö baasiks oleva roboti jaoks on olemas kood DC mootorite juhtimiseks/testimiseks, on olemas kood kaamerast video käivitamiseks ning selle kuvamiseks ning on loodud ka mitu koodijuppi, mille ülesanne on ruumis navigeerida. Käsu argumentidega on võimalik

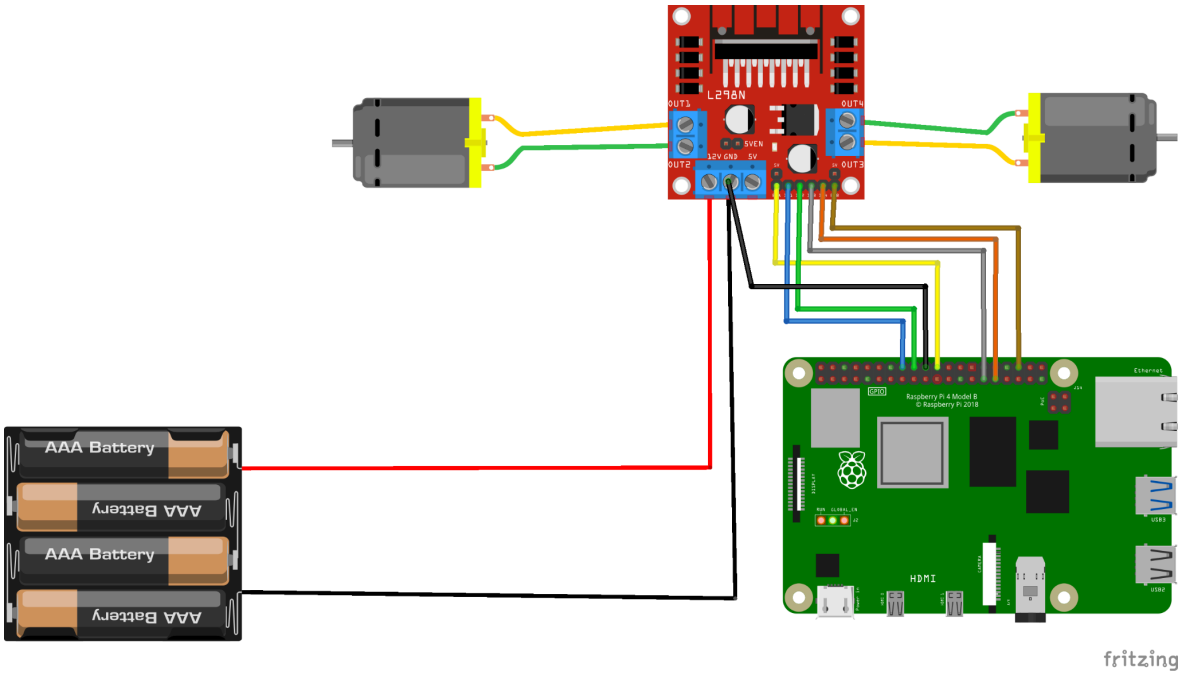
paika panna, alates millisest piksli väärtusest piksleid lävendatakse ja lävendamismeetodeid on samuti erinevaid.

Peale pildi värviformaadi halliks muutmist, hägufiltri ja lävendamise rakendamist võetakse kasutusele funktsioon nimega “Canny edge detector”, mis kasutab mitmetasandilist algoritmi, et tuvastada pildil intensiivsuse muutusi, mis kujutavad endast pildil leitavaid servi. “Canny edge detector” võtab vastu kaks lävendit. Intensiivsuse muutumise piirkonnad, mis jäävad alla esimese lävendväärtuse, ei loeta servadeks ja surutakse algoritmi poolt tulemuspildil alla. Piirkonnad, mis jäävad esimese ja teise lävendi vahele klassifitseeritakse nõrkadeks servadeks ja piirkonnad, mille intensiivsuse muutus ületab teist lävendit, klassifitseeritakse tugevateks servadeks.

Kui pilt on servatuvastusalgoritmiga juba töödeldud, on võimalik kasutada “Hough Line Transform” nimelist teisendust, et tuvastada pildil sirgeid jooni. Käesolev teisendus on võimeline tuvastama sirgeid jooni ka siis, kui need pole ideaalselt sirged, neid pole ideaalselt näha, või kui pildil esineb muid segavaid faktoreid. “Hough Line Transform” otsib pildil olevaid järjestikkuseid punkte. Servatuvastusalgoritmiga leitakse esmalt servad. Iga servapunkt pildil muundatakse parameetriliseks ruumiks nimetatud Hough'i ruumis kõveraks. Selles ruumis esindab iga kõver võimalikku servapunkti pildil. Mida rohkem kõveraid Hough’i ruumis üksteisega mingis ruumi punktis lõikuvad, seda suurem on tõenäosus, et servapunktid, mida kõverad esindavad, moodustavad töödeldaval pildil sirge joone. Robotplatform kasutab “Hough Line Transform”-i, et tuvastada teibiribasid, millest oma navigatsiooniprotsessil lähtuda. Lisaks eelkirjeldatud meetodile on võimalik otsida pildilt ka teatud värvi piksleid, et sel viisil roboti liikumisteekonda suunata.



Joonis 2.1. Pilt olemasolevast Raspberry Pi-l põhinevast robotlahendusest.



Joonis 2.2. Raspberry Pi-l põhineva robotplatformi ühendusskeem Fritzingu tarkvaras.

3. LEVINUMAID MONOPLAATARVUTIPLATVORME JA MIKROKONTROLLEREID

3.1 Raspberry Pi Model 4 B

Esmalt kirjeldan Raspberry Pi Model 4B monoplaatarvutit. Olemasolev robotlahendus kasutab just seda monoplaatarvutit ning seetõttu on sobilikuim alustada just sellest variandist.

Raspberry Pi Model 4B on 64-bitine neljatuumalise 1.5 GHz sagedusel töötava protsessoriga monoplaatarvuti, mis toetab kahte monitori kuni 4K resolutsiooni juures, suudab töödelda videot 4K 60 FPS, valikus on 1, 2, 4 või 8 GB RAM-i. Toetab traadita LAN võrku nii 2.4 GHz kui ka 5.0 GHz sagedusel, arvutil on Bluetooth 5.0, Gigabit Ethernet, kaks USB 3.0 ja kaks USB 2.0 porti ja PoE võimekus (lisaliidesega). Arvutil on 40 piniga GPIO osa, kuhu saab ühendada erinevaid mooduleid. Video ja audio koha pealt on Raspberry Pi-l 2 micro-HDMI porti (kuni 4K 60 FPS võimekus). 2-realine MIPI DSI "display" liides, 2-realine MIPI CSI kaameraliides ja 3.5 mm audio/video "TRRS" port. Saab kasutada H.265 ja H.264 videokompressiooni standardit. Multimeedia võimekusest on olemas ka sisseehitatud OpenGL liides (OpenGL ES, 3.0 graafika). Raspberry Pi 4B-l on ka micro-SD mälukaardi kasutamise võimekus. Sisendvõimsuseks on 15 W (5 V DC nii USB-C kui ka GPIO kaudu, minimaalselt 3 A juures). Kasutustemperatuur 0-50 ºC. Monoplaatarvuti mõõtmed on 85 mm x 56 mm. [1]

Raspberry Pi Model 4B-d peetakse võimekaks robotiplatvormiks, mille abil on võimalik käitada paljusid tarkvaramooduleid, sh. keerukaid süva-õppimisalgoritme [3]. Model 4B arvutusvõimsus on võrreldav odavamapoolse x86 operatsioonisüsteemiga lauaarvutiga [1]. Heaks näiteks Model 4B võimekusest võib pidada 2023 CSITSS konverentsil esitletud robotit, "Digital Assistant Robot", mis on võimeline iseseisvalt liikuma, tundma ära erinevaid inimesi (sh mitmeid samaaegselt) ning interakteeruma nii õpilaste, kui õpetajatega, neid erinevatel viisidel toetades ja abistades õppetööga seotud ülesannetes. [3]. Võimekus kasutada tehislikke närvivõrgustikke videost reaalajas esemete tuvastamiseks illustreerib käesoleva mudeli suurt võimekust.

Teise näitena võib tuua roboti, mis on kokku pandud Diana Belolipetskaja ja Daniil Valme poolt, ent seda kirjeldatakse töö järgmises peatükis põhjalikult.

Raspberry Pi Model 4B tugevustena saab välja tuua: Arvutusvõimsuse ja suuruse suhte, suure liideste arvu, GPIO pinnide suure arvu, mis lubab luua paljude ühendustega keerukat süsteemi ja üleüldise mitmekülgsuse. Model 4B on alternatiividega võrreldes väiksem vajadus lisaliideste järele, kuna need on sellel juba olemas. monoplaatarvutit on võimalik isegi otse monitoridega ühendada ja sisse on ehitatud pildituvastusotstarveteks kasulik OpenCV sardtarkvara.

Model 4B-l on kaks suurt nõrkust: Esiteks, maksab käesolev süsteem rohkem kui paljud lahendused, millega analoogseid ülesandeid täita saab, lihtsama otstarbega robotite jaoks on mõistlikum soetada odavam mikrokontroller täita limiteeritumat kogust ülesandeid. Eestist kohapealt saab soetada Raspberry Pi Model 4B 8 GB varianti Oomipoes 129 euro eest. Välismaalt tellides õnnestus leida pakkumine 62 euro eest [4][8][2]. Võrreldes mikrokontrollerlahendustega, on Raspberry Pi Model 4B oluliselt kallim, ning hariduslikul eesmärgil kasutamiseks on tähtis hoida robotplatvormi hind võimalikult madalal, et selle kättesaadavus oleks maksimaalne. Teiseks, Raspberry Pi 15 W võimsus on kordades suurem mõneti analoogsetel eesmärkidel kasutatavate mikrokontrollerite võimsusest. Suur võimsus vajab mahukamat toitelahendust, mis piirab lahenduste kompaktsust ja maksumust. Lisaks võib välja tuua, et Raspberry Pi 4B on lauaarvutiga võrreldes kompaktne, kuid on siiski oluliselt suurem selle bakalaureusetöö raames võrdluseks kasutatavatest mikrokontrolleritest, mida kaalutakse samalaadse ülesande lahendamisel töö teises osas.

3.2 BeagleBone AI-64

Kuigi käesoleva töö praktiline osa teostatakse praeguse hüpoteesi kohaselt soodsa mikrokontrolleri baasil, on töö autori hinnangul hädavajalik tuua juba käsitletud lahendusele ka üks võrdluspunkt, et saada paremat ülevaadet monoplaatarvutite tugevustest ja nõrkustest. Selleks käsitletakse sellist platvormi nagu BeagleBone AI-64.

BeagleBone AI-64 kasutab enda disainis süsteemikiipi TDA4VM, mis on originaalselt suunitletud täitma autotööstuse vajadusi ning antud kiip on disainitud, pidades silmas töökindlust ja kiirust. Süsteemikiibi eesmärk AI-64s on tõsta süsteemi andmetöötlusvõimsust, graafikavõimekust, video- ja pilditöötlusvõimekust ja toetada veel muid süsteemi funktsioone. AI-64 kasutab 64-bitist Dual Arm Cortex-A72 2.0 GHz protsessorit. Lisaks on monoplaatarvutil palju alamsüsteeme, nagu: programmeeritav tööstuskommunikatsiooni alamsüsteem, töötluskiirendid masinnägemist kasutavate

[2] Hinnad 18.03.2024 seisuga

ülesannete jaoks, kuni kolm mikrokontrollerüksust Dual Arm Cortex-R5F MCU, C71x digitaalne signaaliprotsessor, maatrikskorrutamiskiirendi, mis toetab C71x tööd ja mitmed muud displei-, turva-, ja jõudluseesmärkidega tegelevad alamsüsteemid. Lisaks
on AI-64l kaameraühilduvus (CSI_RX_IF ja CSI_TX_IF), sügavus- ja liikumistöötluse kiirendi, graafikakaart ja kahe-tuumaline video enkooder+dekooder. Välisseadmetena on kasutusel erinevaid perifeerseid ühendusseadmeid (ainult 10 GPIO moodulit), mh välkmäluliidesed ja audioliidesed. BeagleBone AI-64 kasutab 4 GB RAM moodulit ning 16 GB eMMC mälu ja jõuallikas on läbi USB-C või eraldi DC ühenduse (5 V, min 3 A). Seadmel on microSD ühilduvus.

Toodetud 2022. aastal, kirjeldab andmeleht seda monoplaatarvutit üpris protsessorikeskselt. AI-64 töötab Dual Arm Cortex-A72 protsessori baasil ja kasutab mitmekülgset liideste valikut, et kasutaja "saaks kogeda" protsessori võimekust. Andmelehel, peatükk 4, on öeldud, et BeagleBone AI-64 pole mõeldud täielikuks arendusplatvormiks. Samuti on ära toodud, et plaat pole loodud võimekusega koheselt kasutama kõiki selle liideseid ja kasutajat julgustatakse ise kujundama tarkvara ja riistvara, mis võimaldaks kasutada BeagleBone AI-64 protsessori võimekust ja suurt liidestamise võimalust [5].

Otsides artikleid/lahendusi, kus BeagleBone AI-64 on kasutatud, ei leia sisuliselt mitte midagi. AI-64 on nii uus süsteem, et seda pole jõutud veel kasutada või seda kasutavate süsteemide kohta pole jõutud veel artikleid avaldada. BeagleBone AI-64 kasutamist arutati ajakirja "Sustainable Computing: Informatics and Systems" 34. väljaandes. Artikkel, mille kirjutasid A.Albanese, M.Nardello ja D.Brunelli, käsitleb sobivate monoplaatarvutite leidmist mehitamata lennusõidukitele. Autorite lahendatav probleem on mehitamata lennusõidukite jaoks vajaminek komplekt piisavast monoplaatarvuti võimsusest, madalast energiatarvest ja väikestest/kergetest mõõtmetest. Ülesande kontekstis tuuakse välja, et BeagleBone on kõrgema sooritusvõimega, kui Raspberry Pi, aga selle energiatarve on artiklis käsitletud eesmärgiks liiga kõrge. Tuuakse välja, et BeagleBone-l puudub selline arendajate ja kasutajate kommuun nagu Raspberry Pi-l ning arendustegevuse käigus esinevaid probleeme võib olla sellest lähtuvalt keerulisem lahendada. Samuti tuleb välja, et artiklis kirjeldatud lennumasina arendustegevuse käigus ei õnnestunud lennumasinaga kasutatavat kaamerat BeagleBonega liidestada. Sellest võib järeldada, et kuigi BeagleBone AI-64-l on palju erinevaid liideseid ja see on teatud ülesannete täitmiseks väga võimekas, siis võib kohati esineda probleeme riistvaraga liidestamisel. Seevastu

Raspberry Pi paistab silma selle poolest, et seda on võimalik peaaegu, et kõigega liidestada. [9]

BeagleBone AI-64 on süsteem, mille võimekus tehisintellekti kasutavates rakendustes on kindlasti suurem, kui eelkirjeldatud Raspberry Pi 4B puhul [5],[9]. AI-64 on tugevalt spetsialiseeritud just tehisintellekti kasutatavate ülesannete jaoks, kui Raspberry Pi 4B on võrdluseks universaalne, selle kasutusala pole liideste kaudu niivõrd selgelt defineeritud. Võrdlusest järeldub, et kahe erineva monoplaatarvuti sihtturud on erinevad. Siinkohal tuleb tähelepanu pöörata BeagleBone AI-64 kõrgele hinnale. Internetiotsingu tulemusel leitud odavaim AI-64 trükkplaat maksab seisuga 18.03.24 173 eurot, seevastu on võimalik tellida 4 GB RAM-iga Raspberry Pi Model 4B juba alates 62 euro hinnapiirist [6],[8]. Inimene, kes tellib BeagleBone AI-64, on valmis esiteks rohkem maksma ja teiseks, vaeva nägema, et osta või arendada tarkvara/riistvara, et täielikult AI-64 võimekust ja spetsialiseeritud liideseid ära kasutada. BeagleBone AI-64 kasutaja on tõenäoliselt seotud tööstusega või on juba varasemate kogemustega kõrgtaseme hobiarendaja. Kuna AI-64 on kallis, ei pruugi liidestuda nii paljude perifeersete seadetega, kui RaspBerry Pi, sellel on vähem ebaspetsiifilisi GPIO liideseid ja sellel puudub samaväärne tugikommuun, siis on AI-64 hariduslikul otstarbel algtaseme robootika kogemuse saavutamiseks ebasobilik töövahend [9]. Õppe korraldamisel oleks raske põhjendada 170 eurose seadme soetamist, kui piisava funktsionaalsuse saaks ka ligi kolm korda odavamast seadmest.[3]

Albanese et. al artiklis [10] kirjeldati lühidalt ka sellist seadet, nagu Nvidia Jetson Nano. Jetson Nano kasutab "edge- computing" lähenemist, mis on võimaldab teostada osasid protsesse pilves, pakkudes ka suurt andmevahetuskiirust ja suurt võimekust robootikaga seotud ülesannete täitmisel. Lugedes Nvidia lehekülge, hakkas silma, et hinnauuring paigutas Jetsoni hinna lähemale AI-64le kui Raspberry Pi-le. Jetson Nano-l põhinevat lahendust oleks mõistlik ehitada, kui eesmärk oleks tõsta seadme võimsust ja andmetöötluskiirust ning teostuse rahaline pool ei oleks probleemiks. Autor leiab peale mõningat võrdlustegevust, et käesoleva bakalaureusetöö raames ei ole üldse mõistlik kaaluda seadmeid, mis on Raspberry Pi-st kallimad.

Hobirobootika lahendusi teostatakse ka mikrokontrolleritel, mille maksumus on oluliselt väiksem Raspberry Pi Model 4B maksumusest. Kiire hindade võrdlus annab mõista, et mikrokontrollerid on oluliselt odavamad, kompaktsemad ja väiksema

[3] Hinnad 18.03.2024 seisuga

energiatarbega kui monoplaatarvutid, mis teeb need eos eelistatuks käesoleva lõputöö haridusliku eesmärgi täitmisel. Lisaks odavusele, suurusele, energiatarbele ja muudele teguritele on tähtis, et valmiv robot oleks piisavalt võimas, et täita mingis mahus lihtsamaid pildi/videotöötlusülesandeid, mille täpne olemus selgub töö teises, praktilises osas. Töö võrdleva osa teises pooles võrreldakse erinevaid mikrokontrollereid, et mõista, kas mikrokontrolleritel põhinevad robotid võivad olla piisavalt võimekad, et teostada pildi- ja videotöötlusülesandeid lisaks veel roboti enda baasfunktsioonide teostamisele ja kas/milline mikrokontroller võiks olla põhjaks töö teises osas teostatavale robotile.

3.3 Arduino mikrokontrollerid Arduino Uno R3 näitel

Arduino Uno platvorm on tänapäeval levinud hobirobootika platvorm, mille populaarsus on aastate jooksul olulisel määral kasvanud [11]. Sellest tulenevast valiti esimeseks võrreldavaks mikrokontrolleriks ka Eestis, Oomipoes müüdav Arduino Uno Rev3.

Arduino Uno R3 on algajatele suunitletud robootikaplatvorm, mis reklaamib end mitmekülgsena. R3 kasutab laialdaselt kasutusel olevat ATmega328P ja ATMega 16U2 protsessorit taktsagedusega 16 MHz (ligi 100x madalam protsessori taktsagedus, kui näiteks Raspberry Pi 4B-l). Lisaks on mikrokontrolleril 32 KB välkmälu, 2 KB SRAM ja 1 KB EEPROM mälu. Välisseadmetest on R3 plaadil 3 taimerit, 2 8-bitist ja 1 16-bitine, USART liides, SPI liides ja I2C liides, analoogkomparaator ja 6 PWM kanalit. Süsteemi pinge on 2.7 - 5.5 V ja amperaaž ei ületa 100 mA (võimsus kuni 0.55 W). Süsteem saab toidet saada nii läbi USB-B pordi, kui ka läbi 2.1 x 5.5 mm toitepordi. Toitepordi kaudu võib sisendpinge olla vahemikus 6-20 V (sisseehitatud voolumuundur) ja USB-B kaudu 5.5 V. Arduino Uno arendusplaadi suurus on 69 mm x 54 mm. [12]

Artikkel, mis avaldati “Computer Science Review” 40. väljaandes, kirjeldas Arduino-põhiste süsteemide kasutusalasid uuriva metaanalüüsi tulemusi. Artikkel illustreeris Arduino laiapõhjalisust ja mitmekülgsust ning tõi erinevaid näiteid, mis tekitasid ülevaate Arduino-põhiste süsteemide võimekusest. Mikrokontrolleritel põhinevaid lahendusi kasutatakse süsteemide arenduses, tervishoius, haridusvaldkonnas, kaevandustöödel, kodude automatiseerimisel, kaitsevaldkonnas ja paljudel muudel aladel ja paljudes rollides. Käesolevat metaanalüüsi lugedes tekib arusaam, et mikrokontrolleritega saab teostada peaaegu lõpmatult palju erinevaid lahendusi. Samuti tekkis arusaam, et mikrokontrollerlahendusi kasutatakse tihti andmekogumisrollides, kus digitaal- või analoogsignaale on vaja saata mõnda teise

andmekandjasse. Metaanalüüsis oli kirjeldatud kaitsevaldkonna rakendust, mis edastas juhtmevaba öönägemiskaamera videopilti Android-seadmesse, eesmärgiga seda videopilti analüüsida ja saada sealt sõduri jaoks vajalikku luureinformatsiooni. Arduinot kasutavate süsteemide uurimisel pakuvad käesoleva lõputöö raames kõige rohkem huvi kaitsevaldkonna projektid, kuna mitmed metaanalüüsis välja toodud kaitsevaldkonna projektid hõlmavad kaamerapildi edastamist ja töötlemist. Selle bakalaureusetöö praktilise osa eesmärgiks on samuti luua robotit, mis on võimeline lisaks liikumisele edastama ka videopilti, mille põhjal siis roboti liikumist suunata saaks. [11]

Tähelepanuväärne on see, et Arduino platvormid pole piisavalt võimsad, et kaamerapilti ise töödelda ning kaamerapildi töötlemiseks tuleb kasutada lisaseadet. Sama tähelepanek kehtib tõenäoliselt ka teiste mikrokontrollerite puhul. Haridusotstarbelises kasutuses on kõige realistlikumaks kasutatavaks lisaseadmeks klassiruumis olev lauaarvuti. Kuna igas klassiruumis on arvuti tõenäoliselt olemas, siis ei pea praktilises osas kokkupandav robot suutma videotöötlusülesannet ise täita, vaid ta peab olema suuteline seda videopilti edastama, saama video/pilditöötlustarkvaralt andmeid vastu ning neid andmeid kasutama roboti mootorite juhtimiseks. Video enkodeerimiseks, dekodeerimiseks, kompressiooniks ja dekompressiooniks on tähtis võimalikult suur protsessori taktsagedus. Videoandmete ajutiseks salvestamiseks ja vajaliku firmware/software hoiustamiseks on vaja piisavalt RAM ja ROM mälu. Samuti on tarvis andmeedastuseks vajalikke liideseid. Mikrokontrollereid võrreldes on vaja leida parima hinna ja võimekuse suhe kaaludes kirjeldatud parameetreid. Arduino platvorm liidestub või on võimalik lisajuppide abil panna liidestuma väga suure riistvara ampluaaga, mis on asi, millega tuleb mikrokontrollerite võrdluses arvestada. [11]

3.4 ESP32

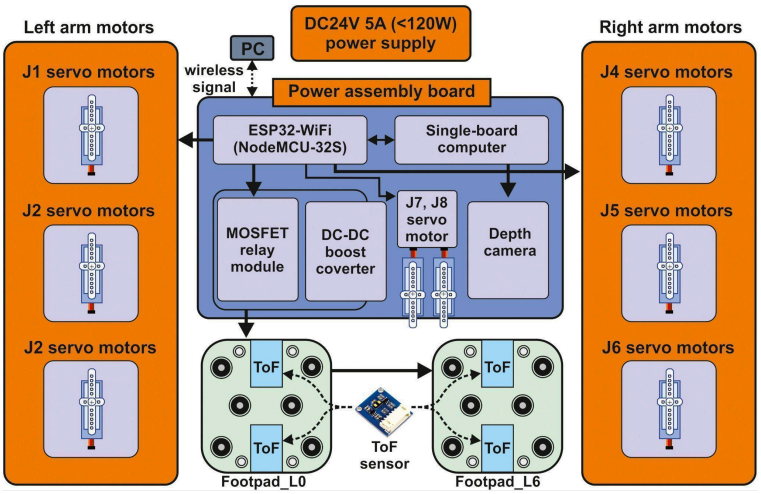
ESP32 on relatiivselt uuem perekond mikrokontrollereid, mis paistavad silma oma suure võimsuse ja suure mälumahu poolest. Olles ESP32 kohta uurinud, tundub see mikrokontrollerite perekond üpris hea kandidaat käesoleva lõputöö praktilise osa realiseerimiseks.

Säilitades relatiivselt soodsat hinda, suudavad ESP32 plaadid pakkuda kuni oluliselt suuremat protsessori taktsagedust ja kordades suuremat RAM ja ROM mälumahtu võrreldes Arduino platvormidega. ESP32 arendusplaadid on mõõtmetelt väikesed,

sobides suurepäraselt liikuvplatvormidele. ESP32 plaatidel on sisseehitatud liidestuvus WiFi ja Bluetoothiga, mis muudab andmete saatmise mugavaks. Täpsemalt on ESP32 perekonna mikrokontrolleritel 240 MHz taktsagedusega 32-bitine Xtensa protsessor, 448 KB ROMi, 520 KB SRAMi ja 16 KB RTC SRAMi (SRAM, mis on püsiv ka väljalülitamise korral). Sarnaselt Arduinoga on ESP32-l mitu ostsillaatorit ja taimerit. [13]

Otsides ESP32 baasil ehitatud robotplatvorme käsitlevaid teadusartikleid, hakkab silma teostatud lahenduste mitmekülgsus ja pealtnäha on mitmed projektid keerukad ja suurt võimsust vajavad. Silma hakkas "Automation in Construction" ajakirja 2023. aasta augusti väljaandes avaldatud artikkel sildu inspekteerivast liikurrobotist, kellel on liikumiseks nii elektromagnetitega jalad kui ka servomootoriga kontrollitav ratas.

"HMICRobot" kasutab tehisintellekti, et liikuda enamike liikurrobotite jaoks keerukatel pindadel. See robot on suuteline ronima üles mööda vertikaalseid seinu, ületama takistusi ja sõitma, kõndima ja ronima. Robot muudab oma liikumisviisi vastavalt vajadusele, kohates takistusi, mida selle antud hetkel kasutusel olev liikumisviis ületada ei luba. Lisaks suurele liikumisvõimekusele, kasutab robot kaamerat ja ToF ("Time of Flight") sensoreid, et hinnata sildade konstruktsiooni seisukorda, esemete kaugust robotist ja roboti asukohta ruumis. Täpsemalt kontrollib ESP32-WIFI moodul 8 erinevat servomootorit, releemooduleid, kolme voolumuundurit, 20 elektromagnetit ja nelja ToF sensori komplekti. [15]



Joonis 3.1. HMICRoboti mehhatroonikasüsteem [15].

ESP32 võimaldab juhtmevabalt juhtida roboti mehhanisme. Mobiiltelefoni või arvutiga on võimalik ühenduda robotiga samasse WAP-i ("Wireless Access Point"). Kasutusel on

WebSocket veebiühendusprotokoll, mis lubab mitme seadme ühendumist ja kontrolli reaalajas. Antud protokolliga vähendatakse andmeühenduse anomaaliate mõju roboti toimimisele. [15] ESP32 platvormil on Arduino mikrokontrollerite ees suur võimekuse eelis, hoides seejuures hinda madalal, kohati isegi madalamal kui Arduino perekonna mikrokontrolleritel. Arduino eeliseks seevastu on suur kasutajate baas. Tõenäoliselt on lihtsam saada enda projektidega abi, kuna kasutajaskond on mingite tüüp-probleemidega kokku puutunud ja suudab neid kerge vaevaga lahendada. Samuti võib olla perifeersete seadmete valik, mida Arduinoga liidestada saaks, suurem. Kuna käesoleva lõputöö eesmärgiks oleva roboti kokkupanekut juhiksid koolide õppejõud, kelle kokkupuude robootikaga on suure tõenäosusega piiratud, siis hariduslikus kasutuses ESP32 platvormi probleemide lahendamiseks tuleks tekitada algajatele kergelt arusaadav juhiste kogum ning valida välja roboti ehitaja poolt heakskiidetud lisaseadmete nimistu, mida saab probleemideta ESP32 platvormiga ühildada.

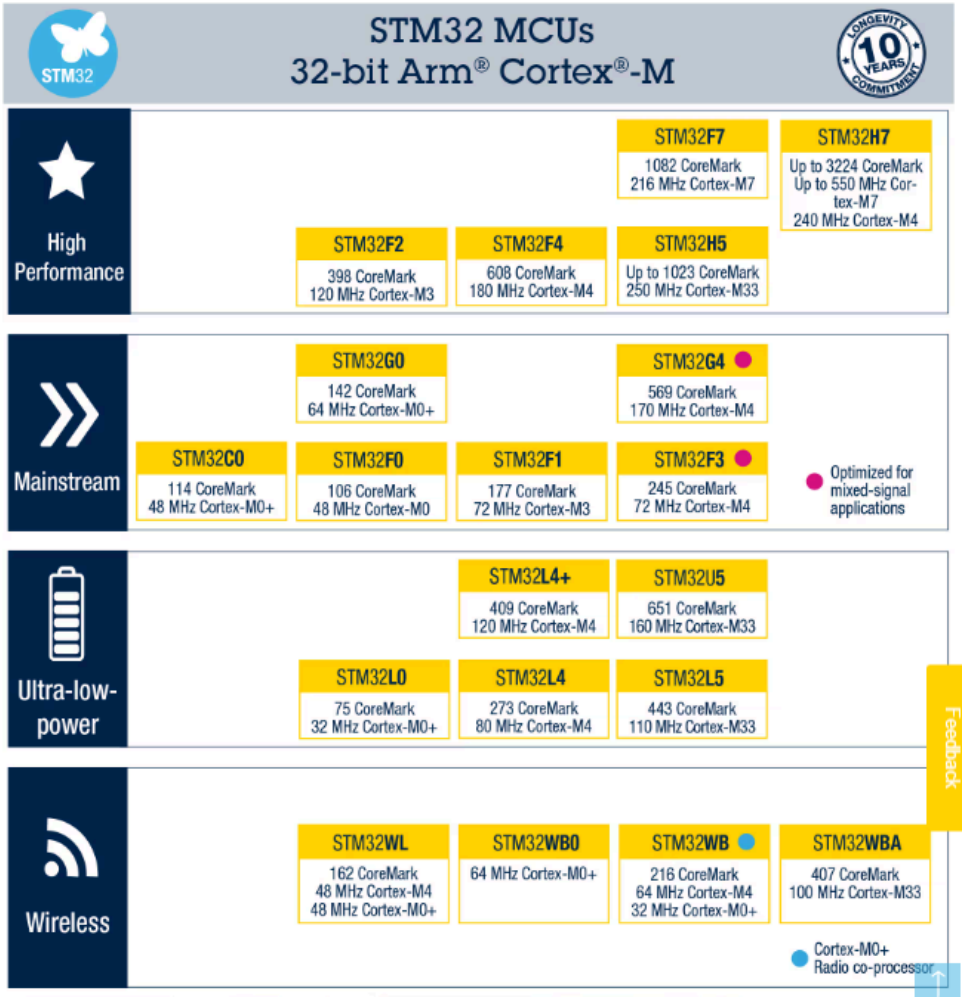
Lõputöö raames tundub ESP32 parim kandidaat töö praktilise osa teostuseks ja seda nimelt suurepärase hinna-kvaliteedi suhte ja sisseehitatud andmeside ühilduvuse tõttu. Eesti robootikatooteid müüvates poodides tootevalikut uurides hakkas silma lahendus, mis integreeris omavahel Arduino UNO R4 ja ESP32 mikrokontrollerid, vähendades mõlema platvormi nõrkuste mõju potentsiaalsele teostusele. Võimsuse probleemi lahendab ESP32, samas kui Arduino paistab silma oma võimekuse poolest paljude komponentidega liidestuda. Hinnaga 37.34 eurot Lemona Electronics elektroonikapoes [4]oleks selline lahendus olemasolevast Raspberry Pi roboti trükkplaadist ligikaudu poole soodsam, samuti oleks selle energiatarve madalam [16]. Väiksem võimsustarve lubaks suunata rohkem võimsust liikumiseks või ehitada robot odavamalt kui alternatiivne Raspberry-t kasutav lahendus. Lisaks on UNO R4 WiFi R3 seeria mikrokontrollerist samm edasi mitte ainult võimsuse, vaid ka funktsionaalsuse poolest. Rohkem perifeerseid seadmeid, sisendpinge max. väärtuse tõus 24 voldini, HID-toetus (võimekus läbi USB edastada klaviatuuri/hiire käsklusi), WiFi ja Bluetooth ühendamise võimekus läbi ESP kiibi, 12x8 LED maatriks ja veel mõned uuendused teevad UNO R4 WiFist tugeva kandidaadi lõputöö praktilise osa teostuseks. Kuna Arduino UNO platvorm on disainitud muuhulgas algajatele nende esimesteks hobiprojektideks, siis räägib ka see antud mikrokontrolleri rakendamise kasuks. [14]

3.5 STM32

Lisaks Arduinole ja ESP32-le, toon võrdlusesse ka kolmanda platvormi, STM32. STMicroelectronics on Šveitsis baseeruv mikroelektroonika tootmisele keskenduv

[4] Hind 18.03.2024 seisuga

ettevõte. Nende STM32 mikrokontrollerplatvorm pakub konkurentidega võrreldes suurt tootevalikut ja erinevaid STM32 seeria mikrokontrollereid iseloomustab suur võimsus ja mälumaht. STM32 mikrokontrollerite võimekus (võimsamad F ja H seeria) on võrreldav ESP32 platvormi kontrolleritega ja kaugelt ületavad Arduino kontrollerite võimsust. [18]



Joonis 3.2. STM32 mikrokontrollerite tootevalik. [18]

STM32 mikrokontrollerite kasuks räägivad väga põhjalikud, mitmesajalehekülgelised andmelehed, kus on kirjeldatud põhjalikult kõiki kontrolleri funktsionaalsusi ja lahendatud tõenäolisemalt esinevaid probleeme. Selline ressurss võiks olla kasulik STM32 arhitektuuril põhineva roboti kokkupanijale, kuid käesoleva ülesande kontekstis on paremaid valikuid. Lugedes STMi mikrokontrollerite andmelehti, siis saab kiiresti selgeks, et andmelehed on koostatud isikutele, kellel on varasem sügav kokkupuude robootikavaldkonnaga. Töö autori hinnangul on STM32 kasutamine lõputöö praktilise osa väljundis liiga riskantne, kuna keerukaid materjale on raske kasutada, kui lõputöö

teostamisel peaks probleeme esinema. Samuti on WiFi ühendusega variantide võimsus oluliselt madalam kui ESP32 platvormi võimsus, andes põhjust eelistada ESP32 mikrokontrollerit kasutavat robotlahendust. STM32 võimsama F-seeria plaadi STM32F4 hind on Elfa Distreleci poest 22,81 eurot[5]. See on odavam kui eelkirjeldatud ESP32 ja UNO R4 komplektlahendus, kuid on madalama võimsusega, ei paku juhtmevaba ühenduvust, vajades seetõttu lisamoodulit, et täita käesoleva bakalaureusetöö raames vajalikke ülesandeid. Samuti ei toeta STM32 kasutuselevõttu põhjalikud, ent keerukad andmelehed. Arduino platvorm on orienteeritud algajatele ning kooliõpilased saavad tõenäoliselt paremini hakkama Arduino platvormi kui STM32 platvormi kasutamisega. [17], [19]

3.6 Võrdlus ja lõpplahenduse valik

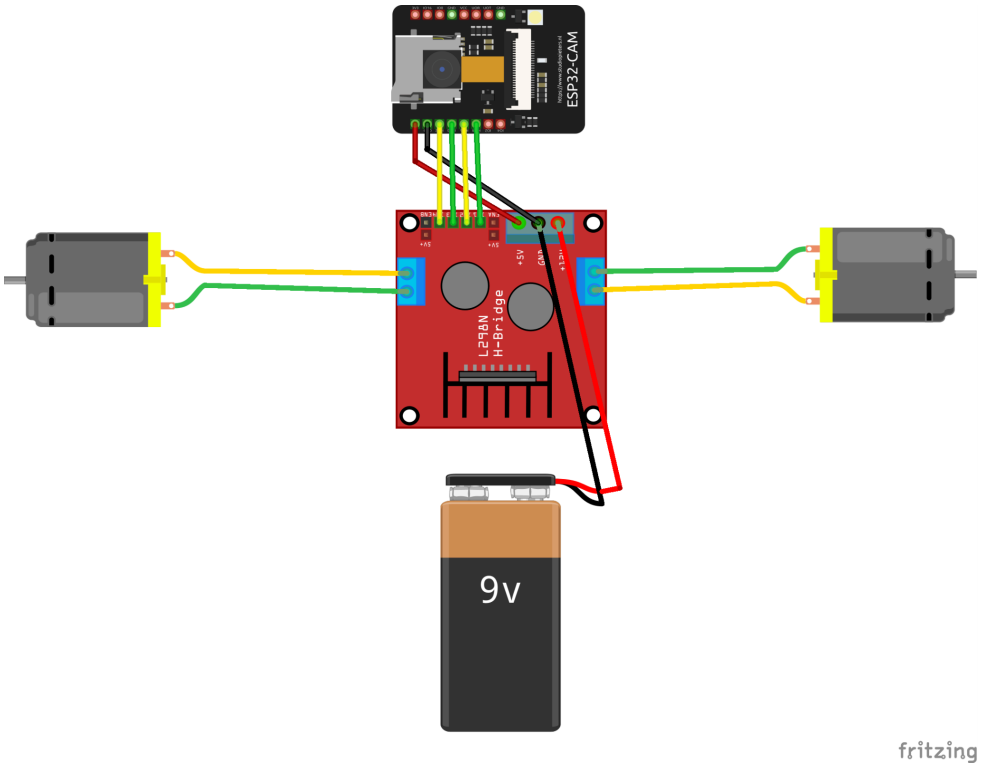
Monoplaatarvutite ja mikrokontrollerite uurimise käigus selgus, et monoplaatarvutid on peaaegu eranditult kallimad kui mikrokontrollerid. Käesoleva ülesande arvutusvõimekuse vajadust arvestades on vajadus võimsuse järele piisavalt madal, et oleks võimalik kasutada mikrokontrollerit, mitte monoplaatarvutit. Mikrokontrollerite hulgast valides otsustas töö autor lõpplahenduse teostamiseks kasutada ESP32-CAM mikrokontrollerit. Edasise uurimise käigus tuli välja, et ESP32-CAMil kardetud liidestumise probleeme ei esine ning kooslus Arduino Uno R4-ga ei ole vajalik. Soodsa hinnaga on võimalik saada nii sõiduki ajuks olev mikrokontroller, kui ka OV2640 kaameramoodul sellega komplektis. Kasutades sellist lahendust, ei ole vaja ehitada eraldi hoidikut kaamerale. ESP32-CAM on piisavalt võimas, et videopilti efektiivselt töödelda. Sellel on olemas juba valmiskirjutatud näidisprogramm, mis suudab luuda veebiserveri, kus striimitakse suhteliselt kvaliteetset ja suhteliselt suure kaadrisagedusega videopilti. Samuti on veebiserveril sisseehitatud seadete riba, kus saab kiirelt videopildi seadeid vastavalt oludele muuta. ESP32-CAM edestab kõiki mainitud konkurente hinna poolest ning samuti kõrvaldab vajaduse eraldi kaameramoodulit soetada. Madala hinna juures on siiski tegu piisavalt võimsa seadmega, et käesoleva ülesande sooritamiseks vajalikud protseduurid on võimalik teostada mõistliku kaadrisageduse juures. Roboti ehitusliku poole juures saab tugevusena välja tuua võimaluse ESP32-CAM roboti esiotsa külge lihtsalt kahepoolse teibi ribaga paigaldada ilma vajaduseta 3D prinditud korpuse või kaamera liigutamise mehhanismi järele.

[5] Hind 18.03.2024 seisuga

4. PRAKTILINE OSA

4.1 Juhtmestus ja struktuur

Lõputöö praktilise osa jaoks kokku pandud roboti komponentideks on: Ülikoolist saadud roboti kere plaat, kaks ratast samast komplektist ja kolmas, väiksem ratas roboti esiotsas (eesmärgiga lihtsustada pööramist), kaks DC mootorit, L298N mootorite draiver, ESP32-CAM mikrokontroller, 9 V 6F22 patarei ning juhtmestus ja erinevad roboti kokkupanekuks vajalikud kruvid ja mutrid. Joonisel 4.1 on roboti ühendusskeem Fritzingu tarkvaras:



Joonis 4.1. Roboti ühendusskeem Fritzingu tarkvaras.

Robot saab toidet 9 V patareist, kuna seda on mugav roboti kere külge paigaldada ja see jääb L298N 7-12 V lubatud sisendpinge vahemikku. L298N annab toidet ESP32-CAM mikrokontrollerile läbi 5 V pini ning edastab juhtimiskäsklusi kahele DC mootorile läbi OUT1-4 pinide. Patarei maandusjuhe on ühendatud L298N GND pini külge ja L298N GND pinist läheb maandusjuhe ESP32-CAM 5 V pini all paiknevasse GND pini. Mootorite juhtimine ja mootorite juhtimiseks vajaliku sisendi tekitamisega tegeleb ESP32-CAM, mis on ühendatud L298B IN1-4 pinidega läbi oma GPIO 12, GPIO 13, GPIO 15 ja GPIO 14 pinide. Plaadi alumise poole külge on kahepoolse teibiga kinnitatud 9 V patarei. Roboti kereplaati läbivad 3D-prinditud plastist kinnitused roboti

kere külgede lähedal kinnitavad DC mootorid koos juhtivate ratastega roboti külge. Roboti nina külge on lühemate poltidega kinnitatud juhtimist abistav kolmas ratas. Roboti "ajuks" olev ESP32-CAM kinnitub kahepoolse teibi ribaga roboti nina külge, kaamera on suunatud otse alla.

4.2 Roboti tööpõhimõte ja teostuse kirjeldus

Töö praktilises osas ehitatud robotplatvorm suudab iseseisvalt navigeeruda siledal pinnasel, kui sellele on tagatud musta värvi aluspind ja roboti liikumissuuna sihis jookseb roboti alt kontrastse teibi riba. Robot jälgib teibi suunda läbi kaamerapildi ja korrigeerib teibi asendist lähtuvalt oma liikumissuunda. Oma eesmärgi saavutamiseks kasutab robot kolmesammulist protsessi, mis on kirjeldatud allpool.

Esimeseks sammuks võib pidada kaamerapildi filmimist ja ESP32-CAM poolt personaalarvutile kättesaadavaks tegemist. ESP32-CAM kaameramoodul on roboti ninal otse alla suunatud ning see filmib videot, mis tehakse PC-le kättesaadavaks ja töödeldavaks. Selleks käivitab ESP32-CAM lokaalses WiFi võrgus veebiserveri, mille kaudu on võimalik filmitavale videopildile ligi pääseda. Selleks kasutatakse Webserver.h teeki, WiFi.h teeki ja esp32cam.h teeki. Programmi käivitamisel kuvatakse kasutajale ühenduse loomise edukust kommenteerivad sõnumid ning antakse IP-aadress, millel veebiserver asub. Samuti on mikrokontroller konfigureeritud kuulama ja kasutajale edastama serverisse saadetavaid sõnumeid. ESP32-CAMi-l toimiv kood on leitav töö lisas nr 2.

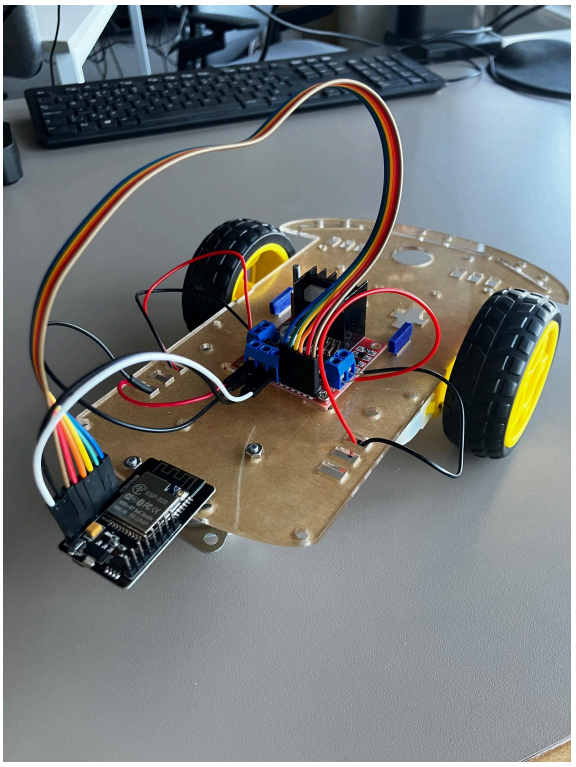
Teiseks sammuks on video tuvastamine ja töötlemine PC poolt, mille eest vastutab arvutis käivitatav Pythoni program (leitav lisa 1-st). Esmalt kasutab arvuti urllib.request teeki, et haarata veebiserverist videopilt, mis tõlgendatakse NumPy "array"-ks ning seejärel tehakse "array" uuesti videopildiks kasutades OpenCV imdecode funktsiooni. Videopilt töödeldakse seejärel OpenCV-ga "grayscale" formaati, sellele rakendatakse pilditöötlusvahend "Gaussian blur" eesmärgiga vähendada ebaolulisi detaile pildil ja seejärel tõlgendatakse pilt binaarformaati läbi "Otsu thresholding" meetodi. "Otsu thresholding" erineb tavalisest OpenCV lävendamisest, kuna "Otsu thresholding" leiab automaatselt sobivad lävendväärtused, mida pilti binaarformaati tõlgendades kasutada. See on kasulik, kuna on soovitav mitmesugustes valgusoludes mustal taustal olevat teipi võimalikult täpse piiritlusega tuvastada. Tavalise lävendamisega oleks vajalik valgusoludest olenevalt lävendamise parameetreid pidevalt muuta. Binaarformaadis pilti kasutab valgete pikslite otsimiseks mõeldud funktsioon, mis otsib viies sihis (vasakule 90° nurga all, vasakule 45° nurga

all, üles, paremale 45° nurga all, paremale 90° nurga all) selgelt piiritletud alal valget värvi piksleid. “Switch” struktuur koodis kontrollib, millistel sihtidel valgeid piksleid tabatakse. Selle eesmärk on tajuda värvilise teibi täpset suundumust roboti asendi suhtes, et robot saaks mööda seda võimalikult täpselt liikuda. Lihtsustatud näitena: Kui valgete pikslite tuvastamise funktsioon tuvastab, et valgeid piksleid asub vasakule diagonaalis suunduvas tuvastamissihis, aga mitte üles, ega paremale diagonaalis suunduvas liikumissihis, siis tuvastab robot, et teip liigub roboti suhtes diagonaalselt vasakule ning on tarvis roboti liikumissuunda korrigeerida, et jätkuvalt teipi enda liikumissuunaga samasihilisena hoida.

Kolmandaks sammuks on juhtimiseks vajaliku informatsiooni edastamine ESP32-CAM-ile, mis kasutab seda informatsiooni, et edastada juhtimiskäsklusi L298N motor driverile. Pythoni programmi “main” funktsioon edastab valgete pikslite tuvastamise funktsioonist return funktsiooniga kättesaadavaks tehtud “int” tüüpi muutujat ESP32-CAMi poolt jooksutatavale veebiserverile, kasutades selleks urllib.parse ja urllib.request teeke. Olenevalt muutuja väärtusest, liigub robot kas teatud määral paremale, vasakule või jätkab otse liikumist.

4.3 Sooritatud katsete kirjeldus ja analüüs

Roboti ehitamine ja sobivate funktsioonide välja arendamine toimus suuresti katse-eksitus meetodil. Väga palju oli vaja testida erinevaid roboti alamsüsteemide implementeerimise võimalusi, et nende hulgast selgitada välja toimiv lahendus. Testimise eesmärgiks oli rohkem toimiva lahenduse leidmine, kui toimivate lahenduste hulgast parima leidmine. Isejuhtimise saavutamine käesoleva lõputöö mikrokontrolleri ja muude riistvaraliste lahendustega on kindlasti võimalik ka mitmeid muid teid pidi saavutada, millest mõned on tõenäoliselt lihtsamini teostatavad või mõne peatükis 2.1 ära toodud võimekuse kriteeriumi alusel oluliselt paremad, kuid autori vähese praktilise robootikakogemuse tõttu on autor rahul ka kõigest toimiva, kuid mitte võimalike hulgast parima lahendusega. Võimalike lahenduste hulgast parima leidmine võib olla kunagi sooritatava magistritöö või mõnes muus võtmes uurimise teemaks. Joonisel 4.2 on kujutatud käesoleva lõputöö praktilise osa tulemina valminud robotit.

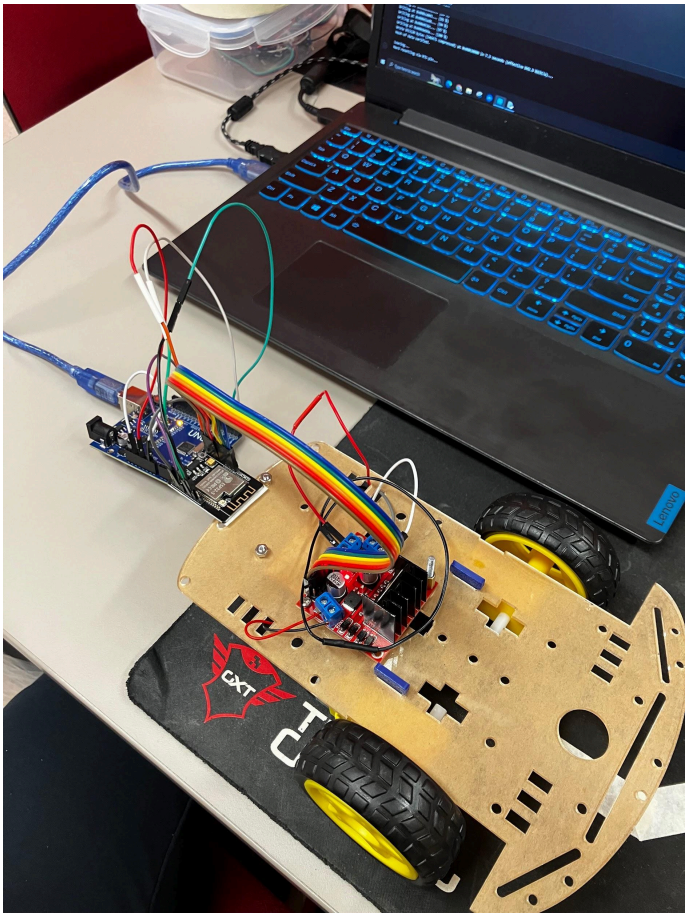


Joonis 4.2. Kokkupandud robotplatvorm.

ESP32-CAM programmeerimiseks kasutati Arduino Uno R3 mikrokontrollerit, mis ühendati USB pesa kaudu arvutiga. ESP32-ga ühenduse saamine oli paras väljakutse ning koodi üles laadimine toimib ainult väga kindlas juhtmete konfiguratsioonis. Samuti oli keeruline sobivate parameetrite Arduino arenduskeskkonnas kindlaks määramine, et koodi oleks võimalik üles laadida. Praktilise osa teostuse ühe esimese eesmärgi täitmine, ESP32-CAM videopildi kuvamine veebiserveris, õnnestus kiirelt, kuna arenduskeskkonna "Examples" sektsioonis oli leitav näidiskood, mis selle eesmärgi saavutas. CameraWebServer.h nimeline näidiskood võimaldas vaadata brauseris video voogedastust ja muuta mitmeid videot käsitlevaid seadeid.

Robotplatvormi loomise käigus sai katsetatud kõiki kolme roboti töö etappi ning sooritatud katsetused ka terviku peal. ESP32-CAM veebiserveri loomiseks sai katsetatud erinevaid koodikonfiguratsioone ning erinevaid teeke, mis on mõeldud tagama juhtmevaba ühenduse lõputöö vajaduste piires. Katsete käigus parandati järkjärgult lahenduse sobivust lõputöös kasutamiseks. Testiti erinevaid seadeid esp32cam teegi näidiskoodi CameraWebServer.h poolt loodavas keskkonnas. Kui selgus, et näidiskoodist tulevat voogedastust ei saa soovitud viisil töödelda, siis leiti alternatiivne lahendus kasutades teisi teeke. Erinevus esialgsest näidiskoodist on muuhulgas see, et ei kasutajale ei pakuta brauseris videopildi seadistamise võimalusi, sest see ei osutu vajalikuks. Kasutajaliides on täielikult teistsugune. Lisaks

kasutatakse “POST request” funktsiooni, et koguda teatud intervalli tagant andmeid veebiserverisse “client”-ina ühendunud arvutilt. Testimise käigus tuli välja, et vaja on paremat kaablit ESP32-CAM ja serial monitori ning koodi üleslaadimise jaoks vajaliku Arduino Uno R3 maanduste ühendamiseks (joonisel 4.3 kujutatud Arduino Uno kasutamine koodi üles laadimiseks ESP32-CAMile), kuna olemasolev tuli videolahenduse testimise käigus pidevalt lahti ja videopildi edastamine katkes. Videopildi kvaliteeti, töötlemist ja edastamist sai testitud nii, et arvuti andis toidet Arduinole läbi USB pordi ja sinna külge oli ühendatud ESP32-CAM, mis sai toidet Arduino 5 V pinilt, Arduino ja ESP maandused olid kokku ühendatud, Arduino RESET pin ja GND pin olid kokku ühendatud ja Arduinolt läks ESP32-CAMile RX ja TX portidest juhe vastavalt UOR ja UOT portidesse. Lõpuks sai maandused kokku ühendatud Oomipoest soetatud uhiuute juhtmetega (alguses sain suurest juhtmekotist peoga võtta ning panin kohati ise sobivad toitejuhtmed emastest ja isastest kokku), ning maanduskaabel ei tulnud enam iga paari katsetuse tagant oma pesast välja.



Joonis 4.3. Arduino Uno R3 kasutamine koodi üleslaadimiseks.

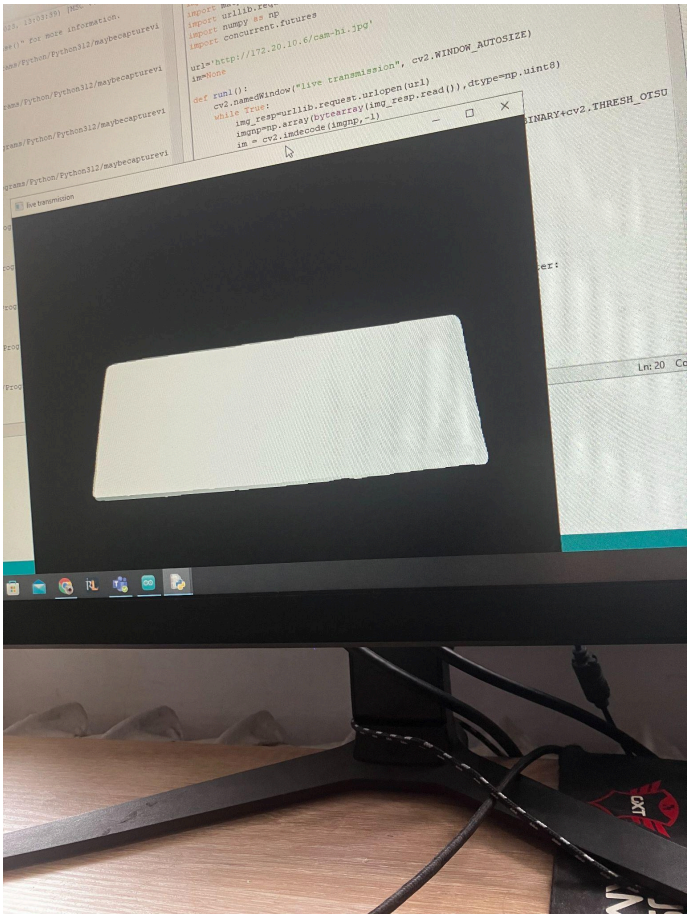
Teise sammu käigus kaaluti erinevaid video töötlemise meetodeid. Esialgu oli soov kasutada töötleva arvutina ESP32-CAM ennast, kuid pika katsetamise käigus ei

õnnestunud leida viisi, kuidas pilti ESP32-CAMiga binaarseks saada. Kaaluti ka võimalusi tuvastada pildilt servasid, kasutades "Canny edge detector" servatuvastusalgoritmi. Esialgsed testversioonid sellise koodi implementeerimiseks said küll loodud, kuid tööle neist ühtegi ei saanud ja esinesid errorid, mida lahendada ei osanud. Peale mõningast üritamist otsustas töö autor, et soovib just nimelt binaarse pildi pealt valgeid piksleid otsida, kuna servade edukas tuvastamine sõltub rohkem valgusoludest. Autor nägi ette, et roboti ESP32-CAM võiks olla suunatud otse roboti alla, liikumissuunaga samas sihis ning sellistes tingimustes ei jõua kaamera sensoriteni piisavalt valgust, et servatuvastust edukalt implementeerida. Kui võtta näiteks D. Belolipetskaja ja D. Valme Raspberry Pi roboti, siis nemad suutsid servatuvastuse edukalt kasutusele võtta, suunates kaameramooduli näitama roboti ette. See võimaldab paremaid valgustingimusi, kuid selle käigus peab töötlev arvuti töötlema oluliselt detailsemat pilti, kui käesoleva lõputöö lahenduse puhul. Antud roboti töötlev monoplaatarvuti on võimsusega 1.5 GHz ja kasutada on gigabaitidesse ulatuvat välkmälu. Võrdluseks on ESP32-CAM mikrokontroller võimsusega 240 MHz ja mälumaht jääb kilobaitide valda. Keeruka pildi töötlemine võib osutuda soodsale kuid väiksema võimekusega mikrokontrollerile üle jõu käivaks ülesandeks. Peale arvukaid katseid servatuvastusega õnnestus töötlevaks arvutiks kasutusele võtta lauaarvuti ning sündis lõpuks realiseeritud kolmesammuline lähenemine.

Teise sammu olemasolevat koodi (lisas Pythoni kood) on võimalik optimeerida, leides katseliselt sobivad väärtused erinevatele pilditöötlust puudutavatele muutujatele (näiteks "Gaussian blur"-i tuuma suurus). Samuti on kindlasti võimalik optimeerida näiteks valgete pikslite otsimise funktsiooni algkoordinaate just ESP32-CAM kaamerapildi jaoks parematesse kohtadesse. Ilmselt on võimalik panna valgete pikslite otsimise funktsiooni toimima kiiremini ning katseliselt tuleb kindlaks teha, täpselt milliste tingimuste täitmisel on parim juhtimisinformatsiooni edastada. Autori hinnangul ei ole algelise, kuid põhitasemel toimiva "switch" tüüpi koodistruktuuri loomine suur väljakutse, kuid optimaalsete väärtuste leidmine koodis on märkimisväärne väljakutse.

Töö praktilise osa teostamise üks suurimaid raskuspunkte oli leida viis, kuidas saaks videopilti binaarseks töödelda ja videopildilt valgeid piksleid otsida. Näidiskoodiga ei õnnestunud arvutile videopilti kättesaadavaks teha (muuhulgas oli näha ainult halli pilti või avanes voogedastuse aken ja sulgus koheselt) ja ESP32-CAM ennast töötleva üksusena kasutada ei õnnestunud. CameraWebServer.h koodis kasutatava esp_camera.h teegi funktsionaalsuste piires pilti töödeldes tekkisid arusaamatud errorid programmi käivitamisel, mida ei õnnestunudki lahendada. Praktilise osa teine

samm õnnestus suured edusammud teha, kui kasutusele võeti ESP32-CAM koodis WebServer.h ja esp32cam.h teegid ning Python koodis urllib.request ja urllib.parse. Kui videopilt sai PC-le kättesaadavaks, siis ei valmistanud raskusi videopildi OpenCV-ga binaarseks töötlemine. Samuti osutus vähese kogemuse programmeerimiskogemuse tõttu keeruliseks valgete pikslite otsimine, kuid koostöös Diana Belolipetskajaga õnnestus kood toimima saada. Joonisel 4.4 on näidis roboti töötluseks kasutatavast binaarseks töödeldud pildist. Pildil on kujutatud sinist teibiriba musta värvi hiirematil.



Joonis 4.4. Binaarseks töödeldud pilt sinisest teibiribast hiirematil.

Töö praktilise osa esmase kokkupanemise käigus juhtus äpardus, mis kahjustas roboti tööprotsessis keskset rolli omavat ESP32-CAMi. Enne roboti esmast kokkupanekut avastasin, et kasutatav L298N mootorite draiver on defektne, nimelt on sellel puudu kaks kruvi. Kruvisid oli võimatu oma plastikutest hoidikutest välja keerata, seega otsustas töö autor võtta kasutusele ülikoolist küsitud sama marki asendusdraiveri. Kahjuks ei märganud autor, et asendusdraiveril oli kahjustada saanud üks takisti. Kui robot oli 4. mail esimest korda täielikult kokku pandud, ei õnnestunud ESP32-CAM programmi käivitada, ning ESP32-CAM ei võtnud toidet taha ka alternatiivsest

toiteallikast. Tähelepanelikul komponentide inspektsioonil leiti draiverilt vigane takisti ja tunda oli ka kerget kärsahaisu. Lühis kahjustas roboti toimimise keskset komponenti ja töö edasine teostus jäi ootama uue ESP32-CAM tarnet.

Roboti esmase katsetamise käigus leidis aset olukord, kus programmikoodi käivitamisel hakkasid DC mootorid tugevalt undama, kuid rattad ringi ei käinud. Veaotsingu käigus selgus, et programmikood töötas korrektselt, kuid mootorite lahtivõtmisel tuli välja, et mootoritest on puudu vajalikud hammasrattad ning ülekannet ei saanudki toimuda. Uute, komplektsete mootoritega õnnestus robotit edukalt juhtida.

Lõputöö esitamise tähtajaks on robot kokku pandud ja edukalt on implementeeritud roboti tööks vajalikud alamsüsteemid. ESP32-CAM tarkvara suudab edukalt edastada videopilti, mida Pythoni programm personaalarvutis kätte ja saab ja töötleb, edastades töötluse tulemusel tuvastatud pikslite suuna tagasi ESP32-CAM tarkvarale, mis kasutab seda sisendit oma mootorite juhtimiseks. Roboti juhtimiseks vajalikud ESP32-CAM ja Pythoni koodid on leitavad lisadest 1 ja 2. ESP32-CAM programmi pidevalt tsüklilises töös oleva põhifunktsiooni tööjärjestus on esmalt kontrollida, kas klient (PC) on serverile midagi saatnud. Juhul kui on, siis edastatakse sisend "switch" struktuuri, kus vastavalt *int*-muutuja väärtusele sõidab robot madalal liikumiskiirusel samas suunas, pöörab veidi vasakule, pöörab veidi paremale või peatub täielikult programmi töö. Juhul kui kaamera pikslite otsimise keskpunkt paikneb koheselt valgetel pikslitel, liigub robot samas suunas edasi. Kui valgeid piksleid leitakse vasakult/paremalt poolt, keerab robot end ligikaudu 10 kraadi vasakule või paremale, mis saavutatakse läbi pöördesuunas paikneva mootori kiiruse vähendamise. Kui kaamera valgeid piksleid ei tuvasta, lõpetab Pythoni programm töö, kuna vastasel juhul hakkaks Pythoni koodi kasutatav "Otsu thresholding" mõne hetke pärast leidma heledamaid piksleid ka tumedalt taustalt. "Otsu thresholding" tehnika tugevuseks on keskkonnas parimate võimalike lävendväärtuste tuvastamine, kuid ühtlasi on see keskkonna drastilise muutumise korral ka selle suurimaks nõrkuseks.

Iga paari sekundi tagant kontrollitav juhtimissisendi väärtus, roboti väga madal liikumiskiirus ja iga keeramise suhteliselt väike nurgamuutus tagavad võimalikult täpse teibiriba jälgimise. Põhjaliku testimise järel on võimalik veelgi optimeerida täpset juhtimissisendi kontrollimise tempot. Liikumiskiirusele on võimalik leida selline kiirus, mis on sellise kiirusvahemiku maksimaalne väärtus, mille piires on võimalik efektiivselt navigeeruda teibist tehtud rajal, mille teibi pöördenurgad ei ületa kunagi 90 kraadi. Üle 90 kraadise pöördenurgaga teibirajal robot sõita ei suuda, kuid

tehniliselt oleks võimalik lisada robotile valgete pikslite otsimise suundi juurde. Seejuures tuleb arvestada ka programmi töö kiirusega ning tagada roboti sujuv juhtimine. Antud roboti töö juhtimine toimub läbi juhtmevaba võrgu (WLAN), mille puhul on ühendusprobleemide esinemise tõenäosus suurem kui juhtmega ühenduste puhul ning seetõttu on tähtis hoida edastatavad andmemahud võimalikult madalad.
Lõputöö juhendajale esitamise hetkeks on juhtimisega probleeme, mis tulenevad kaamera ebatäpsest asetusest roboti liikumissuuna suhtes, ning kaamera ei ole maapinnaga täpselt paralleelne, vaid on suunatud natuke roboti alla. Autoril on idee seoses kaamera paigutusega roboti ninal, mis vajab edasist katsetamist. Lisaks kaamera asetusest tulenevale probleemile on olnud keerukas leida juhtimisväljundi edastamiseks sobivat intervalli ning sellega koos toimivat juhtimiskäskude tõlgendamise süsteemi. Täpsemalt on juhtimiskäskude tõlgendamise süsteemis probleemseks kohaks programmikoodis olevad viiteväärtused (“delay”), mis oleksid sobilikud, et robot suudaks ennast võimalikult täpselt juhtida. Sellest tulenevalt ei ole saavutatud tulemust, kus robot suudaks järjepidevalt alati tumedal pinnal olevat teibiriba jälgida. Autori hinnangul on võimalik edasise optimeerimise ja arendustööga sellise tulemuseni siiski jõuda.

Antud tööd on võimalik tulevikus mitmes suunas edasi arendada. Tuleviku uurimisteemaks saaks olla näiteks erinevalt konfigureeritud, sama platvormi (ESP32-CAM) kasutavate isesõitvate robotite võrdlus, eesmärgiga leida seatud võimekuskriteeriumite alusel parim variant ja üritada leida lahendusele ka praktilisi kasutusi väljaspool haridusvaldkonda. Samuti saaks uurimisteemaks olla valitud parameetrite maksimaalne võimalik optimeerimine kindlaksmääratud riistvaraliste piirangute juures. ESP32 seeriale keskenduv mikrokontrollerite kasutusalade uurimine võib tekitada uusi ideid tänapäevaste, minevikuga võrreldes kasvanud võimekusega mikrokontrollerite kasutamisel erinevates valdkondades. Mikrokontrollerite üks suurimaid eeliseid on nende relatiivselt madal hind ning minevikuga võrreldes oluliselt kasvanud töötlusvõimsused võivad potentsiaalselt võimaldada kasutuselevõttu uues võtmes, rollides, mida on varasemalt täitnud kõrgema hinnaklassiga seadmed.

Roboti kokkupaneku ja testimise järgseks tegevuseks on antud lõputööle lisandiks mõeldud kokkupanekujuhiste kirjutamine. Esialgu Windowsi-põhiste kokkupanekujuhiste puhul on fookuses lihtne ja arusaadav kõnepruuk ning selgelt arusaadavad ja hea kvaliteediga pildid. Juhismaterjalid on üles laetud GitHubi keskkonda, kus on kõigil võimalik neile ligi pääseda. Juhismaterjalide kasutamist toetab YouTube video, kus roboti kasutamist ja kokkupanekut demonstreerib töö autor selliselt, et sellest saavad aru ka varasema robootikakogemuseta õpilased ning

kokkupanekuprotsess on seejuures kergelt jälgitav. Juhismaterjalid on saadaval nii eesti kui ka inglise keeles, et tagada kättesaadavus võimalikult suurele audientsile. Tulevikus on võimalik avalikustada koostamisjuhiseid ka teist sorti robotitele, nagu näiteks D. Belolipetskaja ja D. Valme poolt kokku pandud robotile, et pakkuda õpilastele mitmekülgsemaid võimalusi enesearenguks.

KOKKUVÕTE

Käesoleva lõputöö põhiväljundiks oleva praktilise osaga seotud eesmärkide täitmine õnnestus osaliselt. Põhilisel määral suudeti saavutada esmane eesmärk luua alternatiivne robotplatvorm D. Belolipetskaja ja D. Valme poolt ehitatud Raspberry Pi platvormil põhinevale robotile. Isesõitev robot õnnestus ehitada oluliselt odavamalt ja seejuures lihtsama struktuuriga. Põhieesmärgi saavutamine näitab autori hinnangul potentsiaali edasiseks arendustööks ehitatud robotplatvormiga ja kasutamiseks hariduslikul eesmärgil tulevikus. Käesoleva lõputöö käigus ehitatud platvormi võimekus ei ole hetkel siiski võrreldav edasiarenduse baasiks olnud roboti võimekusega. Juhtimisega esineb palju probleeme ja olukordi, mille käigus robot käitub ettearvamatult. Probleemide ulatus on selline, et roboti sõitmine mööda tumedal taustal olevat teibiriba õnnestub ei õnnestu igal katsetusel. Paremaks on vaja teha nii roboti juhtimise koodi kui ka video töötlemise koodi. Samuti oleks kasulik välja töötada elegantsem ja töökindlam lahendus ESP32-CAM kinnitamiseks roboti esiosa külge kui praegune implementatsioon kahepoolse teibi ribaga. Optimeerimiseks on ruumi palju ning autori hinnangul on võimalik uue roboti võimekus viia lähedale vana roboti võimekusele.

Kokkupandud roboti hariduslikul eesmärgil kasutamiseks on vaja teha veel palju tööd. Robot on vaja panna piisavalt hästi funktsioneerima, et iga eduka sõidu kõrval ei oleks mitut ebaõnnestunud sooritust. Roboti koostu on vaja ehitada turvalisemaks, kuna hetkel on võimalik roboti ninas paiknev ESP32-CAM kerge vaevaga mõne ettejääva takistuse otsa puruks sõita. Selleks oleks võimalik paigaldada roboti ninale 3D-prinditud “bumper” või integreerida selline “bumper” mingitsorti kavalasse kinnitusmehhanismi, millega on võimalik ESP32-CAM kindlamalt ja ilusamini roboti nina külge kinnitada, nii et seejuures on tagatud roboti kaameramooduli maaga täpselt paralleelne asetus. Samuti tuleb parandada viisi, millega roboti komponendid on kereks oleva plaadi külge kinnitatud. Hetkel on roboti koost kokku pandud sisuliselt kätte juhtunud poolsuvalise pikkusega poltide ja mutrite abil. Roboti kokku panemiseks võiks teada olla poltide arv, pikkus, diameeter ja kinnitusmutrite arv ja tüüp.

Lõputöö tegemise käigus sai erinevaid töö teostamiseks vajaminevaid oskuseid, mida varasemate aastate käigus ülikoolis omandatud on, aegamööda värskendatud või hoopiski nullist omandatud. Iga töö etapi realiseerimisega kaasnes arvestatav väljakutse, kuid järgemööda said põhilised raskuspunktid ületatud. Lõputööga tegelemist oleks võinud alustada varem, kuna äpardused seadsid lõpuks ohtu lõputöö

õigeaegse valmimise. Roboti ehitamine oli põnev ja meeldiv protsess, mille käigus tekkinud probleemide ületamine tekitas suurt rahulolu. Roboti ehitamise protsessi huvipakkuv laad tekitas kindlustunde, et ülikoolis sai valitud õige eriala. Kuna varasem praktiline kogemus võrreldavas mastaabis antud erialal puudus, siis õnnestus töö autoril läbi projekti osaliselt eduka valmissaamise ületada enda suhteliselt madalaid ootusi.

KASUTATUD KIRJANDUSE LOETELU

[1] "Raspberry Pi Model 4B," Raspberry Pi Datasheets, https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf?%5C_gl=1%5C*1282u2o%5C*%5C_ga%5C*MTMxMDk0NTY1NS4xNzAxNTIwMTYy%5C*%5C_ga%5C_22FD70LWDS%5C*MTcwOTc0NDIzMy4zLjEuMTcwOTc0NDI0NS4wLjAuMA (accessed Mar. 18, 2024).

[2] F. -D. Secuianu and C. Lupu, "Implementation of a home appliance mobile platform based on computer vision: self-charging and mapping," *2018 22nd International Conference on System Theory, Control and Computing (ICSTCC)*, Sinaia, Romania, 2018, pp. 464-468, doi: 10.1109/ICSTCC.2018.8540685, https://ieeexplore.ieee.org/document/8540685 (accessed Mar. 18, 2024).

[3] L. K. Raj, C. S. Bijapur and G. H. S, "Digital Assistant Robot for Academic Fraternity Using Deep Learning," *2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, Bangalore, India, 2023, pp. 1-8, doi: 10.1109/CSITSS60515.2023.10334220, https://ieeexplore.ieee.org/document/10334220 (accessed Mar. 18, 2024).

[4] "Raspberry Pi 4 mudel B moodul 1.5ghz 8GB," Oomipood, https://www.oomipood.ee/product/3369503_raspberry_pi_4_mudel_b_moodul_1_5ghz_8gb (accessed Mar. 18, 2024).

[5] D. Khatri, "Beaglebone AI-64 System reference manual," Farnell Datasheets, https://www.farnell.com/datasheets/3745931.pdf (accessed Mar. 18, 2024).

[6] "102110646 - SBC, Beaglebone AI-64, TDA4VM, ARM cortex-A72, 64 bit, 8MB RAM, 4GB RAM, 16GB eMMC, WIFI," Farnell, https://uk.farnell.com/beagleboard/beaglebone-ai-64/beagleboard-arm-cortex-a72/dp/3923739 (accessed Mar. 18, 2024).

[7] "Beaglebone® AI-64," AliExpress, https://de.aliexpress.com/i/1005005921425255.html?gatewayAdapt=glo2deu (accessed Mar. 18, 2024).

[8] "Amazon.com: Raspberry Pi 4 model B 2019 Quad Core 64 bit WIFI ...," Amazon, https://www.amazon.com/Raspberry-Model-2019-Quad-Bluetooth/dp/B07TC2BK1X (accessed Mar. 18, 2024).

[9] A. Albanese, M. Nardello, and D. Brunelli, "Low-power deep learning edge computing platform for resource constrained lightweight compact uavs," Sustainable Computing: Informatics and Systems, https://www.sciencedirect.com/science/article/abs/pii/S2210537922000609 (accessed Mar. 18, 2024).

[10] "Get started with Jetson Nano Developer Kit," NVIDIA Developer, https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit (accessed Mar. 18, 2024).

[11] H. K. Kondaveeti, N. K. Kumaravelu, S. D. Vanambathina, S. E. Mathe, and S. Vappangi, "A systematic literature review on prototyping with Arduino: Applications, challenges, advantages, and limitations," Computer Science Review, https://www.sciencedirect.com/science/article/abs/pii/S1574013721000046 (accessed Mar. 18, 2024).

[12] "Arduino® Uno R3," Arduino Datasheets, https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf (accessed Mar. 18, 2024).

[13] "ESP32 series datasheet," ESP32 Datasheets, https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf (accessed Mar. 18, 2024).

[14] "Arendusplaat UNO R4 WiFi Arduino," Lemona, https://www.lemona.ee/arendusplaat-uno-r4-wifi-arduino.html (accessed Mar. 18, 2024).

[15] T.-H. Lin, A. Putranto, P.-H. Chen, Y.-Z. Teng, and L. Chen, "High-mobility inchworm climbing robot for Steel Bridge Inspection," Automation in Construction, https://www.sciencedirect.com/science/article/pii/S0926580523001656 (accessed Mar. 18, 2024).

[16] "Raspberry pi 4B, 4x 1,5 GHz, 4 GB RAM, WIFI & BT,...," Arvutitark, https://arvutitark.ee/arvutid-ja-lisad/lauaarvutid/raspberry-pi/raspberry-pi-4b-4x-15-ghz-4-gb-ram-wifi-bt-soc-mini-m-1272985 (accessed Mar. 18, 2024).

[17] “Datasheet - STM32F722xx stm32f723xx,” STM32 Datasheets, https://www.st.com/resource/en/datasheet/stm32f722ic.pdf (accessed Mar. 18, 2024).

[18] “STM32 32-bit ARM cortex mcus,” STMicroelectronics, https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html (accessed Mar. 18, 2024).

[19] “Nucleo-F411RE: St M32 Nucleo Board STM32F411RET6: Elfa Distrelec Eesti,” Elfa Distrelec Eesti, https://www.elfadistrelec.ee/et/stm32-nucleo-board-stm32f411ret6-st-nucleo-f411re/p/30009240 (accessed Mar. 18, 2024).

LISAD

Lisa 1. Pythoni kood (08.05.24).

```
import cv2
import urllib.request
import urllib.parse
import numpy as np
import concurrent.futures
import time

url = 'http://172.20.10.8/cam-hi.jpg'


def white_pixels_detection(image, edges):
    height, width, _ = image.shape
    search_point_x = width // 2
    search_point_y = int(height * 0.78)
    border_size = 5
    borders = cv2.copyMakeBorder(edges, border_size, border_size, border_size,
border_size,
                                 cv2.BORDER_CONSTANT, value=128)

    directions = [(0, -1), (-1, 0), (-1, -1), (1, 0), (1, -1)]
    direction_found = None

    for direction in directions:
        x = search_point_x
        y = search_point_y
        while 0 <= x < width and 0 <= y < height:
            if borders[search_point_y, search_point_x] == 255:
                direction_found = 0 #already white
                break
            if borders[y, x] == 255:
                if direction == (-1, -1) or direction == (-1, 0):
                    direction_found = 1 #left
                    break
                elif direction == (1, -1) or direction == (1, 0):
```

```
                direction_found = 2 #right
                break
            else:
                direction_found = 3 #no white detected
                break

        x += direction[0]
        y += direction[1]

    return direction_found



def process_image():
    img_resp = urllib.request.urlopen(url)
    imgnp = np.array(bytearray(img_resp.read()), dtype=np.uint8)
    im = cv2.imdecode(imgnp, -1)
    grayim = cv2.cvtColor(im, cv2.COLOR_BGR2GRAY)
    blurredim = cv2.GaussianBlur(grayim, (7, 7), 0)
    ret, th = cv2.threshold(blurredim, 0, 255, cv2.THRESH_BINARY +
cv2.THRESH_OTSU)
    direction_found = white_pixels_detection(im, th)
    return direction_found


if __name__ == '__main__':

    print("started")
    with concurrent.futures.ProcessPoolExecutor() as executor:
        while True:
            future = executor.submit(process_image)
            direction_found = future.result()
            print("Case:", direction_found)
            data_to_send = urllib.parse.urlencode({'data':
direction_found}).encode('utf-8')
            url = 'http://172.20.10.8/'
            response = urllib.request.urlopen(url, data=data_to_send)
            response_data = response.read().decode('utf-8')
            print(response_data)
```

time.sleep(0.5)

Lisa 2. ESP32-CAM kood (08.05.24).

```
#include <WebServer.h>
#include <WiFi.h>
#include <esp32cam.h>

// Define motor control pins
const int ENA = 4; // Enable Motor A, connected to GPIO 4
const int IN1 = 2; // Motor A IN1, connected to GPIO 2
const int IN2 = 14; // Motor A IN2, connected to GPIO 14
const int IN3 = 15; // Enable Motor B, connected to GPIO 15
const int IN4 = 13; // Motor B IN3, connected to GPIO 13
const int ENB = 12; // Motor B IN4, connected to GPIO 12

// Function declarations
void continueDriving();
void driveDiagonallyLeft();
void turnLeft();

void driveDiagonallyRight();
void turnRight();
void stopMotors();
void handlePostRequest();

const char* WIFI_SSID = "Hottttspotttt";
const char* WIFI_PASS = "jimmucat";

WebServer server(80);

static auto loRes = esp32cam::Resolution::find(320, 240);
static auto midRes = esp32cam::Resolution::find(350, 530);
static auto hiRes = esp32cam::Resolution::find(800, 600);

void serveJpg() {
```

```
  auto frame = esp32cam::capture();
  if (frame == nullptr) {
    Serial.println("CAPTURE FAIL");
    server.send(503, "", "");
    return;
  }
  Serial.printf("CAPTURE OK %dx%d %db\n", frame->getWidth(),
frame->getHeight(),
                static_cast<int>(frame->size()));

  server.setContentLength(frame->size());
  server.send(200, "image/jpeg");
  WiFiClient client = server.client();
  frame->writeTo(client);
}

void handleJpgLo() {
  if (!esp32cam::Camera.changeResolution(loRes)) {
    Serial.println("SET-LO-RES FAIL");
  }
  serveJpg();
}

void handleJpgHi() {
  if (!esp32cam::Camera.changeResolution(hiRes)) {
    Serial.println("SET-HI-RES FAIL");
  }

  serveJpg();
}

void handleJpgMid() {
  if (!esp32cam::Camera.changeResolution(midRes)) {
    Serial.println("SET-MID-RES FAIL");
  }
  serveJpg();
}
```

```
void setup() {

  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENB, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);

  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

  Serial.begin(115200);
  Serial.println();
  {
    using namespace esp32cam;
    Config cfg;
    cfg.setPins(pins::AiThinker);
    cfg.setResolution(hiRes);
    cfg.setBufferCount(2);
    cfg.setJpeg(80);

    bool ok = Camera.begin(cfg);
    Serial.println(ok ? "CAMERA OK" : "CAMERA FAIL");
  }
  WiFi.persistent(false);
  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_SSID, WIFI_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
  }
  Serial.print("http://");
  Serial.println(WiFi.localIP());
  Serial.println("  /cam-lo.jpg");
```

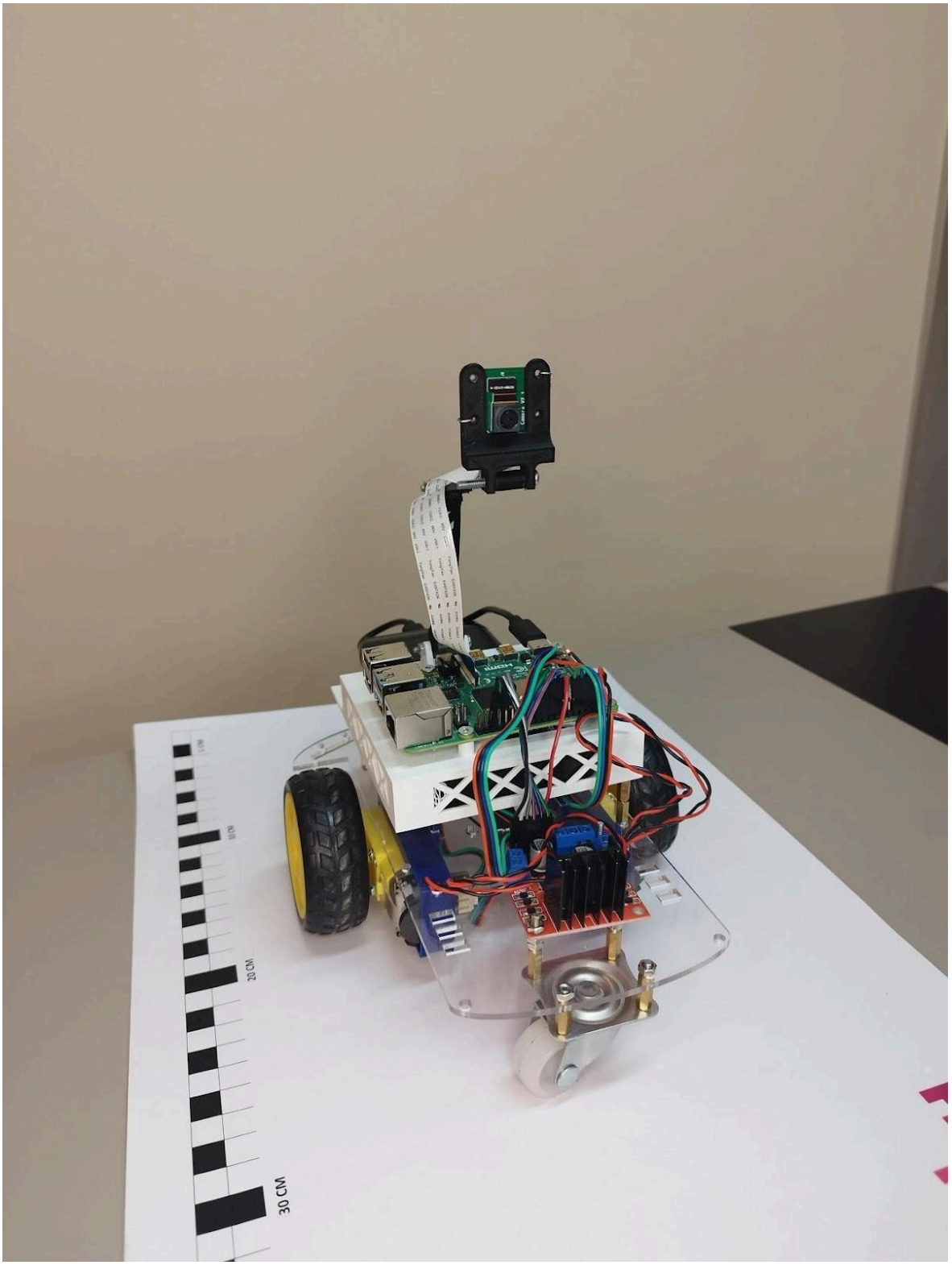
```
  Serial.println("  /cam-hi.jpg");
  Serial.println("  /cam-mid.jpg");

  server.on("/cam-lo.jpg", handleJpgLo);
  server.on("/cam-hi.jpg", handleJpgHi);
  server.on("/cam-mid.jpg", handleJpgMid);
  server.on("/", HTTP_POST, handlePostRequest); // Handle POST request

  server.begin();
}

void loop() {
  server.handleClient();
}

void handlePostRequest() {
  String data_received = server.arg("data"); // Retrieve data from POST
request

  if (data_received != "") {
    Serial.println("Data received: " + data_received);

    // Process the received data and control motors accordingly
    int command = data_received.toInt(); // Convert received data to integer
command

    // Control motors based on received command
    switch (command) {
      case 0:
        // Continue driving in the same direction
        continueDriving();
        break;
      case 1:
        // Drive diagonally left
        driveDiagonallyLeft();
        break;
      case 2:
```

```
        // Drive diagonally righ



        driveDiagonallyRight();
        break;
      default:
        // Invalid command
        Serial.println("Invalid command received");
    }

    // Send response back to the client
    server.send(200, "text/plain", "Command executed");
  } else {
    server.send(400, "text/plain", "No data received"); // Bad request
  }
}

void continueDriving() {
  // Set both motors to move forward
  analogWrite(ENA, 120);
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  analogWrite(ENB, 135);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  delay(200);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
  delay(1000);

}

void driveDiagonallyLeft() {
  // Set motor A to move forward
  analogWrite(ENA, 200);
```

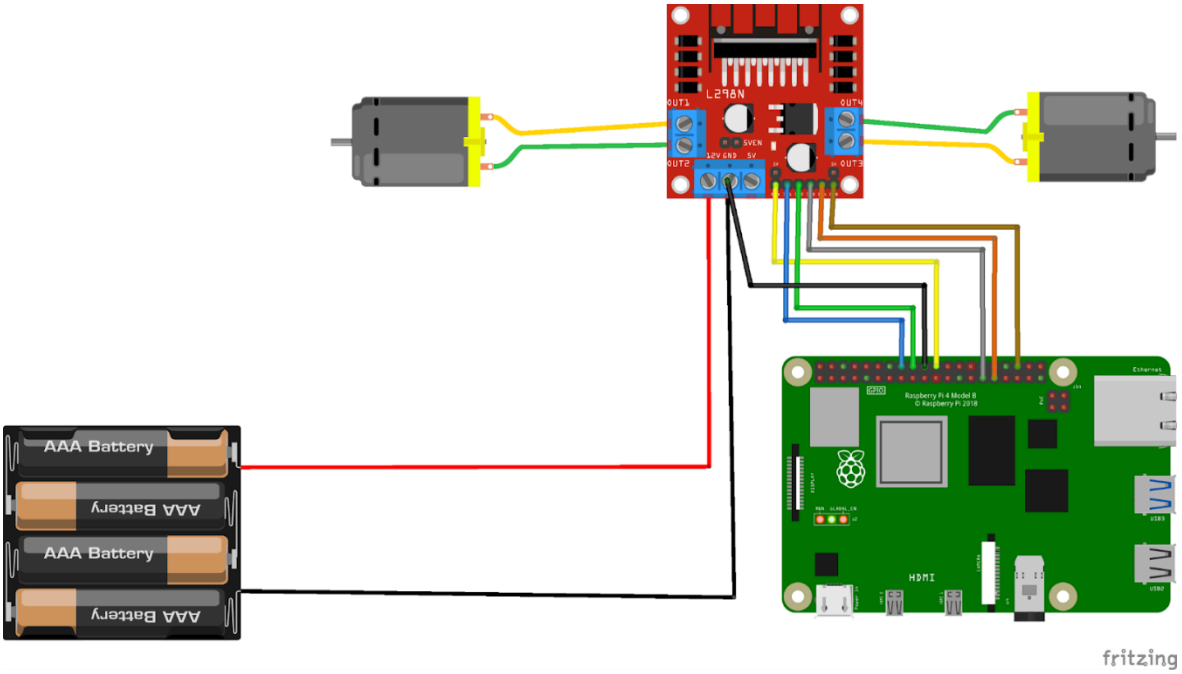
```
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  // Set motor B to move forward slightly faster to turn diagonally left
  digitalWrite(ENB, LOW); // Adjust speed as needed
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
  delay(200);


  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

  // Delay to allow the robot to turn
  delay(1000); // Adjust the delay as needed

  // Stop both motors
  stopMotors();
}

void driveDiagonallyRight() {
  // Set motor A to move forward slightly faster to turn diagonally right
  digitalWrite(ENA, LOW); // Adjust speed as needed
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  // Set motor B to move forward
  analogWrite(ENB, 215);
  digitalWrite(IN3, HIGH);
  digitalWrite(IN4, LOW);
  delay(200);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);

  // Delay to allow the robot to turn
```

```
  delay(1000); // Adjust the delay as needed

  // Stop both motors
  stopMotors();
}

void stopMotors() {
  // Stop both motors
  digitalWrite(ENA, LOW);
  digitalWrite(ENB, LOW);
}
```

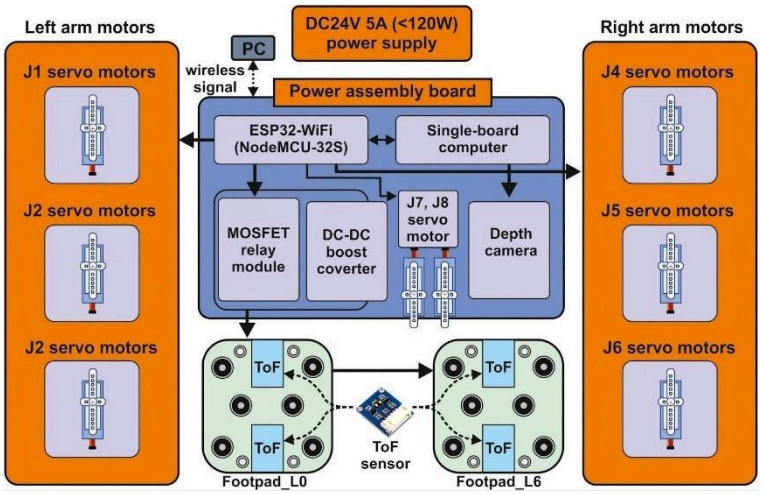
Lisa 3. Pilt olemasolevast Raspberry Pi-l põhinevast robotlahendusest.



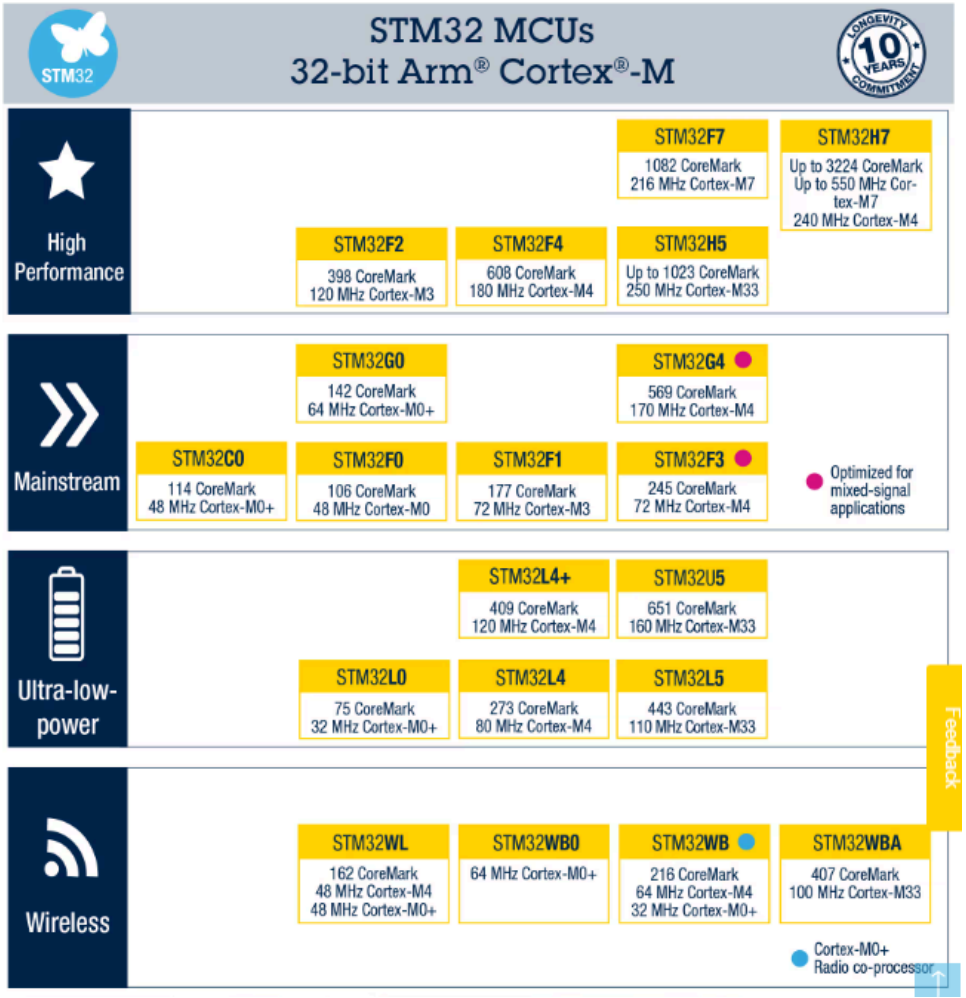
Lisa 4. Raspberry Pi robotplatformi ühendusskeem Fritzingu tarkvaras.



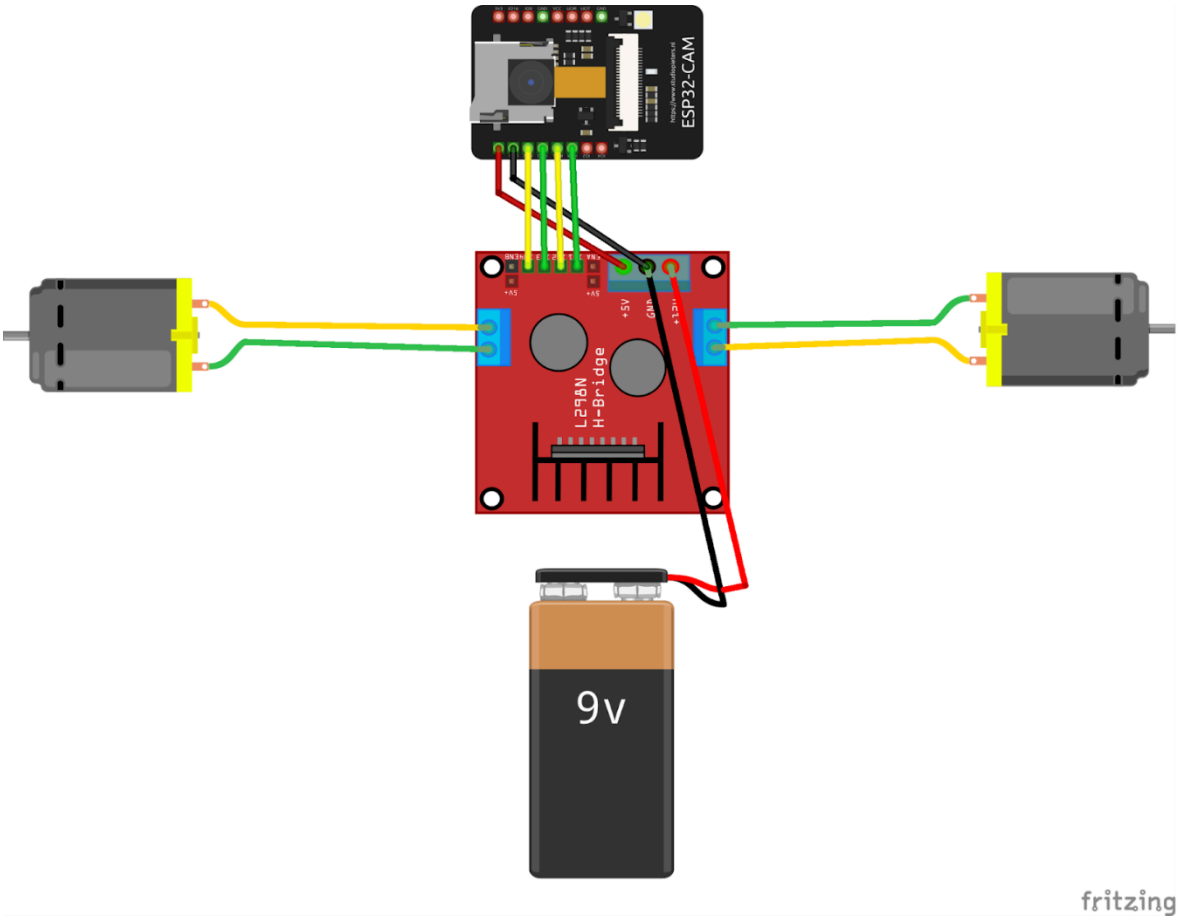
Lisa 5. HMICRoboti mehhatroonikasüsteem [15]



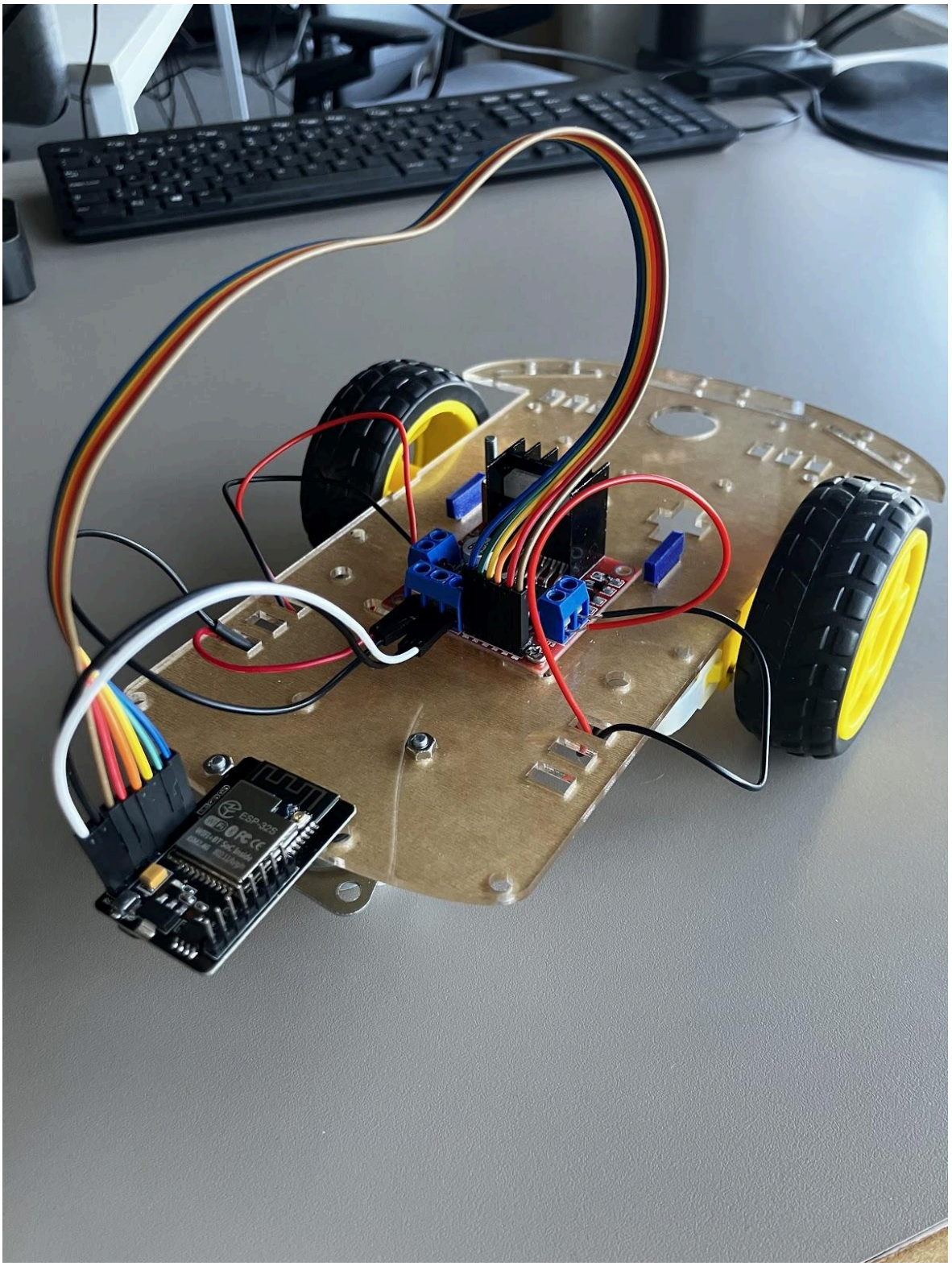
Lisa 6. STM32 mikrokontrollerite tootevalik. [18]



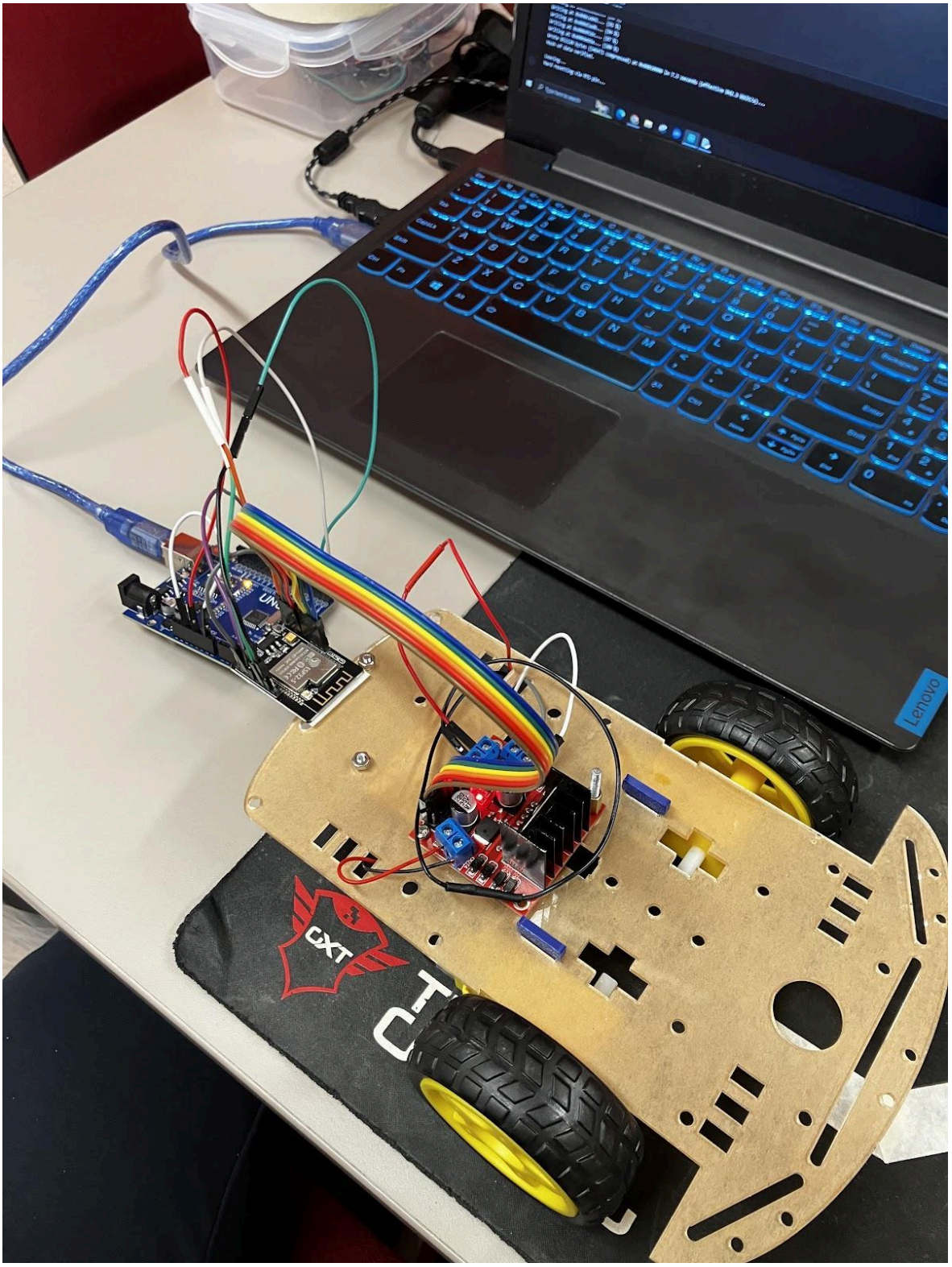
Lisa 7. Roboti ühendusskeem Fritzingu tarkvaras.



Lisa 8. Kokkupandud robotplatvorm.



Lisa 9. Arduino Uno R3 kasutamine koodi üleslaadimiseks.



Lisa 10. Binaarseks töödeldud pilt sinisest teibiribast hiirematil.