

TARTU ÜLIKOOL  
Arvutiteaduse instituut  
Informaatika õppekava

**Argo Mändla**

# **BrickPi võrdlus teiste robotikaplatvormidega**

**Bakalaureusetöö (9 EAP)**

Juhendaja(d): Anne Villems

Taavi Duvin

Alo Peets

Tartu 2017

## **BrickPi võrdlus teiste robotikaplatvormidega**

### **Lühikokkuvõte:**

Töö tutvustab BrickPi robotika platvormi, mis sai alguse läbi ühisrahastus-projekti. BrickPi võimaldab kasutada LEGO Mindstorms EV3 ja LEGO Mindstorms NXT andureid ja mootoreid väljaspool nende tootjapoolset komplekti. Käesoleva töö raames kirjeldatakse BrickPi positiivseid ja negatiivseid omadusi ning võrreldakse neid LEGO Mindstorms robotikaplatvormide omadustega. Kirjeldatakse kahte erinevat operatsioonisüsteemi, mis toetavad BrickPi trükkplaati. Lisaks on juhend nende installeerimiseks. Juhendites kirjeldatakse, kuidas seadistada BrickPi nii, et oleks võimalik alustada programmeerimisega. Töö lõpus esitletakse robotit, mis kasutab LEGO Mindstorms EV3 mootoreid ja LEGO komplekti klotse ning lisaks kasutatakse mittestandardset LEGO Mindstorms platvormide ajudega ühilduvat andurit.

### **Võtmesõnad:**

Robotics, Python, BrickPi, Raspberry Pi, LEGO Mindstorms EV3, LEGO Mindstorms NXT, Raspbian for Robots

### **CERCS:**

P170 - Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

## **BrickPi Comparison with other Robotics Platforms**

### **Abstract:**

This paper describes a robotics platform BrickPi that started through crowdfunding. BrickPi enables using LEGO Mindstorms EV3 and LEGO Mindstorms NXT sensors and motors outside their original kit. This paper also describes BrickPi's pros and cons compared to LEGO Mindstorms robotics platforms. Furthermore, two different operating systems are presented, that support BrickPi circuit board. The paper includes manual written in Estonian, which describes how to install either of the operating systems, and everything that is needed to start programming on BrickPi right-away. The paper also contains a demo robot, that uses LEGO Mindstorms EV3 motors and LEGO bricks. The robot uses a sensor, that is a non-standard LEGO Mindstorms sensor, but is compatible with the LEGO Mindstorms robotics platforms.

### **Keywords:**

Robotics, Python, BrickPi, Raspberry Pi, LEGO Mindstorms EV3, LEGO Mindstorms NXT, Raspbian for Robots

### **CERCS:**

P170 - Computer science, numerical analysis, systems, control

# Sisukord

Sissejuhatus .....	5
1 BrickPi tutvustus .....	7
1.1 BrickPi trükkplaadi ajaloost .....	7
1.2 BrickPi generatsioonide erinevused .....	8
1.3 BrickPi kirjeldus .....	10
1.4 BrickPi programmeerimine .....	11
2 Konkureerivate platvormide tutvustus ja võrdlus .....	12
2.1 LEGO Mindstorms NXT tutvustus.....	12
2.2 LEGO Mindstorms EV3 tutvustus .....	14
2.3 Põhilised erinevused platvormide lõikes .....	15
3 Arenduskeskkonnad ja sobilikud platvormid .....	18
3.1 Tööks vajalikud programmid.....	18
3.1.1 Etcher .....	19
3.1.2 PuTTY .....	19
3.1.3 WinSCP.....	19
3.1.4 PyCharm .....	20
3.1.5 WinRAR .....	20
3.1.6 Raspbian for Robots.....	20
3.1.7 ev3dev .....	21
3.2 Ev3dev .....	21
3.2.1 Ev3dev installeerimine.....	21
3.2.2 Ühendumine ev3dev virtuaalmasinasse .....	25
3.3 Raspbian for Robots .....	28
3.3.1 Raspbian for Robots installeerimine .....	28
3.3.2 Ühendumine Raspbiani virtuaalmasinasse .....	29
3.3.3 BrickPi konfigureerimine.....	32
4 Demorobot.....	34
4.1 Näidised Raspbian for Robots operatsioonisüsteemis.....	34

4.2 Näidisrobot - Aardeotsijarobot .....	37
4.2.2 Lähtekoodi kirjeldus .....	42
Kokkuvõte .....	48
Kasutatud kirjandus .....	49
Lisad .....	51
I Lähtekood .....	51
II Litsents .....	56

# Sissejuhatus

Bakalaureuse töö eesmärk on tutvustada Dexter Industries Inc poolt loodud BrickPi trükkplaati kõigile eesti keelt oskavatele robootikahuvilistele. Ka üritatakse jõuda otsusele, miks peaks või ei peaks asendama LEGO Mindstorms[20] EV3 või NXT platvormi juhtkontroller BrickPi trükkplaadi lahendusega.

Antud bakalaureusetöö esimeses peatükis kirjeldatakse, mida endast kujutab BrickPi trükkplaat. Lisaks tuuakse välja, mis on erinevat töö valmimise hetkeks välja lastud kolmel erineval BrickPi trükkplaadi generatsioonil. Vaadeldakse, mis programmeerimiskeeltes on võimalik BrickPi trükkplaati programmeerida

Teises peatükis keskendutakse tähtsaimate konkureerivate robootikaplatvormide kirjeldamisele ja võrreldakse neid omadusi BrickPi poolt pakutavate omadustega. Konkureerivateks platvormideks on loetud LEGO Mindstorms NXT ja LEGO Mindstorms EV3. Kaudselt on ka konkureerivad platvormid Matrix ja Tetrrix. Antud töös on teadlikult puudu nende kirjeldus ning võrdlus kuna Matrix'i ja Tetrrix'i platvormid on kõige paremini ära kirjeldatud 2016 aasta kevadel edukalt kaitstud Sander Orava bakalaureusetöös „MATRIX, TETRIX ja VEX robootikaplatvormid“ [1].

Bakalaureuse töö kolmas peatükk sisaldab juhiseid, kuidas alustada programmeerimist kasutades BrickPi trükkplaati. Kirjeldatakse lühidalt erinevaid vabavaralisi ja ka tasulisi programme, mis lihtsustavad programmeerimist kasutades riistvarana BrickPi'd.

Neljandas peatükis näidatakse ühte võimalikku robotit, mida on võimalik luua kasutades BrickPi trükkplaati. Näidetena on välja toodud osa programmi lähtekoodist, mis on spetsiaalselt kirjutatud antud roboti jaoks. Näidisroboti programmeerimisel on valitud keeleks Python. Seda põhjusel, et Python on üks lihtsamaid programmeerimiskeeli, mida õppida. Lisaks on selle keelega kõige lihtsam alustada kodeerima õppimist. Ka on Tartu Ülikoolis programmeerimist õpetavas baasaines MTAT.03.100 Programmeerimise[2] õpetatavaks keeleks Python. Pythonit kasutatakse ka Tartu Ülikooli kursusel „Programmeerimisest maalähedaselt,“ [3], mis on mõeldud inimestele, kes pole kunagi varem elus programmeerimisega kokku puutunud. Lisaks on tekkinud võimalus ev3dev operatsioonisüsteemi tulekuga programmeerida LEGO Mindstorms EV3 aju kasutades Pythonit. Seetõttu võib lugeja, kes on programmeerinud juba Pythonis LEGO Mindstorms EV3, tõmmata paralleele, kuidas näeks asi välja BrickPi peal.

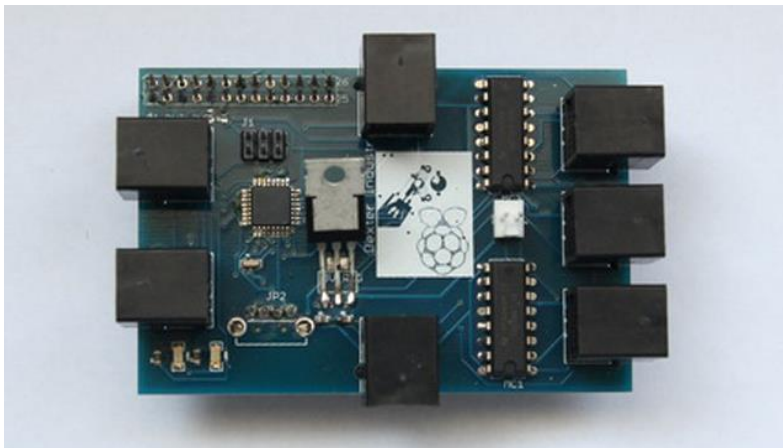
Bakalaureuse töö lisades on välja toodud kogu lähtekood, mis on autori poolt kirjutatud näidis-roboti lahenduse jaoks.

# 1 BrickPi tutvustus

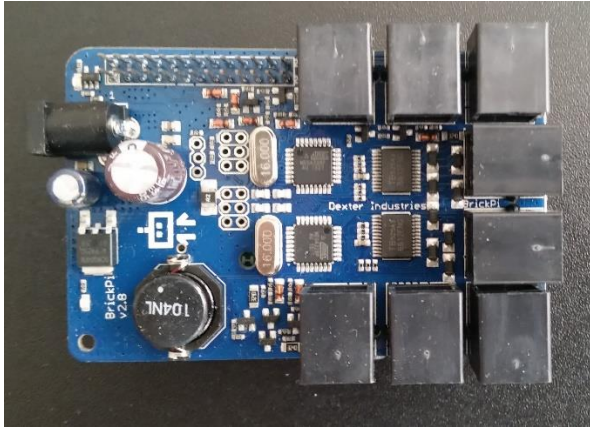
BrickPi on trükkplaat, millest avalikkus kuulis esmakordselt 2013 maikuu. Järgnevas peatükis kirjeldatakse, kust sai alguse BrickPi trükkplaat. Lugeja saab teada, mis versioonid on sellest välja lastud ja mis on nende põhilised erinevused. Peatükis selgitatakse ka välja, mis on BrickPi's positiivsed ja negatiivsed omadused.

## 1.1 BrickPi trükkplaadi ajaloost

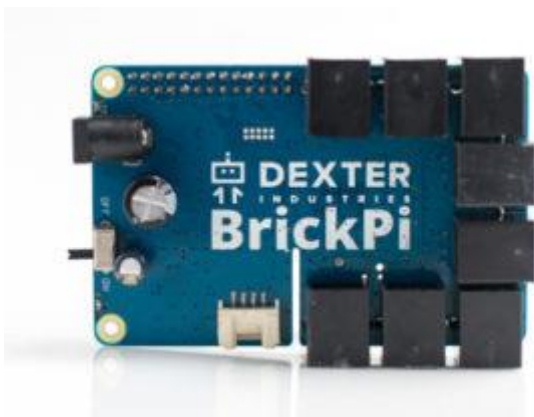
BrickPi sai alguse läbi Dexter Industries Inc - firma, mis asub Ameerika Ühendriikides Staffordi linnas Marylandi osariigis. Dexter Industries lõi ühisrahastus-portaalis Kickstarter[4] projekti 8. mail 2013, et tutvustada avalikkusele oma uut projekti ning saada ka sellele rahastust. Vähem kui 2 kuuga saadi kokku 127 538 \$ ja seda 1702 toetaja abiga. Nende eesmärk oli koguda vaid 1889 \$. Esimestele toetajatele läksid BrickPi arendusplaadid teele 7. septembril 2013. Nüüdseks on võimalik kõnealust trükkplaati tellida läbi Dexter Industries kodulehe ning hetkel maksab see 99,99 \$. Joonistel 1.1-1.3 on näha erinevaid BrickPi generatsioonide trükkplaate.



**Joonis 1.1** Esimese generatsiooni trükkplaat. [4]



**Joonis 1.2** Teise generatsiooni BrickPi trükkplaat.



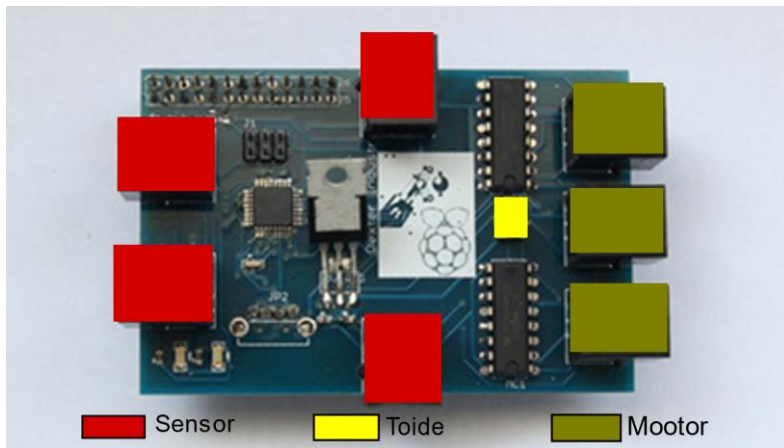
**Joonis 1.3** Kolmanda generatsiooni BrickPi trükkplaat. [16]

Kui Dexter Industries lasi välja kolmanda generatsiooni BrickPi'st, siis kirjutati ümber ka teegid, mille abil suudab riistvara paremini suhelda tarkvaraga.

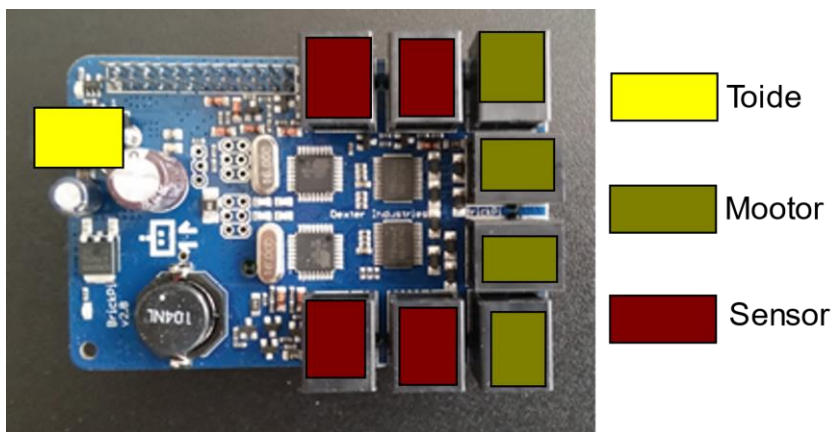
## 1.2 BrickPi generatsioonide erinevused

Esimese generatsiooni trükkplaadil on 4 porti sensorite jaoks ja 3 porti mootorite jaoks. Esimene generatsioon kannab nime BrickPi. Selle pordid on kujutatud joonisel 1.4. Teise generatsiooni trükkplaat kannab nime BrickPi+ ning selle pordid on kirjeldatud joonisel 1.5. BrickPi+ trükkplaat omab sensorite jaoks 4 porti ja mootorite jaoks 4 porti.

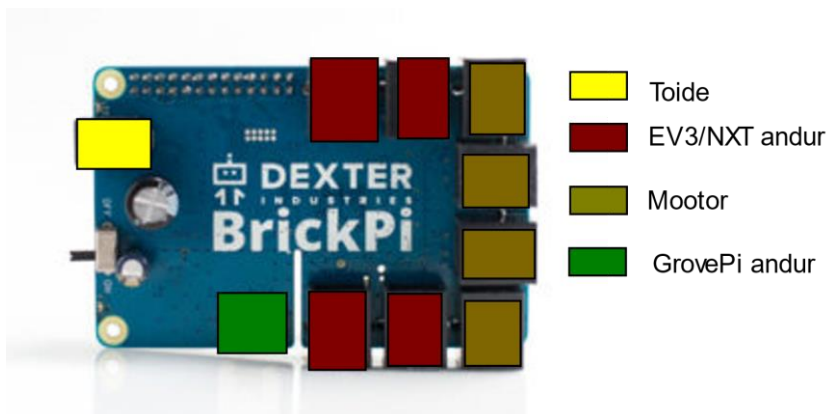




**Joonis 1.4** BrickPi portide asetus.



**Joonis 1.5** BrickPi+ portide asetus



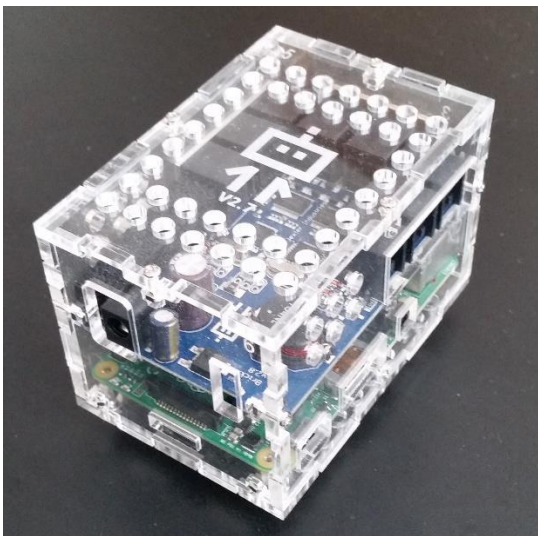
**Joonis 1.6** BrickPi3 portide asetus

Dexter Industries on jätkanud trükkplaadi arendamisega ning välja lasknud ka kolmanda generatsiooni trükkplaadist. See kannab nimetust BrickPi3. BrickPi3 trükkplaadil on 4 porti

sensorite jaoks, 4 porti mootorite jaoks ja lisaks on 1 port GrovePi I2C sensori jaoks, mida toodab ja arendab Dexter Industries. Joonisel 1.6 on näidatud BrickPi3 portide asetus.

### 1.3 BrickPi kirjeldus

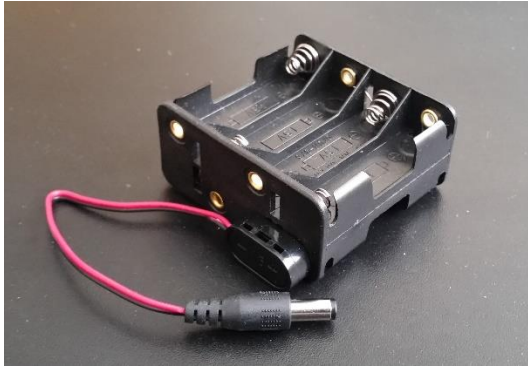
BrickPi on arendusplaat, mille abil saab muuta Raspberry Pi arvuti LEGO Mindstorms klotsidest ja anduritest/mootoritest valmis ehitatud roboti ajuks. BrickPi loob liidese, mille abil saab Raspberry Pi suhelda LEGO Minstorms EV3 sensorite ja mootoritega. BrickPi töötab ka LEGO Mindstorms NXT sensorite ja mootoritega. Olenevalt BrickPi generatsioonist on tal 3 kuni 4 porti mootorite jaoks ja 3 kuni 5 porti sensorite jaoks.



**Joonis 1.7** RaspberryPi ja BrickPi+ kabis.

Üks suurimaid plusse BrickPi puhul on see, et programmeeritavate portide arvu on lihtne lisada. BrickPi trükkplaadid on üksteise otsa laotavad ja iga järgmine trükkplaat suhtleb Raspberry Pi'ga kasutades BrickPi'd, millega see on ühendatud. BrickPi küljes on viigud mille abil saab üks BrickPi teisega suhelda.

Üheks suurimaks miinuseks on see, et BrickPi ei saa iseseisvalt asendada tervet EV3 aju, vaid selleks on vaja siiski ka Raspberry Pi arvutit. Dexter Industries toodab ka karp, mis on disainitud spetsiaalselt LEGO Mindstorms komplektiga kaasas olevate klotside eripärasid arvesse võttes. Sellesse karp on puuritud augud võttes arvesse LEGO klotside tihvtide ja ristvõllide asetust.



**Joonis 1.9** Toiteblokk BrickPi komplektis

Ning antud karbi abiga on võimalik uus aju kinnitada LEGO klotsidest ehitatud roboti raami külge. Karp ise on suuruselt võrreldav LEGO Mindstorms EV3 komplekti ajuga. BrickPi'ga kaasatulevasse karpi mahub nii BrickPi, kui ka selle külge kinnituv Raspberry Pi. Sellesse karpi aga ei mahu üle ühe BrickPi trükkplaadi. BrickPi saab voolu komplektiga kaasa tulnud raamis (joonis 1.9) asuvatest patareidest. Raami mahub 8 AA patareid.

## 1.4 BrickPi programmeerimine

Dexter Industries poolt on loodud teegid, mis toetavad BrickPi'd kolmes programmeerimiskeeles. Nendeks on C, Python ja Scratch. Nende teekide versioonid Dexter Industries poolt loodud operatsioonisüsteemis, mida kirjeldatakse peatükis 3, on:

- Python – 2.7.9
- C - (Debian GLIBC 2.19-18+deb8u6) 2.19
- Scratch – 1.4

Lisaks toetab BrickPi programmeerimist kasutades Javat, BlocklyTalkyt, NodeJS, Ruby, Erlang või Wylidrin. Küll on tugi viimati nimetatud programmeerimiskeeltele loodud kogukonna liikmete endi poolt ning ei ole garanteeritud, et kõik töötab eeldatud viisil.

BrickPi trükkplaadi teiselt generatsioonilt üleminek kolmanda generatsiooni trükkplaadile vajab uusi teadmisi, sest kolmanda generatsiooni jaoks on Dexter Industries kirjutanud uued teegifailid, mis ei ole tagasiühilduvad vanema generatsiooni trükkplaadiga. Seetõttu vajavad programmid, mis on tehtud kasutades vanemat BrickPi'd ja selle toeks loodud teegifaile, põhjalikke muudatusi, et töötaksid ka kolmanda generatsiooni BrickPi'ga.

## 2 Konkureerivate platvormide tutvustus ja võrdlus

BrickPi esmane eesmärk on pakkuda alternatiivset juhtkontroller LEGO klotsidest ja anduritest/mootoritest valmistatud robotile. Peatüki järgmistes alapunktides kirjeldame, millised on lähimate konkureerivate robotikaplatvormide juhtkontrollerid. Kuna BrickPi on otsene konkurent LEGO Mindstorms NXT ja LEGO Mindstorms EV3 platvormide juhtkontrolleritega, siis just neid kirjeldatakse. Peatüki lõpus tuuakse välja põhilisemad erinevused LEGO ajude ja BrickPi vahel. Kindlasti on BrickPi platvormil ka teisi konkurente, aga kuna need on kaugemad konkurendid ja vähem levinud, siis jäävad antud töö skoobist välja

### 2.1 LEGO Mindstorms NXT tutvustus

LEGO Mindstorms NXT [17] (edaspidi NXT) on robotika platvorm, mis muutus avalikkusele esmakordselt kättesaadavaks 2006 aasta juulis. NXT platvorm on otsene järeltulija LEGO RCX platvormile. NXT komplekte on kahte tüüpi. Üks on kommertslikumal eesmärgil komplekteeritud komplekt ja teine on hariduslikul eesmärgil. Haridusliku eesmärgiga komplekt sisaldab rohkemaid andureid võrreldes kommertskomplektiga. Kuid kommertskomplekt sisaldab rohkemaid klotse roboti ehitamiseks. Põhiline keel NXT programmeerimise jaoks on NXT-G, kuid tootja poolt on toetatud ka valikuline LabVIEW tarkvara NXT'le. Kasutajate poolt on loodud ka keeled nagu NXC, NBC, leJOS NXJ ja RobotC. Varem nimetatud ametlikud keeled (LabVIEW ja NXT-G) on graafilised programmeerimiskeeled, mis tähendab, et robotit on võimalik programmeerida lohistades töölaual erinevaid ikooni. Need ikoonid on sisult väiksed programmijupid.



**Joonis 2.1** LEGO Mindstorms NXT juhtblokk. [17]

Aastal 2009 tuli välja ka uuendatud versioon NXT platvormist, mille nimetuseks sai NXT 2.0.

Uuendatud platvormil on:

- 32-bitine Atmel AT91SAM7S256 mikrokontroller (ARM7)
- 8-bitine Atmel ATmega48 mikrokontroller lisaks põhilisele
- Mälu 64 KB RAM ja 256 KB Flash
- 4 porti sensorite jaoks
- 3 porti servomootorite jaoks
- USB port
- LCD 100 x 64 piksline displei
- Sinihammas V2.0
- Kõlar
- Neli juhtnuppu
- Toetab ühendust Android seadmega
- Toiteks sobib kas 6 AA patareid või NXT taaslaetav aku

Võrreldes vanema platvormiga parandati uue komplektitarkvara märkimisväärselt ning valgussensor asendati värvisensoriga. Uus tarkvara oli oluliselt usaldusväärsem, jooksis harvemini kokku ning lisas uusi funktsioone. Uus tarkvara oli ühilduv ka vanemate LEGO Mindstorms NXT komplektide ajudega.

## 2.2 LEGO Mindstorms EV3 tutvustus

LEGO Mindstorms EV3 [18] on otsene järeltulija NXT 2.0 platvormile. Platvormi nimetuses EV on tulnud inglisekeelsest sõnast *Evolution* ja number 3 tähistab Mindstorms toodete sarjas kolmandat platvormi RCX ja NXT järel. Müüki jõudis EV3 2013 septembris. Võrreldes NXT platvormiga, on EV3 kontrolleri võimsam ARM9 tehnoloogia protsessor. Ka EV3 platvormi müüakse kahes variatsioonis. Üks on mõeldud kodukasutajale ning teine on mõeldud haridusasutustele. Hariduslikul eesmärgil loodud variant sisaldab rohkemaid andureid, samas ehitamiseks sobivaid klotse on komplektis vähem.



**Joonis 2.2** LEGO Mindstorms EV3 juhtblokk [21]

LEGO Mindstorms EV3 platvormil saab taaskasutada LEGO Mindstorms NXT platvormi andureid ning mootoreid, kuid EV3 andurid ei ole kasutatavad NXT platvormil. Küll aga saab NXT platvormi aju programmeerida kasutades EV3 jaoks mõeldud arendus keskkonda ning saab kasutada ka EV3 mootoreid.

EV3 tehnilised eripärad on:

- Texas Instruments Sitara AM1808 mikrokontroller (ARM9)
- Mälu 64 MB RAM ja 16 MB Flash
- Laiendatav mälu kuni 32GB kasutades microSHDC kaarti
- USB port lisaseadme tarbeks
- 4 porti sensorite jaoks

- 4 porti servomootorite jaoks
- USB port
- Sinihammas V3.0
- LCD 178 x 128 piksline displei
- Kõlar
- Kuus juhtnuppu
- Toetab ühendust Android ja iOS seadmega
- Taaslaetav aku

LEGO Mindstorms EV3 puhul saab asendada toiteallikana kaasas olevat akut ka kuue AA patareiga.

## 2.3 Põhilised erinevused platvormide lõikes

Omadus	BrickPi	LEGO Mindstorms EV3	LEGO Mindstorms NXT
Sensorite pesasid	4-5	4	4
Mootori pesasid	3-4	4	3
Toetatud keeled	C, Python, Java, Scratch, BlocklyTalky, NodeJS, Ruby, Erlang, Wylidrin	Scratch, Microsoft Visual Basic, LabVIEW, Java, C	Ada, NXT-G, Python, LabVIEW, Perl, Objective-C, Java, NQC, C, Haskell, Ruby, Simulink, Microsoft Visual Programming Language,
Toiteallikas	8x AA patareid	2050mAh suurune aku, 6x AA patareid	2100mAh aku, 6x AA patareid

Üks väikseid eeliseid BrickPi kasutamisel LEGO Mindstorms kontrolleri ees on see, et programmeeritavate portide lisamine on lihtsam. BrickPi trükkplaatide puhul on BrickPi'sid võimalik üksteise otsa lisada, olemasolevaid porte hõivamata. Selleks tuleb vaid uus BrickPi trükkplaat asetada eelmisele. Samas LEGO Mindstorms EV3 puhul tuleb selleks kasutada porti multiplekserit. NXT platvormiga ühilduvat multiplekserit on kujutatud joonisel 2.3.



**Joonis 2.3** Multiplekser [22]

Multiplekser hõivab ühe olemasoleva porti, et lisada juurde kaks või neli porti. Küll aga LEGO Mindstorms EV3 robotitele pakutakse erinevaid multipleksereid, mille abil saab ühte porti näiteks laiendada kaheks või isegi neljaks. Kuna ka LEGO Mindstorms robotika ajud toetavad multipleksereid, ei oma BrickPi selles portide laiendamise lihtsuse näol märkimisväärset eelist BrickPi ees.

Eelis, mida omas BrickPi pikalt LEGO Mindstorms platvormide ees, oli võimalus kergelt robotit programmeerida kasutades programmeerimiskeelt Python. See eelis on ära kadunud, kuna on valminud operatsioonisüsteem ev3dev, mis baseerub Linux Debianil. Seda operatsioonisüsteemi on kerge installeerida LEGO Mindstorms EV3 ajule. Programmeerimise kohta Pythonis kasutades LEGO Mindstorms EV3 on hea lugeda 2017 kevadel Henry Maalinn poolt kaitstavat bakalaureusetööd „LEGO MINDSTORMS EV3-e programmeerimine Pythonis“[15].



Suurimateks miinusteks BrickPi'1 võrreldes LEGO Mindstorms robotikaplatvormidega on:

- Hind
- Seadistamise keerukus
- Vajab LEGO klotse, andureid, mootoreid.
- Vajab lisariistvara

Ostes LEGO Mindstorms EV3 või NXT robotikaplatvormi, saadakse kõik, mida on vaja, et alustada kohe programmeerimist. BrickPi tavakomplekti tellides saadakse ainult BrickPi trükkplaat. Lisaks tuleb muretseda Raspberry Pi ja LEGO klotsid, andurid ja mootorid eraldi. Hetkel maksab BrickPi trükkplaat 99,99 USD [5]

# 3 Arenduskeskkonnad ja sobilikud platvormid

BrickPi vajab tarkvara jaoks Raspberry Pi'd. Kogu tarkvara jookseb Raspberry Pi pealt, BrickPi on vaid ühendusliides LEGO EV3 ühilduvate sensorite/mootorite ja tarkvara vahel. Antud töös vaatleme kahte operatsioonisüsteemi, mis toetavad programmeerimisel BrickPi trükkplaati ning installeeruvad Raspberry Pi peale:

1. Ev3dev - toetab nii Java kui Python programmeerimiskeelt
2. Raspbian for Robots

Ev3dev on operatsioonisüsteem, mida toetavad nii LEGO Mindstorms EV3 enda ajud kui ka Raspberry Pi ja BrickPi kombinatsioon. Raspbiani for Robots on Raspbiani versioon, mida antud töös kasutatakse. See on Dexter Industries, kes on BrickPi trükkplaadi tootja, poolt modifitseeritud versioon Raspbian Jessie'st. Silmapaistvaim erinevus tavalisest Raspbian versioonis on see, et see sisaldab näidisprojekte C, Pythoni ja Scratch keeltes. Samuti on vajalikud teegi eelinstalleeritud eelpoolmainitud programmeerimiskeeltes kirjutatud programmide kirjutamiseks ja jooksutamiseks.

## 3.1 Töökäes vajalikud programmid

Bakalaureuse tööd tehes oli vaja kasutada järgmisi programme.

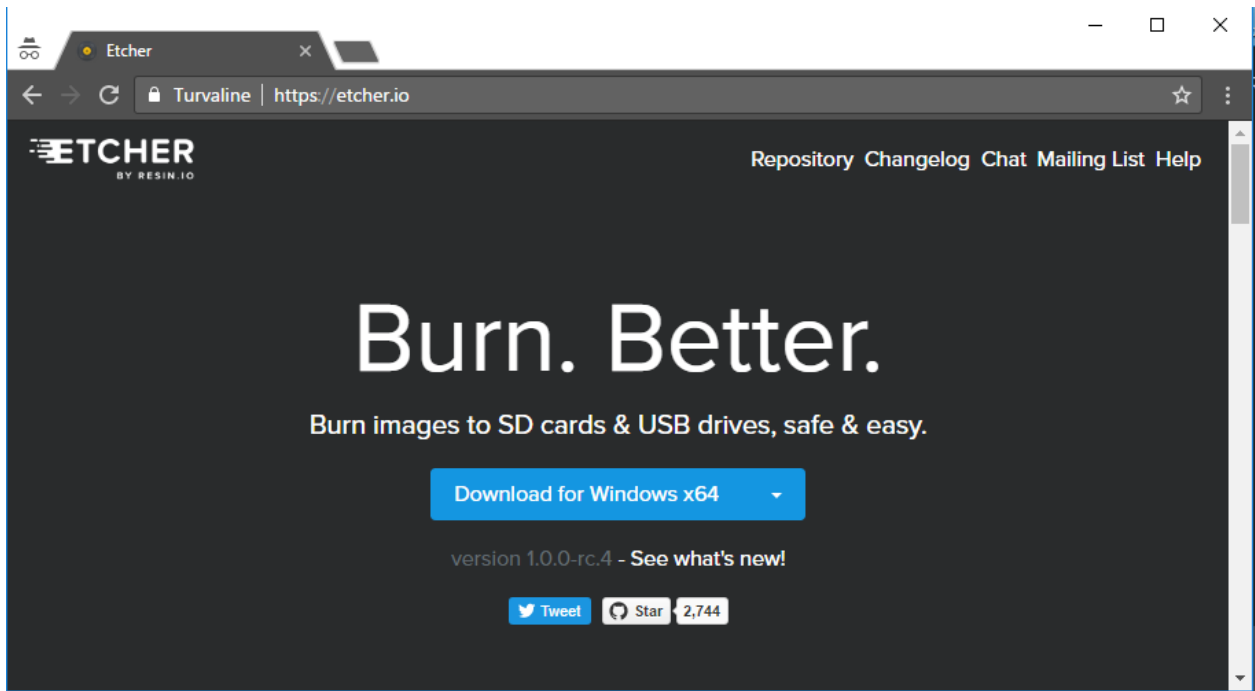
1. Etcher
2. PuTTY
3. WinSCP
4. PyCharm
5. WinRAR

Lisaks on vaja kujutisfaile, mis sisaldavad Raspberry Pi jaoks vajalikku operatsioonisüsteemi.

Antud tööd on kasutatud kahte operatsioonisüsteemi – Raspbian for Robots by Dexter Industries ja ev3dev.

### 3.1.1 Etcher

Etcher on vabavaraline programm, mis võimaldab kirjutada kujutisfaile mälukaartidele ja USB pulkadele. Antud programmi kasutati ev3dev ja Raspbiani kujutisfailide kirjutamiseks SD mälukaardile. Etcherit on võimalik alla laadida aadressilt <http://etcher.io>. Joonisel 3.1 on kujutatud veebilehe tõmmis Etcheri kodulehelt.



**Joonis 3.1** Etcheri koduleht, kust võimalik programm alla laadida.

Laadige endale sobilik versioon programm alla ja installeerige see meelepärasesse kohta.

### 3.1.2 PuTTY

PuTTY on vabavaraline alternatiivne käsurea programm, mis toetab erinevad võrgusuhtluse protokolle, nagu näiteks SSH, SCP, Telnet, rlogi, jne. PuTTY abil on võimalik luua ühendus virtuaalmasinaga. Antud programmi sai kasutatud et ühenduda Raspbiani/ev3dev virtuaalmasinasse, mis jookseb Raspberry Pi peal.

### 3.1.3 WinSCP

WinSCP on vabavaraline programm, mis lihtsustab failide liigutamist lokaalsest masinast virtuaalmasinasse ning vastupidi. Selle abil on võimalik näiteks kopeerida lokaalsesse

masinasse kõik lähtekoodi failid, juhendid ja muu vajalik Raspberry Pi peal jooksvast operatsioonisüsteemist.

### **3.1.4 PyCharm**

PyCharm[6] on ettevõtte JetBrains poolt toodetud integreeritud programmeerimiskeskond, mis on spetsialiseeritud arendustööks kasutades Pythoni programmeerimiskeelt. Antud töös on kasutatud versiooni PyCharm Ultimate 2017.1. Võrreldes Community versiooniga programmist, on Ultimate suurim eelis see, et võimaldab harrastada kaugpääsuga programmeerimist. See tähendab, et muudetavad failid ei asu mitte lokaalses masinas, vaid asuvad Raspberry Pi'l. Samuti on PyCharmile sisse ehitatud käsureaprogrammi tugi. PyCharm hõlbustab ilma lisaamme tegemata otse integreeritud programmeerimiskeskonnast uue versiooni failist üles laadida Raspberry Pi'sse ning seal ka kohe seda käivitada.

### **3.1.5 WinRAR**

WinRAR on levinuim failide arhiveerimisprogramm. Kuna enamus juhendeid ja õpetusi virtuaalmasinas on kokku pakitult, siis on vaja nende avamiseks WinRAR programmi. Lisaks on kujutisfail, millel on vajalik operatsioonisüsteem, et Raspberry Pi saaks suhelda BrickPi trükkplaadiga, samuti kokku pakitud WinRAR abil. Seegi tuleb enne lahti pakkida, et Etcheri abil SD kaardile kirjutada.

### **3.1.6 Raspbian for Robots**

Dexter Industries poolt on ette valmistatud Raspbiani versioon, mida nimetatakse Raspbian for Robots. See lihtsustab nende poolt toodetavate trükkplaatide üles seadmist. Sama versiooni Raspbianist saab lisaks BrickPi'le, kasutada ka teiste Dexter Industries toodete puhul. Näiteks Arduberry[7], GoPiGo[8] ja GrovePi[9] arendusplaatide puhul.

Arduberry lihtsustab arduino andurite ühendamist Raspberry Pi'ga. GoPiGo on Dexter Industries poolt toodetav täiskomplekt, millest on võimalik valmis ehitada algeline robotauto. GrovePi on elektroonika plaat, mis võimaldab erinevaid Grove andureid kasutades programmeerida Raspberry Pi peal.

Raspbian for Robotsil on järgmised versioonid programmeerimiskeelte teekidest:

- Python – 2.7.9
- Java - 1.8.0\_65
- C - (Debian GLIBC 2.19-18+deb8u6) 2.19
- Scratch – 1.4

Kuna Scratch 1.4 on üsna vana versioon ja arenduses on ka juba Scratch 3.0 [10], siis uuriti ka võimalusi Scratch 2.0 installeerimiseks Raspberry Pi platvormile. Scratch 2.0 aga vajab Adobe Air installatsiooni, mida Raspberry Pi ei toeta. Seda seetõttu, et Adobe Air ei toeta ARM tüüpi protsessoreid, mida kasutatakse Raspberry Pi peal. Adobe Air'ita aga ei ole võimalik Scratch 2.0 kasutada. Lisaks on Scratch 2.0 väga ressursinõudlik, mistõttu teda ei oleks otstarbekas Raspberry Pi peal kasutada.

### **3.1.7 ev3dev**

Ev3dev on alternatiivne operatsioonisüsteem, mida kasutada Raspbian for Robots asemel Raspberry Pi's, et see saaks suhelda BrickPi trükkplaadiga. Ev3dev põhineb Linuxi Debiani[11] versioonil. Linux Debian tasuta saadaolev alternatiivne operatsioonisüsteem Windows operatsioonisüsteemile. Enne ev3dev installeerimist tuleb veenduda, et on alla tõmmanud õige versioon kujutisfailist. Valikus on neid 4. Üks on mõeldud EV3 riistvarale installeerimiseks, kaks neist on mõeldud Raspberry Pi riistvarale. Siinkohal tuleb vaadata jällegi, mis Raspberry Pi versiooni kasutatakse. Ja neljas versioon on mõeldud BeagleBone jaoks.

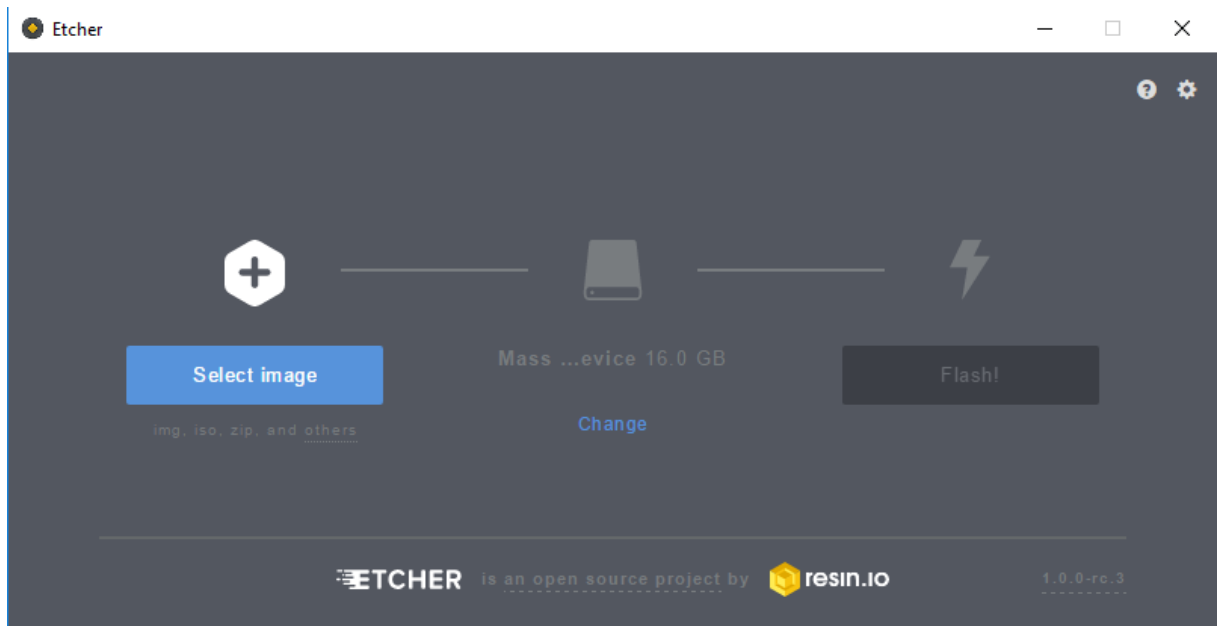
## **3.2 Ev3dev**

### **3.2.1 Ev3dev installeerimine**

Enne installeerimist on vaja vähemalt 2GB mälu mahuga SD kaarti. Kindlasti veenduge et mälukaart ei oleks suurem kui 32GB, sest sellisel juhul ei ole seda võimalik ev3dev operatsioonisüsteemiga kasutada. Ev3dev installeerimiseks:

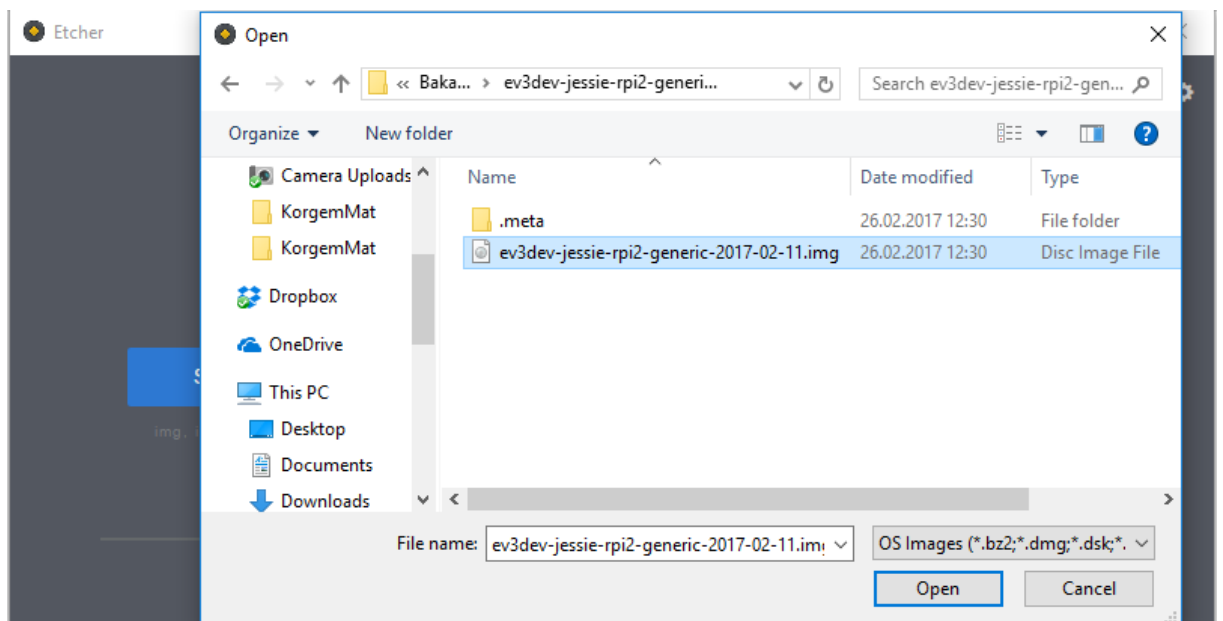
1. Laadi alla ev3dev arhiivifail nende kodulehelt[12] ja paki see lahti, et saada kujutisfail.

2. Käivita Etcher. Avaneb joonisel 3.2 kujutatatu.



**Joonis 3.2** Etcheri käivitamine

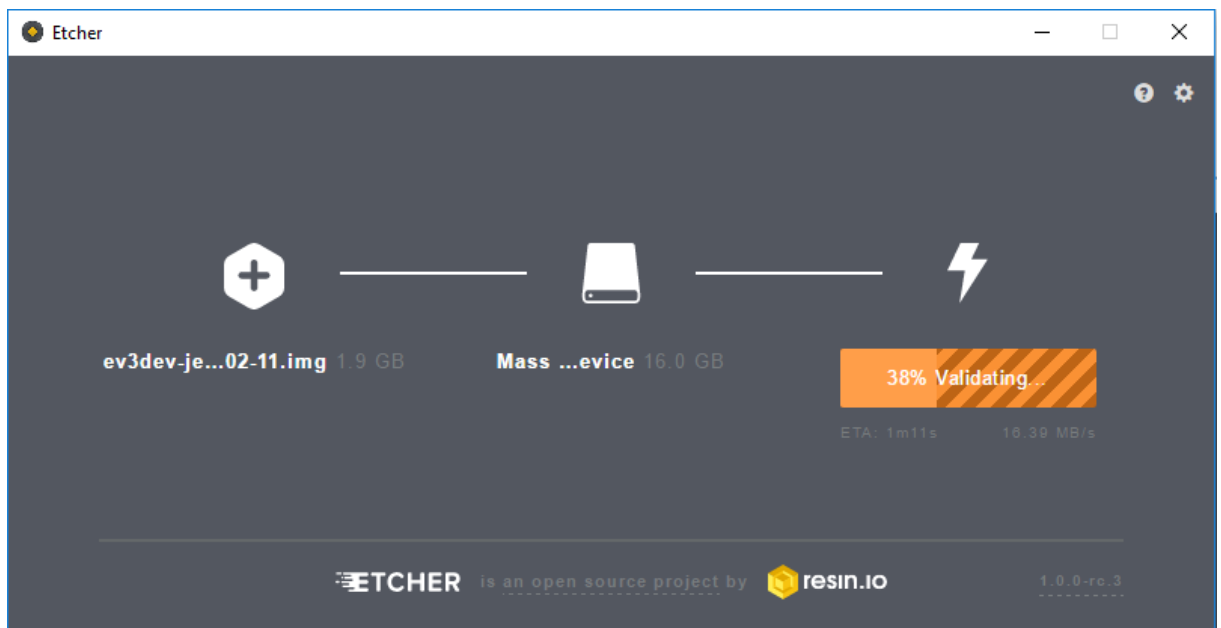
3. Vali õige kujutisfail, mida soovid SD kaardile kirjutada



**Joonis 3.3** Kujutisfaili valimine

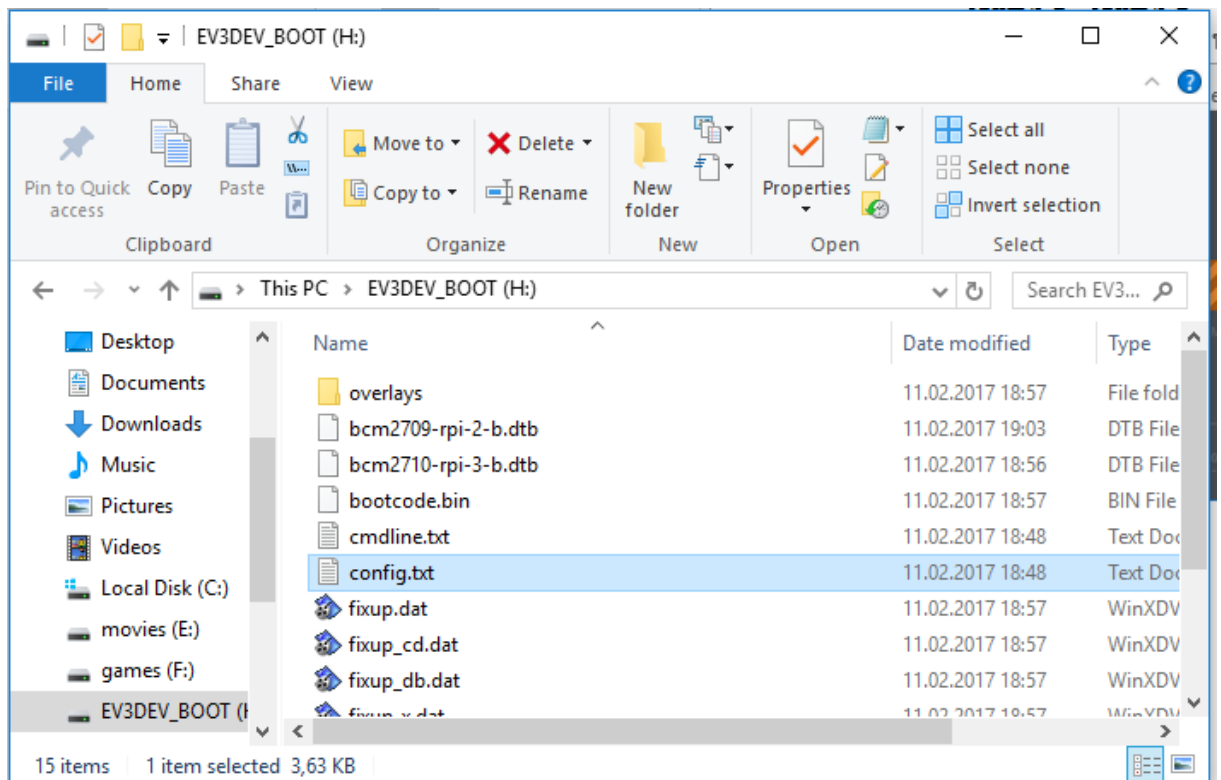
4. Vajuta „Flash“

- Oota, kuni Etcher kirjutab SD kaardile ja teeb ka failide valideerimise



**Joonis 3.4** Kujutisfaili kirjutamine ja valideerimine

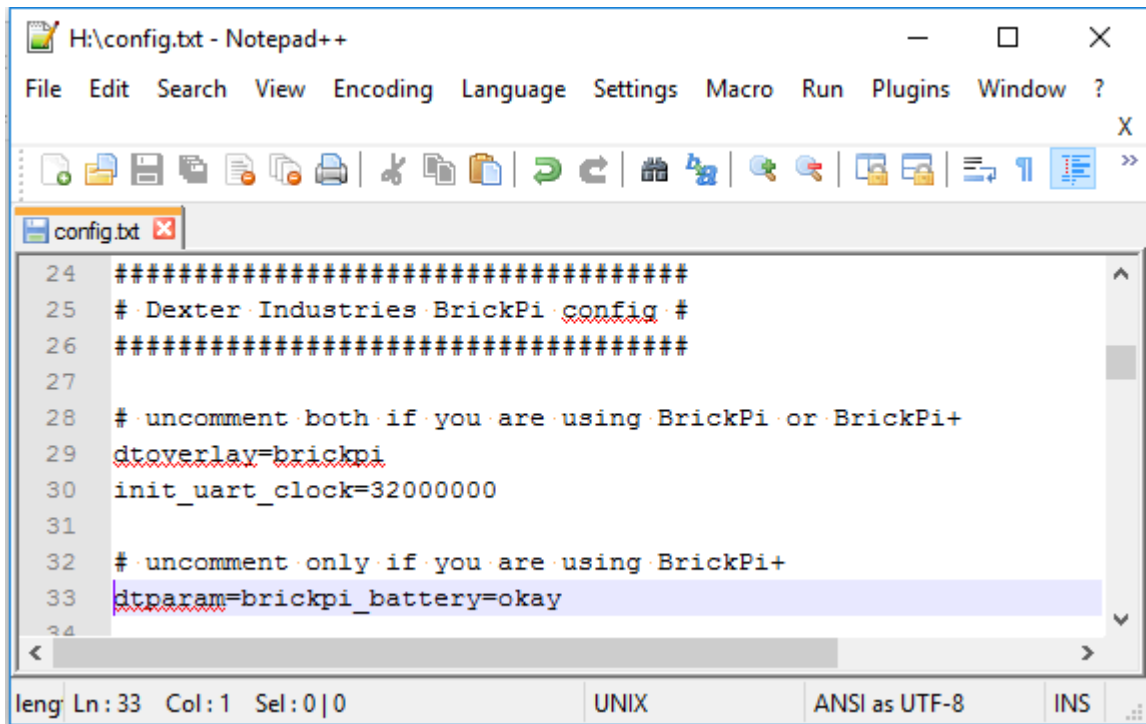
- Kui installeerimine on lõppenud, ava läbi kataloogi lehitseja mälukaardi sisu ja otsi fail „config.txt“



**Joonis 3.5** Faili „config.txt“ asukoht

- Failis „config.txt“, eemalda trellide märk (#) ridade eest, kus seadistatakse muutujad
  - dtoverlay=brickpi
  - init\_uart\_clock=32000000

c. dtparam=brickpi\_battery=okay



```
H:\config.txt - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
config.txt
24 #####
25 # Dexter Industries BrickPi config #
26 #####
27
28 # uncomment both if you are using BrickPi or BrickPi+
29 dtoverlay=brickpi
30 init_uart_clock=32000000
31
32 # uncomment only if you are using BrickPi+
33 dtparam=brickpi_battery=okay
34
leng Ln: 33 Col: 1 Sel: 0 | 0 UNIX ANSI as UTF-8 INS
```

**Joonis 3.6** Failis „config.txt“ muudetavad muutujad

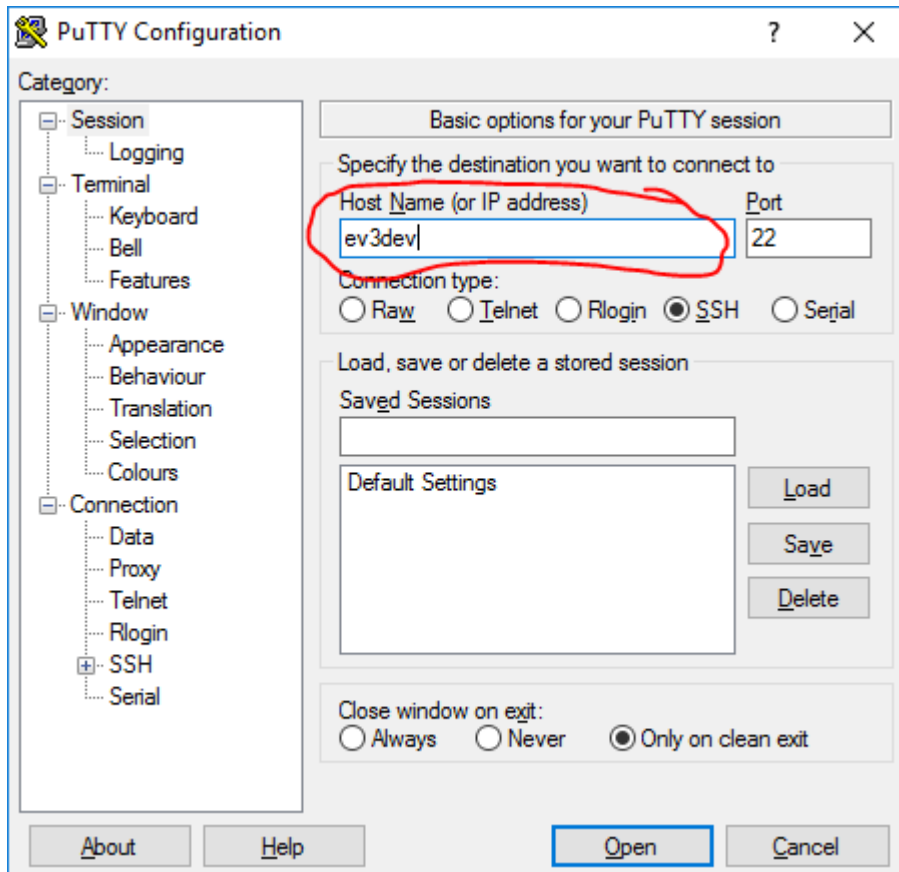
8. Võta mälukaart USB seadmes ja sisesta see Raspberry Pi mälukaardipesa
9. Käivita Raspberry Pi



### 3.2.2 Ühendamine ev3dev virtuaalmasinasse

Esimesel korral ev3dev virtuaalmasinasse ühendamisel on soovituslik seadistada ka WiFi tongel, et ei peaks olema koguaeg võrgu kaablit taga. Selleks:

1. Ühenda võrgukaabel
2. Ava PuTTY ja ühenda aadressile „ev3dev“



**Joonis 3.7** PuTTY'ga ühendamine virtuaalmasinasse

3. Sisene kasutades kasutajatunnust „robot“ ja parooli „maker“
4. Mine võrgühenduste haldurisse kasutades käsku  
**\$ connmanctl**
5. Võimalda wifi ühendus olles võrgühenduste halduris:  
**> enable wifi**
6. Võimalda agent käsuga:  
**> agent on**
7. Leia üles saadaval olevad võrgud kasutades käsku:  
**> services**

8. Peale seda kuvatakse saadaval olevad võrgud.

```
robot@ev3dev: ~  
connmanctl> services  
*AO Wired          ethernet_b827eb1672fa_cable  
WhySoSerious      wifi_74da385850fe_576879536f536572696f7573_managed_psk  
1ae8d4            wifi_74da385850fe_316165386434_managed_psk  
LIANA             wifi_74da385850fe_4c49414e41_managed_psk  
059886           wifi_74da385850fe_303539383836_managed_psk  
1b396a           wifi_74da385850fe_316233393661_managed_psk  
720286           wifi_74da385850fe_373230323836_managed_psk  
kebab24          wifi_74da385850fe_6b656261623234_managed_psk  
connmanctl>
```

**Joonis 3.8** Saadaval olevad WiFi võrgud

9. Ühendamiseks kindlasse wifi võrku sisesta käsk:

> **connect** <võrk>

```
robot@ev3dev: ~  
connmanctl> connect wifi_74da385850fe_576879536f536572696f7573_managed_psk
```

**Joonis 3.9** WiFi võrku ühendamine

10. Sisesta võrgu parool. Töös kasutasin võrku WhySoSerious, mille parool on MullaKaitse123

```
robot@ev3dev: ~  
Agent RequestInput wifi_74da385850fe_576879536f536572696f7573_managed_psk  
  Passphrase = [ Type=psk, Requirement=mandatory ]  
Passphrase? MullaKaitse123  
Connected wifi_74da385850fe_576879536f536572696f7573_managed_psk  
connmanctl>
```

**Joonis 3.10** WiFi võrku ühendamisel parooli sisestamine

11. Välju võrguühenduste haldurist:

> **exit**

12. Et teada saada IP, millega saab läbi wifi võrgu ühendada masinasse kasutades

PuTTY't, kasuta **ifconfig** käsku. Seda vaja kasutada **root** õigustes. Root õiguste parool on „maker“. Sisesta käsk

**\$ sudo ifconfig**

13. Otsi **wlan0** alt IP. Selle IP abil on võimalik PuTTY'ga ühendada masinasse ka siis, kui füüsiline võrgukaabel on seadme tagant lahti ühendatud.

```
robot@ev3dev: ~  
[sudo] password for robot:  
eth0      Link encap:Ethernet  HWaddr b8:27:eb:16:72:fa  
          inet addr:192.168.1.208 Bcast:192.168.1.255 Mask:255.255.255.0  
          inet6 addr: fe80::ba27:ebff:fe16:72fa/64 Scope:Link  
          inet6 addr: 2001:7d0:87c6:df80:ba27:ebff:fe16:72fa/64 Scope:Global  
          UP BROADCAST RUNNING MULTICAST DYNAMIC MTU:1500 Metric:1  
          RX packets:5231 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:2104 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:1000  
          RX bytes:373689 (364.9 KiB)  TX bytes:252147 (246.2 KiB)  
  
lo        Link encap:Local Loopback  
          inet addr:127.0.0.1 Mask:255.0.0.0  
          inet6 addr: ::1/128 Scope:Host  
          UP LOOPBACK RUNNING MTU:65536 Metric:1  
          RX packets:35 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:35 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:1  
          RX bytes:3203 (3.1 KiB)  TX bytes:3203 (3.1 KiB)  
  
wlan0     Link encap:Ethernet  HWaddr 74:da:38:58:50:fe  
          inet addr:192.168.1.210 Bcast:192.168.1.255 Mask:255.255.255.0  
          inet6 addr: 2001:7d0:87c6:df80:76da:38ff:fe58:50fe/64 Scope:Global  
          inet6 addr: fe80::76da:38ff:fe58:50fe/64 Scope:Link  
          UP BROADCAST RUNNING MULTICAST DYNAMIC MTU:1500 Metric:1  
          RX packets:787 errors:0 dropped:49 overruns:0 frame:0  
          TX packets:204 errors:0 dropped:1 overruns:0 carrier:0  
          collisions:0 txqueuelen:1000  
          RX bytes:99750 (97.4 KiB)  TX bytes:23634 (23.0 KiB)
```

**Joonis 3.11** Virtuaalmasina IP leidmine

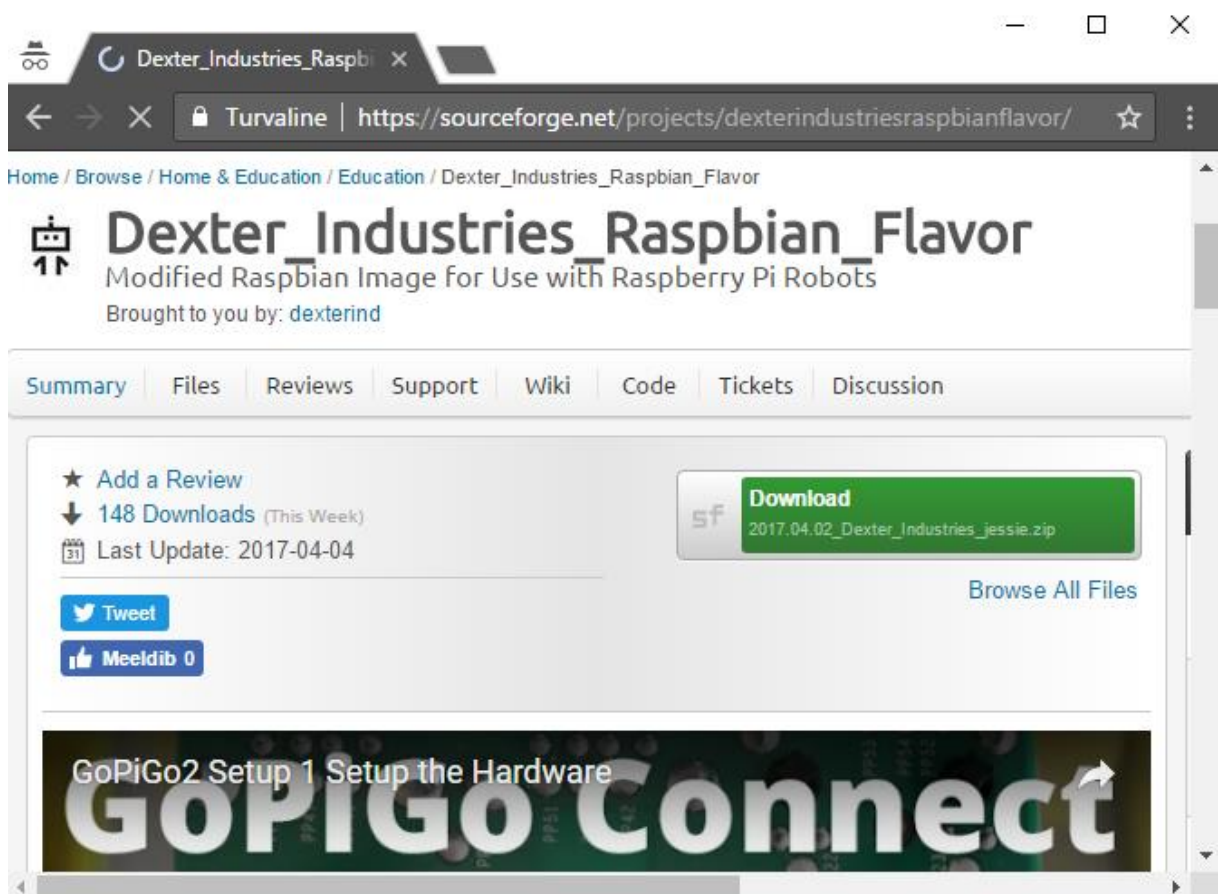
Nüüd on võimalik seadme tagant lahti ühendada võrgukaabel, ka kasutades PuTTY't, saab kasutades IP 192.168.1.210 ligi ev3dev virtuaalmasinasse, mis jookseb Raspberry Pi peal jooksvasse virtuaalmasinasse.

## 3.3 Raspbian for Robots

### 3.3.1 Raspbian for Robots installeerimine

Enne installeerimist on vaja vähemalt 8GB mälu mahuga SD kaarti ning seadet sellele mälukaardile kirjutamiseks. Selleks, et saada Raspbian operatsioonisüsteemiga mälukaart, tuleb:

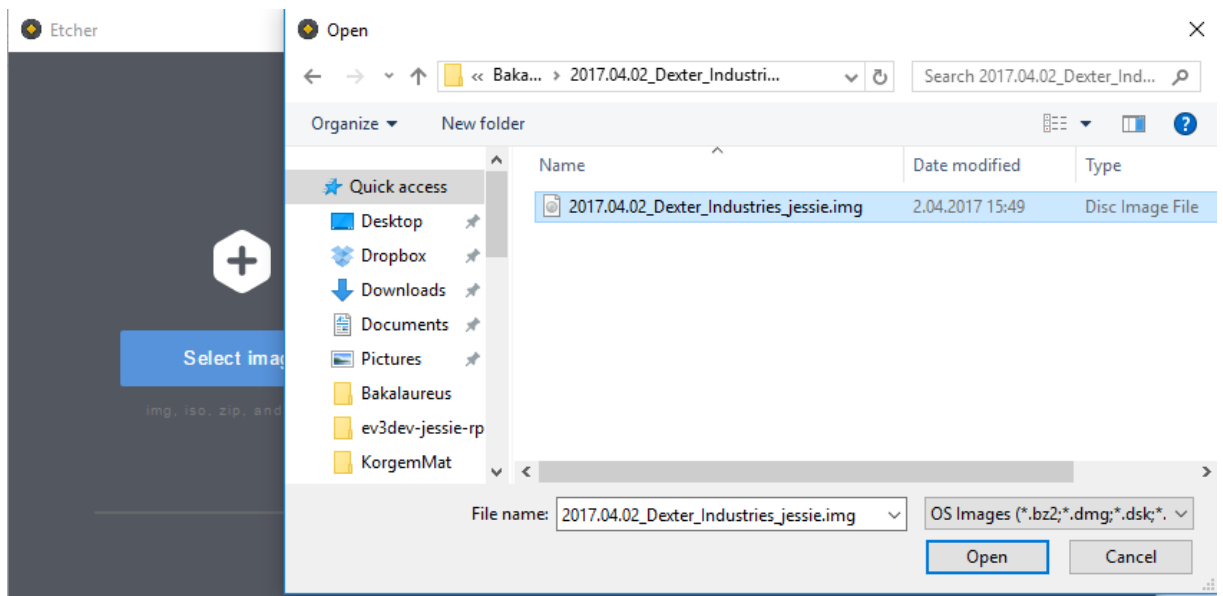
1. Alla laadida Raspbian for Robots kujutisfailiga arhiivfail kas Google Drive aadressilt[13] või Sourceforge lehelt[14].



**Joonis 3.12** Veebilehe tömmis kujutisfaili allalaadimislingilt.

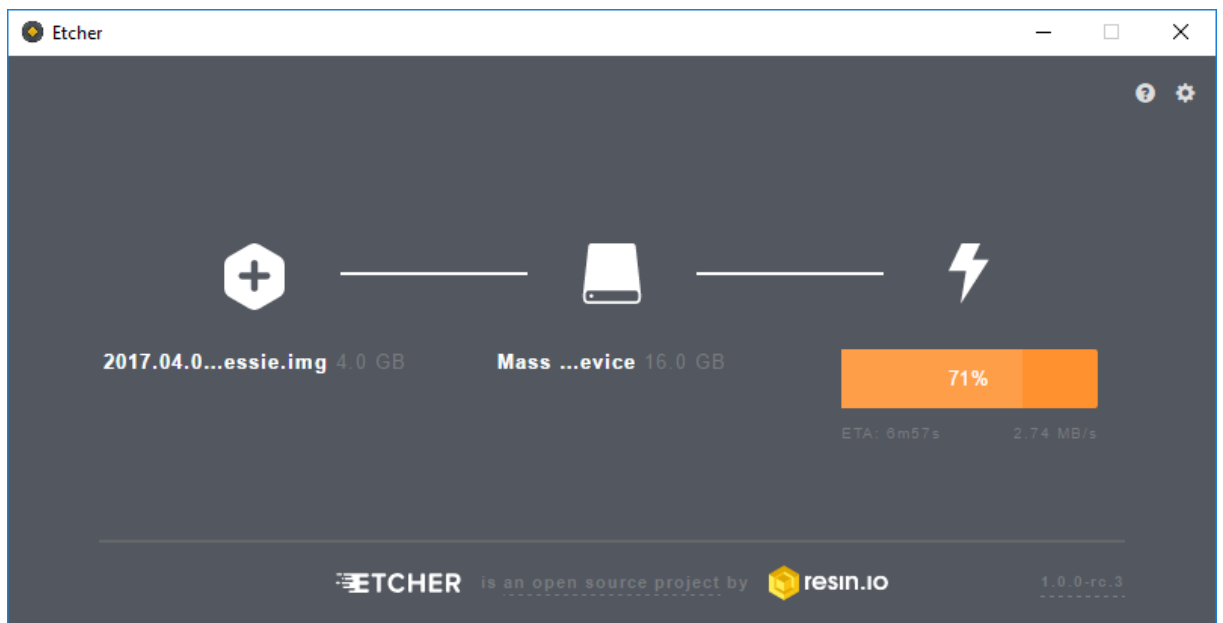
2. Lahti pakkida alla laaditud arhiivifail.
3. Sisestada SD-kaardile kirjutamise seade koos SD-kaardiga USB pessa
4. Käivitada Etcher

5. Valida kujutisfail, mida soovid SD kaardile kirjutada



**Joonis 3.13** Kujutisfaili valimine

6. Vajutada „Flash“
7. Oodata kuni Etcher kirjutab SD kaardile ja teeb ka failide valideerimise



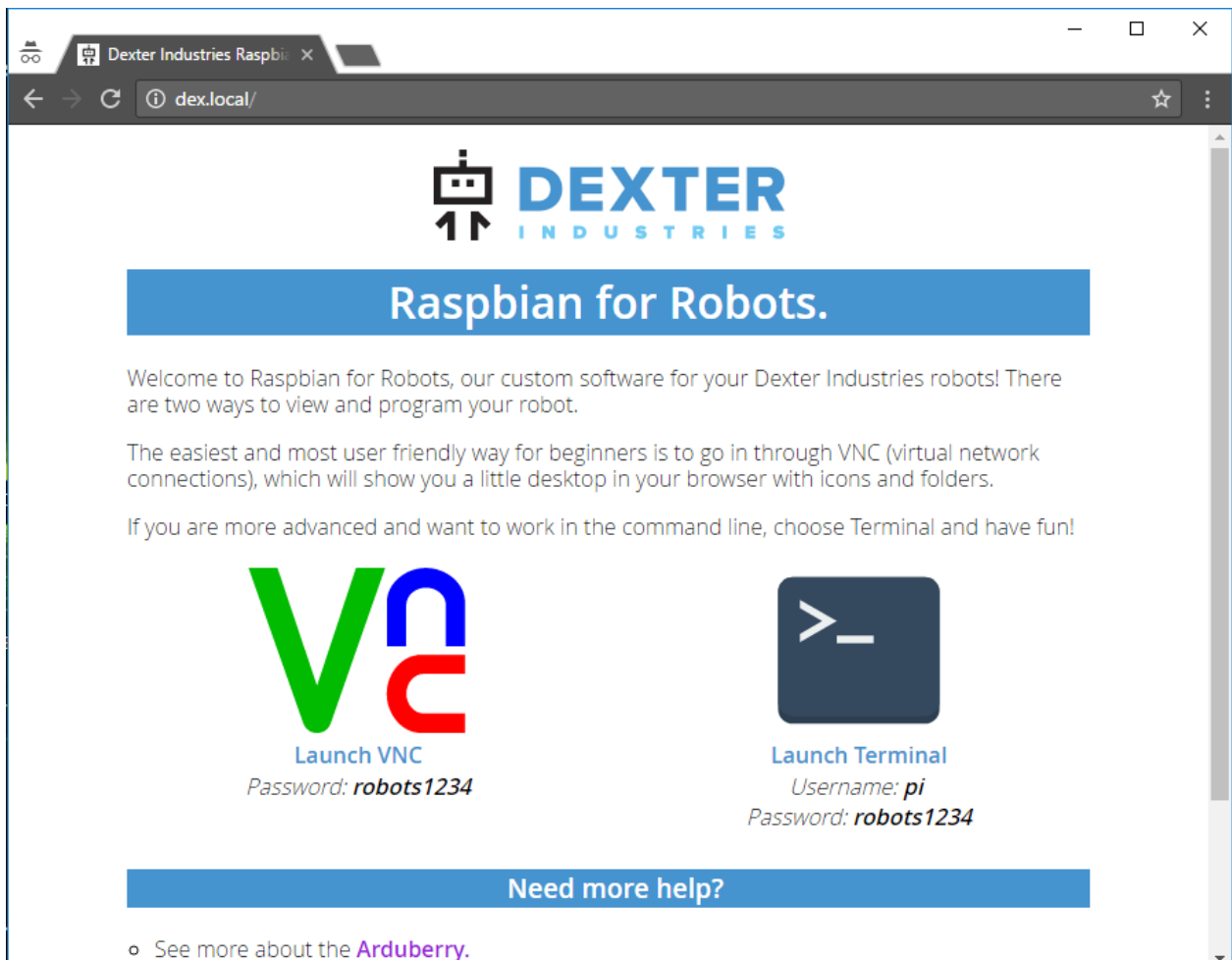
**Joonis 3.14** Kujutisfaili kirjutamine ja valideerimine

8. Võtta mälukaart USB seadmes ja sisesta see Raspberry Pi mälukaartipessa
9. Käivitada Raspberry Pi

### 3.3.2 Ühendumine Raspbiani virtuaalmasinasse

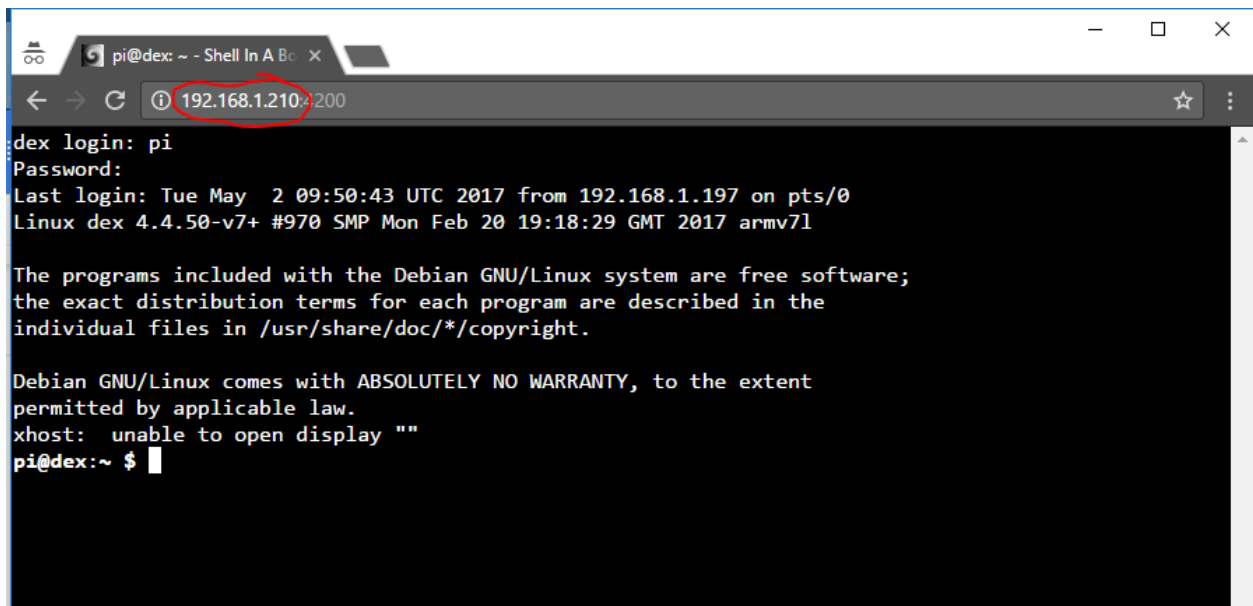
Dexteri poolt modifitseeritud Raspbian for Robots operatsioonisüsteem toetab ka kohe peale installatsiooni lõppu ühendamist Raspberry Pi peal jooksvasse virtuaalmasinasse. Ühendada

saab virtuaalmasinasse kasutades nii käsurea tööriista kui ka VNC. Kasutades endale meelepärast veebibrauserit avage aadress <http://dex.local>. Avatud aadressil avaneb võimalus kasutada brauseri vahendusel nii VNC kui käsureatööriista. Vaata joonist 3.15. Tasub tähele panna, et arvuti, mida kasutatakse, peab olema samas võrgus ning esmakordsel ühendamisel peab olema Raspberry Pi ühendatud samasse võrku kaabli abil.



**Joonis 3.15** Aadressilt dex.local avanev leht.

Seadmesse ühendades on kasutajatunnuseks “pi” ja selle parooliks on “robots1234”. Esmakordsel ühendamisel soovitaks kasutada käsureaprogrammi. Jooniselt 3.15 vajuta “Launch Terminal” ning logi sisse. Seal saab teada ka IP aadressi (vaata joonis 3.16), mis on antud seadmele määratud. Selle IP abil on võimalik kasutada ka meelepärast käsurea programmi, et ühendada sellesse seadmesse.



**Joonis 3.16** Brauseris avanev käsurea programm ning seadmele määratud IP.

Esmalt soovitaks seadistada raspberry pi kasutama WiFi võrku. See võimaldab ühendada seadmesse ilma, et seadmel peaks olema võrgukaabel ühendatud. Selleks sisesta WiFi pulk ühte Raspberry Pi USB portidest. Järgmiseks kasutades terminali akent, näiteks seda, mis on välja toodud joonisel 3.16 ning sisesta järgmised käsud:

1. Lisa enda WiFi võrgu sätteid seadme konfiguratsiooni kasutades käsku:

```
$ sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

2. Lisa avanenud faili lõppu:

```
network={  
    ssid="<võrgunimi>"  
    psk="<võrguparool>"  
}
```

```
GNU nano 2.2.6      File: /etc/wpa_supplicant/wpa_supplicant.conf

country=GB
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
    ssid="WhySoSerious"
    psk="MullaKaitse123"
}

^G Get Help      ^O WriteOut      ^R Read File      ^Y Prev Page      ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is       ^V Next Page      ^U UnCut Text    ^T To Spell
```

**Joonis 3.17** Info WiFi võrgu kohta failis „wpa\_supplicant.conf“

3. Vajuta CTRL+X ja siis “Yes”, et muudatused salvestuksid.
4. Konfigureeri uuesti võrgu seadmed kasutades käsku

**\$ sudo wpa\_cli reconfigure**

```
pi@dex:~ $ sudo wpa_cli reconfigure
Selected interface 'wlan0'
OK
pi@dex:~ $
```

**Joonis 3.18** WiFi edukalt seadistatud

Võib juhtuda, et Raspberry Pi’le määratakse peale seda ka uus IP. Kui see peaks juhtuma, siis uue IP saab uuesti kätte kasutades joonisel 3.16 näidatud viisil.

Võrk, mida mina kasutasin, kasutab nime “WhySoSerious” ja selle parool on “MullaKaitse123”. Peale seda võib võrgukaabli eemaldada ja seadmesse on võimalik ühendada üle WiFi. Kuid seadmed peavad olema ka selleks jätkuvalt samas võrgus.

### 3.3.3 BrickPi konfigureerimine

Raspbian for Robots on loodud selliselt, et see toetaks mitmeid erinevad Dexter Industires tooteid. Ka BrickPi trükkplaadist on välja lastud mitu erinevat generatsiooni. Selleks, et Raspbian for Robots oleks seadistatud vastavalt kasutatavale seadmele, on Dexter Industries loonud käsuraaskriptid, mis lihtsustavad konfigureerimist. Antud töös on kasutatud trükkplaati,



mis kasutab nime BrickPi+. Selle plaadi puhul tuleb minna kataloogi Dexter/BrickPi+/Setup\_Files. Selleks sisesta käsk:

```
$ cd Dexter/BrickPi+/Setup_Files/
```

Ning kasuta käsku superõigustega ja siis vajuta „Enter“

```
$ sudo ./install.sh
```



```
pi@dex: ~/Dexter/BrickPi+/Setup_Files $ sudo ./install.sh
Dexter
Industries
BrickPi+

Welcome to BrickPi Installer.
Please ensure internet connectivity before running this script.

NOTE: Raspberry Pi will reboot after completion.
Special thanks to Joe Sanford at Tufts University. This script was derived from his work. Thank you Joe!

Press ENTER to begin...
```

**Joonis 3.17** Seis peale installeerimisskripti jooksmise lõppu ja enne „Enter“ vajutamist et jooksutada seadistamise skript. Kui skript lõpetab, Raspberry Pi teeb restardi.

Juhul kui kasutad BrickPi3, siis skriptini jõudmiseks kasuta käsku:

```
$ cd Dexter/BrickPi3/Install/
```

Skripti käivitamine aga käib samamoodi.

# 4 Demorobot

Järgnevas peatükis kirjeldatakse, kuidas lugeja leiab võimalikult lihtsalt üles näidisprogrammid, mis lihtsustab kasutajal programmeerimise alustamist kasutades BrickPi riistvara. On eeldatud, et lugeja on omandanud BrickPi ja Raspberry Pi trükkplaadid ning installeerinud Raspbian for Robots operatsioonisüsteemi, et oleks lugejale kättesaadav näidiskood, mis on avaldatud Dexter Industries poolt.

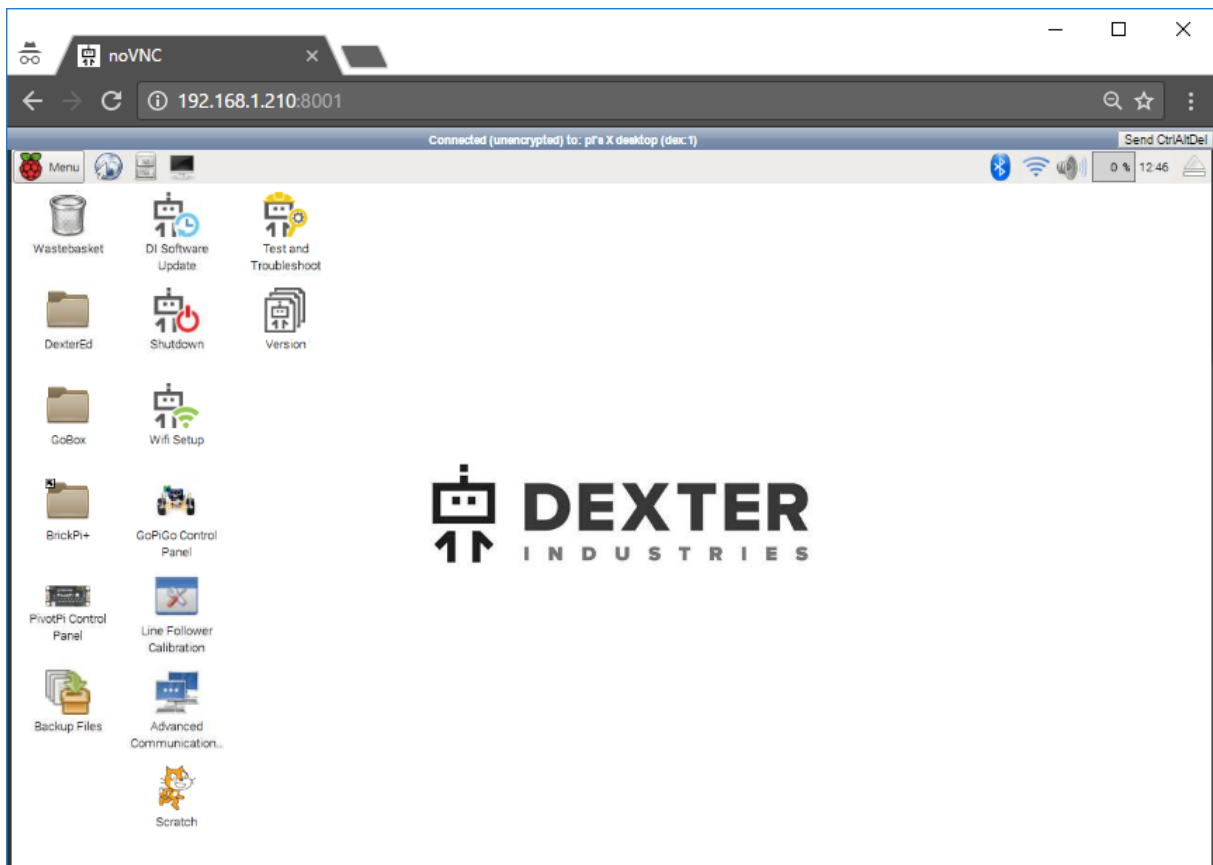
Peatüki teises pooles demonstreeritakse demorobotit, mille ülesandeks on otsida aaret. Demoroboti kokkupanekul on kasutatud järgmist riistvara:

- BrickPi+ trükkplaat
- Raspberry Pi 2 Model B v1.1 trükkplaat
- Sixaxis PlayStation 3 pult
- LEGO Mindstorms EV3 komplekt
- AbsoluteIMU Sensor for Mindstorms NXT / EV3
- WiFi tongel
- 8GB SD kaart
- Sinihamba tongel

Lisaks on kirjas robotis kasutatava programmi lähtekood, mille abil robot täidab oma ülesannet.

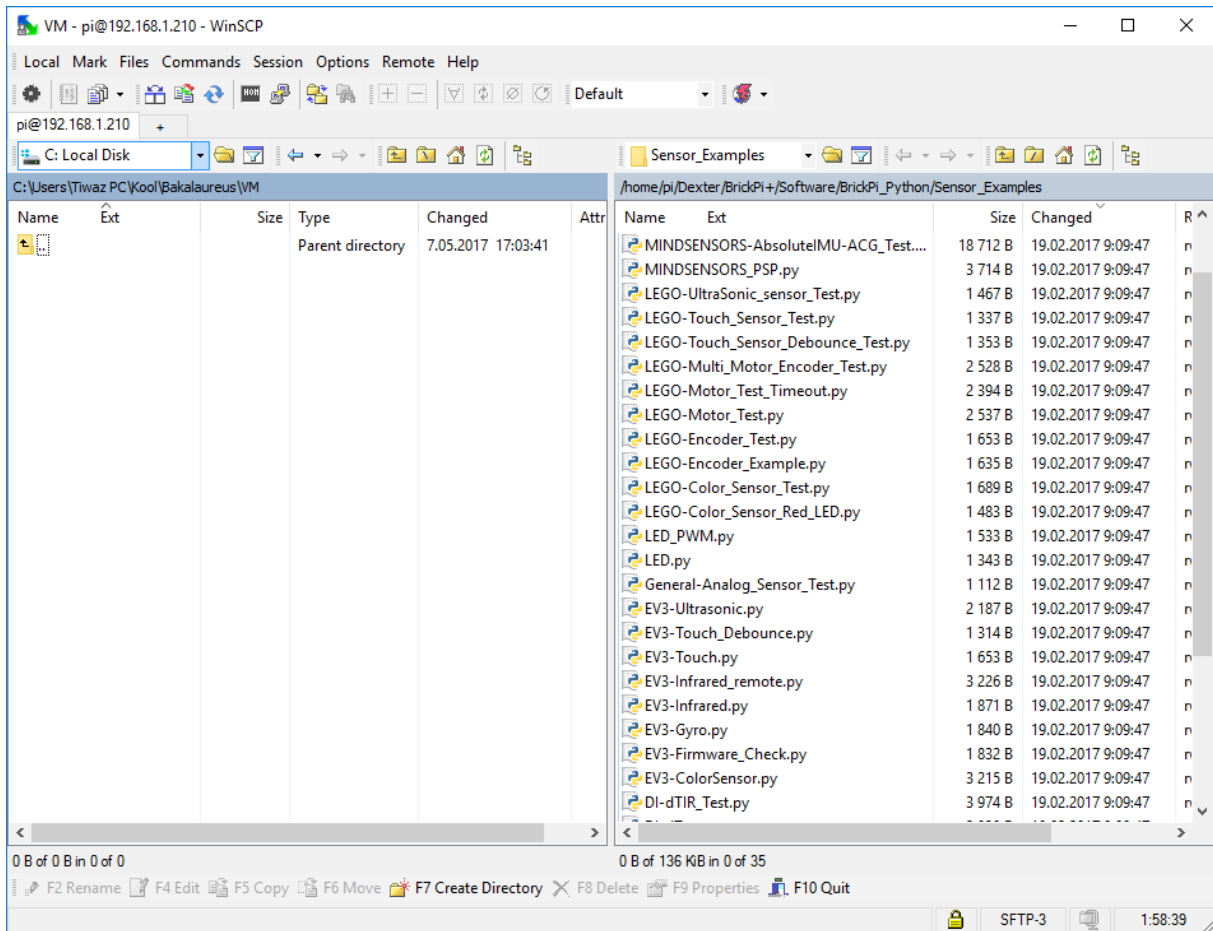
## 4.1 Näidised Raspbian for Robots operatsioonisüsteemis

Avades veebibrauseri abil aadress <http://dex.local>, avaneb vaade, mis on kujutatud joonisel 4.1. Selle kaudu on võimalik uurida, mis failid ja juhendid on antud masinasse installeeritud. Küll aga on soovituslik juhendite arhiivifailid ja lähtekoodi failid tõsta WinSCP programmi abil endale lokaalsesse masinasse. Raspbian for Robots toetab VNC (ingl k. *Virtual Network Computing*), mis on klient-server rakendus, mis lubab kasutades server rakendust jagada oma töölauda klientrakendustele. Antud kontekstis on serveriks Raspbian for Robots operatsioonisüsteemi kasutatav Raspberry Pi. VNC võimaldab võrguülest töölauda jagamist.

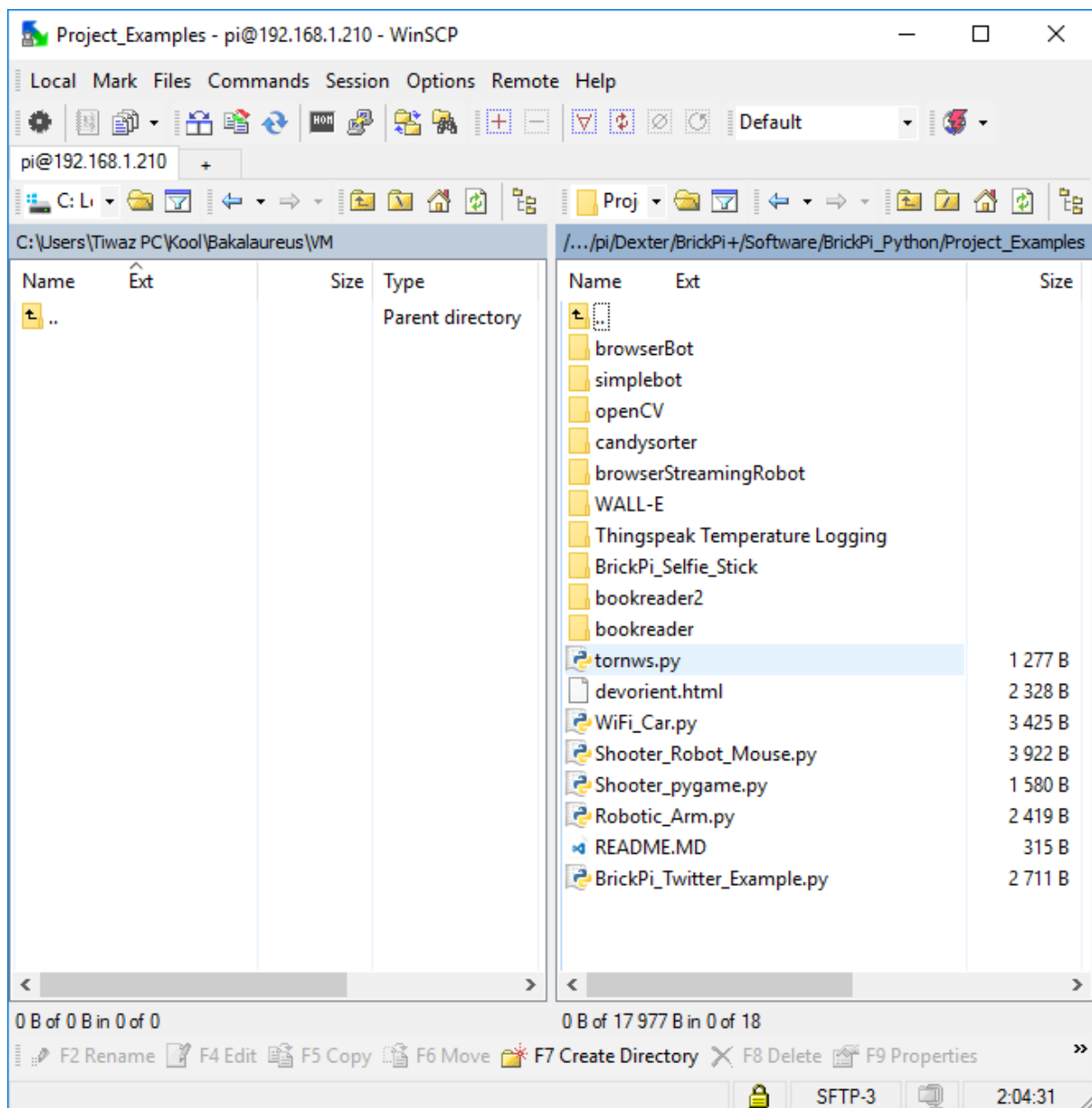


**Joonis 4.1** VNC kaudu avanev vaade Raspberry Pi's jooksvast operatsioonisüsteemi töölauast.

Raspian for Robots sisaldab näidisprojekte ja ka näidised, kuidas erinevaid andureid kasutada. Lahendatud on testid kasutamaks Absolute IMU andurit ja ka mitmeid LEGO enda andureid ja mootoreid. Lähtekoodi failid lahendustest on nimetatud andurite järgi (joonis 4.2).



**Joonis 4.2** Lähtekoodi failid andurite näidete kohta.



**Joonis 4.3** Kataloog, kus asuvad näidisprojektid

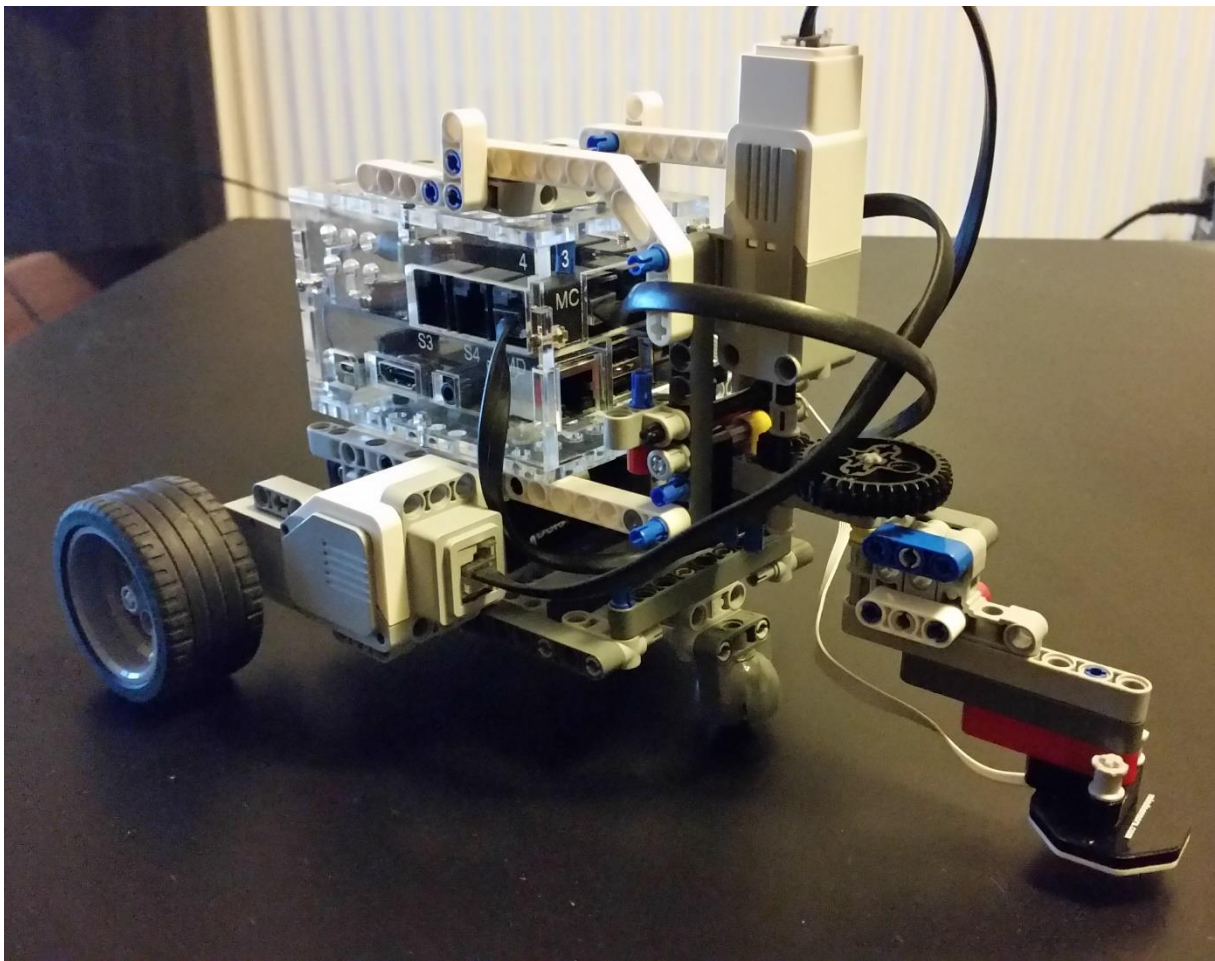
Mõned näidisprojektid sisaldavad ka detailset kirjeldust, kuidas mingit robotit ehitada. Kõikide robotite kohta aga sellist juhendit ei ole. Joonisel 4.3 on näidatud näidisprojektide asukohad.

## 4.2 Näidisrobot - Aardeotsijarobot

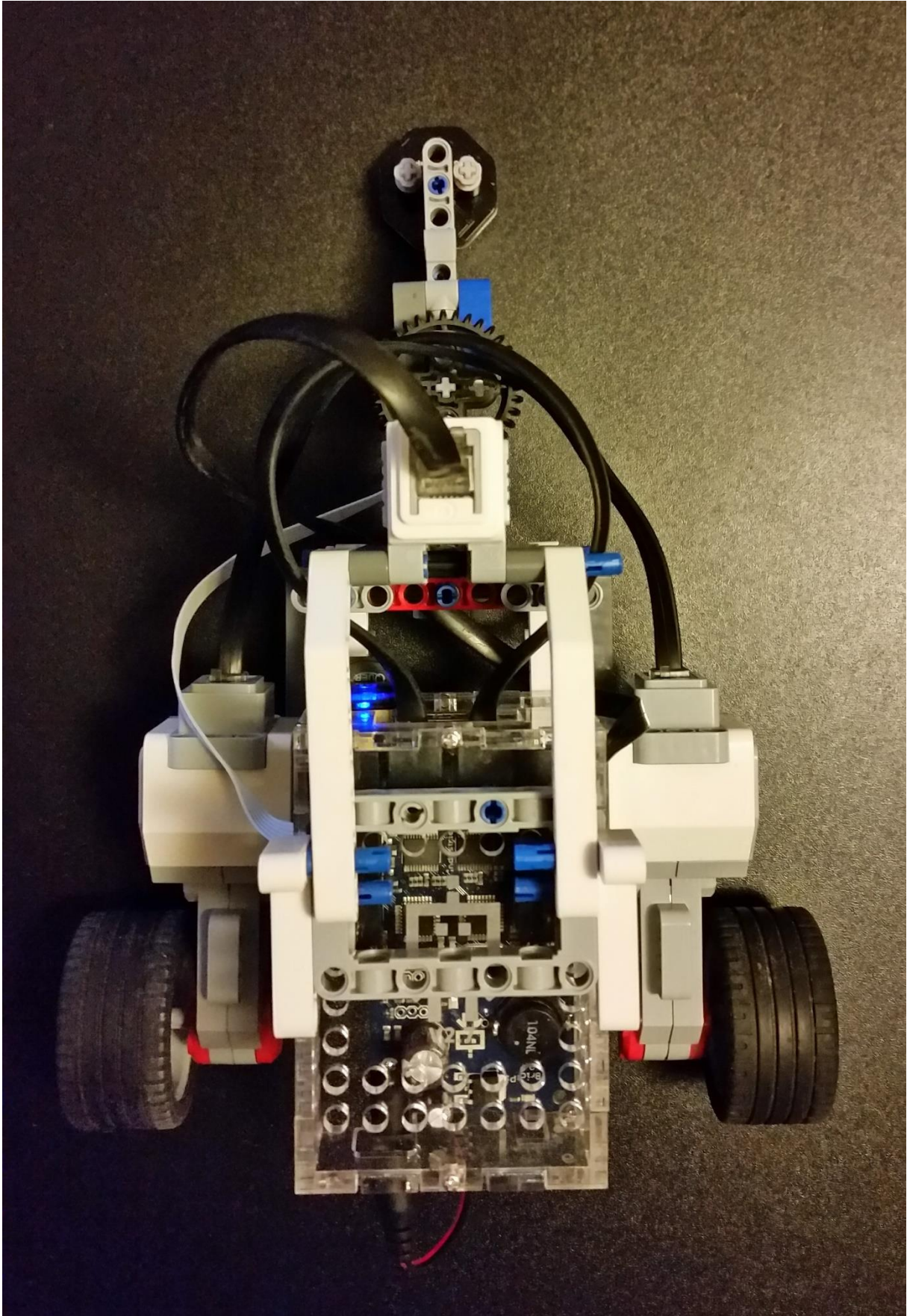
Näidisrobotiks (joonised 4.4-4.6) on maapinnaga kolme puutepunkti omav robot. Kaks puutepunktid on rattad ja kolmas on metallkuul, mis lihtsustab robotil pöörämist. Roboti lähtekood on programmeeritud kasutades Python'i programmeerimiskeelt. Robot kasutab ära Sixaxis PlayStation 3 DualShock pulti, mille abil on robotit võimalik juhtida, enne kui ta lülitakse aarde otsimise režiimi. PlayStationi puldist on kasutatavad järgmised nupud:

- Vasak analoog – liigutab robotit edasi-tagasi, pöörab robotit
- Parem analoog – liigutab andurit hoidvat varrast vasakule-paremale
- X – katkestab programmi töö ja väljub programmist
- Kolmnurk – lülitab roboti ümber aardeotsimise režiimi
- Ring – seadistab vaikeväärtused aardeotsimise režiimi jaoks

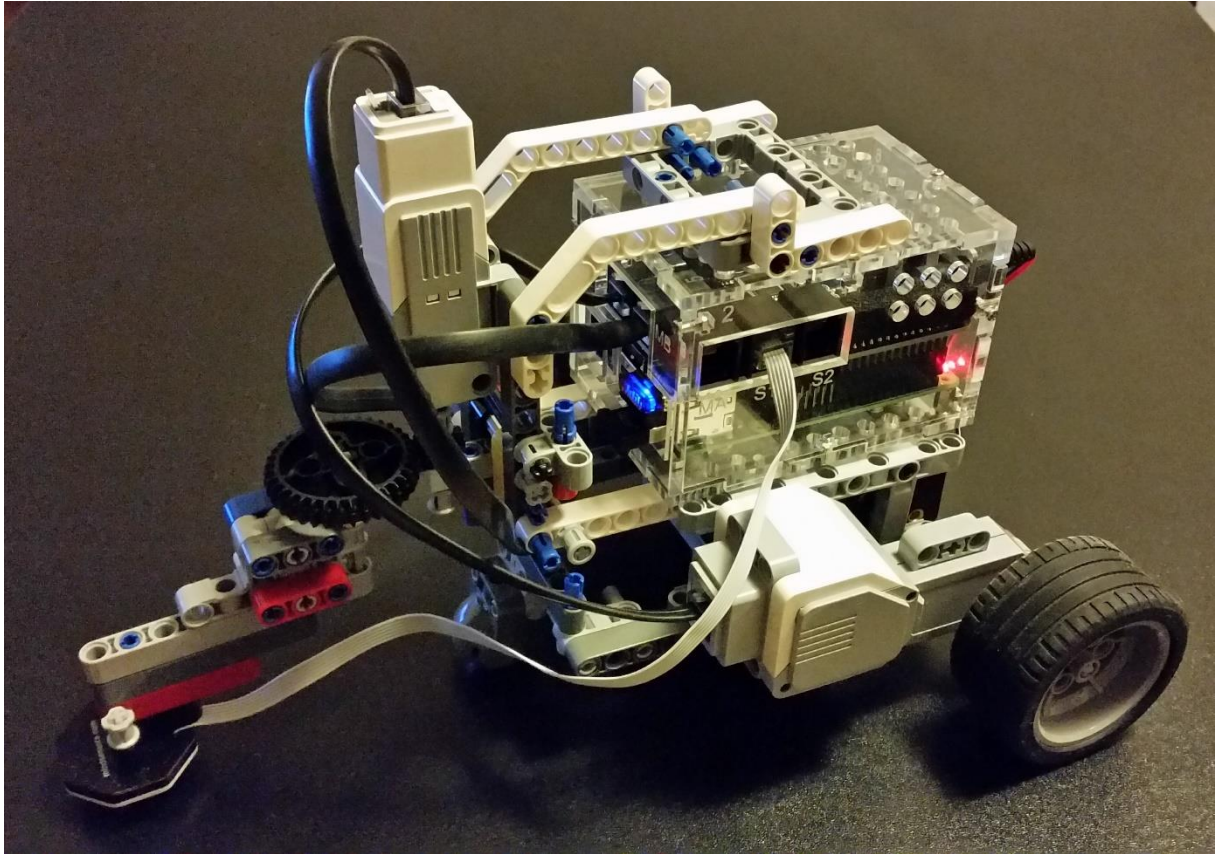
Roboti mõistes aardeks on suvaline metallist objekt, mida saab magnetiga mõjutada. Sellel põhjusel ei sobiks aardeks reaalsed rahalist väärtust omavad mündid, kuna need ei ole magnetiseeritavad.



**Joonis 4.4.** Aardeotsija robot paremvaatest



**Joonis 4.5.** Aardeotsimisrobot pealtvaatest.



**Joonis 4.6.** Aardeotsimisrobot vasakvaatest.

Robot kasutab aarde otsimiseks Absolute IMU-ACG(joonis 4.7) sensorit, mis sisaldab endas mitmeid andureid. Aardeotsimisel kasutatakse sensorite komplektist vaid magnetvälja andurit. Andur mõõdab magnetvälja kolmemõõtmeliselt, ehk kasutades X-, Y- ja Z-telge. Näidisrobotil asub andur roboti ees ja see liigub aeglaselt vasakule-paremale kasutades keskmist mootorit EV3 komplektist.



**Joonis 4.7** Absolute IMU-ACG sensor [19]



Näidisrobotis kasutatud programm hoiab meeles eelmist mõõtmistulemust ja võrdleb seda uuega. Juhul kui programm tuvastab ühel kolmest samal ajahetkel mõõdetavast X/Y/Z-telje mõõtmistulemusest sedavõrd suure erinevuse eelmisega, siis robot lõpetab aardeotsimise režiimi ning jääb seisma. Peale seda on jätkuvalt robotit võimalik juhtida kasutades PlayStation 3 pulti.

Näidisroboti mõistes on mõõtmistulemuste anomaaliaks sellised mõõtmistulemused, mis on suuremad, kui on ette antud programmi käivitamisel. Kui parameetrit ette ei anta, siis arvestatakse anomaaliaks mõõtmistulemust mis on suurem kui arvu 1000 absoluutväärtus ühikut. Kuna mõõtmistulemused on tugevalt sõltuvad keskkonnast, ei pruugi igas keskkonnas olla selline vaikeväärtus sobiv, seega on kasutajal võimalik ette anda oma väärtus.

Üks võimalik rakendus roboti aarde otsimise oskusele, on panna robot otsima üles põrandale asetatud metallist eset. Selleks on soovituslik tõsta roboti andur maapinnast 5mm kõrgemale, kui on otsitava eseme kõrgus, kui see on asetatud maapinnale. Kui robot saab mõõtetulemuse, mis on suurem kui ette määratud parameeter, siis jääb robot seisma ja lõpetab oma programmi. Kasutajal on võimalik ka programm ise lõpetada, kasutades puldil X nuppu, korrigeerida sisendparameetrit ning siis proovida uuesti. Testimiseks võib asetada roboti teekonnale ka esemeid, mis ei ole metallist. Selliseid objekte robot ootuspäraselt ei leia ja liigub edasi.

Aardeotsimis robotis kasutatakse järgmist riistvara:

- BrickPi+
- Raspberry Pi 2 Model B v1.1
- LEGO Mindstroms EV3 komplekt
  - Kaks suurt mootorit
  - Üks keskmine mootor
  - Klotsid
- Absolute IMU-ACG sensor – sisaldab 5 erinevat andurit, robot kasub vaid magnetväljaandurit
  - Kaldeandur
  - Kiirendusandur
  - Kompass
  - Magnetväljaandur

- Güroskoop
- 8 x AA patareid

## 4.2.2 Lähtekoodi kirjeldus

Programm kompileerub kasutades Python 2.7.9 versiooni. Python'i teekides, mis pole loodud Dexter Industries poolt, on kasutusel:

- `__future__`
- Threading

Lisaks kasutatakse Dexter Industries poolt kirjutatud teeke:

- BrickPi
- Ps3

BrickPi teek on ühilduv vaid BrickPi esimese ja teise generatsiooni trükkplaatidega. Kui kasutada kolmanda generatsiooni trükkplaati, siis selleks on Dexter Industries kirjutanud uue teegi. Teeki „ps3“ on vaja suhtluseks Sixaxis PlayStation 3 puldiga. Sinna tegin ühe väikese muudatuse, et programmi käivitamisel ei tekiks käitusviga.

---

```
1.     def __init__(self):
2.         # Make the stdout buffer as 0, because of bug in Pygame which keeps on printing
           debug statements
3.         # http://stackoverflow.com/questions/107705/python-output-buffering
4.         sys.stdout = os.fdopen(sys.stdout.fileno(), 'w', 0)
5.
6.         pygame.init()
7.         pygame.display.set_mode((1, 1))
8.         pygame.joystick.init()
9.         ps3.joystick = pygame.joystick.Joystick(0)
10.        ps3.joystick.init()
11.        ps3.joystick_count = pygame.joystick.get_count()
12.        ps3.numaxes = ps3.joystick.get_numaxes()
13.        ps3.numbuttons = ps3.joystick.get_numbuttons()
```

Programm on kirjutatud mitmelõimeliseks. Iga lõim vastutab mingi roboti funktsionaalsuse eest. Kui programm alustab, käivitatakse paralleelselt jooksma kaks lõime.

---

```
1.     def tekitaLoimed():
2.         thread_2 = threading.Thread(target=ps3RemoteTask, args=[])
3.         thread_3 = threading.Thread(target=valuesUpdater, args=[])
4.         thread_2.start()
5.         thread_3.start()
```

Esimene lõim kuulab sisendeid, mis tulevad Sixaxis PlayStation 3 puldist. Olenevalt selle sisendist, kas robotit liigutatakse, või minnakse aardeotsijarežiimi, milleks käivitatakse lisaks kolmas ja neljas lõim.

```
217.     Hoolitseb PS3 puldi sisendi t66tluse eest ja k2ivitab vajadusel
218. #aardeotsimise režiimi
219. def ps3RemoteTask():
220.     global breakAllThreads
221.     global aardeotsimisReziim
222.     global lopeta
223.     p = ps3()
224.
225.     while True:
226.         p.update()
227.         if p.cross:
228.             print("X - Katkesta programm")
229.             breakAllThreads = True
230.             lopeta = True
231.             break
232.         elif p.circle:
233.             if breakAllThreads:
234.                 print("O - Taaseadistame")
235.                 breakAllThreads = False
236.                 aardeotsimisReziim = False
237.         elif p.triangle:
238.             if not aardeotsimisReziim:
239.                 print("K2ivita aardejaht!")
240.                 aardeotsimisReziim = True
241.                 thread_0 = threading.Thread(target=motorsTask, args=[])
242.                 thread_1 = threading.Thread(target=getMagnetic, args=[])
243.
244.                 #alustame automaatse edasi liikumise ja anduri liigutamise
245.                 thread_0.start()
246.                 #alustame andmete m66tmist
247.                 thread_1.start()
248.             else:
249.                 BrickPi.MotorSpeed[PORT_B] = 0
250.                 BrickPi.MotorSpeed[PORT_C] = 0
251.                 x=(p.a_joystick_left_x+1)*90
252.                 y=(p.a_joystick_left_y+1)*90
253.                 speedLeft=0
254.                 speedRight=0
255.                 if x != 90:
256.                     if x > 90:
257.                         speedLeft += int(x* (-1))
258.                     else:
259.                         speedRight += int((180 - x)* (-1))
260.                 if y != 90:
261.                     if y > 90:
262.                         speedRight += int(y)
263.                         speedLeft += int(y)
264.                     else:
265.                         speedLeft+= int((180 - y)*(-1))
266.                         speedRight += int((180 - y) * (-1))
267.
268.
269.
270.                 BrickPi.MotorSpeed[PORT_B] = checkSpeed(speedRight)
271.                 BrickPi.MotorSpeed[PORT_C] = checkSpeed(speedLeft)
272.
273.                 speedS = 0
274.                 secondM = (p.a_joystick_right_x+1)*90
275.
```

```

276.         turnable = 40
277.         if secondM != 90:
278.             if secondM >90:
279.                 speedS = turnable
280.             else:
281.                 speedS = (-1) * turnable
282.
283.         BrickPi.MotorSpeed[PORT_D] = speedS
284.         time.sleep(.01)
285.

```

Teine lõim pidevalt kutsub välja BrickPiUpdateValues() meetodit BrickPi teegist. See on vajalik selleks, programm saaks kätte andurite mõõtetulemused läbi BrickPi ja et mootorid muudaksid oma seisu vastavalt programmi sisenditele.

```

198.         Hoolitseb, et kogu aeg oleksid BrickPi'st oleks saadaval v2rsked anduri andme
199.         d
200. #ja uuendab mootorite seisu BrickPi's
201. def valuesUpdater():
202.     global lopeta
203.     while True:
204.         BrickPiUpdateValues()
205.         if lopeta:
206.             break

```

Kolmas lõim loeb magnetvälja mõõtetulemusi andurist ja lõpetab töö siis, kui leiab anomaalia magnetväljas, või läbi juhtpuldi katkestatakse aardeotsija režiim.

```

1. def getMagnetic(motors_task):
2.     global breakAllThreads
3.     print("Starting sensors")
4.     prev_x = 0
5.     prev_y = 0
6.     prev_z = 0
7.
8.     while True:
9.         if breakAllThreads:
10.            break
11.         mfx = 0 #
12.         mfy = 0
13.         mfz = 0
14.         try:
15.             if (BrickPi.Sensor[I2C_PORT] & (0x01 << I2C_DEVICE_INDEX)):
16.
17.                 # NOTE: The magnetic field values returned by the AbsoluteIMU-ACG
18.                 # are 16bit signed integers. See the note on reading acceleration.
19.                 mfx = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][0]
20.                     | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][1] << 8)
21.                 if (mfx > 0x7FFF): # 32767
22.                     mfx -= 0xFFFF # 65535
23.                 mfy = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][2]
24.                     | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][3] << 8)
25.                 if (mfy > 0x7FFF):
26.                     mfy -= 0xFFFF
27.                 mfz = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][4]
28.                     | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][5] << 8)
29.                 if (mfz > 0x7FFF):

```

```

30.             mfz -= 0xFFFF
31.
32.
33.             if prev_x == 0:
34.                 prev_x = mfx
35.                 prev_y = mfy
36.                 prev_z = mfz
37.
38.             if mfx == 0 or mfy == 0 or mfz == 0:
39.                 prev_x = mfx
40.                 prev_y = mfy
41.                 prev_z = mfz
42.
43.             if test(prev_x, mfx):
44.                 print("found x", prev_x, mfx)
45.             if test(prev_y, mfy):
46.                 print("found y", prev_y, mfy)
47.             if test(prev_z, mfz):
48.                 print("found z", prev_z, mfz)
49.
50.             prev_x = mfx
51.             prev_y = mfy
52.             prev_z = mfz
53.         except TypeError:
54.             None

```

Neljas lõim hoolitseb roboti ja anduri liikumise eest robot liigutab kangi, millel paikneb andur. Robot liigub edasi anduri pikkuse jagu kasutades kahte suurt mootorit. Neljas lõim töötab vaid aardeotsijarežiimis.

```

168. Programm liigutab anduriga kangi vasakule ja paremale.
169.     #Peale igat edasi-tagasi liigutamist liigub edasi 2.2 sekundit.
170.     #iga kord kui mõni mootor on oma p66rdetsykli lõpetanud
171.     #seisatakse mootoris lyhikeseks ajahetkeks, et roboti liikumise inertis
172.     #ei mõjutaks andmete lugemist
173.     def motorsTask():
174.         global breakAllThreads
175.         while True:
176.             if breakAllThreads:
177.                 break
178.             kiirusEdasi = -50
179.             kiirusAndurPoore = 40
180.             BrickPi.MotorSpeed[PORT_B] = kiirusEdasi
181.             BrickPi.MotorSpeed[PORT_C] = kiirusEdasi
182.
183.             time.sleep(0.3)
184.
185.             BrickPi.MotorSpeed[PORT_B] = 0
186.             BrickPi.MotorSpeed[PORT_C] = 0
187.
188.             BrickPi.MotorSpeed[PORT_D] = kiirusAndurPoore
189.
190.             time.sleep(2.2)
191.             BrickPi.MotorSpeed[PORT_D] = 0
192.             time.sleep(0.3)
193.             BrickPi.MotorSpeed[PORT_D] = kiirusAndurPoore*(-1)
194.             time.sleep(2.2)
195.             BrickPi.MotorSpeed[PORT_D] = 0
196.             time.sleep(0.5)

```

Keerukaim osa koodist on parameetrite seadistamine selliselt, et BrickPi kaudu oleks võimalik saada õige anduri andmed. Selleks sai vaadatud Dexter Industries poolt koostatud testprogrammi näidisesest kopeeritud anduri seadistus. Lähtefailiks sellele oli MINDSENSORS-AbsoluteIMU-ACG\_Test.py. Jooniselt 4.2 on näha faili asukoht Raspian for Robots operatsioonisüsteemis. Järgnev kood baseerub Dexter Industries koodil.

```

29. #####Absolute IMU ACG seadistamine p2rineb Dexter Industries n2itefailist###
   #####
30. # Connection information. For this test program the AbsoluteIMU-ACG should be
31. # connected to port 1 of the BrickPi, and it should be the only device on that port.
32. I2C_PORT          = PORT_1 # I2C port the AbsoluteIMU-ACG is connected to.
33. I2C_SPEED         = 0      # Delay for as little time as possible. Usually about 100
   k baud
34. I2C_DEVICE_INDEX = 0      # AbsoluteIMU-ACG is device 0 on this I2C bus
35.
36. # Default I2C address of MindSensors AbsoluteIMU-ACG
37. AIMU_I2C_ADDR = 0x22
38.
39. # Command register address
40. AIMU_CMD_REG = 0x41
41.
42. # Command codes.
43. AIMU_CMD_RANGE_2G_250          = 0x31 # Accelerometer 2G and Gyro 250 degrees p
   er second range
44. AIMU_CMD_RANGE_4G_500          = 0x32 # Accelerometer 4G and Gyro 500 degrees p
   er second range
45. AIMU_CMD_RANGE_8G_2000         = 0x33 # Accelerometer 8G and Gyro 2000 degrees
   per second range
46. AIMU_CMD_RANGE_16G_2000        = 0x34 # Accelerometer 16G and Gyro 2000 degrees
   per second range
47. AIMU_CMD_BEGIN_COMPASS_CALIBRATION = 0x43 # 'C'
48. AIMU_CMD_END_COMPASS_CALIBRATION  = 0x63 # 'c'
49. AIMU_CMD_RESET_ALL_CALIBRATION    = 0x52 # 'R'
50. AIMU_CMD_BEGIN_GYRO_CALIBRATION   = 0x47 # 'G'
51. AIMU_CMD_END_GYRO_CALIBRATION     = 0x67 # 'g'
52.
53. AIMU_REG_VRSN = 0x00 # Software version number (ascii), 8 bytes. Should read somethi
   ng like: 'Vn.nn'
54. AIMU_REG_VID  = 0x08 # Vendor Id (ascii), 8 bytes. Should read: 'mndsnsrs'
55. AIMU_REG_DID  = 0x10 # Device Id (ascii), 8 bytes. Should read: 'AbsIMU'
56. AIMU_REG_GLVL = 0x19 # Undocumented, found in nxc code as: 'IMU_ReadGLevel'. One byt
   e.
57. AIMU_REG_TILT = 0x42 # Tilt data is 3 bytes
58. AIMU_REG_ACC  = 0x45 # Acceleration data is 6 bytes (3 LSB,MSB pairs for x, y and z)
59. AIMU_REG_CMPH = 0x4B # Compass heading data is 2 bytes (LSB,MSB)
60. AIMU_REG_MFR  = 0x4D # Raw magnetic field data is 6 bytes (3 LSB,MSB pairs for x, y
   and z)
61. AIMU_REG_GYRO = 0x53 # Gyro data is 6 bytes (3 LSB,MSB pairs for x, y and z)
62. AIMU_REG_FLTR = 0x5A # One byte filter value (0-7), default is 4
63.
64. # Some scale factors
65. ROTATION_SPEED_TO_MILLIDEG_PER_SECOND = 8.75
66.
67. # Setup the communication with BrickPi
68. BrickPiSetup()
69.
70. # Type of sensor and connection details
71. BrickPi.SensorType [I2C_PORT] = TYPE_SENSOR_I2C
72. BrickPi.SensorI2CSpeed [I2C_PORT] = I2C_SPEED

```

```

73. BrickPi.SensorI2CDevices [I2C_PORT] = 1
74.
75. BrickPi.SensorSettings [I2C_PORT][I2C_DEVICE_INDEX] = 0
76. BrickPi.SensorI2CAddr [I2C_PORT][I2C_DEVICE_INDEX] = AIMU_I2C_ADDR
77.
78. # Set the sensitivity of the accelerometer and gyro to 2G and 250 degrees/second
79. BrickPi.SensorI2CWrite [I2C_PORT][I2C_DEVICE_INDEX] = 2
80. BrickPi.SensorI2CRead [I2C_PORT][I2C_DEVICE_INDEX] = 0
81. BrickPi.SensorI2COut [I2C_PORT][I2C_DEVICE_INDEX][0] = AIMU_CMD_REG
82. BrickPi.SensorI2COut [I2C_PORT][I2C_DEVICE_INDEX][1] = AIMU_CMD_RANGE_2G_250
83. #####Absolute IMU ACG seadistamine p2rineb Dexter Industries n2itefailist###
    #####

```

Samuti pärineb magnetvälja mõõtetulemuste lugemine Dexter Industries poolt kirjutatud näitefailist.

```

140. #andmete lugemine p2rineb Dexter Industries poolt n2itefailist
141. if (BrickPi.Sensor[I2C_PORT] & (0x01 << I2C_DEVICE_INDEX)):
142.     mfx = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][0]
143.           | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][1] << 8)
144.     if (mfx > 0x7FFF):
145.         mfx -= 0xFFFF
146.     mfy = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][2]
147.           | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][3] << 8)
148.     if (mfy > 0x7FFF):
149.         mfy -= 0xFFFF
150.     mfz = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][4]
151.           | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][5] << 8)
152.     if (mfz > 0x7FFF):
153.         mfz -= 0xFFFF
154.

```

# Kokkuvõte

Töö käigus anti ülevaade BrickPi trükkplaadist. Toodi välja, kuidas see ettevõtmine alguse sai ja kuidas see Dexter Industries tootesari on muutunud aja möödudes - selle esimesest generatsioonist kuni kolmanda generatsioonini välja. Lisaks toodi välja BrickPi lähimad ja tähtsamad konkurendid – LEGO Mindstorms EV3 ja LEGO Mindstorms NXT. Võrdluse tulemusel on autori isiklik veendumus, et võttes arvesse hinda, seadistamise keerukust ja ka programmeerimise keerukust, ei ole BrickPi hea lahendus, mida kasutada asendamaks LEGO klotsidest ehitatud robotit BrickPi trükkplaadi ja Raspberry Pi komplektiga. Autori isikliku arvamuse järgi, omab LEGO Mindstorms EV3 enamaid eeliseid, kuna see võimaldab programmeerida kasutades ev3dev operatsioonisüsteemi Pythonis ja soovi korral on võimalik alati tagasi liikuda graafilist programmeerimist toetava operatsioonisüsteemi peale. BrickPi küll toetab ka graafilist programmeerimist, eriti silmas pidades populaarse programmeerimiskeele Scratch tuge. Küll aga BrickPi toetab ainult Scratch 1.4 versiooni, mis on vananenud. Scratch 2.0 ei ole võimalik installeerida Raspberry Pi peale Adobe Air poolt nõutavate tehniliste piirangute tõttu. Samuti on teada antud, et käib arendus Scratch 3 kallal.

Töö raames valmisid eestikeelsed juhendid, mille abil on võimalik üles hõlpsalt üles seada keskkond, kus õppida programmeerimist kasutades BrickPi robotikaplatvormi. Lisaks on võimalik võtta malli näidisrobotist, mis valmis lõputöö raames või siis uurida iseseisvalt erinevaid näidisroboteid, mis on valminud Dexter Industries poolt. Need juhendid ja koodilõigud on kättesaadavad kujutisfailist, millel on Raspbian for Robots operatsioonisüsteem.

Töö autor osales kahe aasta vältel erinevatel robotika üritustel, et laiendada oma teadmisi seoses Eesti koolirobootikaga. Aastate 2016 ja 2017 kevadel osales ta FIRST LEGO League poolfinaalides ja finaalides korraldajate meeskonnas vabatahtlikuna. Lisaks aitas vabatahtlikuna 2016 Robomiku linnalahingu üritust läbi viia. Sama aasta kevadel oli ka abis korraldamas Junior FIRST LEGO League. Nimetatud üritustel vabatahtlikuna osalemine võimaldas autoril otse suhelda inimestega, kes tegelevad LEGO Mindstorms platvormide robotite ehitamise ja programmeerimisega. See omakorda võimaldas autoril paremini aru saada, kuidas on võimalik kasutada LEGO Mindstorms platvorme, mis on nende piirangud ja kuidas oleks võimalik võrrelda BrickPi platvormi LEGO Mindstorms platvormidega. See omakorda võimaldas välja kujundada isiklikku arvamuse LEGO Mindstorms aju asendamise otstarbekusest BrickPi platvormiga.



# Kasutatud kirjandus

- [1] Sander Orava bakalaureuse töö „MATRIX, TETRIX ja VEX robotikaplatvormid“  
<https://www.tuit.ut.ee/sites/default/files/tuit/atprog-courses-bakalaureuset55-loti.05.029-sander-orav-text-20160601.pdf> (11.05.2017)
- [2] Aine MTAT.03.100 Programmeerimise koduleht  
<https://courses.cs.ut.ee/2016/programmeerimine/fall> (11.05.2017)
- [3] Kursuse „Programmeerimisest maalähedaselt“ koduleht  
<https://www.ut.ee/et/mooc/programmeerimisest-maalahedaselt> (11.05.2017)
- [4] BrickPi projekti veebileht Kickstarteris: <https://www.kickstarter.com/projects/john-cole/brickpi-lego-bricks-with-a-raspberry-pi-brain> (27.11.2016)
- [5] BrickPi ostuinfo <https://www.dexterindustries.com/shop/brickpi-advanced-for-raspberry-pi/> (27.11.2016)
- [6] JetBrains poolt loodud integreeritud programmeerimiskeskond PyCharm  
<https://www.jetbrains.com/pycharm/> (11.05.2017)
- [7] Arduberry koduleht Arduberry kohta saab lugeda  
<https://www.dexterindustries.com/arduberry/> (11.05.2017)
- [8] GoPiGo koduleht <https://www.dexterindustries.com/gopigo/> (11.05.2017)
- [9] GrovePi koduleht <https://www.dexterindustries.com/product-category/robots/grovepi/>  
(11.05.2017)
- [10] Scratch 3.0 kirjeldus [https://wiki.scratch.mit.edu/wiki/Scratch\\_3.0](https://wiki.scratch.mit.edu/wiki/Scratch_3.0) (11.05.2017)
- [11] Linux Debian <https://www.debian.org/> (11.05.2017)
- [12] Ev3dev veebileht kujutisfaili allalaadimiseks <http://www.ev3dev.org/downloads/>  
(11.05.2017)
- [13] Google Drive link Raspbian for robots kujutisfailile  
<https://drive.google.com/file/d/0B0WChwP4CnLBOFNNQIFObFcxWUU/view> (11.05.2017)
- [14] Sourceforge portaali link Raspbian for robots kujutisfailile  
<https://sourceforge.net/projects/dexterindustriesraspbianflavor/> (11.05.2017)

- [15] Henry Maalinn bakkalaureusetöö „LEGO MINDSTORMS EV3-e programmeerimine Pythonis“ (11.05.2017)
- [16] BrickPi3 <https://www.dexterindustries.com/new-brickpi3-lego-mindstorms/> (11.05.2017)
- [17] LEGO Mindstorms NXT [https://en.wikipedia.org/wiki/Lego\\_Mindstorms\\_NXT](https://en.wikipedia.org/wiki/Lego_Mindstorms_NXT) (11.05.2017)
- [18] LEGO Mindstorms EV3 [https://en.wikipedia.org/wiki/Lego\\_Mindstorms\\_EV3](https://en.wikipedia.org/wiki/Lego_Mindstorms_EV3) (11.05.2017)
- [19] Absolute IMU ACG sensor <https://www.generationrobots.com/en/401339-absolute-imu-acg-sensor-gyro-accelero-compass-mindsensors.html> (11.05.2017)
- [20] LEGO Mindstorms [https://en.wikipedia.org/wiki/Lego\\_Mindstorms](https://en.wikipedia.org/wiki/Lego_Mindstorms) (11.05.2017)
- [21] LEGO Mindstorms EV3 aju <https://shop.lego.com/en-US/EV3-Intelligent-Brick-45500> (11.05.2017)
- [22] Multiplekser  
[http://mindsensors.ddns.net/index.php?module=pagemaster&PAGE\\_user\\_op=view\\_page&PAGE\\_id=135](http://mindsensors.ddns.net/index.php?module=pagemaster&PAGE_user_op=view_page&PAGE_id=135) (11.05.2017)

# Lisad

## I Lähtekood

```
1. from __future__ import print_function
2. from __future__ import division
3.
4. from BrickPi import * #BrickPi teek, mille abil võimalik juhtida robotit
5. from ps3 import * #ps3 teek, mille abil võimalik PlayStation pulti kasutada ro
  botis
6.
7. import threading
8.
9. BrickPiSetup() #Seadistatakse suhtlus Raspberry Pi ja BrickPi vahel
10.
11. #Mootorid tuleb ykshaaval sisse lylyitada, et neid kasutada saaks
12. #Pordid on n2htavad BrickPi ymbriskarbil
13. BrickPi.MotorEnable[PORT_B] = 1
14. BrickPi.MotorEnable[PORT_C] = 1
15. BrickPi.MotorEnable[PORT_D] = 1
16.
17. #globaalne boolean, et l6imed teaksid l6petada.
18. breakAllThreads = False
19.
20. #Vaikimisi anomaalite eristamiseks
21. threshold = 1000
22.
23. #defineerib aardeotsimisreziimi
24. aardeotsimisReziim = False
25.
26. #voimaldab taasalustada aarde otsimiset
27. lopeta = False
28.
29. #####Absolute IMU ACG seadistamine p2rineb Dexter Industries n2itefailist##
  #####
30. # Connection information. For this test program the AbsoluteIMU-ACG should be
31. # connected to port 1 of the BrickPi, and it should be the only device on that port.
32. I2C_PORT = PORT_1 # I2C port the AbsoluteIMU-ACG is connected to.
33. I2C_SPEED = 0 # Delay for as little time as possible. Usually about 100
  k baud
34. I2C_DEVICE_INDEX = 0 # AbsoluteIMU-ACG is device 0 on this I2C bus
35.
36. # Default I2C address of MindSensors AbsoluteIMU-ACG
37. AIMU_I2C_ADDR = 0x22
38.
39. # Command register address
40. AIMU_CMD_REG = 0x41
41.
42. # Command codes.
43. AIMU_CMD_RANGE_2G_250 = 0x31 # Accelerometer 2G and Gyro 250 degrees p
  er second range
44. AIMU_CMD_RANGE_4G_500 = 0x32 # Accelerometer 4G and Gyro 500 degrees p
  er second range
45. AIMU_CMD_RANGE_8G_2000 = 0x33 # Accelerometer 8G and Gyro 2000 degrees
  per second range
46. AIMU_CMD_RANGE_16G_2000 = 0x34 # Accelerometer 16G and Gyro 2000 degrees
  per second range
47. AIMU_CMD_BEGIN_COMPASS_CALIBRATION = 0x43 # 'C'
48. AIMU_CMD_END_COMPASS_CALIBRATION = 0x63 # 'c'
49. AIMU_CMD_RESET_ALL_CALIBRATION = 0x52 # 'R'
50. AIMU_CMD_BEGIN_GYRO_CALIBRATION = 0x47 # 'G'
51. AIMU_CMD_END_GYRO_CALIBRATION = 0x67 # 'g'
```

```

52.
53. AIMU_REG_VRSN = 0x00 # Software version number (ascii), 8 bytes. Should read something like: 'Vn.nn'
54. AIMU_REG_VID = 0x08 # Vendor Id (ascii), 8 bytes. Should read: 'mndsnsrs'
55. AIMU_REG_DID = 0x10 # Device Id (ascii), 8 bytes. Should read: 'AbsIMU'
56. AIMU_REG_GLVL = 0x19 # Undocumented, found in nxc code as: 'IMU_ReadGLevel'. One byte.
57. AIMU_REG_TILT = 0x42 # Tilt data is 3 bytes
58. AIMU_REG_ACC = 0x45 # Acceleration data is 6 bytes (3 LSB,MSB pairs for x, y and z)

59. AIMU_REG_CMPH = 0x4B # Compass heading data is 2 bytes (LSB,MSB)
60. AIMU_REG_MFR = 0x4D # Raw magnetic field data is 6 bytes (3 LSB,MSB pairs for x, y and z)
61. AIMU_REG_GYRO = 0x53 # Gyro data is 6 bytes (3 LSB,MSB pairs for x, y and z)
62. AIMU_REG_FLTR = 0x5A # One byte filter value (0-7), default is 4
63.
64. # Some scale factors
65. ROTATION_SPEED_TO_MILLIDEG_PER_SECOND = 8.75
66.
67. # Setup the communication with BrickPi
68. BrickPiSetup()
69.
70. # Type of sensor and connection details
71. BrickPi.SensorType [I2C_PORT] = TYPE_SENSOR_I2C
72. BrickPi.SensorI2CSpeed [I2C_PORT] = I2C_SPEED
73. BrickPi.SensorI2CDevices [I2C_PORT] = 1
74.
75. BrickPi.SensorSettings [I2C_PORT][I2C_DEVICE_INDEX] = 0
76. BrickPi.SensorI2CAddr [I2C_PORT][I2C_DEVICE_INDEX] = AIMU_I2C_ADDR
77.
78. # Set the sensitivity of the accelerometer and gyro to 2G and 250 degrees/second
79. BrickPi.SensorI2CWrite [I2C_PORT][I2C_DEVICE_INDEX] = 2
80. BrickPi.SensorI2CRead [I2C_PORT][I2C_DEVICE_INDEX] = 0
81. BrickPi.SensorI2COut [I2C_PORT][I2C_DEVICE_INDEX][0] = AIMU_CMD_REG
82. BrickPi.SensorI2COut [I2C_PORT][I2C_DEVICE_INDEX][1] = AIMU_CMD_RANGE_2G_250
83. #####Absolute IMU ACG seadistamine p2rineb Dexter Industries n2itefailist###
#####
84.
85.
86. if(BrickPiSetupSensors()):
87.     print ("BrickPiSetupSensors failed for setting sensitivity.")
88.     sys.exit(0)
89.
90. if(BrickPiUpdateValues()):
91.     print ("BrickPiUpdateValues failed for setting sensitivity.")
92.     sys.exit(0)
93.
94. # The sensor user manual says that we have to wait at least 50 milliseconds
95. # for the sensor to reconfigure itself after changing the sensitivity.
96. time.sleep(0.06) # 60ms
97.
98. # Read the magnetic field information from the sensor, ten times.
99. BrickPi.SensorSettings [I2C_PORT][I2C_DEVICE_INDEX] = BIT_I2C_SAME
100. BrickPi.SensorI2CWrite [I2C_PORT][I2C_DEVICE_INDEX] = 1
101. BrickPi.SensorI2CRead [I2C_PORT][I2C_DEVICE_INDEX] = 6
102. BrickPi.SensorI2COut [I2C_PORT][I2C_DEVICE_INDEX][0] = AIMU_REG_MFR
103. if(BrickPiSetupSensors()):
104.     print ("BrickPiSetupSensors - seadistamine eba6nnestus")
105.     sys.exit(0)
106.
107. if(BrickPiUpdateValues()):
108.     print ("BrickPiUpdateValues - Ei saa andmeid sensorist k2tte")
109.     sys.exit(0)
110.
111. # Sensor peab ootama v2hemalt 50ms peale seadistusparameetrite muudatust.
112. time.sleep(0.06) # 60ms

```

```

113.
114.#M22rame sensoriks magnetv21ja anduri
115.#p2rineb Dexter Industries poolt n2itefailist
116.BrickPi.SensorSettings [I2C_PORT][I2C_DEVICE_INDEX] = BIT_I2C_SAME
117.BrickPi.SensorI2CWrite [I2C_PORT][I2C_DEVICE_INDEX] = 1
118.BrickPi.SensorI2CRead [I2C_PORT][I2C_DEVICE_INDEX] = 6
119.BrickPi.SensorI2COut [I2C_PORT][I2C_DEVICE_INDEX][0] = AIMU_REG_MFR
120.
121.if( BrickPiSetupSensors() ):
122. print ("BrickPiUpdateValues - magnetv21ja andmete lugemine eba6nnestus")
123. sys.exit( 0 )
124.
125.
126.def kontrolliLeidu(f):
127. global threshold
128. return f >threshold or f < threshold*(-1)
129.
130.def getMagnetic():
131. global breakAllThreads
132. print("K2ivitan sensorid")
133. while True:
134. if breakAllThreads:
135. break
136. mfx = 0
137. mfy = 0
138. mfz = 0
139. try:
140. #andmete lugemine p2rineb Dexter Industries poolt n2itefailist
141. if (BrickPi.Sensor[I2C_PORT] & (0x01 << I2C_DEVICE_INDEX)):
142. mfx = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][0]
143. | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][1] << 8)
144. if (mfx > 0x7FFF):
145. mfx -= 0xFFFF
146. mfy = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][2]
147. | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][3] << 8)
148. if (mfy > 0x7FFF):
149. mfy -= 0xFFFF
150. mfz = (BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][4]
151. | BrickPi.SensorI2CIn[I2C_PORT][I2C_DEVICE_INDEX][5] << 8)
152. if (mfz > 0x7FFF):
153. mfz -= 0xFFFF
154.
155.
156. if kontrolliLeidu(mfx):
157. print("Leidsin metallise eseme! Jee!")
158. breakAllThreads = True
159. if kontrolliLeidu(mfy):
160. print("Leidsin metallise eseme! Jee!")
161. breakAllThreads = True
162. if kontrolliLeidu(mfz):
163. print("Leidsin metallise eseme! Jee!")
164. breakAllThreads = True
165. except TypeError:
166. None
167.
168.#Programm liigutab anduriga kangi vasakule ja paremale.
169.#Peale igat edasi-tagasi liigutamist liigub edasi 2.2 sekundit.
170.#iga kord kui m6ni mootor on oma p66rdetsykli l6petanud
171.#seisatakse mootoris lyhikeseks ajahetkeks, et roboti liikumise inertis
172.#ei m6jutaks andmete lugemist
173.def motorsTask():
174. global breakAllThreads
175. while True:
176. if breakAllThreads:
177. break
178. kiirusEdasi = -50

```

```

179.     kiirusAndurPoore = 40
180.     BrickPi.MotorSpeed[PORT_B] = kiirusEdasi
181.     BrickPi.MotorSpeed[PORT_C] = kiirusEdasi
182.
183.     time.sleep(0.3)
184.
185.     BrickPi.MotorSpeed[PORT_B] = 0
186.     BrickPi.MotorSpeed[PORT_C] = 0
187.
188.     BrickPi.MotorSpeed[PORT_D] = kiirusAndurPoore
189.
190.     time.sleep(2.2)
191.     BrickPi.MotorSpeed[PORT_D] = 0
192.     time.sleep(0.3)
193.     BrickPi.MotorSpeed[PORT_D] = kiirusAndurPoore*(-1)
194.     time.sleep(2.2)
195.     BrickPi.MotorSpeed[PORT_D] = 0
196.     time.sleep(0.5)
197.
198. #Hoolitseb, et kogu aeg oleksid BrickPi'st oleks saadaval v2rsked anduri andmed
199. #ja uuendab mootorite seisu BrickPi's
200. def valuesUpdater():
201.     global lopeta
202.     while True:
203.         BrickPiUpdateValues()
204.         if lopeta:
205.             break
206.
207. #kontrollitakse, et PlayStationi puldist tulev soovitatav kiirus
208. #oleks vahemikus -255 <= s <= 255.
209. #Teised v22rtused ei ole lubatud, sest mootor ei toeta neid.
210. def checkSpeed(s):
211.     if s >255:
212.         return 255
213.     elif s < -255:
214.         return -255
215.     return s
216.
217. #Hoolitseb PS3 puldi sisendi t66tluse eest ja k2ivitab vajadusel
218. #aardeotsimise reziimi
219. def ps3RemoteTask():
220.     global breakAllThreads
221.     global aardeotsimisReziim
222.     global lopeta
223.     p = ps3()
224.
225.     while True:
226.         p.update()
227.         if p.cross:
228.             print("X - Katkesta programm")
229.             breakAllThreads = True
230.             lopeta = True
231.             break
232.         elif p.circle:
233.             if breakAllThreads:
234.                 print("O - Taasseadistame")
235.                 breakAllThreads = False
236.                 aardeotsimisReziim = False
237.         elif p.triangle:
238.             if not aardeotsimisReziim:
239.                 print("K2ivita aardejaht!")
240.                 aardeotsimisReziim = True
241.                 thread_0 = threading.Thread(target=motorsTask, args=[])
242.                 thread_1 = threading.Thread(target=getMagnetic, args=[])
243.
244.     #alustame automaatse edasi liikumise ja anduri liigutamise

```

```

245.         thread_0.start()
246.         #alustame andmete m66tmist
247.         thread_1.start()
248.     else:
249.         BrickPi.MotorSpeed[PORT_B] = 0
250.         BrickPi.MotorSpeed[PORT_C] = 0
251.         x=(p.a_joystick_left_x+1)*90
252.         y=(p.a_joystick_left_y+1)*90
253.         speedLeft=0
254.         speedRight=0
255.         if x != 90:
256.             if x > 90:
257.                 speedLeft += int(x* (-1))
258.             else:
259.                 speedRight += int((180 - x)* (-1))
260.         if y != 90:
261.             if y > 90:
262.                 speedRight += int(y)
263.                 speedLeft += int(y)
264.             else:
265.                 speedLeft+= int((180 - y)*(-1))
266.                 speedRight += int((180 - y) * (-1))
267.
268.
269.
270.         BrickPi.MotorSpeed[PORT_B] = checkSpeed(speedRight)
271.         BrickPi.MotorSpeed[PORT_C] = checkSpeed(speedLeft)
272.
273.         speedS = 0
274.         secondM = (p.a_joystick_right_x+1)*90
275.
276.         turnable = 40
277.         if secondM != 90:
278.             if secondM >90:
279.                 speedS = turnable
280.             else:
281.                 speedS = (-1) * turnable
282.
283.         BrickPi.MotorSpeed[PORT_D] = speedS
284.         time.sleep(.01)
285.
286. def tekitaLoimed():
287.     thread_2 = threading.Thread(target=ps3RemoteTask, args=[])
288.     thread_3 = threading.Thread(target=valuesUpdater, args=[])
289.     thread_2.start()
290.     thread_3.start()
291.
292.
293. if __name__ == '__main__':
294.     if (len(sys.argv) > 1):
295.         threshold = int(sys.argv[1])
296.         print ("Vaimikisi m66teerisuse parameeter on=" + str(threshold))
297.         tekitaLoimed()

```

## II Litsents

### Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, **Argo Mändla**,

*(autori nimi)*

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose  
**BrickPi võrdlus teiste robotikaplatvormidega**,  
*(lõputöö pealkiri)*

mille juhendajateks on Anne Villems, Taavi Duvin ja Alo Peets,

*(juhendaja nimi)*

- 1.1.reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
- 1.2.üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace'i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **11.05.2017**