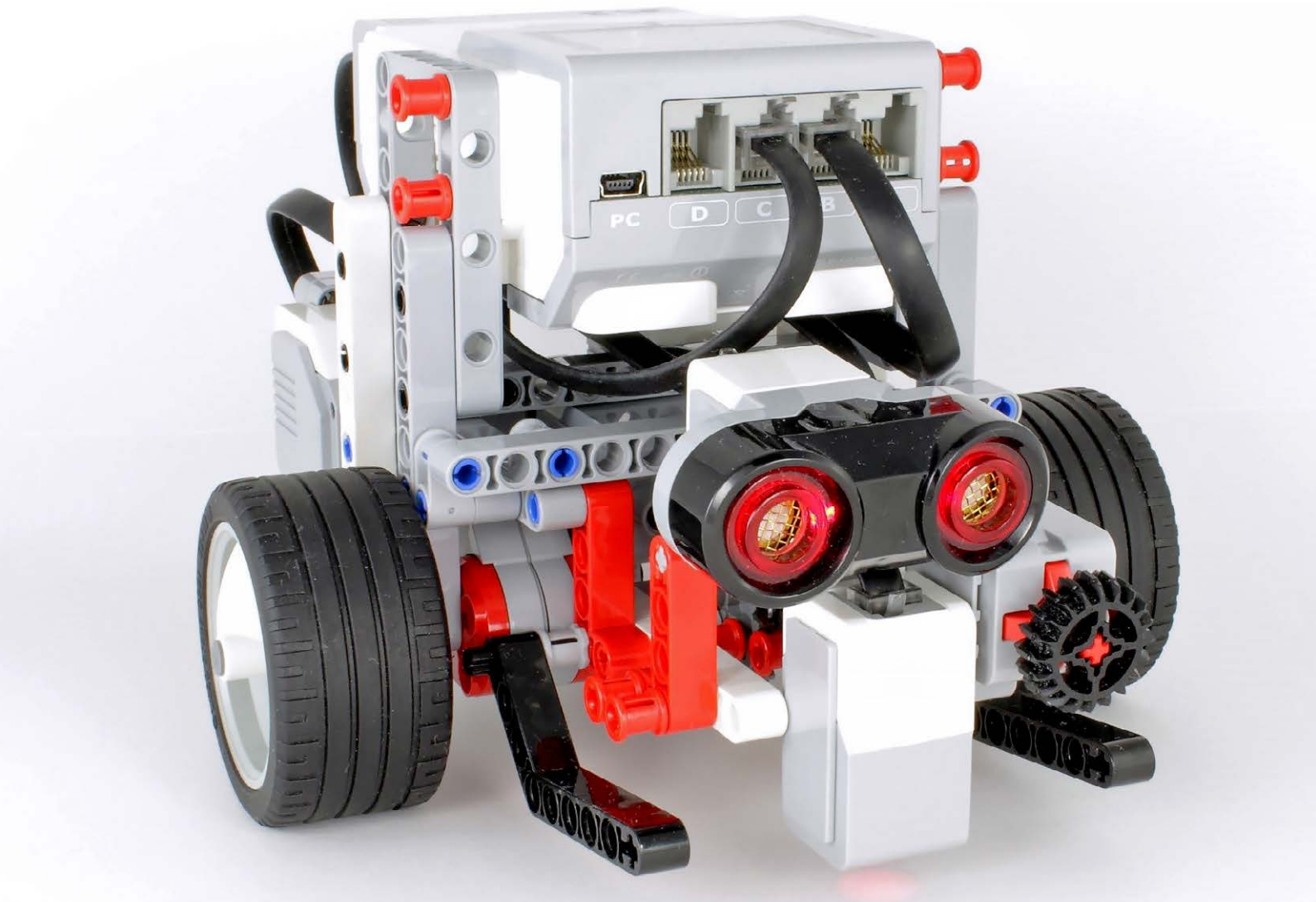
Innokas!

Asimov

Ensimmäiset robottini



Pekka Pihola

ASIMOV Ensimmäiset robottini

ISBN 978-951-51-1931-5

Kirjoittaja	Pekka Pihola
Kuvat	Pekka Pihola, ellei toisin mainittu
Pedagoginen ohjaus ja konsultointi	Tiina Korhonen, Kati Sormunen ja Minna Kukkonen, Innokas-verkosto
Kustantaja	Innokas-verkosto
Jakelu	Innokas-verkosto
Päivitysversio	1
Julkaisupäivä	29.1.2016

Asimov

Ensimmäiset robottini

Sisällys

Esipuhe

Istuessani aamulla syömässä aamupalaa ojensin maidon tyttärelleni tämän pyynnöstä. Tilanne on kaikille tuttu, mutta mitä siinä oikeastaan tapahtui? Millainen kyseinen tilanne olisi robottien maailmassa?

Robottien maailmassa tilanne menisi näin

1. C3PO esittää puhekielisen pyynnön ”Saisinko maitoa?”
2. R2-D2 vastaanottaa pyynnön ääni-antureillaan, tulkitsee sen sisällön ja
3. keskeyttää muut askareensa.
4. Seuraavaksi R2-D2 etsii kohteen ja määrittelee sen tarkan sijainnin.
5. Nopean mutta tarkan laskelman perusteella R2-D2 tekee pienen käännöksen ja ojentaa tarttujan kohti kohdetta,
6. Tarttuminen kohteeseen pitää tehdä juuri oikealla voimalla, pakkausta murskaamatta, mutta riittävän lujasti.
7. Seuraavaksi R2-D2 nostaa maidon, kääntyy ja määrittelee samalla C3PO:n tarkan sijainnin.
8. Kohteen siirto päättyy sen varovaiseen laskemiseen C3PO:n ulottuville.
9. Nyt pitää irrottaa tarttuja ja loitontaa sitä varovasti kohteesta.
10. Lopuksi R2-D2 piippailee C3PO:lle ääniviestin ”Tässä, ole hyvä”, ennen paluuta an keskeytyneiden askareidensa pariin.

(toimintakuvaus kertoo vaihe vaiheelta suoritettavasta tehtävästä)

Ohjelmoinnin ja robotiikan opiskelussa on kyse tutkimisesta, oivaltamisesta ja hyviksi havaittujen ratkaisujen soveltamisesta. Jokainen meistä tulee opettelemaan eläessään noin kolmea puhuttua kieltä ja ohjelmoimaan erilaisia laitteita ja robotteja kymmenillä eri ohjelmointikielillä ja tavoilla.

Ohjelmoinnin opiskelua voidaan verrata vieraan kielen opiskeluun. Puhutut kielet sisältävät paljon käsitteitä ja kuvaavia ilmauksia. Laajan sanaston oppimisessa on oma työnsä, mutta sen avulla asiat voidaan ilmaista lyhyesti ja tehokkaasti. Sen sijaan koneiden ymmärtämissä kielissä, kuten tietokoneiden ja robottien ohjelmoinnissa käytetty sanasto on olematon. Kaikki yksinkertaiseltakin tuntuvat asiat

pitää kertoa niille hyvin tarkasti.

Vaikka ohjelmointikieliä on tuhansia, niiden peruskäsitteistö ja -sanasto ovat hyvin vakiintuneita. Eroja ohjelmointikielien välille on syntynyt lähinnä siksi, että on haluttu kehittää johonkin tiettyyn käyttötarkoitukseen yleisiä ohjelmointikieliä paremmin sopivia kieliä. Käytännössä näihin kieliin on yleensä lisätty joukko jotakin tiettyä käyttötarkoitusta tukevia komentoja ja ominaisuuksia. Joillakin kielillä on esimerkiksi muita helpompaa ohjelmoida nettisivuja ja toisilla taas vaikkapa pelejä.

Tämä kirja ohjaa sinut rakentamaan ensimmäiset robottisi LEGO:n EV3 -laitteilla skeä ohjelmoimaan rakentamasi robotit graafisella LEGO:n EV3 -kielellä. Graafinen EV3-ohjelmointikieli perustuu oikeaan ja laajasti teollisuuden laitteissa käytettävään ohjelmointikieleen (LabWiew). Kolmatta vuosikymmentä jatkunut LEGO Mindstorms -laitteiden käyttö opetustyössä on puolestaan tuonut niihin ominaisuuksia, joiden ansiosta ne soveltuvat opetuskäyttöön erityisen hyvin. EV3 -ohjelmointikieli soveltuu kaikille ohjelmoinnin opetuksen tasoille esikoulusta aina yliopistoon saakka.

LEGO –robotiikka voi tukea monien oppiaineiden opetusta koulussa ja tarjoaa monipuolisia mahdollisuuksia oppiainerajoja ylittävien kokonaisuuksien toteuttamiseen kouluissa ja kerhoissa. Geometriaan, mittayksiköihin, koordinaatistoon, laskutoimituksiin ja ongelmanratkaisutaitojen kehittymiseen liittyvät tehtävät tukevat muun muassa matematiikan opetusta. Valo-oppia, kitkaa ja voimaa sivuavat harjoitukset puolestaan tukevat fysiikan opetusta.

Kun päästään robotin perusperiaatteiden opiskelusta eteenpäin, opittuja ohjelmoinnin ja robotiikan taitoja voidaan soveltaa erilaisissa eheyttävissä projekteissa. Tällainen projekti voi olla esimerkiksi robotiikkakilpailu. Teknisissä käsitöissä voidaan valmistaa erilaisia kisaharjoituksiin liittyviä harjoitusalustoja ja kilpalajeihin liittyviä tehtäväratoja tai valmistaa kehittyneimpiin kilparobotteihin omia antureita ja muita tarvittavia osia. Tekstiilityössä puolestaan voidaan ommella tai kutoa robottien vaatteita ja yhdistää niihin ohjelmoitavaa robotiikkaa ja rekvisiittaa. Kuvataiteessa voidaan tehdä mediaesityksiä, julisteita, kollaaseja ja musiikissa voidaan säveltää ja ohjelmoida roboteille omaa musiikkia. Äidinkieltä ja ilmaisutaitoja puolestaan tarvitaan muun muassa töiden suunnittelussa, projektin eri vaiheiden raportoinnissa ja töiden esittelyssä. Vain opettajien ja oppilaiden mielikuvitus on rajana tällaisia projekteja toteutettaessa.

Tämän kirjan avulla

- Opit rakentamaan robotteja
- Tutustut moottorien ja anturien toimintaan ja rakenteeseen sekä käytät niitä erilaisissa tehtävissä
- Opit laatimaan robottien toimintakuvauksia ja ohjelmoimaan niiden avulla
- Opit ohjelman suorittamiseen vaikuttavien haarautumisten, silmukoiden ja ehdollisien ohjelman osien käyttöä
- Opit muuttujien ja muistipaikkojen käyttöä ohjelmissa, sekä erilaisien tietotyyppien ja niiden muunnoksien käyttämistä
- Opit käyttöliittymän, eli näyttöruudulla esitettävien kuvien, tekstien ja numeeristen tietojen, sekä ohjaimen painikkeiden käytön ohjelmoimista
- Kilvoittelet ja suunnittelet oman pelin
- Opit tekemään ja käyttämään ohjelmissa käytettäviä kuvia ja symboleja, musiikkia ja äänimerkkejä
- Opit ohjelmoimaan matemaattisia yhtälöitä ja vertailuja

Tämä on Ensimmäiset robottini - kirjan ensimmäinen versio, jota tullaan muokkaamaan ja täydentämään vuoden 2016 aikana. Tavoitteenamme on tuottaa Asimov –sarjaan myös muita ohjelmointiin ja robotiikkaan liittyviä oppaita.

Tervetuloa tutustumaan robottien maailmaan ja innostumaan ohjelmoinnista!

Helsingissä 29.1.2016

Pekka Pihola,

Tiina Korhonen, Kati Sormunen ja Minna Kukkonen

Kirjan käyttäjälle

Otsikot

Tässä kirjassa käytetään kahdenvärisiä otsikoita

Harjoituksia ja tehtäviä, vihreät otsikot

Tämän kirjan harjoitukset on laadittu käytettäväksi perusopetuksessa tai kerhoissa. Useimmat niistä soveltuvat myös itsenäiseen ohjelmointiin ja robotiikkaan tutustumiseen vaikka kotona. Kirjan ensimmäinen versio pitää sisällään seitsemän harjoituskokonaisuutta. Kukin harjoituskokonaisuus sisältää muutamia tehtäviä.

Harjoitusten tekemistä tukevat kirjan loppupuolelta löytyvät neljä toimilohkokokonaisuutta. Graafisissa ohjelmointikielissä käskyt esitetään helposti omaksuttavien kuvakkeiden eli toimilohkojen muodossa. Toimilohkokonaisuudesta löytyvät tässä kirjassa käytettyjen toimilohkojen yksityiskohtaiset suomenkieliset kuvaukset, sekä niiden käyttöä koskevat tiedot.

Yleisiä ohjeita ja lähtötietoja harjoitusten tekemiseen, siniset otsikot

Tämän kirjan robotit on rakennettu EV3 Education rakennussarjan osista. Robottien ohjelmoinnissa käytetään EV3-paketin mukana saatavaa opetukseen tarkoitettua LEGO MINDSTORMS Education EV3 lisenssipohjaista ohjelmointiohjelmaa tai ilmaista EV3:n kuluttajaversion ohjelmointiohjelma. Kieli on molemmissa sama. Lisensoitu Education-versio sisältää kuluttajaversion lisäksi myös mm. fysiikan tuntien laboratoriotöihin soveltuvia tiedonkeruu ja –esitysominaisuuksia.

Muita vinkkejä

Etkot

Tehtävät edellyttävät usein pientä ennakkovalmistautumista, jotta ne voidaan viedä läpi sujuvasti. Näihin ennalta tehtäviin valmisteluihin kuuluu aina myös robottien akkujen lataaminen. Tästä ei mainita erikseen enää myöhempänä.

Infosivut

Infosivut ovat asiapitoisia alustuksia ohjelmoinnista, robotiikasta ja itse tehtävästä.

Tehtävät

Kutakin aihealuetta kohden on 1-3 tehtävää ja 1-2 syventävää lisätehtävää nopeimmille tekijöille.

Rakennustehtävissä esitettyjen kuvallisten ohjeiden noudattaminen on yleensä tärkeää myös seuraavien ohjelmointitehtävien onnistumisen kannalta. Myöhemmin ohjeita on vähemmän ja rakentaminen tapahtuu enemmän innovoimalla oman luovan suunnittelun, ongelmanratkaisun, omien ideoiden ja opittujen käytäntöjen soveltamisen pohjalta.

Laaditut ohjelmat kannattaa tallentaa, sillä niistä voi käydä myöhemmin kopioimassa uusiin ohjelmiin hyviä, jo toimiviksi havaittuja osuuksia.

Jatkot

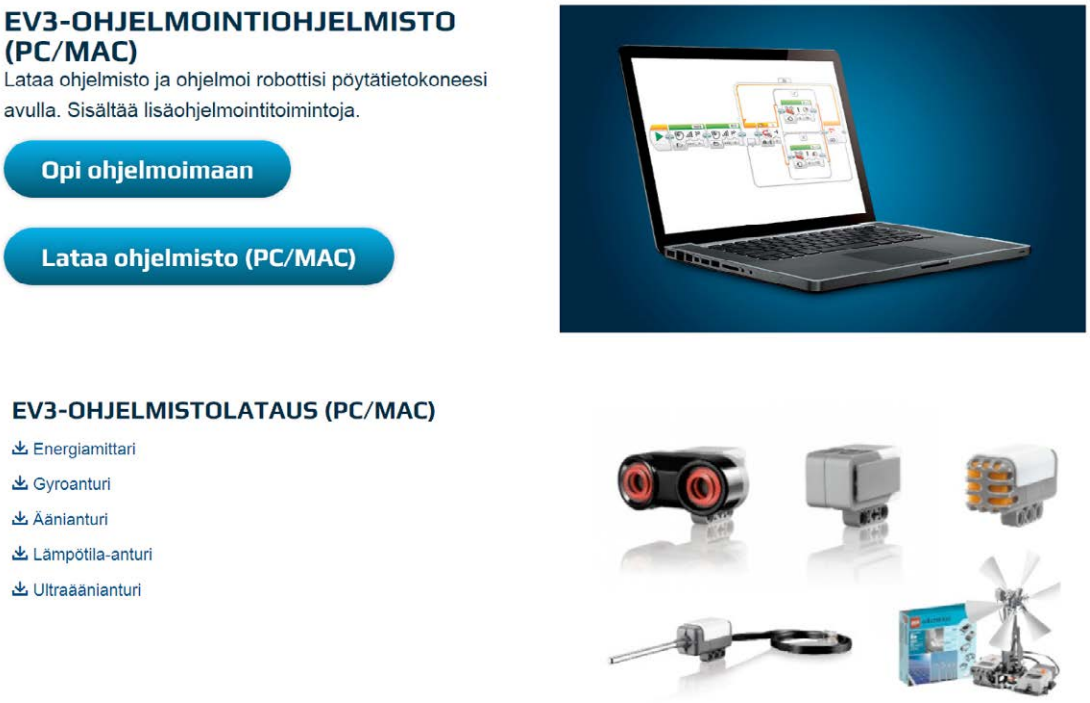
Jatkoihin kuuluu kotitehtäviä sekä vinkkejä seuraavaan aiheeseen tutustumiseen jo ennalta. Kotitehtäväksi voidaan antaa tehtyä harjoitusta tarkastelevia kysymyksiä ja/tai seuraavaan harjoitukseen valmistavia tehtäviä.

1. Rakennetaan monitoimirobotti

Rakennetaan monitoimirobotti. Monitoimirobottia käytetään ensimmäisissä harjoituksissa.

Etkot

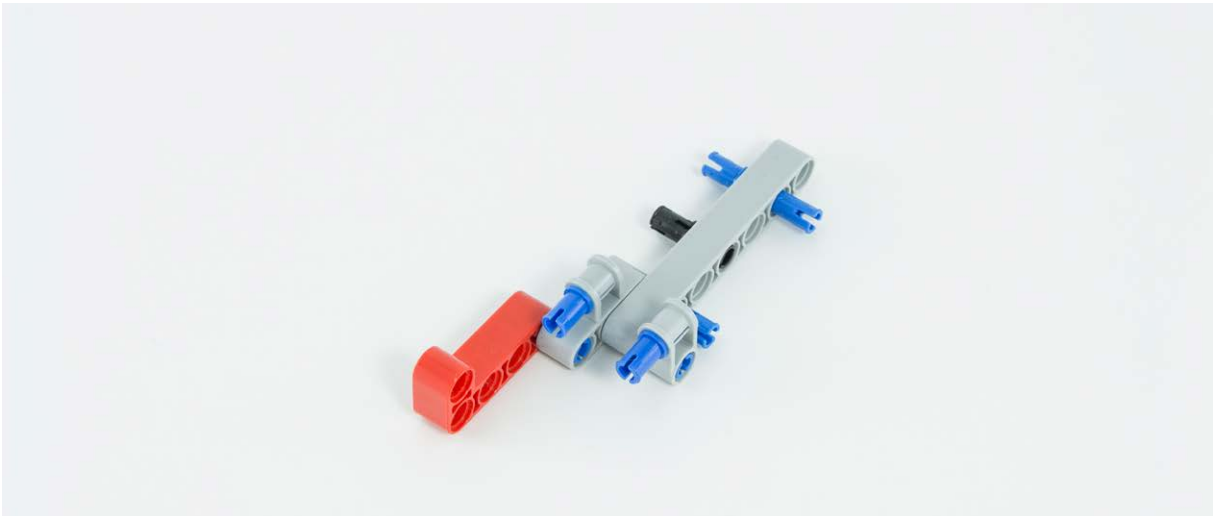
Opetuksessa käytettäville tietokoneille tulee olla asennettuna EV3:n ohjelmointi-ohjelman koululisensoitu versio tai uusin kuluttajaversio, joka löytyy osoitteesta www.lego.com, Tuotteet, Mindstorms ja Downloads (EV3 -ohjelmointiohjelman ilmainen Home Edition –versio). Samalla voi ladata myös suomenkielisen käyttöohjeen, rakenteluohjeita, sekä kuluttajaversiosta puuttuvat anturitiedostot (alempi kuva).

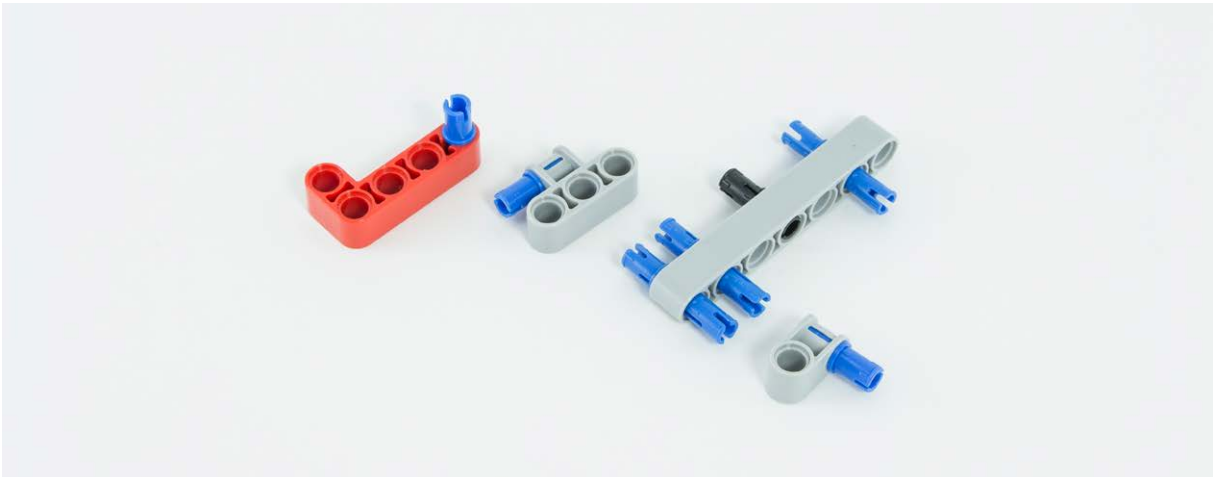


Kuvakkaappauksia LEGO Mindstorms laitteiden ohjelmistojen lataussivustolta. Kuvakaappaukset lähteestä www.lego.com

Tehtävä 1. Rakenna monitoimirobotti ohjeen mukaan

Rakenna monitoimirobotti seuraavan ohjeen mukaan. Aloitetaan vasemmasta rungon puolikkaasta, etsi tarvittavat osat ja rakenna kuvien mukaan.

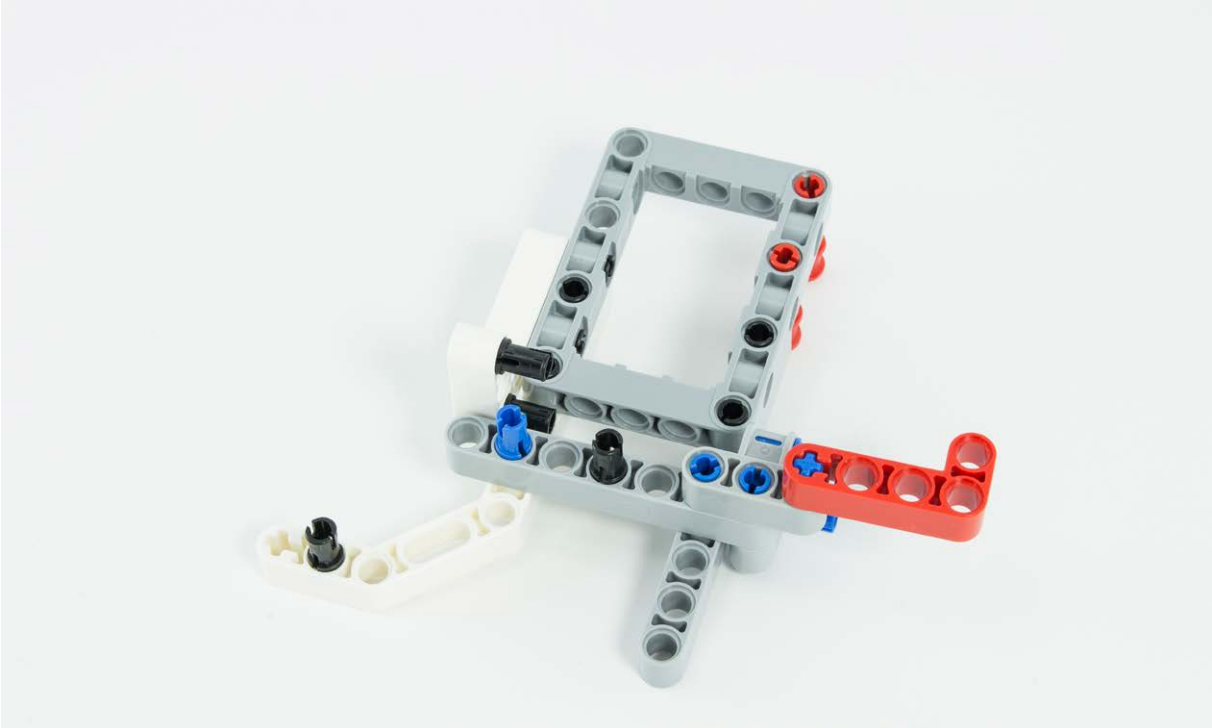




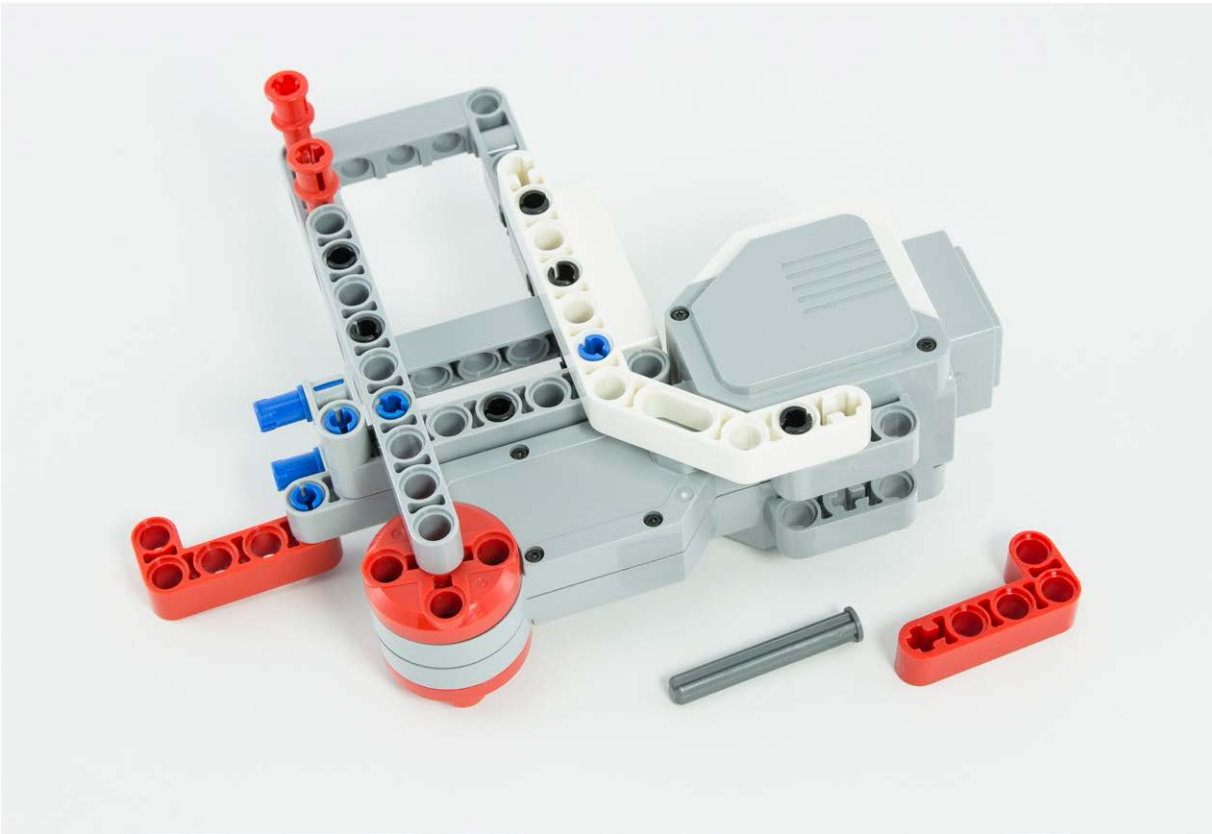
LEGO-järjestelmän perusmitta on ”nuppi” = 8 mm. Tässä kuvassa esimerkiksi harmaa palkki on 7 nuppia pitkiä. Kerää osat ja rakenna kuvan mukaan.

Pieniä liitostappeja kutsutaan ”pinneiksi”. Pinnejä on kahta perustyyppiä ja montaa eri mallia. Ne tunnistaa toisistaan helpoiten värin perusteella. Harmaat ja vaalean ruskeat edustavat vapaasti pyörivien pinnien ryhmää, joita käytetään kun halutaan jonkin osan liikkuvan herkästi.

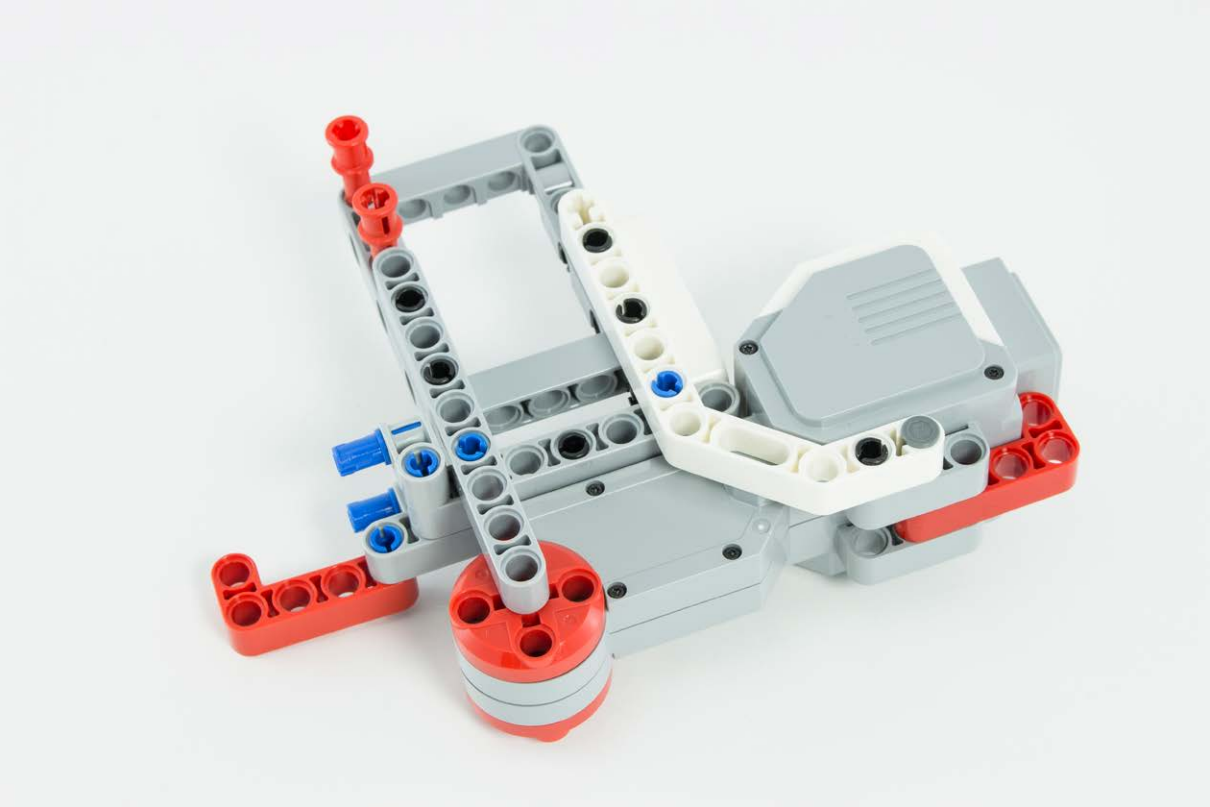
Useimmat pinnit kuuluvat kitka-elementeillä varustettujen pinnien ryhmään, kuten mustat, siniset ja kuvan punaiset vetopinnit. Joissakin pinneissä, kuten lyhyissä sinisissä pinneissä, toinen pääty on muotoiltu LEGO-akselin muotoiseksi.



Vasen rungon puolikas alkaa tässä jo hahmottua. Punaisia vetopinnejä käytetään kun halutaan liittää jotakin rakenteeseen vasta myöhemmin, tai kun halutaan tehdä rakenne josta jokin osa voidaan irrottaa helposti. Tai molempia.

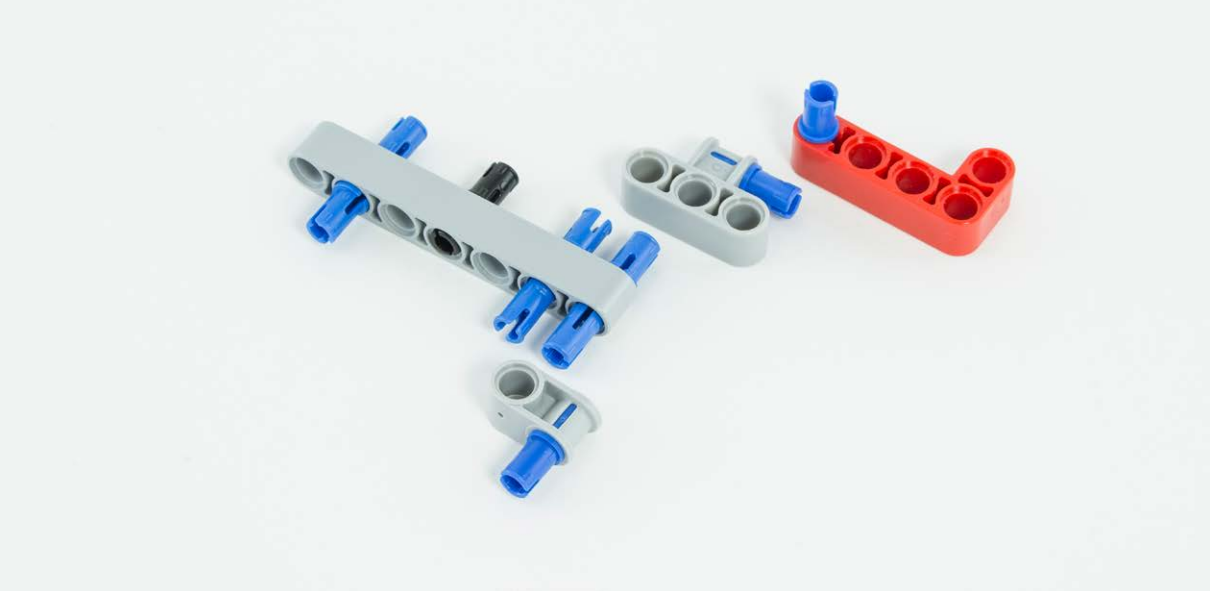


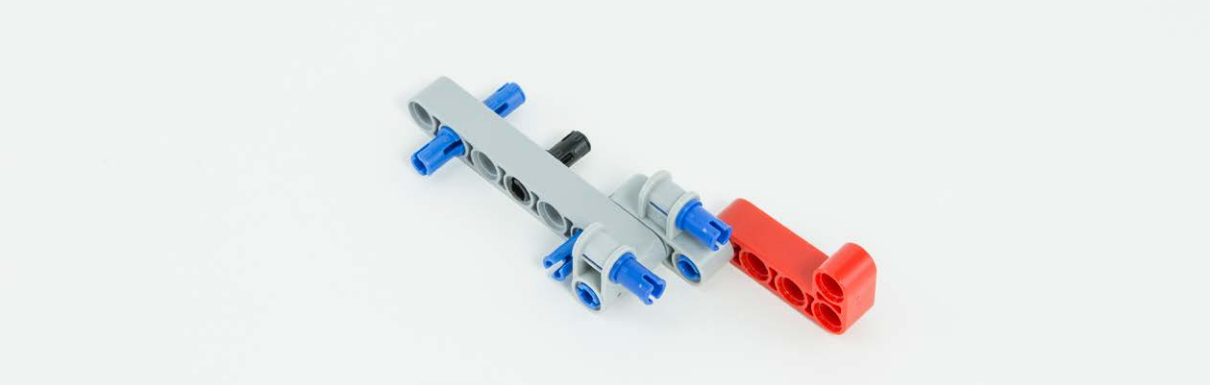
Kiinnitetään vielä moottori runkorakenteeseen ja punainen kulmapalkki moottoriin.

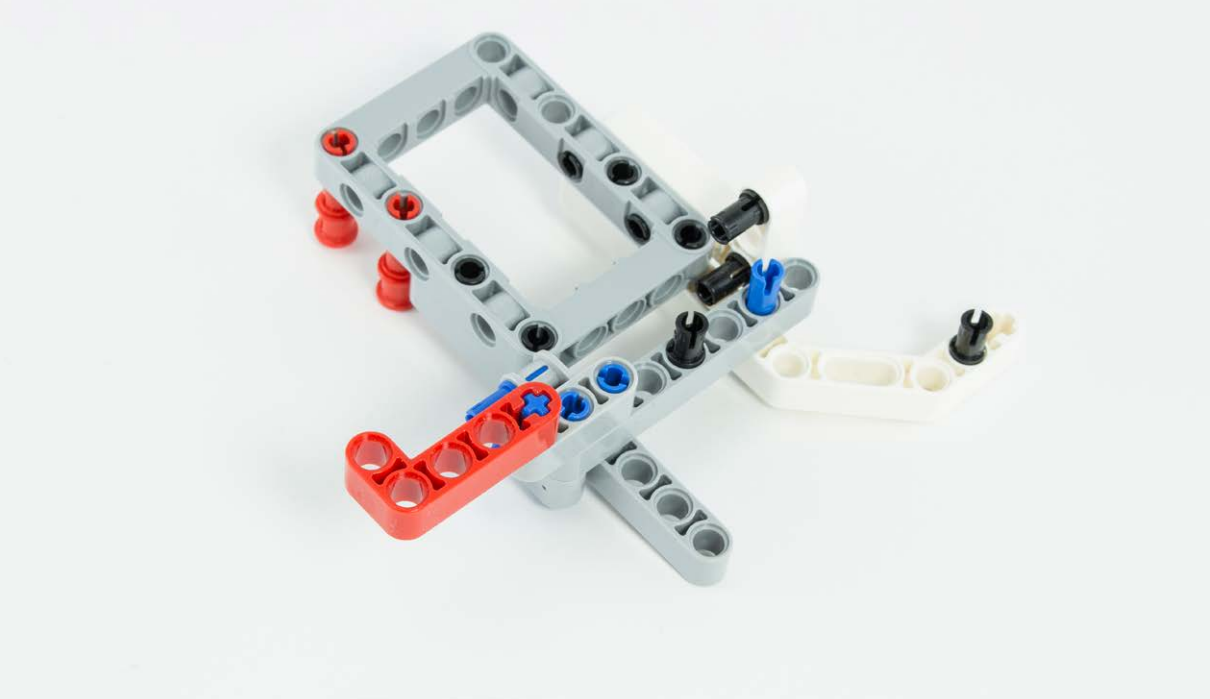


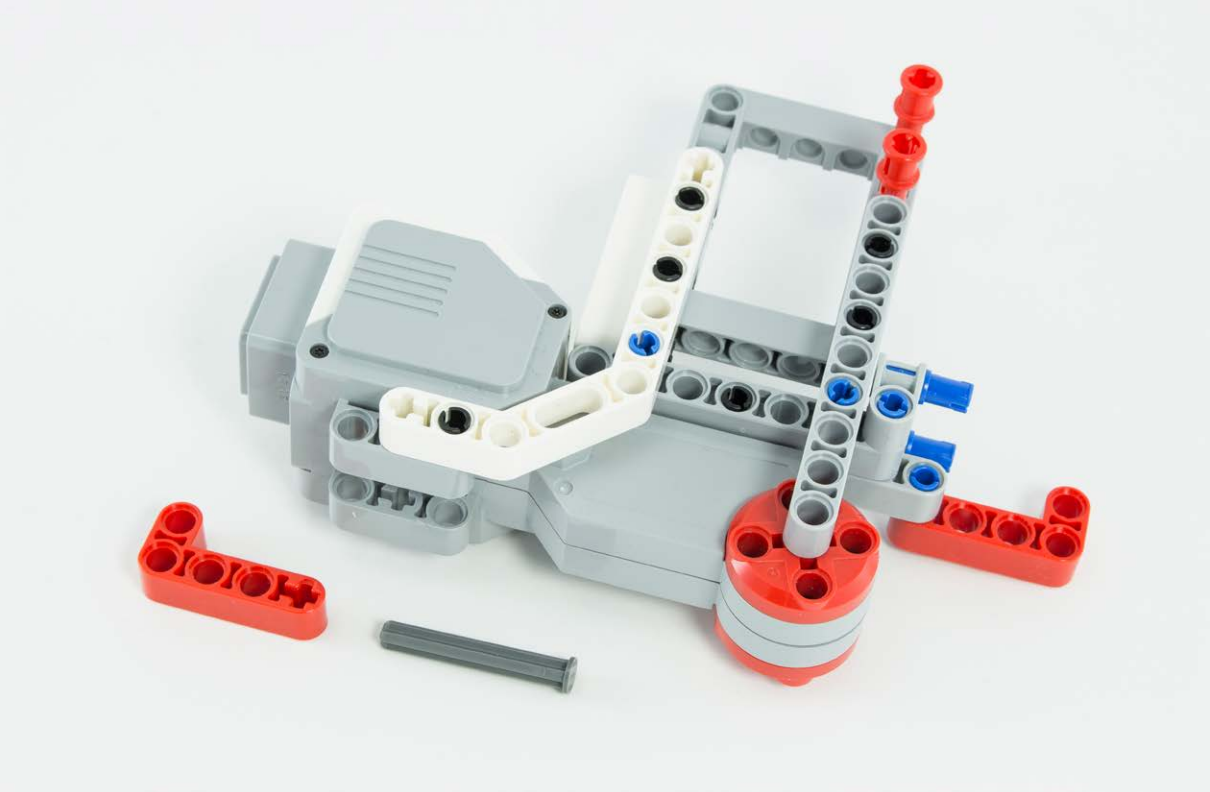
Vasen rungonpuolikas on nyt valmis.

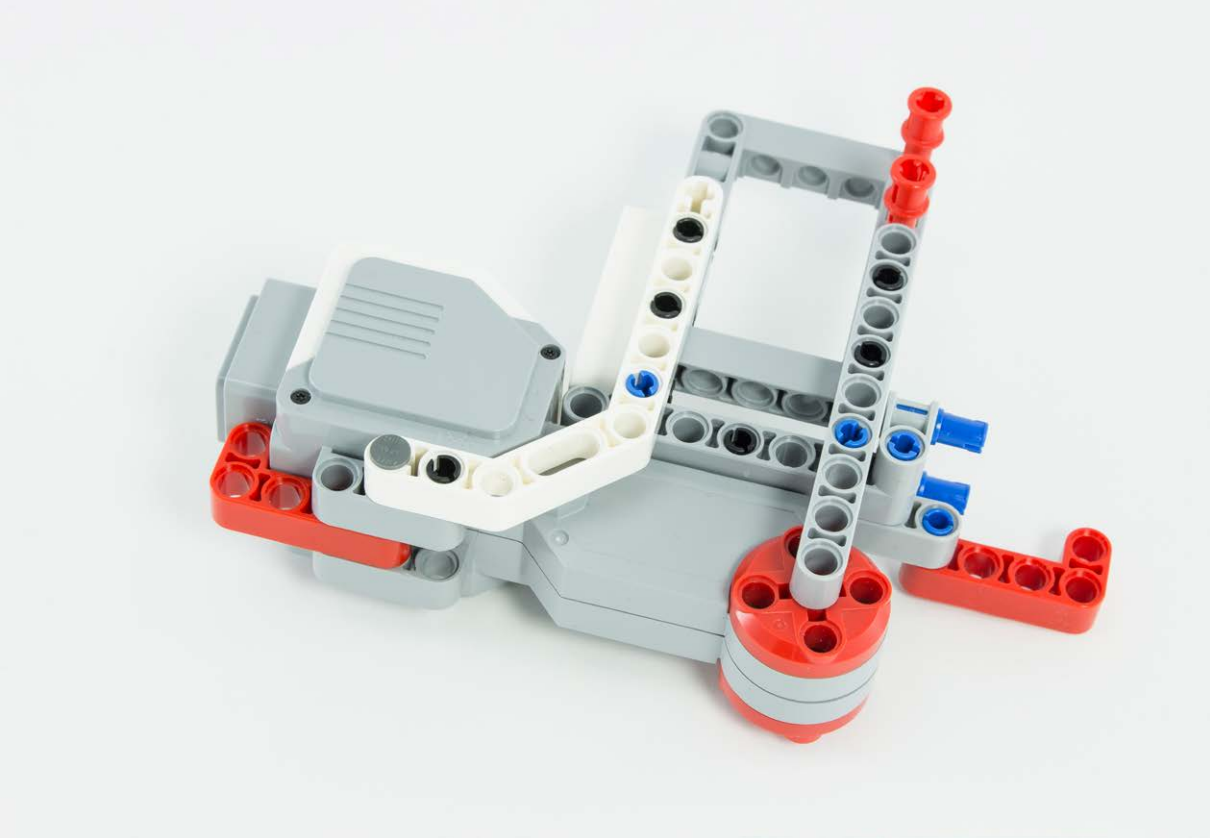
Oikeasta rungonpuolikkaasta tulee vasemman peilikuva.









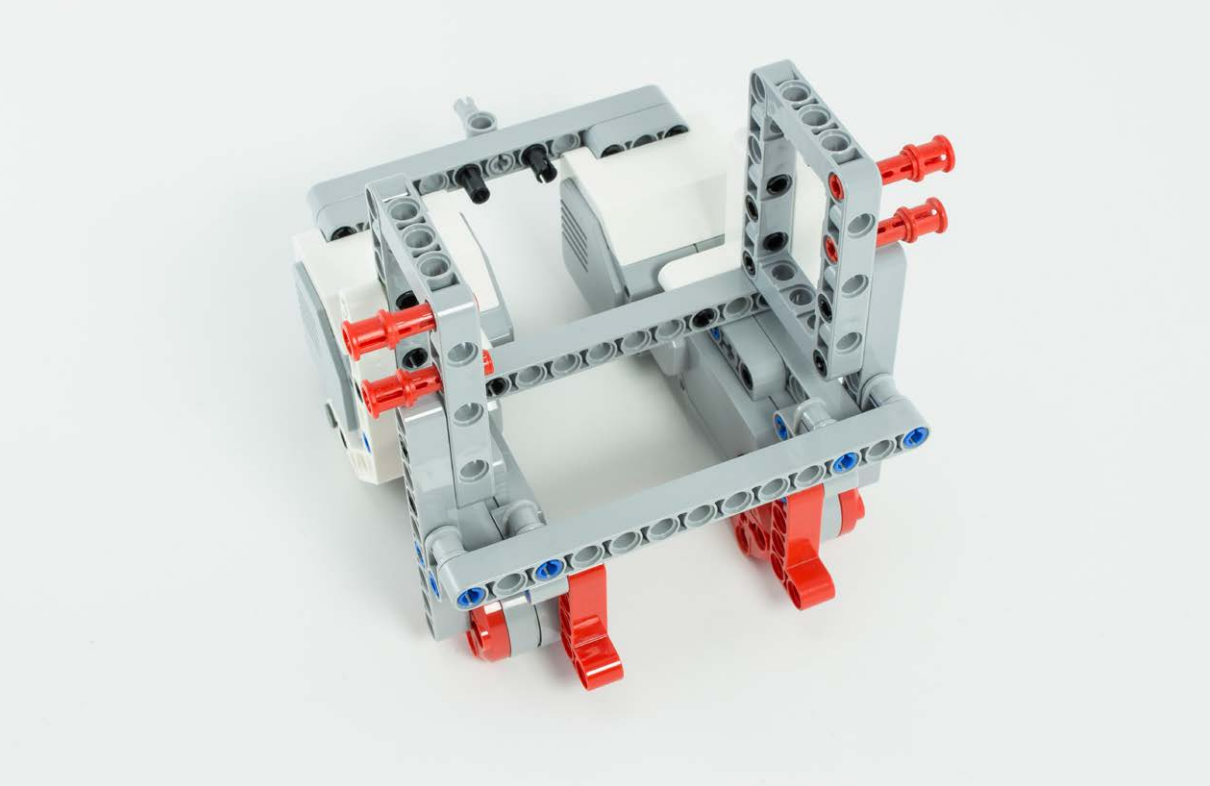


Oikea rungonpuolikas on nyt valmis.

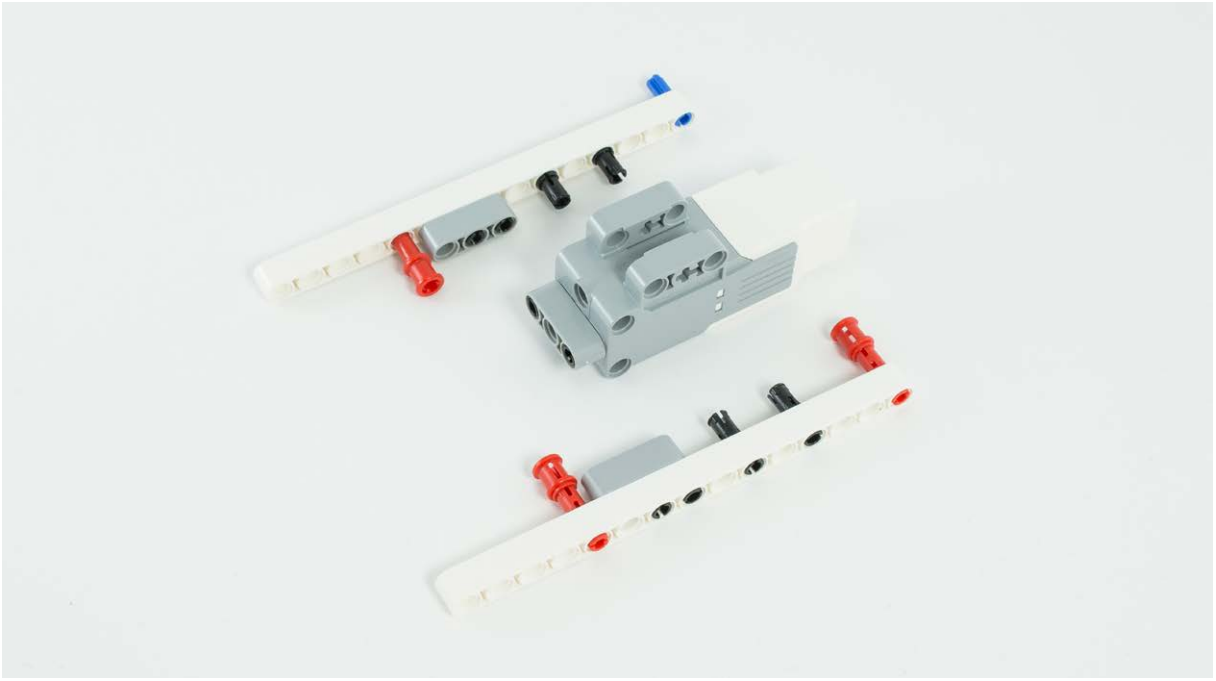
Tähän robottiin tulee kaksi isoa pyörää, mutta koska emme ole (vielä) tekemässä tasapainoilijaa, tarvitaan näiden lisäksi ainakin yksi pyörä tms. Hyviä vaihtoehtoja on useita. Tällä kertaa käytetään varmatoimista ja herkästi pyörivää tukipyörää (musta akseli on 8 nuppia pitkä).

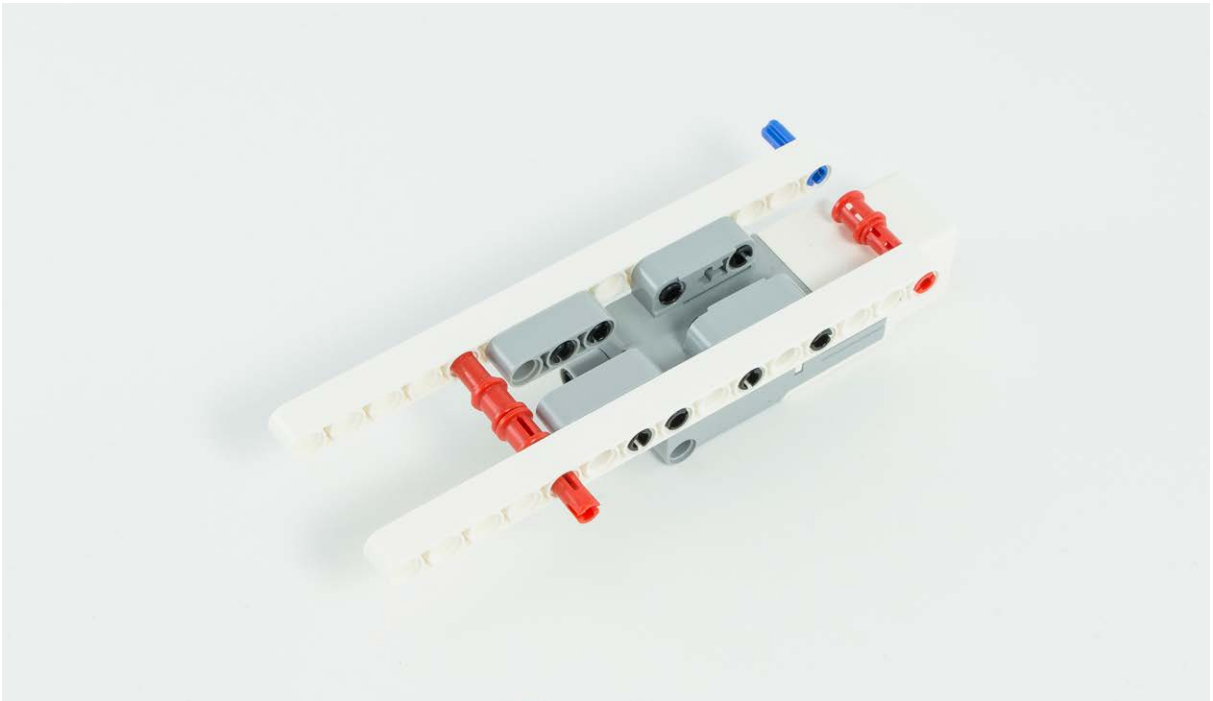
Nyt tukipyörä on paikalleen asentamista vaille valmis. Seuraavaksi yhdistetään rungonpuolikkaat toisiinsa kolmella palkilla.

Ensimmäisellä kiinnitetään moottorit toisiinsa. Kuvassa näkyy myös tukipyörän akselin ylempi kiinnike. Toinen 11-nuppia pitkä palkki tulee keskelle, kiinni valkoisiin kulmapalkkeihin.

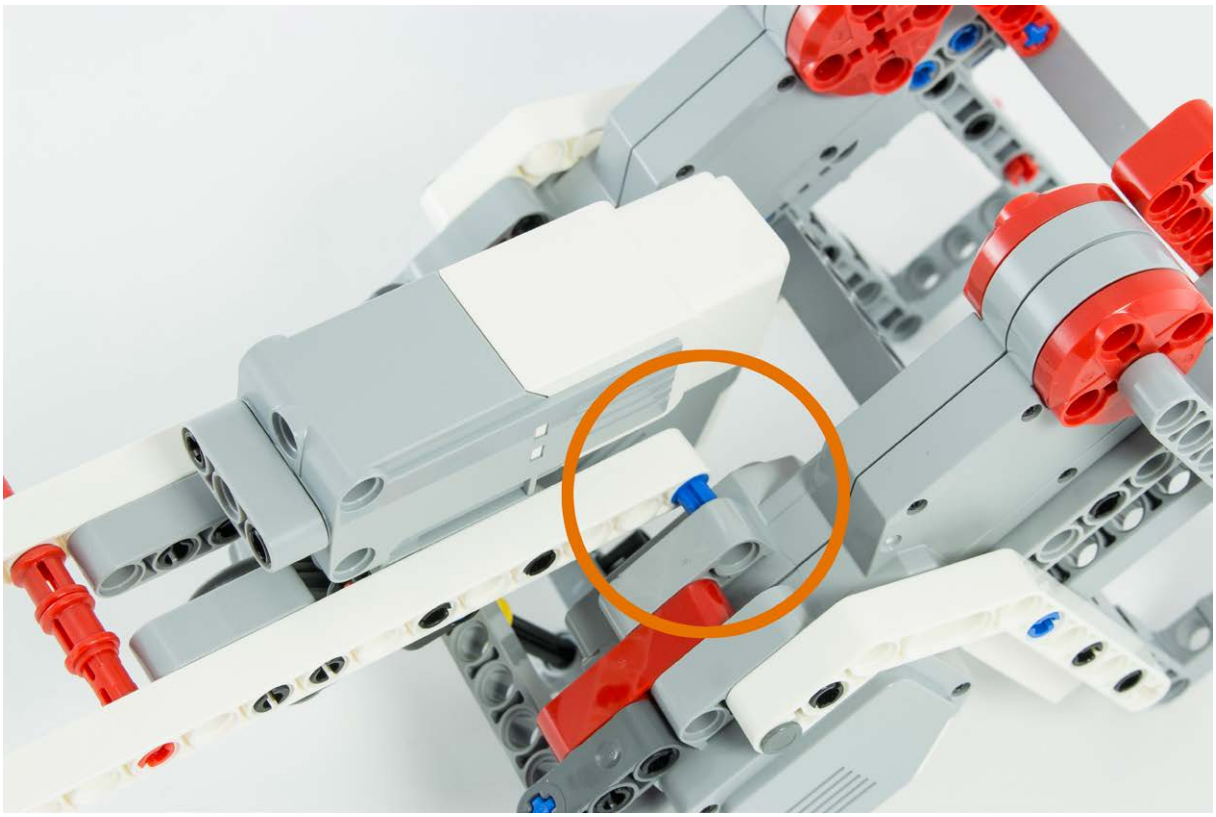


Ja kolmas, 13 nupin mittainen palkki tulee kiinni toiseen päähän (siniset pinnit). Tukipyörä kiinnitetään robotin alle. Kolmas moottori kiinnitetään valmiiksi robotin alle tulevaan kelkkaan, vaikka sitä ei tarvitakaan vielä ihan ensimmäisissä harjoituksissa.



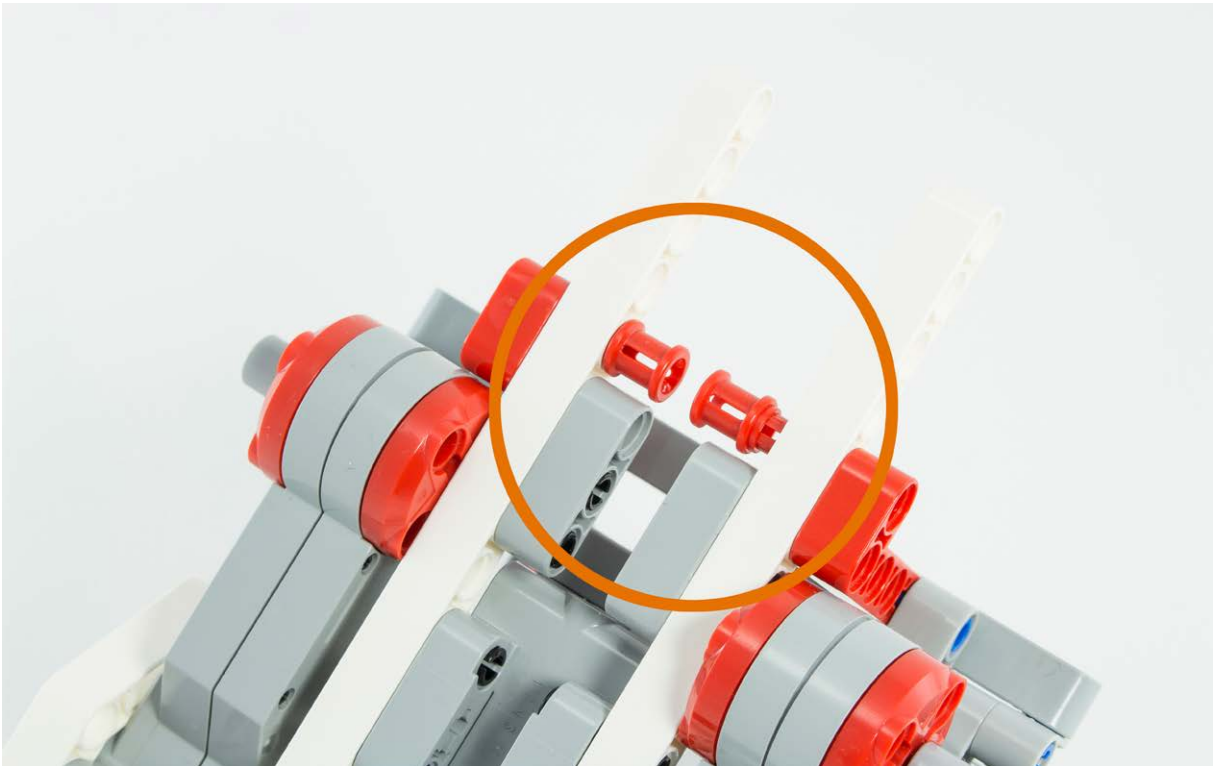


Kelkka kiinnitetään robotin alle pujottamalla ensin sininen pinni reikään. Katso tarkasti oikea kohta (kuvasta).

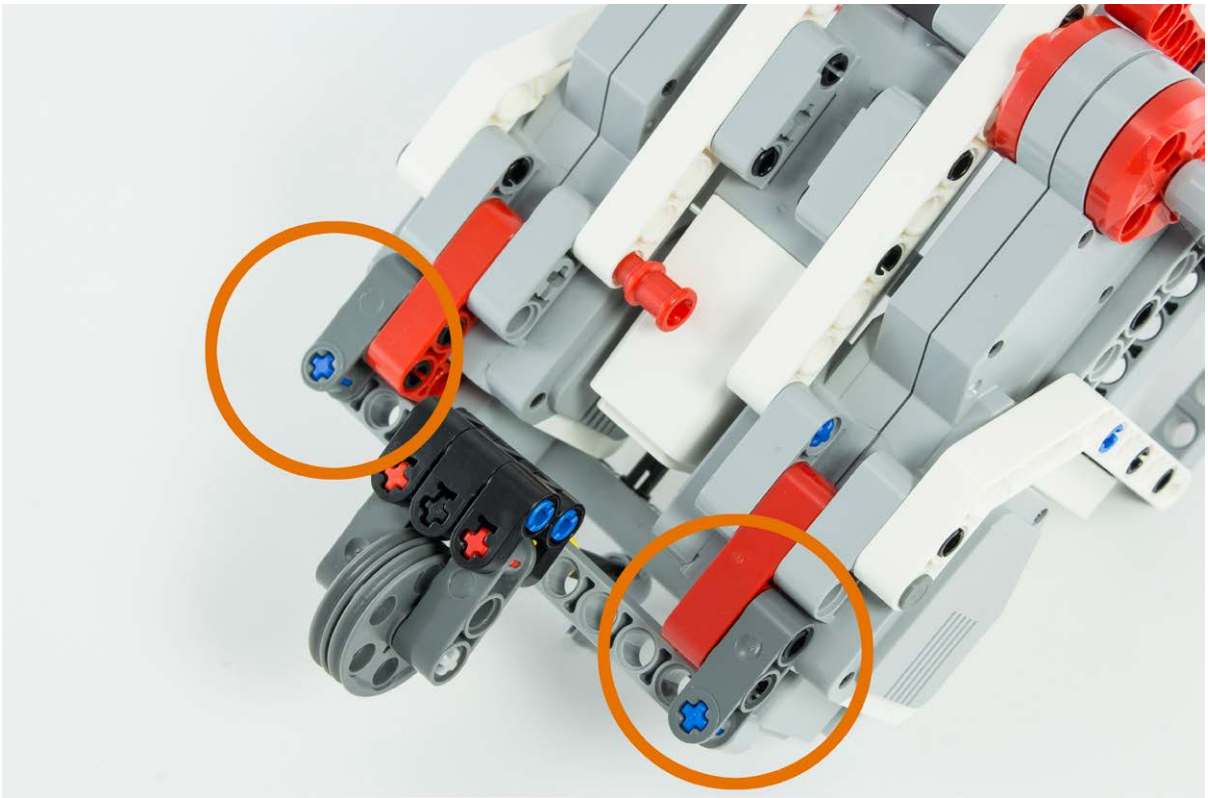


Kelkka käännetään kuvan osoittamalla tavalla ja punainen vetopinni, kuvassa vasemmalla, työnnetään sisään.

Kuvassa ylhäällä näkyy miten kelkan toinen pääty kiinnitetään vetopinneillä punaisiin kulmapalkkeihin. Kiinnitetään myös toinen aisa (kuvassa ylhäällä).



Tukipyörä kiinnitetään kääntämällä harmaat liitin-osat (pinneineen) kiinni kuvan osoittamalla tavalla.



Kokoa ja kiinnitä renkaat, akselit ovat 8 nuppia pitkiä. Näiden EV3 Education sarjan mukana tulevien renkaiden ulkohalkaisija on n.56 mm.

Yhtä pyörän, ja tässä tapauksessa myös moottorin pyörähdystä vastaava matka, voidaan laskea kertomalla pyörän halkaisija pii:llä (π).

Ympyrän kehän pituuden kaava: $x = 2\pi r$

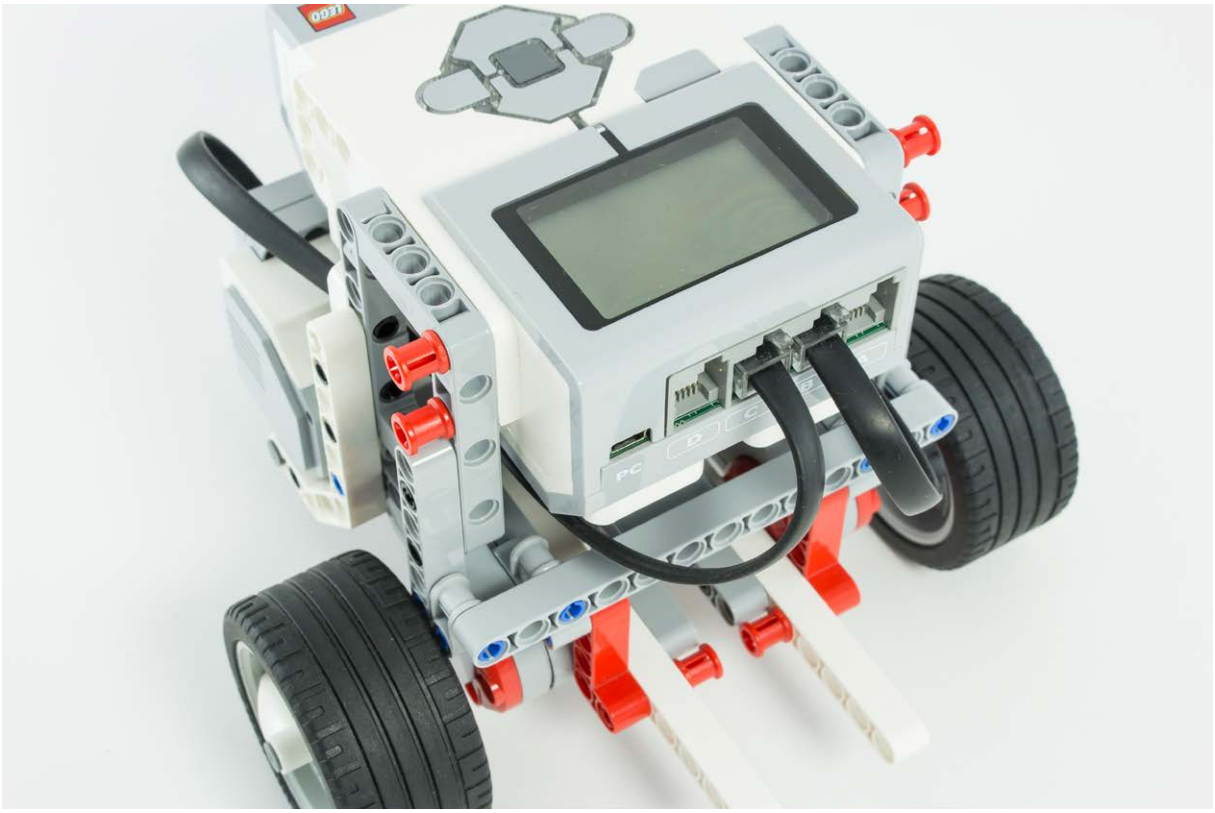
jossa r = säde (halkaisija = 2 kertaa säde)

ja π = vakio, pyöristettynä noin 3.14

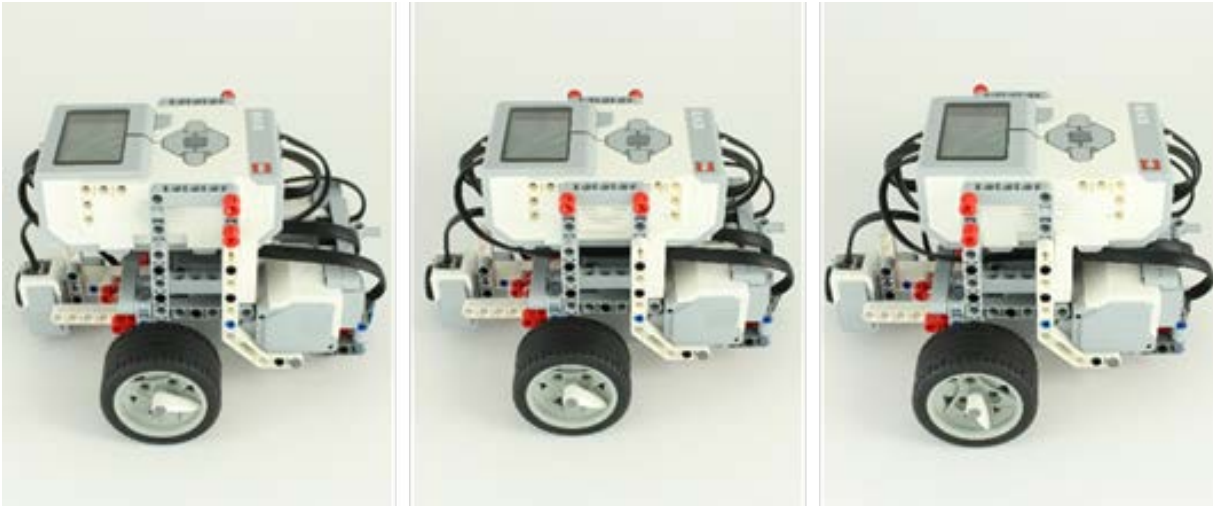
Yhden moottorin kierroksen aikana kuljettavan matkan pituudeksi saadaan siten laskemalla 56 mm x 3.14 ≈ 176 mm. Tuloksen voi tarkastaa vaikka vierittämällä rengasta pöydällä tasan yhden kierroksen verran ja mittaamalla sen kulkema matka viivottimella.

Tämän Robotin painavin osa on sen ohjainyksikkö (ks. viereinen sivu). Moottorien toiminnan ja kulumisen kannalta on edullista, mitä vähemmän ne joutuvat kantamaan robotin painosta. Tässä rakennettavan robotin painosta enin osa välittyy nyt pystypalkkien kautta ohi moottorien, suoraan pyörien akseleihin. Robotin ollessa painavampi on sen pyörät syytä tukea molemmilta puolilta.

Runko on nyt valmis. Seuraavaksi kiinnitetään ohjain runkoon työntämällä neljä punaista vetopinniä sisään, ja laitetaan pyörät paikoilleen. Robotin tulisi kulkea nyt jotakuinkin vaakasuorassa asennossa.



Tähän robottiin liittyy yksi erikoisuus. Käyttäjä voi muuttaa sen painopisteen paikkaa hyvin helposti, vaihtamalla ohjaimen kiinnityspinnien paikkoja.



Ohjain on nyt kuten vasemmassa kuvassa, vetävien pyörien päällä. Siirretään nyt osa sen painosta tukipyörän päälle, oikeanpuoleisen kuvan mukaisesti.

Robotin tulisi näyttää nyt tältä.

Kytketään seuraavaksi moottorit sopivan mittaisilla kaapeleilla ohjaimen portteihin **B** ja **C,** siten että kaapelit eivät risteä. Vasen **B:**hen ja oikea **C**:hen.

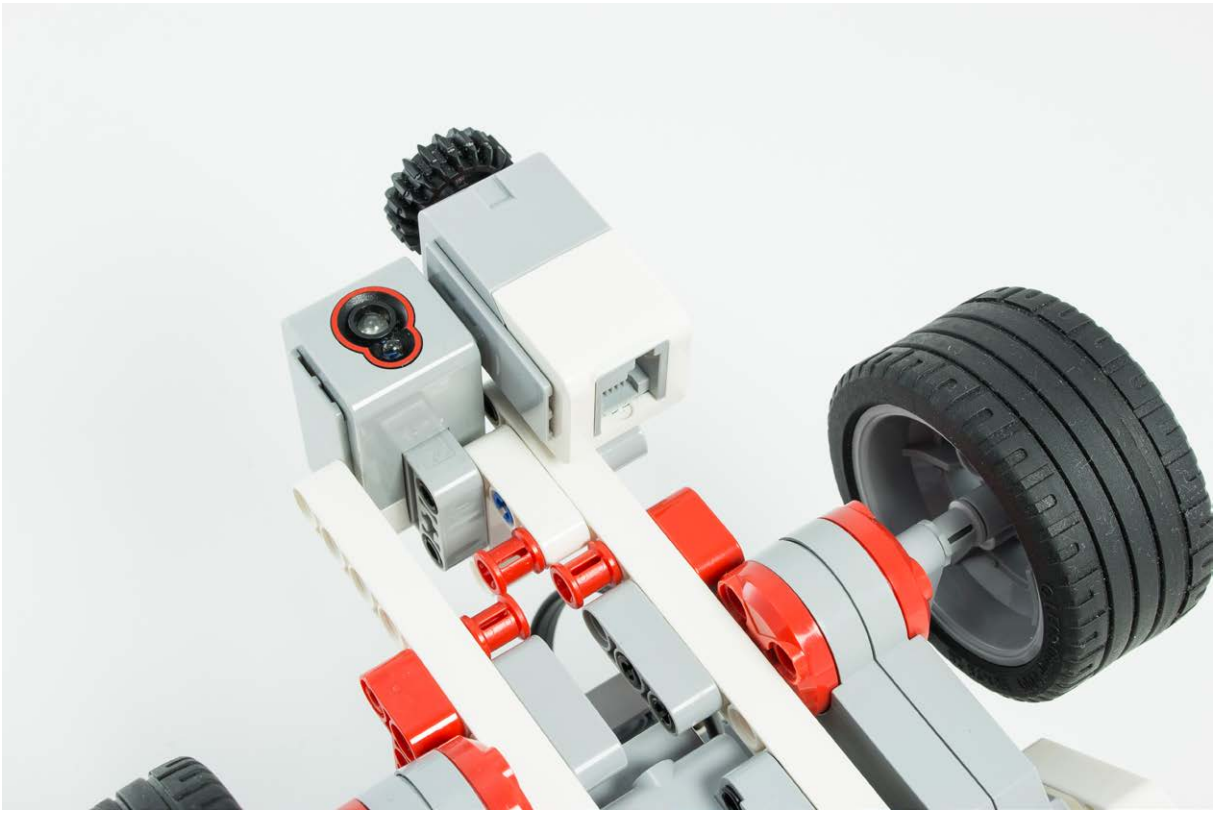
Vinkki: Kaapelien pujottaminen rakenteiden suojaan, rungon ja ohjaimen väliin käy helpommin, jos irroittaa ensin ohjaimen.

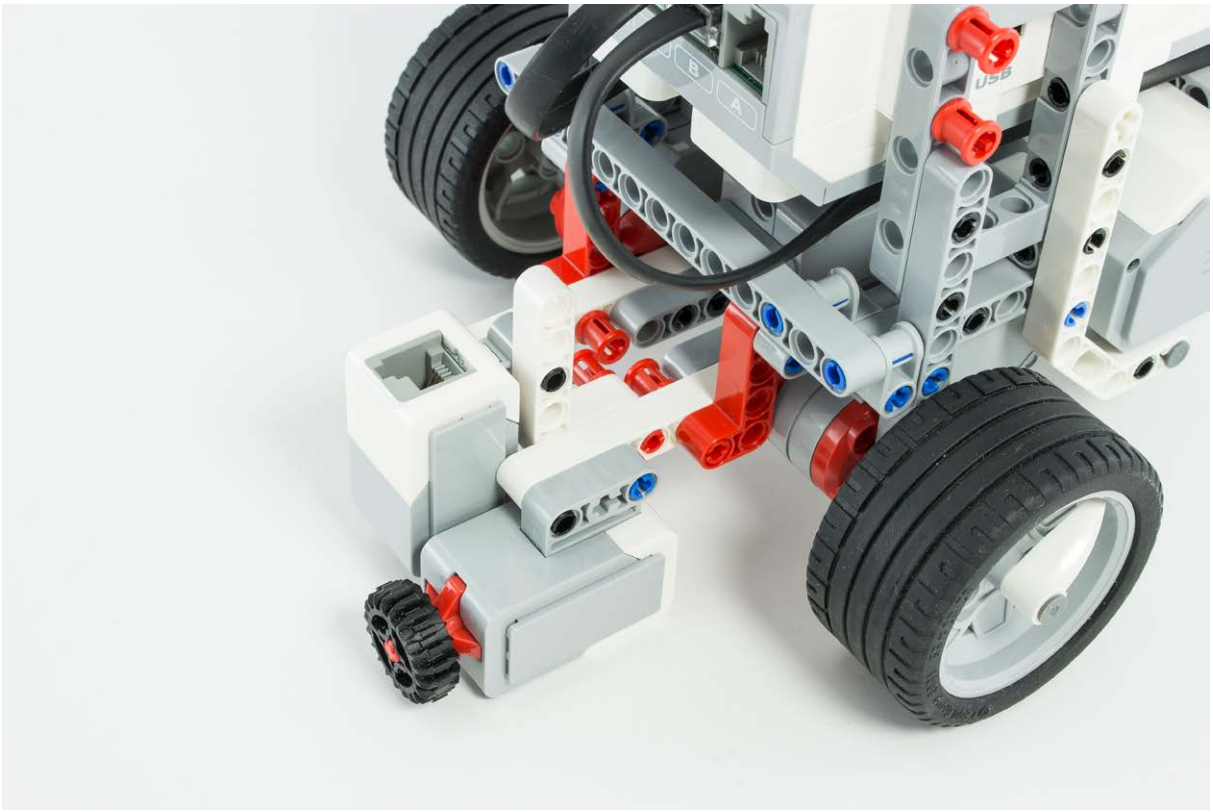
Ohjelmointikaapelin mini USB portti on saman rivin vasemmassa reunassa (PC).

Vinkki: Robotteja rakennettaessa on hyvä ottaa huomioon ohjelmointikaapelin ja akun latauskaapelin liittämisen / paristojen vaihtotyön helppous.

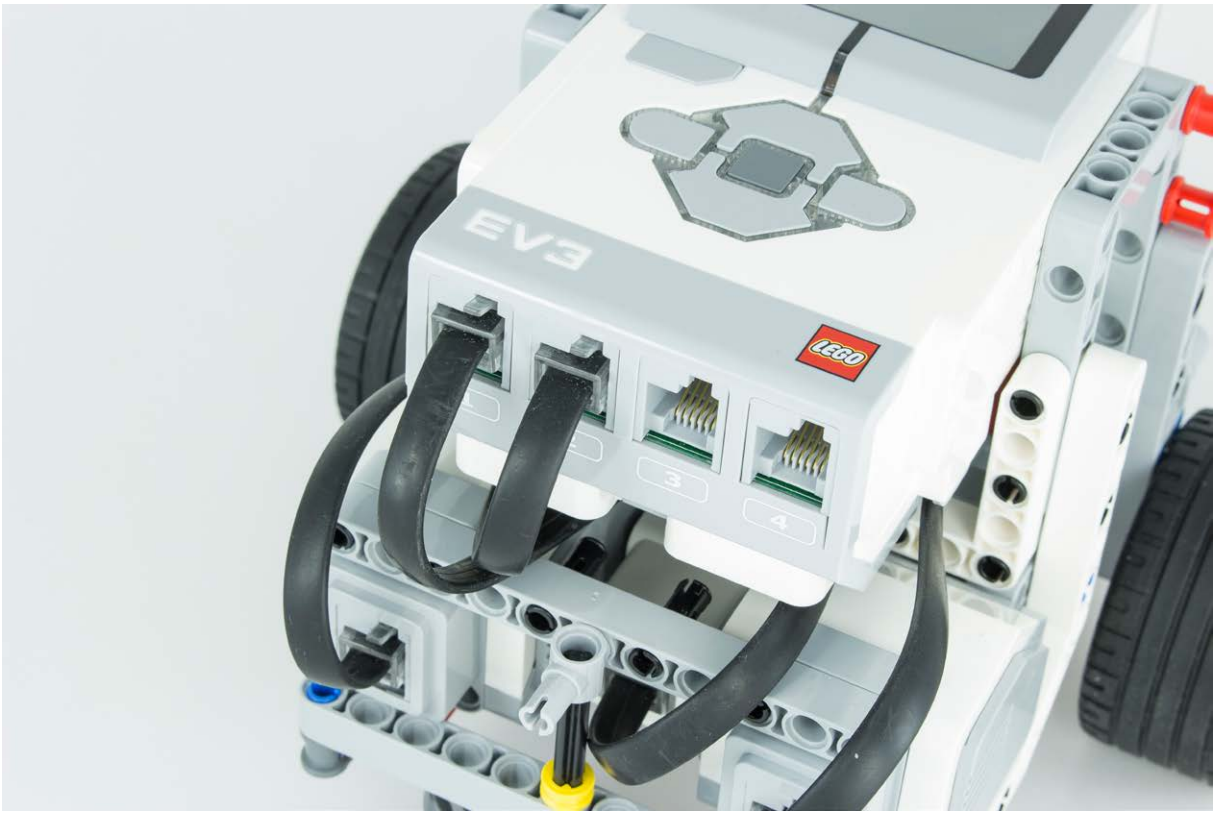
Lopuksi robottiin kiinnitetään vielä kaksi anturia, värejä ja valoa tunnistava valokenno (vasemmalla), sekä kosketusta tunnistava painonappi.

Valokennon kiinnittetään keskelle, ”aisojen” väliin ja kosketusanturi sen viereen Huomaa, että käyttötilanteesta riippuen voit kiinnittää antureita aina tilanteen edellyttämällä tavalla.





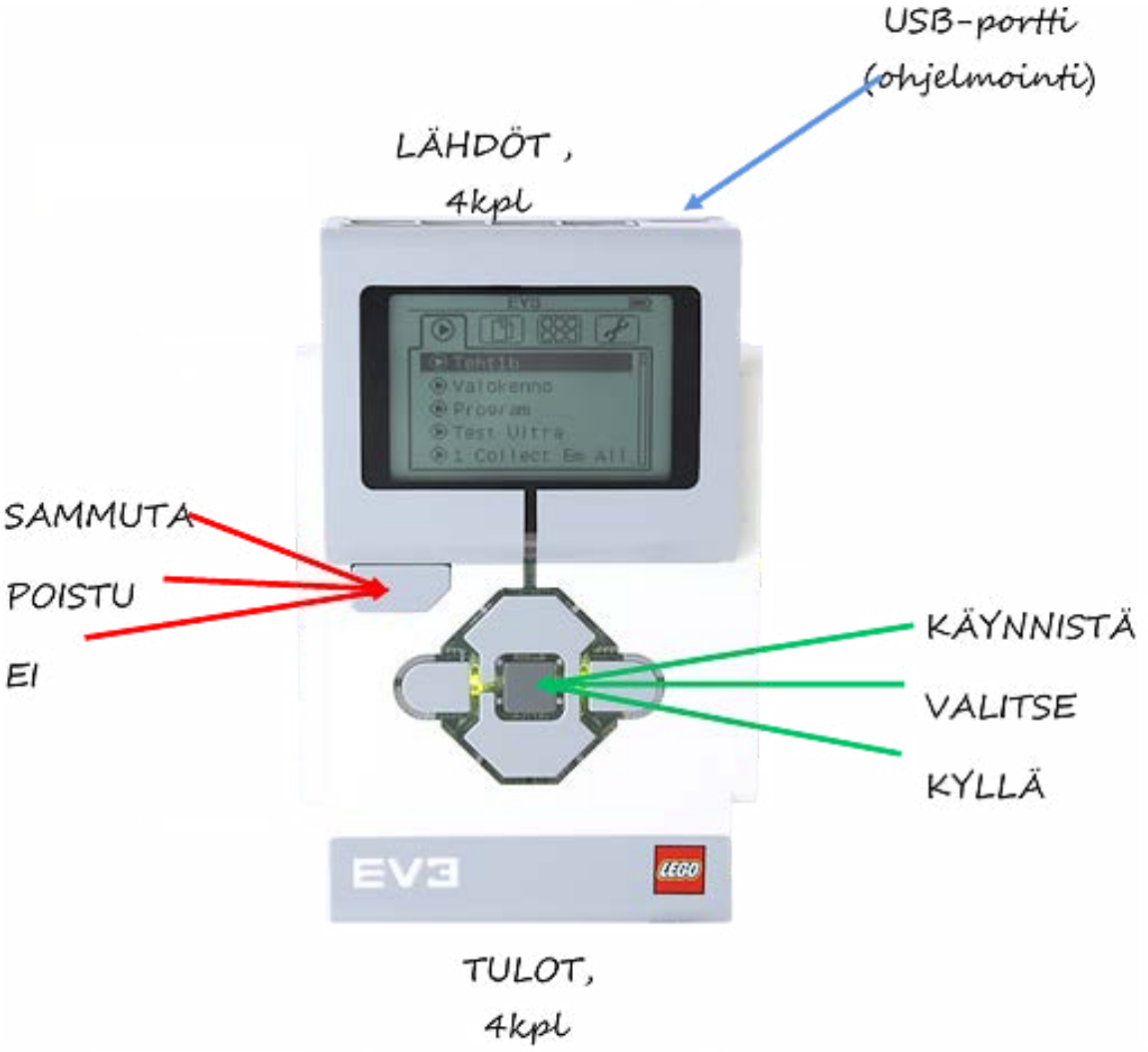
Kosketusanturi kiinnitetään valokennon viereen kuten kuvassa, aisan alapuolelle, tai yläpuolelle. Kytketään vielä antureihin kaapelit.



Kytke valokenno porttiin 1 ja kosketusanturi porttiin 2.

Ja sitten asiaan...

Ohjaimen tärkeimmät liitynnät ja painikkeet



Prosessori

Ohjelmoitavien laitteiden, kuten EV3:n ja tietokoneen “aivoina” on prosessori (CPU, *Central Processing Unit*).

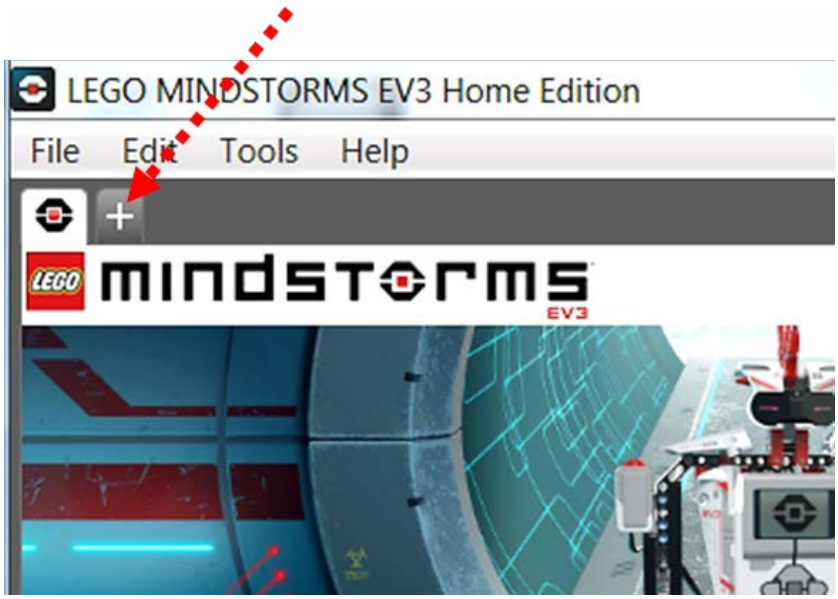
Se on kaikkea käskyjen suorittamista ohjaava ja laskentaa suorittava komponentti, joka toteuttaa sille annettuja käskyjä juuri kuten ne on kirjoitettu. Ohjelmoitaessa on siis oltava tarkka ja huolellinen, jotta prosessori tekisi juuri oikeat asiat, oikealla tavalla ja oikeassa järjestyksessä.

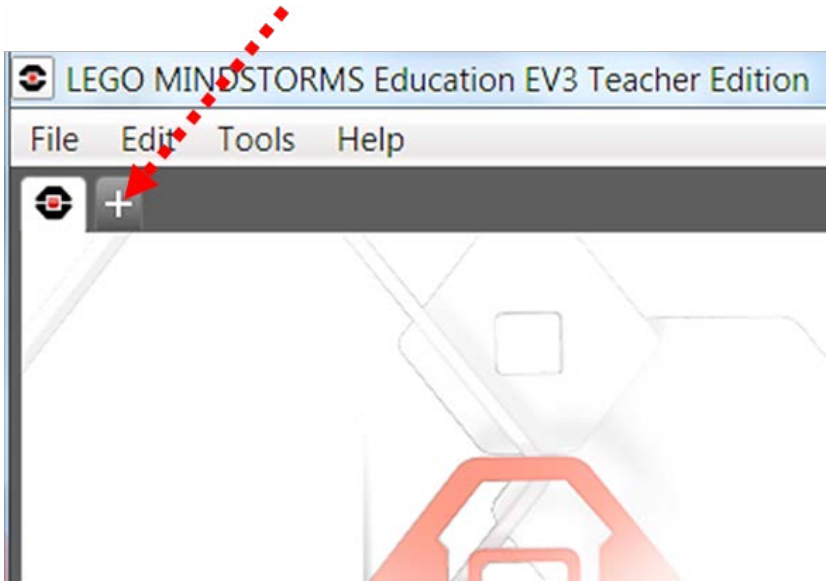
Ohjelmoitavia LEGO Mindstorms robotti-ohjaimia eri aikakausilta.

2. Ensimmäiset ohjelmani, nopeat ja helpot

Ohjelmoijan työpöytä

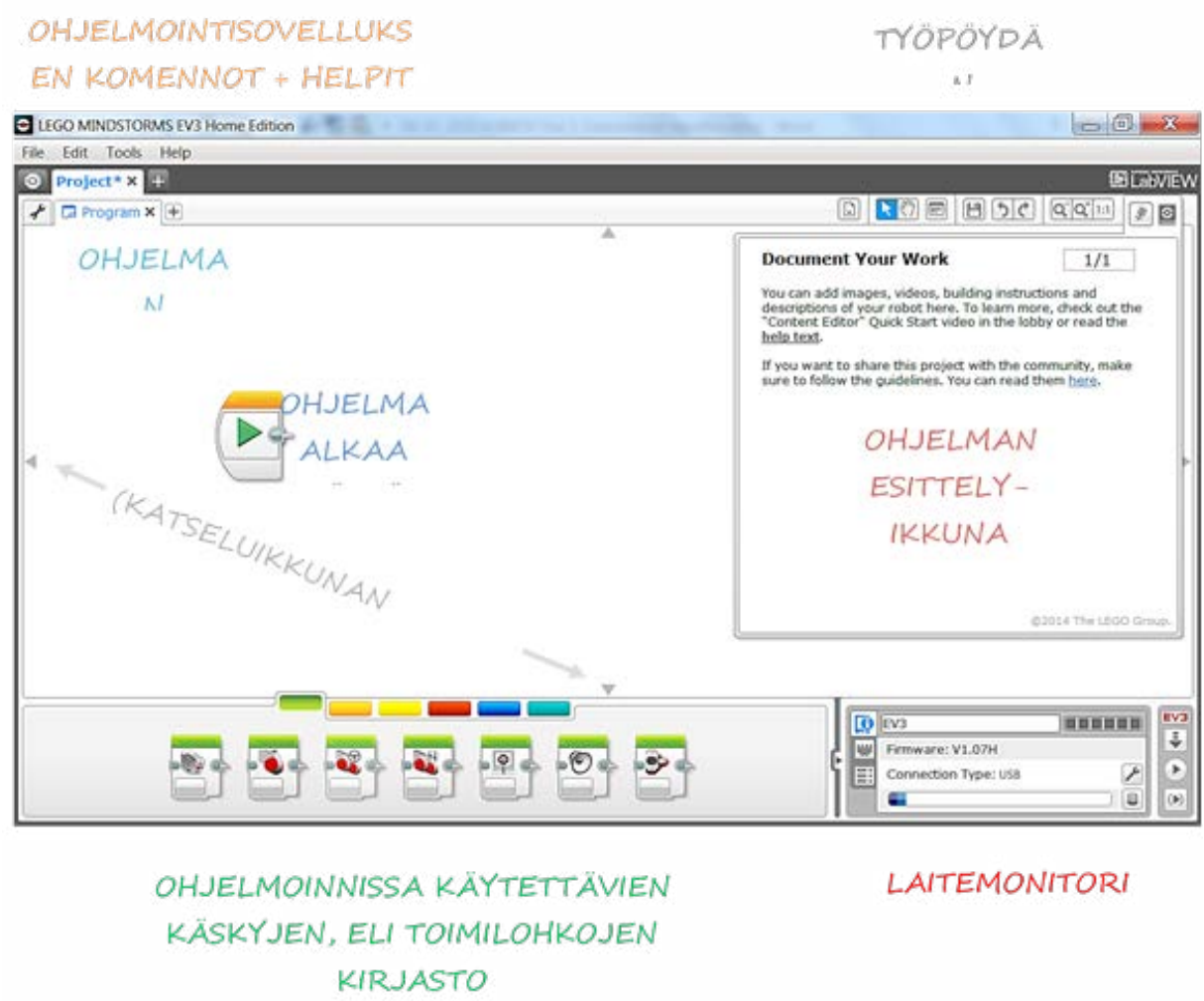
1. Avaa ohjelmointiohjelma, LEGO MINDSTORMS EV3
2. Näytölle avautuu ohjelmiston etusivu
3. Ohjelmoijan työpöydälle pääset helpoiten klikkaamalla plus (+) –merkkiä, ikkunan vasemmassa ylänurkassa (katso kuvat alla)



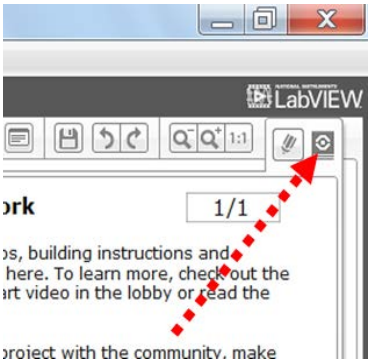


Työpöydän sisältö

Ohjelmoijan työpöydällä on monenlaisia tietoja ja valintoja. Tekemisen myötä ne tulevat nopeasti tutuiksi. Kuvassa alla on esitelty työpöydän pääkohdat.



Vinkki: Ohjelman esittelyikkuna vie paljon tilaa työpöydällä. Milloin sitä ei tarvita, se voidaan piilottaa klikkaamalla työpöytä-komentojen oikeassa reunassa olevaa EV3:n symbolia.



Tehtävä 2. Tarkastetaan moottorien toiminta

Käsketään kokeeksi robottia kulkemaan ensin pieni matka eteenpäin kaartaen samalla hiukan oikealle.

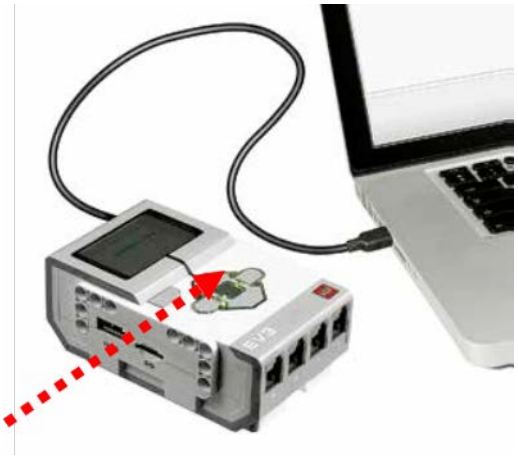
1. Poimitaan hiirellä toimilohkojen kirjastosta Move Steering –toimilohko ja liitetään se Ohjelman aloitus –toimi-lohkoon (kaksi moottoria, ohjauspyörän kuva).

2. Klikataan mustan nuolen alla olevaa numeroa, ja kirjoitetaan ruutuun numero 20.

Ensimmäinen ohjelma on nyt valmis. Seuraavaksi se pitää siirtää tietokoneelta robotin ohjaimeen.

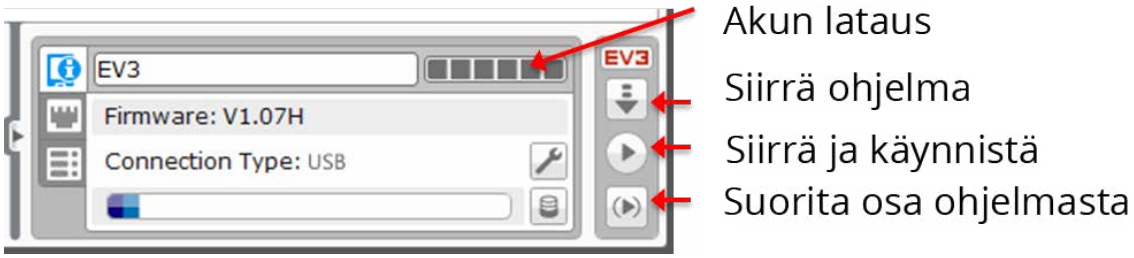
3. Kytke USB-kaapelin mini-USB liitin EV3 yksikön PC-porttiin (D-portin vieressä). USB-kaapelin toinen pää kytketään tietokoneeseen.
4. Käynnistä sitten ohjain käynnistys-painikkeesta (keskellä) ja odota merkkiääntä.

Tähän menee tovi, n. 40s.



(kuva: The LEGO Group)

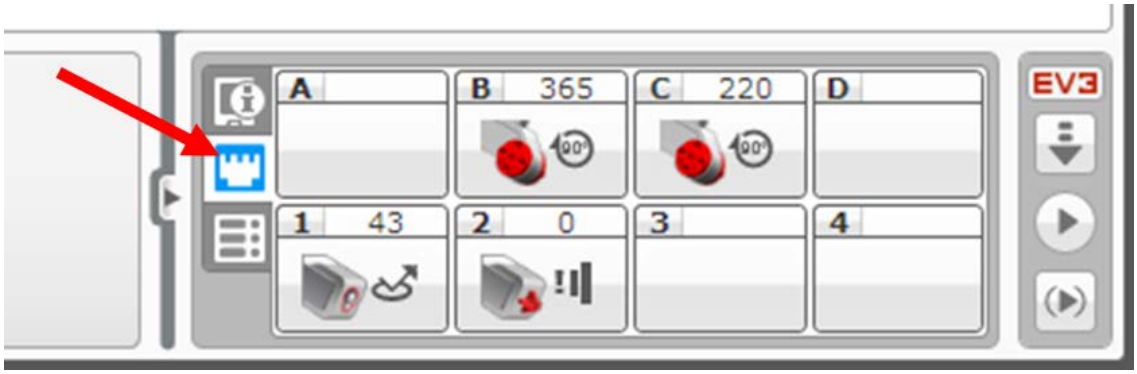
Laitemonitorin tulee nyt herätä eloon.



Jos näin ei tapahtunut, tarkasta USB-kaapelin liitännät.

5. Laske robotti lattialle irroittamatta johtoa
6. Klikkaa joko
   - Ohjelman aloitus –toimilohkon vihreää kolmiota, tai
   - Laitemonitorin oikealle osoittavaa mustaa kolmiota.
7. Jos robotti ajoi pienen matkan eteenpäin kaartaen samalla hiukan oikealle, kaikki toimii. Muussa tapauksessa – tarkasta kytkennät ja yritä uudestaan.

Vinkki: USB-kaapelin ollessa kytkettynä antureiden mittaamia tietoja voi tarkastella ohjelmointiohjelman laitemonitorin portti-näkymässä (klikkaa punaisen nuolen osoittamaa kuvaketta).



Kokeile pyöräyttää moottoria varovasti, paina kosketusanturia ja kokeile miten valokennon mittaama heijastuvan valon määrä muuttuu erilaisilla alustoilla.

Tehtävä 3. Se on kunnari!

1. Tehdään lattialle pesäpallokenttä pienoiskoossa. Pesien paikat voidaan merkitä käsillä olevilla pienillä esineillä, kuten vaikka teippirulla, mukilla, jne. Sopiva matka pesien välillä on luokkaa 0.5 – 1 metriä.
2. Ennen kuin lähdetään kiertämään koko kenttää, kokeilaan ensin käydä kärkkymässä ykkösellä. Eli, otetaan ensin tuntumaa kenttään ohjelmoimalla robotti kiertämään ykköspesän ja palaamaan saman tien kotiin.
3. Pura robotille annettava tehtävä pieniin osiin, eli kirjoita

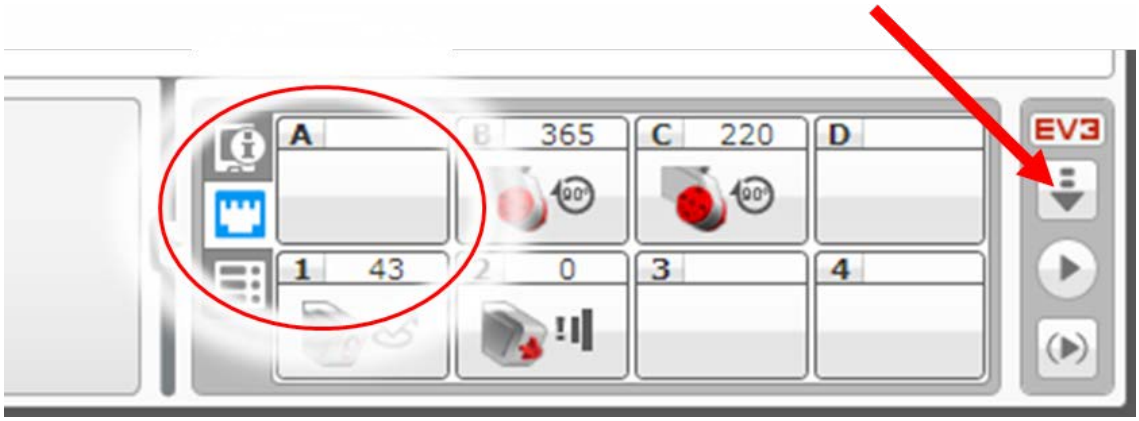
   Toimintakuvaus:

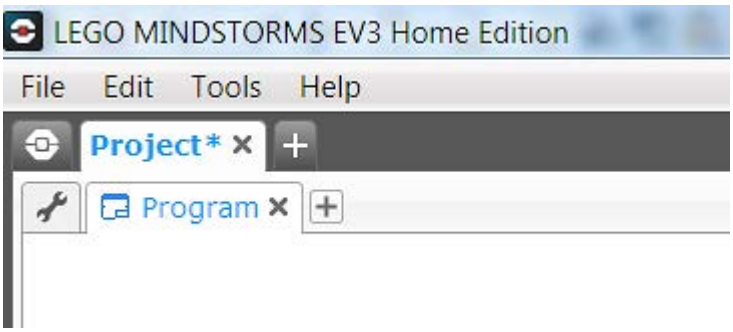
   - Aja suoraan pesän rinnalle
   - Aja kaartaen pesän ympäri
   - Aja suoraan kotipesään

4. Ohjelma voidaan kirjoittaa esimerkiksi käyttäen kolmea ”Move Steering” toimilohkoa

5. Siirrä ohjelma kuvan osoittamaa kuvaketta klikkaamalla. Muuten robotti käynnistyy heti ja pahimmassa tapauksessa ryntää alas pöydältä.

6. Irroita ohjelmointikaapeli, ohjelma on nyt tallennettu ohjaimen muistiin **”Project”** –kansioon nimellä **”Program”.**





7. Haetaan ohjaimen muistista, **Tiedostokansioista**, sen navigointipainikkeita käyttämällä **”Project”** –kansio ja sieltä ohjelma nimellä **”Program”.**

8. Viedään robotti lähtöviivalle, otetaan tarkka sihti ja käynnistetään ohjelma ohjaimen keskimmäisellä painikkeella.

9. Pienet hienosäädöt ja kun ykköspesän kierto alkaa sujua, voidaan lähteä juoksemaan kunnaria.

Vinkki: Rakenna ohjelmaa etappi kerrallaan, käsky kerrallaan.

Vinkki 2: Kokeile myös ”Move Tank” toimilohkon käyttöä.

Vinkki 3: Toimilohkojen esittelyt löydät kirjan lopusta.

Tehtävä 4. Seurataan viivaa

Tehdään robotille ohjelma, jonka avulla se osaa seurata viivaa. Tarkemmin sanoen mustan viivan vasenta reunaa. Viivan tunnistamiseen käytetään robotin etuosaan kiinnitettyä valokennoa.

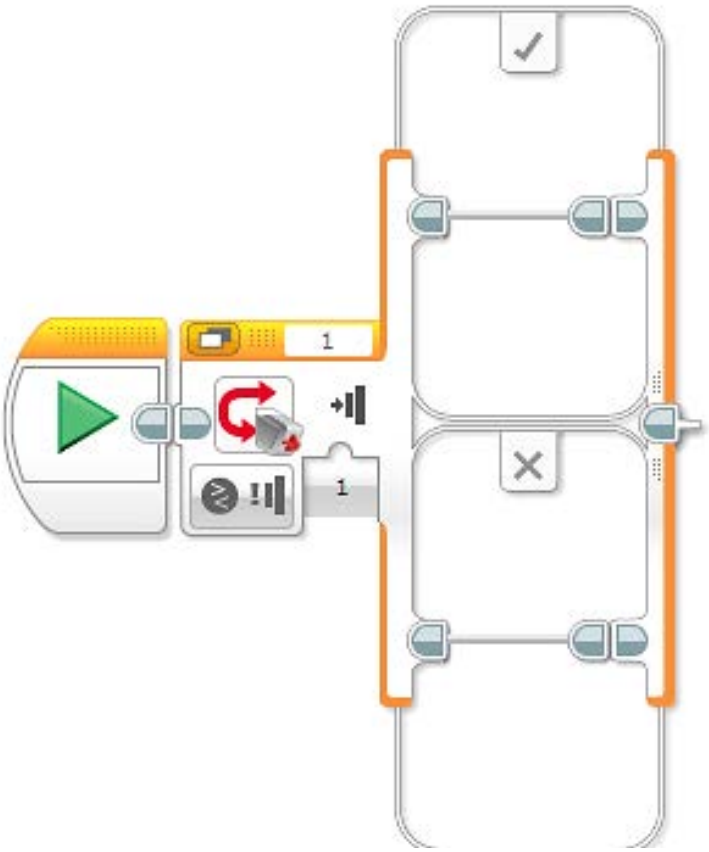
1. Ensiksi tehdään testirata mustalla 1-2 cm leveällä PVC-teipillä (sähkömiesten yleisesti käyttämä eristysteippi, löydät sitä mm. rautakaupasta sähkötarvikeosastolta).

Radan voi merkitä suoraan vaalealle lattialle, valkoiselle kartongille (sopivaa suojakartonkia saa rautakaupasta), tai vaikka vahvan lahjapaperin kääntöpuolelle.

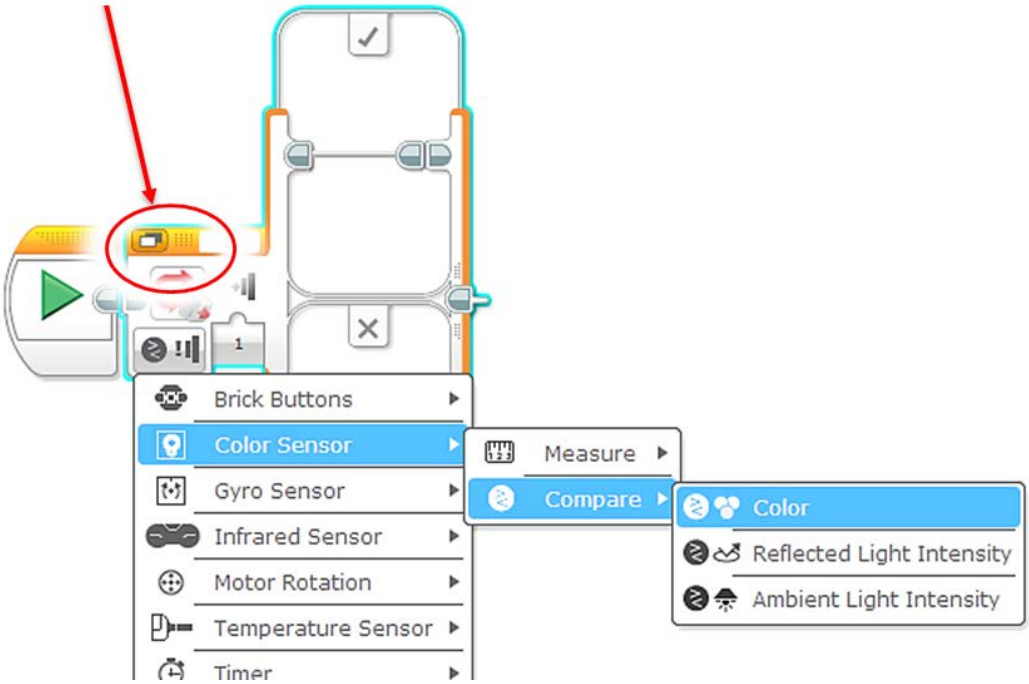
2. Seuraavaksi tehdään **Toimintakuvaus.**
   Robottimme on tarkoitus seurata mustan viivan vasenta reunaa.
   ? Mitä robotin pitää tehdä, jos valokenno näkee mustaa?
   Sen tulee kaartaa vasemmalle. Entä kun se ei näe mustaa?

Ohjelman pitää siis valita (vasemmalle vai oikealle) valokennon tuottaman tiedon perusteella. Tämä voidaan tehdä If-Then –toimilohkolla (Switch), jonka löydät toimilohkojen kirjastosta oranssilta välilehdeltä.

Ohjelmoidaan viivanseuraaja. Aloitetaan työ hakemalla työpöydälle If-Then –toimilohko (Switch).

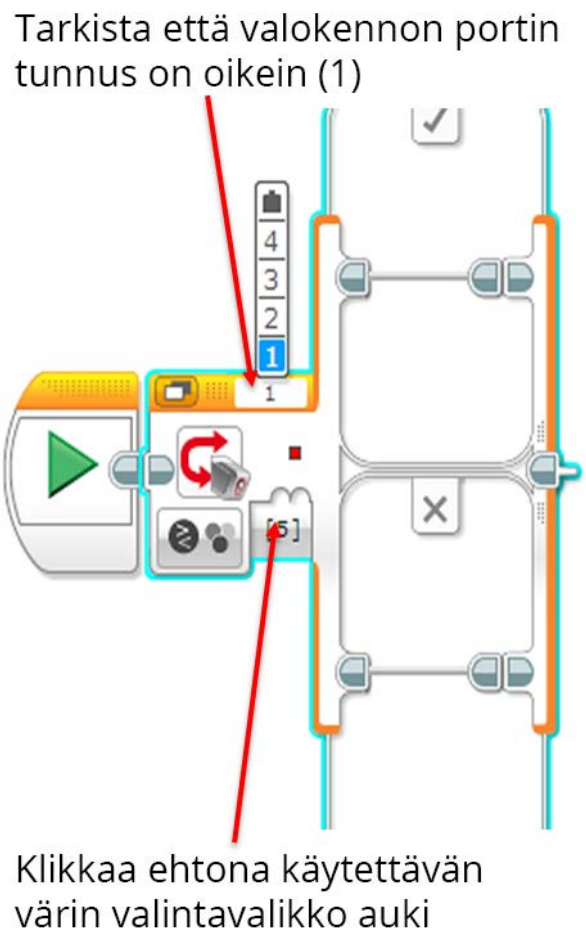


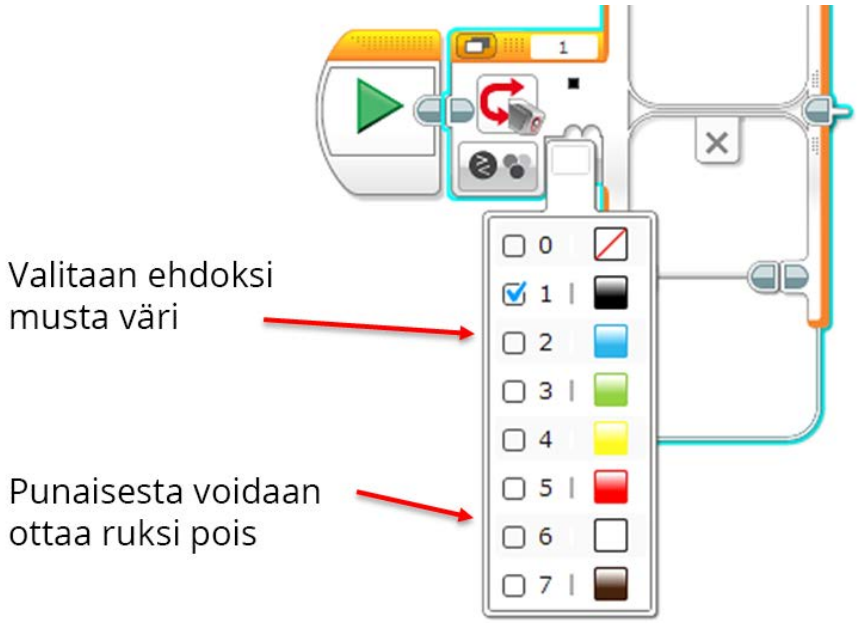
Klikkaa auki valinnan ehdon määrittely, toimintatapa

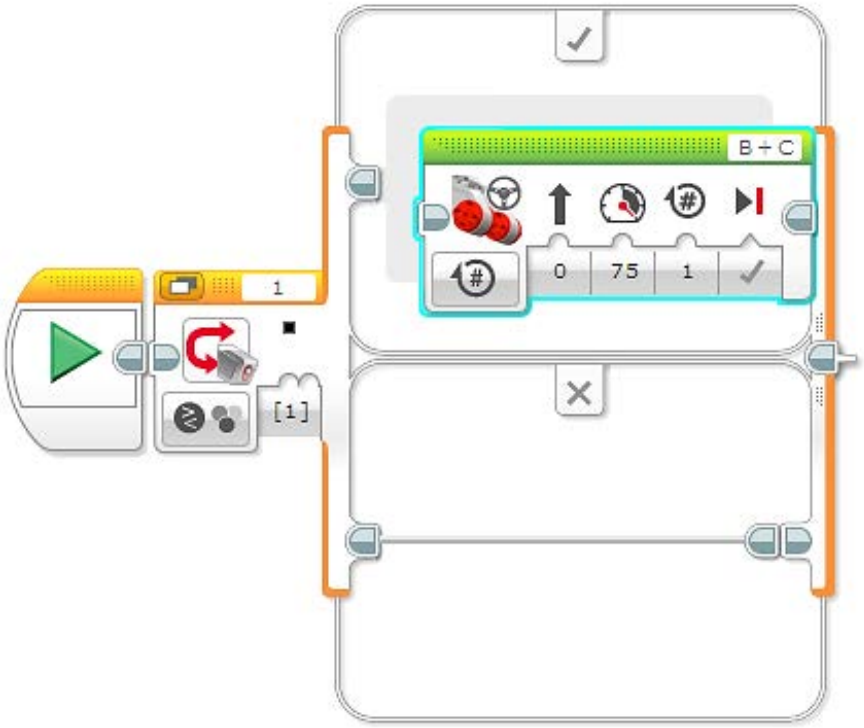


Valitse: Color Sensor (Väri- ja valokenno) -> Compare (Vertaa) -> Color (Väri)

Tarkista että valokennon portin tunnus on oikein (1)







Seuraavaksi lisätään If-Then –toimilohkon sisään, sen ylemmälle polulle yksi Move Steering –toimilohko. Ohjelman suoritus kulkee ylempää polkua pitkin kun ehto toteutuu, eli valokenno näkee jotakin mustaa.

Vinkki: Kun viet If-Then –toimilohkon ylle Move Steering toimilohkon, ohjelma kasvattaa tilan automaattisesti tarpeeksi isoksi.

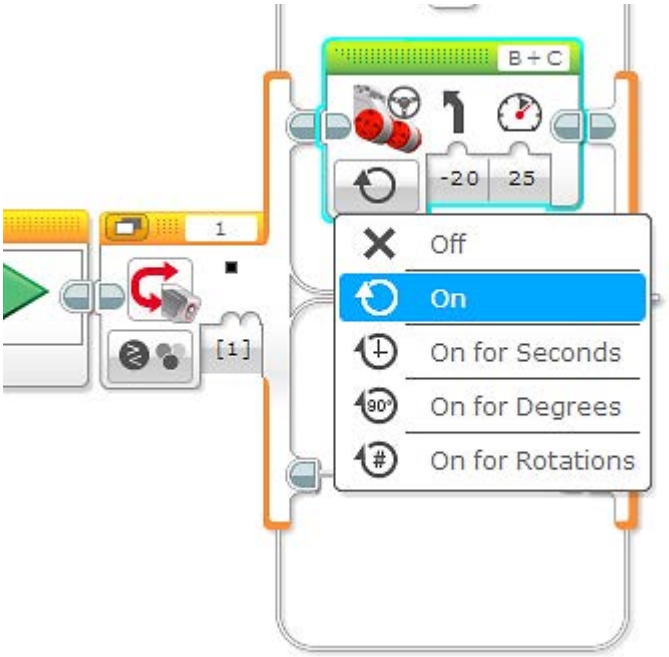
Vinkki: Alla olevassa kuvassa näkyy miten lisättävän toimilohkon taakse ilmestyy ”varjokuva”, kun viet sen lähelle ohjelmaan jo kytkettyä toimilohkoa. Varjon näkyessä voit vapauttaa toimilohkon hiiren otteesta ja ohjelma ohjaa sen tarkasti varjon

Seuraavaksi mietitään miten Move Steering –toimilohkon asetukset, eli parametrit tulisi valita. Käytä tässä apuna kirjan lopusta löytyvää toimilohkon esittelyä.

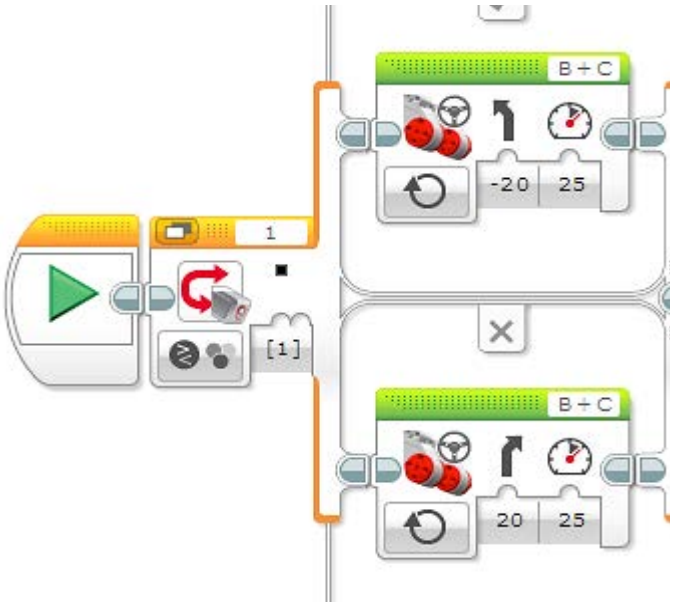
Vinkki: Moottorien tulisi käydä keskeytyksettä ja nopeudesta kannattaa ottaa aluksi pois ihan reilusti. Jos ”rattia” käännetään liikaa, robotista tulee rauhaton. Ja jos rattia käännetään liian vähän, se ei käänny riittävästi (tai nopeutta on liikaa).

If-Then –toimilohko suorittaa ylemmän polun toimilohkot, käskyt, kun annettu ehto täyttyy (valokennon edessä on jotakin mustaa). Muuten ohjelman suoritus kulkee alempaa polkua pitkin.

- Valitaan toimintatavaksi ”On”, joka käynnistää moottorit. Käyntiaikaa, tai kuljettavaa matkaa ei määritellä.
- Ohjataan vasemmalle, maltillisesti.
- Nopeus on laskettu 25:een.

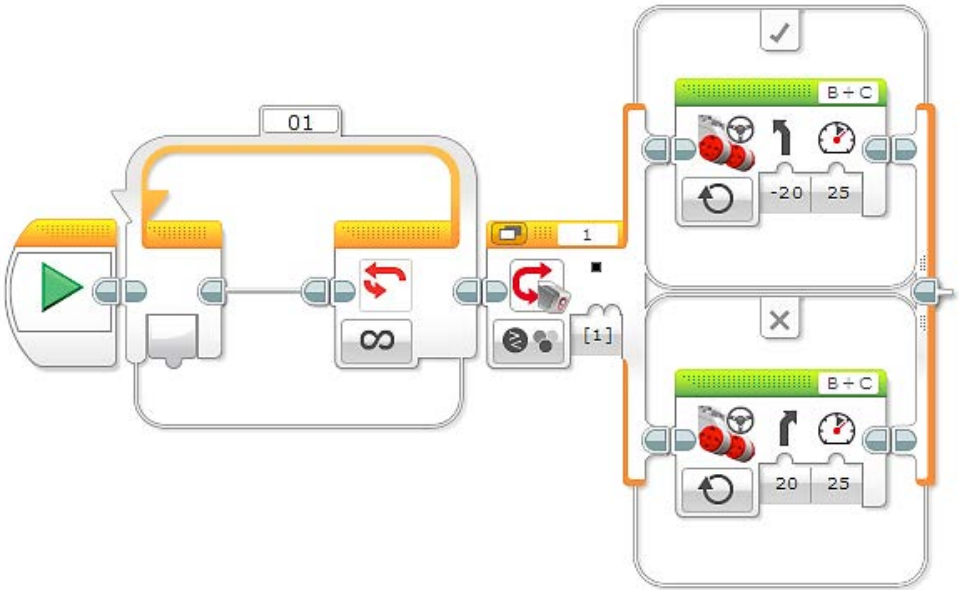


Lisätään If-Then –toimilohkon alemmalle polulle (kun valokenno ei näe mustaa) muuten samanlainen Move Steering –toimilohko, mutta tämän tulee kaartaa oikealle



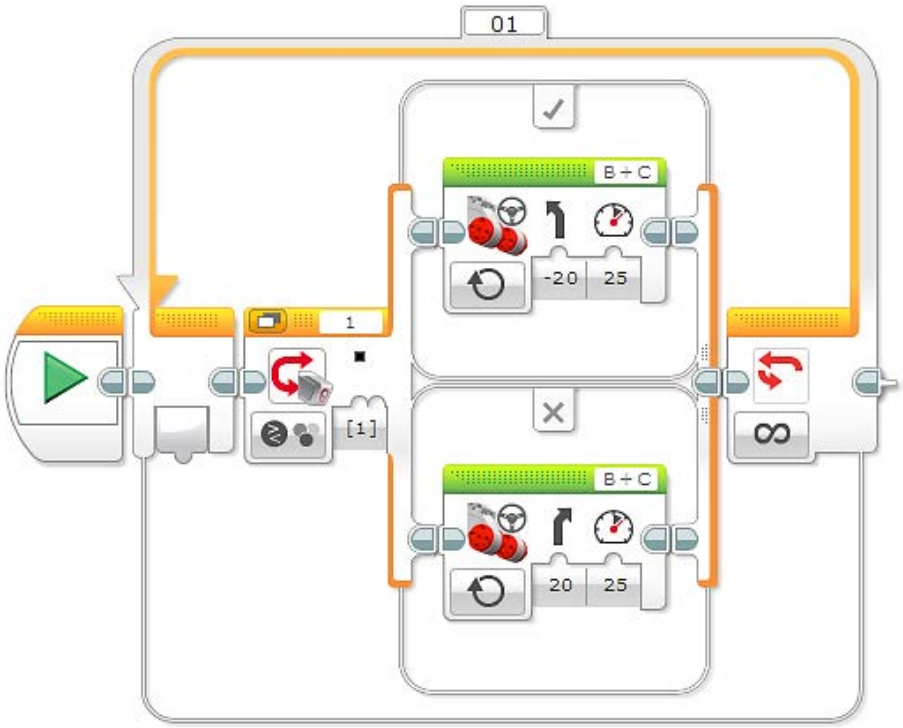
Ohjelma näyttää nyt suunnilleen tältä, mutta se ei vielä toimi kuin yhden sekun-nin murto-osia kestävän suorituskerran verran.

Miten robotti saadaan toistamaan tätä lyhytä ohjelmanpätkää esim. minuutin ajan?

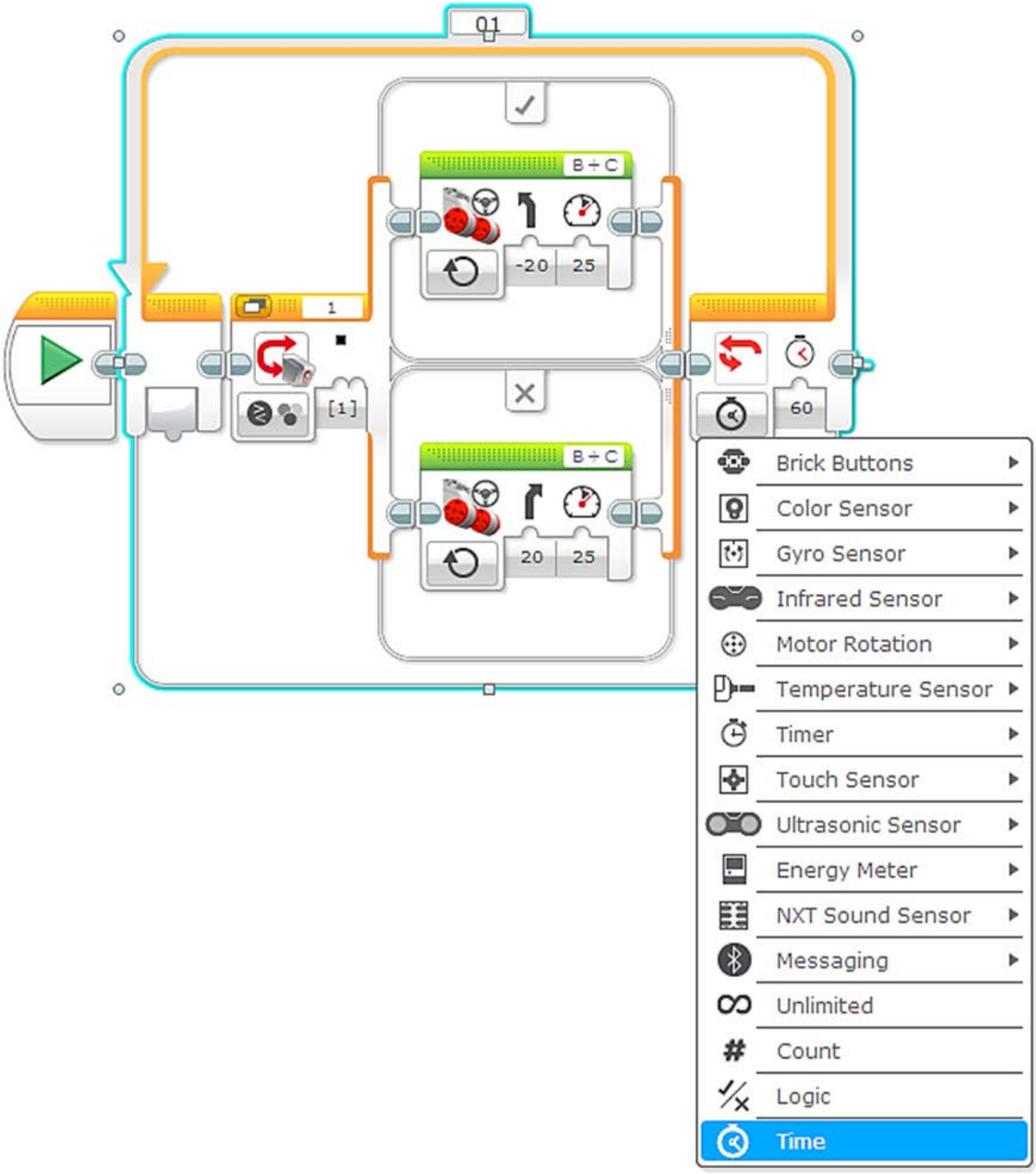


Lisätään ohjelmaan Silmukka –toimilohko (Loop).

Seuraavaksi ohjelma pitää saada silmukan sisään. Se tapahtuu tarttumalla hiirellä If-Then –toimilohkon otsikkokentästä kiinni (nuolen osoittamasta kohdasta) ja hinaamalla se silmukan sisään.



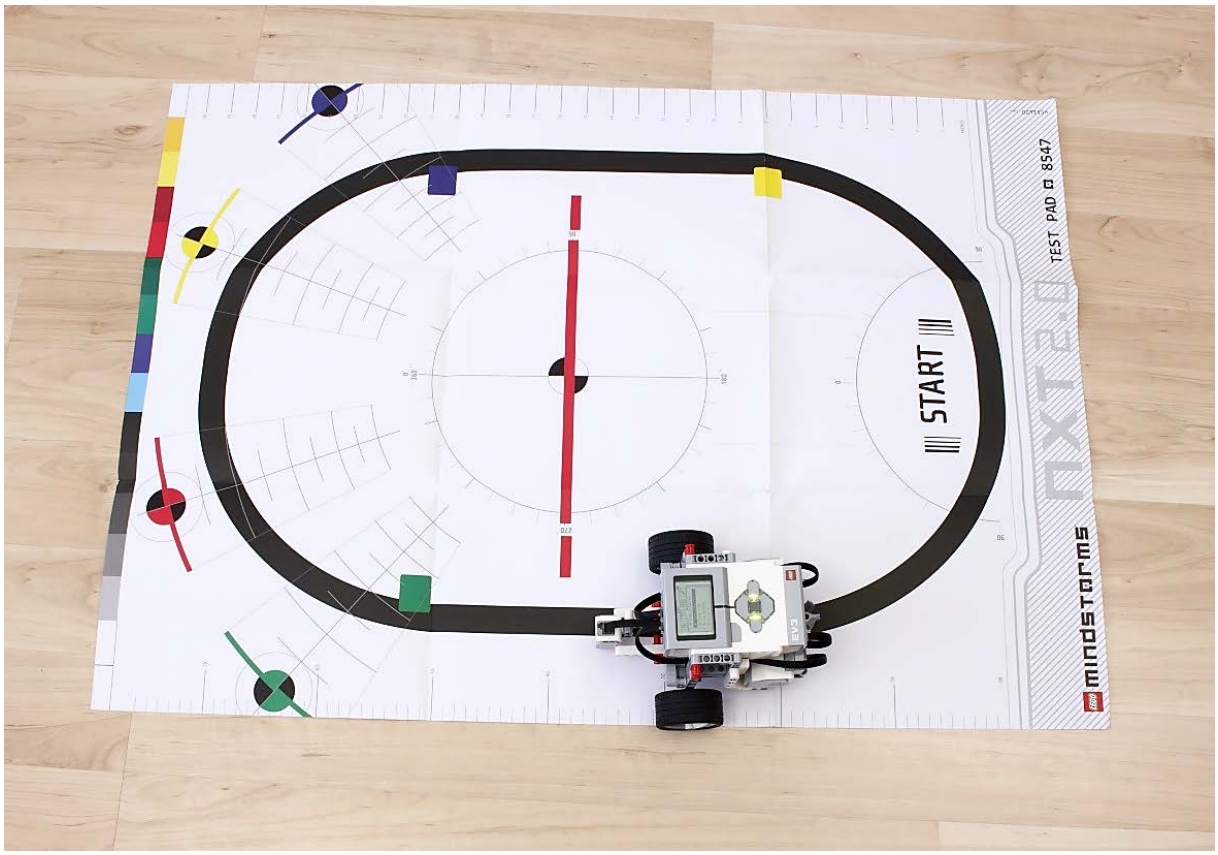
Lopuksi voidaan vielä asettaa ehto silmukasta poistumiselle (1 minuutti).



Asetetaan ehdoksi **Aika** (Time) ja **kestoksi 60s**.

Siirrytään kokeilemaan robottia testiradalle. Tässä kohdassa tarvitaan varmasti vielä pientä hienosäätöä, paljonko sitä ”rattia” voidaan kääntää ja miten kovaa kannattaa ajaa.

Robotin tulisi nyt seurata mustan viivan vasenta reunaa hiukan mutkitellen.



Esimerkki valmiista testiradasta. Tämän mallin hyviä puolia ovat sen siirrettävyys ja käyttöönottamisen helppous.

3. Graafisen ohjelmoinnin perusteet

Tähän mennessä on rakennettu robotti ja kokeiltu sen ohjaamista. Ennen seuraavia tehtäviä on syytä käydä vielä läpi muutamia perusasioita.

Työpöydästä

- Ohjelmatiedostojen hallinta, Projektit ja Ohjelmat
- Työpöydän komennot, katseluikkuna
- Laitemonitori

Ohjelman kirjoittamisesta

- Ohjelman kulku, Polku ja rinnakkainen suoritus
- Toimilohkot ja niiden Parametrointi
- Laiteportin valinta
- Tietotyypit ja tietojohtimet
- Hyvät ohjelmointikäytännöt

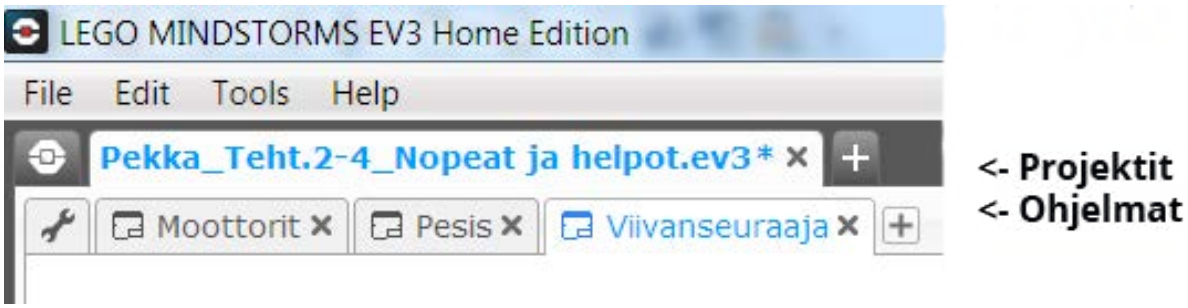
Laitteista

- Servomoottori ja Pulssianturi

Koska tähän lukuun ei sisälly varsinaisia tehtäviä, esiteltäviin asioihin kannattaa sitäkin suuremalla syyllä perehtyä myös kokeilemalla.

Ohjelmatiedostojen hallinta, Projektit ja Ohjelmat

EV3:n ohjelmatiedostorakenteeseen kuuluu kaksi tasoa, Projektit ja Ohjelmat.

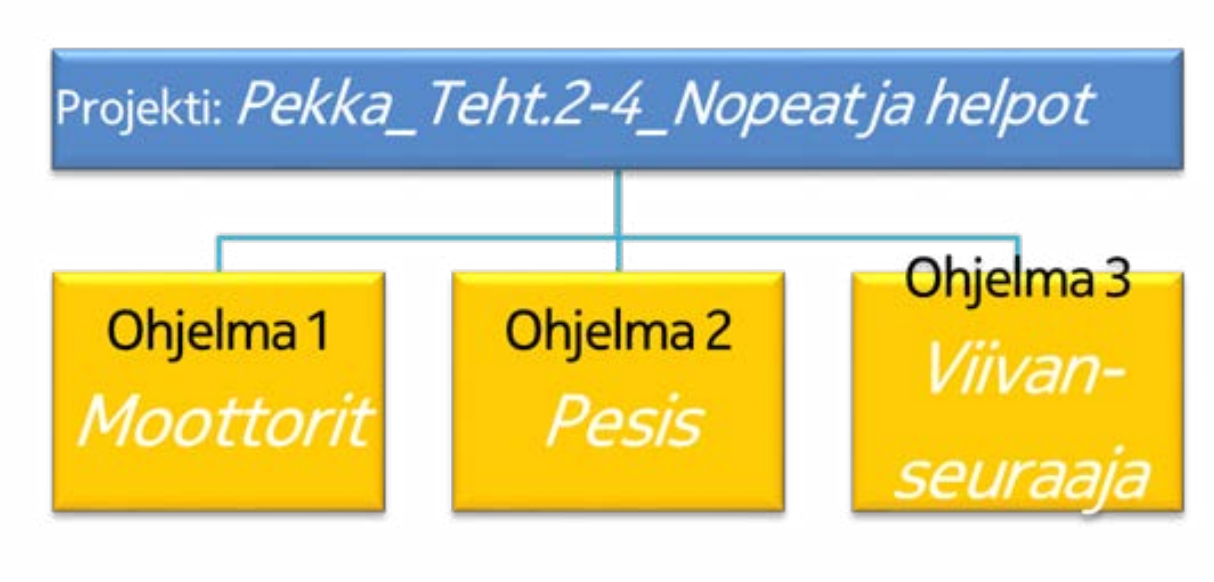


Projekti on ylätaso joka voi pitää sisällään useita ohjelmia, sekä erilaisia projektiin liittyviä muita tiedostoja, kuten kuvia tai musiikkia.

Tässä järjestelmässä ohjelma on aina projektin osa. Siksi ohjelman tallentaminen tehdään tallentamalla projekti ja vastaavasti tallennettu ohjelma avataan avaamalla projekti, johon se kuuluu.

Nimeäminen

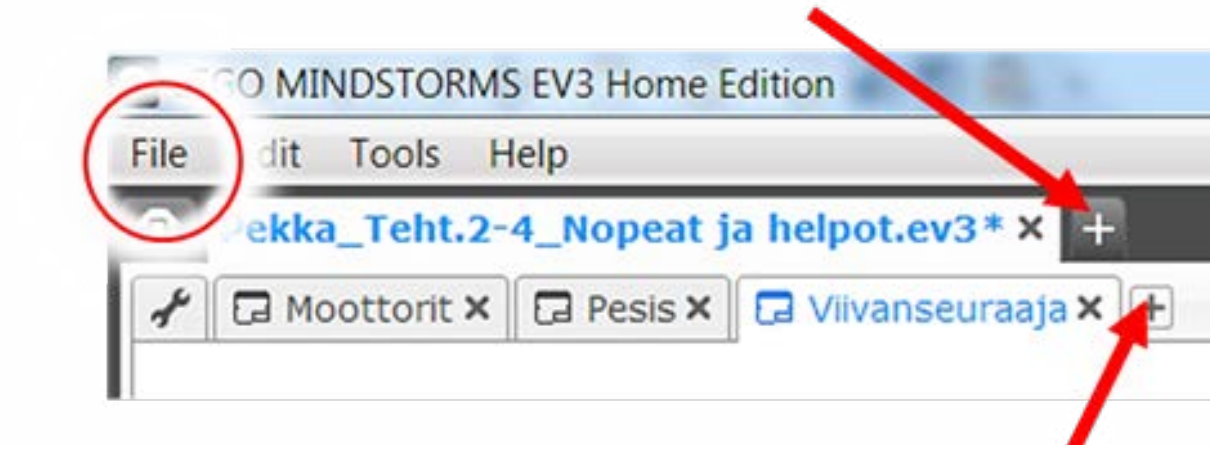
Jos ohjelmointiin käytettävällä tietokoneella on monia käyttäjiä, projektin nimeen on syytä lisätä jokin käyttäjän tunniste. Esimerkiksi oma etunimi, tai nimikirjaimet



(lisäksi pitää sopia, mihin ohjelmat voi tallentaa).

Uusi projekti

Helpoin tapa perustaa uusi projekti on klikata Projektit-rivin perässä olevaa plus-merkkiä (+). Työpöydällä voi pitää yhtä aikaa avoinna useita projekteja. Tämä on kätevää, jos esimerkiksi halutaan kopioida jonkin ohjelman osa vanhasta projektista uuteen projektiin (Copy -> Paste).



Uusi ohjelma

- Klikkaa Ohjelmat-rivin perässä olevaa plus-merkkiä **(+).**
- Nimeä ohjelma, tuplaklikkaa ohjelman nimeä ja kirjoita (ei ääkkösiä!)
- **Huom!** Ohjelman nimen maksimipituus on 26 merkkiä, sallitut merkit ovat: abcdefghijklmnopqrstuvxyzABCDEFGHIJKLMNOPQRSTUVXYZ -_1234567890

Ohjelman (Projektin) tallentaminen

- Uusi projekti -> klikkaa **File** (Tiedosto), valitse **Save As** (Tallenna Nimellä).
- Jo aiemmin tallennettu projekti -> klikkaa **File** (Tiedosto), valitse **Save** (Tallenna).
- **Yhteiskäyttökoneita käytettäessä tallentamisesta on sovittava (tikku/pilvi/...)**
- **Huom!** Projektin nimen maksimipituus on 32 merkkiä, sallitut merkit ovat: abcdefghijklmnopqrstuvxyzABCDEFGHIJKLMNOPQRSTUVXYZ -_1234567890

Tallennetun ohjelman (Projektin) avaaminen

- Klikkaa **File** (Tiedosto), valitse **Open Project..** (Avaa Projekti) ja hae projekti

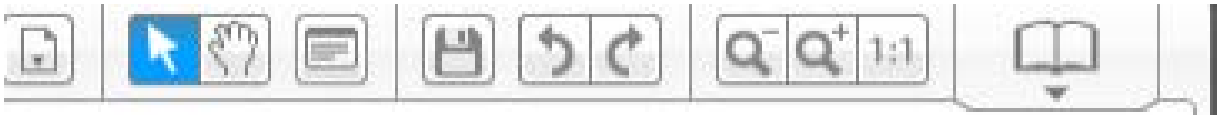
jonka haluat avata.

Ohjelmakirjasto

- Ohjelmoinnissa pyritään aina hyödyntämään kaikkea jo aiemmin tehtyä ja hyväksi havaittua koodia. Tämän kirjan ohjelmista saat omalle ohjelmakirjastolle hyvän alun. Keräämällä aina tilaisuuden tullen käteviä pikku ohjelmanpätkiä kirjastoksi, rakennat itsellesi arvokkaan apuvälineen.

Työpöydän komennot, katseluikkuna

Ohjelmointitaitojen myötä ohjelmiin tekee mieli tehdä lisää ja älykkäämpiä ominaisuuksia. Ohjelmat sisältävät enemmän käskyjä, kohta kaikki toimilohkot eivät mahdu näyttöruudulle. Ohjelmoinnin työskentelytila on onneksi paljon suurempi, kuin mitä näyttöruudun ikkuna.



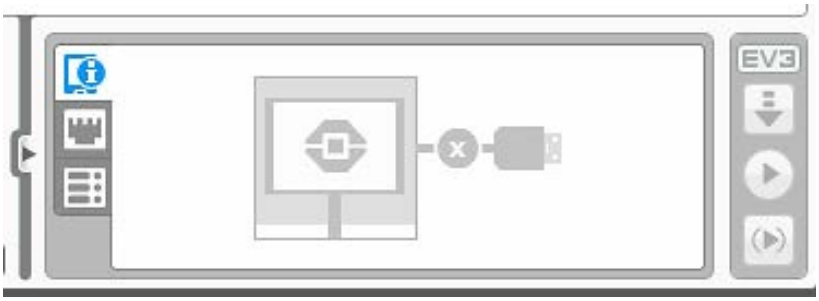
Työpöydän oikeassa ylänurkassa on työpöydän komentojen valikko.

	Valintapainike: Projektin tiedostojen hallinta / Työpöytä
	**Ohjelmointimoodi**
	**Näyttöikkunan siirto-moodi** (panorointi)
	**Kommenttikentän** lisääminen
	**Tallenna** projekti (Save)
	**Peruuta** / Tee uudestaan
	Katselunäkymän **Zoomaus**
	Avaa ohjelman esittely-ikkuna, projektikohtaiset muistiinpanot
	Jos katseluikkunan reunan takana on jotakin piilossa, reunan viereen ilmestyy pieni harmaa kolmio.

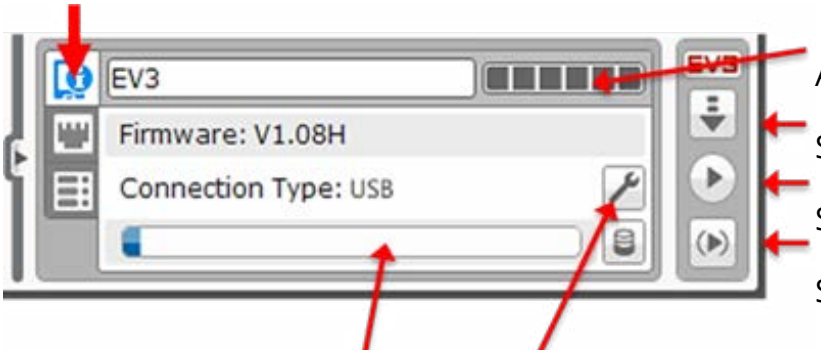
Katseluikkunaa voi siirtää (panoroida) näitä kolmioita klikkaamalla, näppäimistön nuolinäppäimillä, rullahiiren rullalla (ylös/alas), sekä hiirellä kun näyttöikkunan siirto-moodi on valittu käyttöön.

Laitemonitori

Laitenäkymä, kun laitetta ei ole kytketty tietokoneeseen



Laitenäkymä



Akun lataus

Siirrä ohjelma

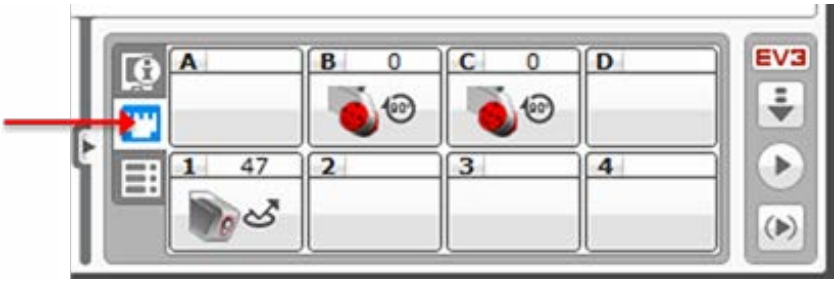
Siirrä ja käynnistä

Suorita osa ohjelmasta

Muistia, käytetty/vapaa

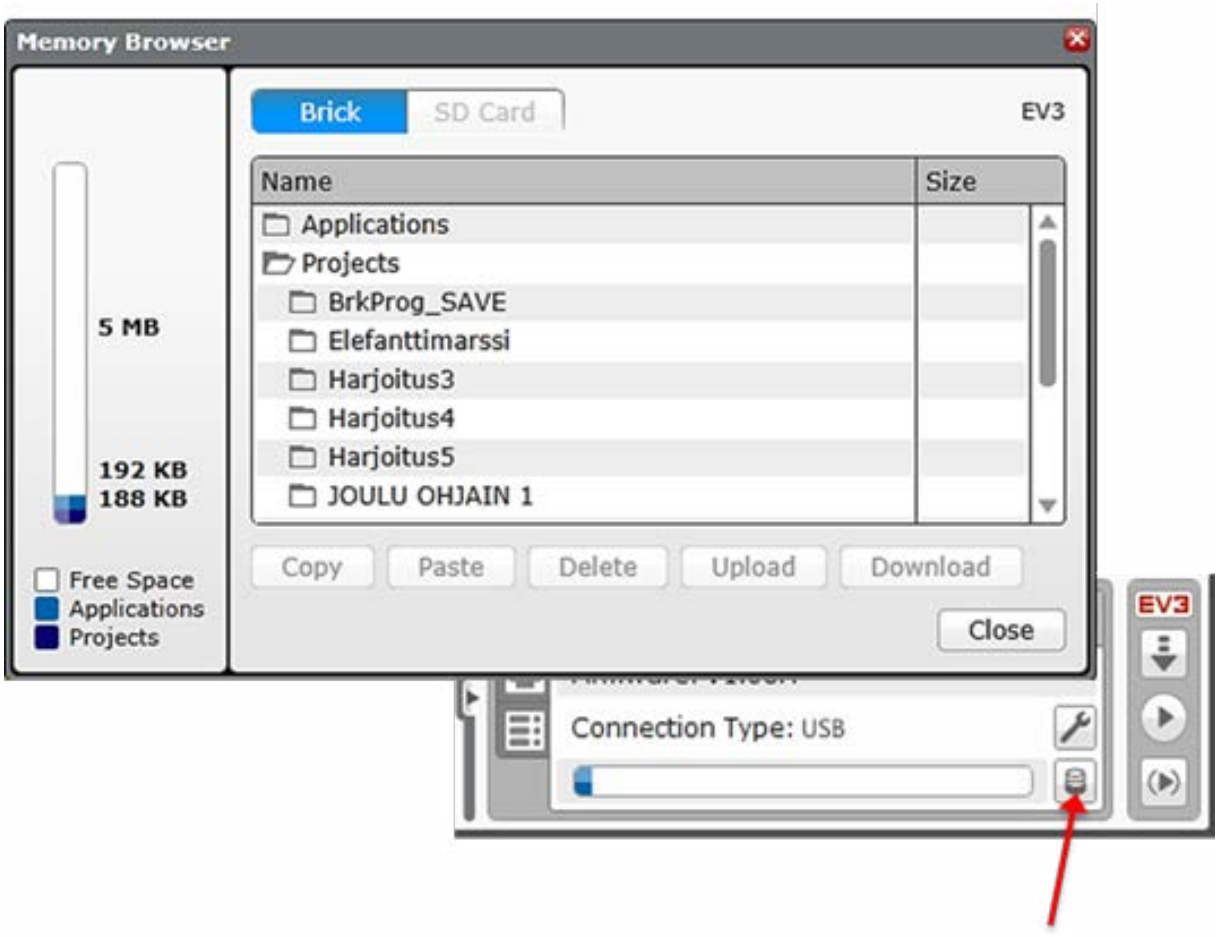
Langattomien yhteyksien asetukset

Porttinäkymä, kytketyt laitteet ja niihin liittyvät mittaustiedot



Yhteysnäkymä

Ohjaimen tiedostojenhallinta



Ohjaimen tiedostojenhallinta avautuu klikkaamalla nuolen osoittamaa kuvaketta.

EV3 ohjaimessa on omaa muistia 16MB, mikä riittää varsin pitkälle. Ennalta ladattut esimerkkiohjelmat, ääni ja kuvatiedostot vievät tästä 4 MB, mutta ne voidaan tarvittaessa poistaa, sekä palauttaa (Firmware update).

Jos tämä muisti ei riitä, ohjaimen sivusta löytyy paikka mikro SD-kortille, max 32 GB.

Ohjelman kulku, Polku ja rinnakkainen suoritus

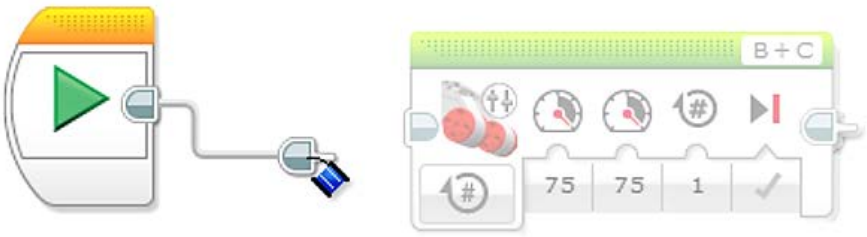
”Polku vie suunnistajaa rastilta rastille,
ohjelmaa käskyltä käskylle.”

Toimilohkoja voidaan kytkeä yhteen myös viivoilla EV3:n ohjelmoinnissa. Tietotekniikan yleiskielisissä (englanti) kirjoituksissa ohjelma-ja toimilohkoja toisiinsa kytkevistä viivoista käytetään usein sanaa wire, joka voidaan tulkita langaksi, johtimeksi, tai vaijeriksi. Suomenkielistä sanaa polku voidaan kuitenkin pitää tässä asiayhteydessä vähintään yhtä kuvaavana.

Polku on ohjelmassa yksisuuntainen reitti, jolla voidaan yhdistää kaksi ohjelmalohkoa, jota pitkin ohjelman suoritus kulkee. Joskus polku haarautuu, mutta aina sillä on alku ja loppu. EV3 pystyy suorittamaan useamman rinnakkain kulkevan polun käskyjä yhtä aikaa.

Polun piirtäminen

Vie hiiren osoitin ohjelmalohkon oikeassa reunassa olevan ”liittimen” päälle (hiiren osoitin vaihtuu lankarullaksi). Paina hiiren vasen painike alas ja vedä lankarulla ohjelmassa seuraavaksi suoritettavan lohkon vasemman reunan liittimeen. Tämä vaatii tarkkaa kättä. Liitos syntyy vasta kun liitin vaihtaa värinsä siniseksi.



- Polun saa pois klikkaamalla sen loppua.
- Ohjelmalohkot saa siirtymään yhteen ja niiden välissä olleen polun katoamaan klikkaamalla polun alkua.

Vinkki: Ohjelmaa voi kirjoittaa riveittäin, kuin tekstiä, piirtämällä aina polku edellisen rivin lopusta uuden rivin alkuun.

Toimilohkot ja niiden parametrointi

”Ohjelma kertoo laitteelle mitä sen pitää tehdä, ja miten”

Tyypillisesti ohjelma on jonoon laitetuja käskyjä, toimilohkoja, joita järjestyksessä suorittamalla laite saa tehdyksi sille asetetut tehtävät. Ohjelmassa voi olla osia, joita toistetaan useita kertoja. Siinä voi olla myös vaihtoehtoisia tehtäviä ja joskus sitä pyydetään suorittamaan useampaa käskyjonoa rinnan, saman-aikaisesti.

EV3:n ohjelmointi perustuu graafisiin kuvakkeisiin, joita kutsutaan toimilohkoiksi. Toimilohkoja käyttämällä me käskemme sitä tekemään erilaisia asioita. Käytettävissä on yhteensä noin 40 erilaista toimilohkoa, ja tarvittaessa ohjelmoija voi suunnitella vielä lisäksi omia erikoistoimilohkoja.

Toimilohkon avulla voimme käskeä vaikka robottia käynnistämään moottorin. Tämän lisäksi me yleensä haluamme myös kertoa robotillemme miten kauan tämän moottorin tulee pyöriä / milloin sen tulee pysähtyä. Varmasti tärkeitä tietoja ovat myös millä nopeudella me haluamme moottorin pyörivän, sekä tulisiko sen pyöriä eteen, vai taakse päin. Kaikkia tällaisia tietoja kutsutaan parametreiksi.

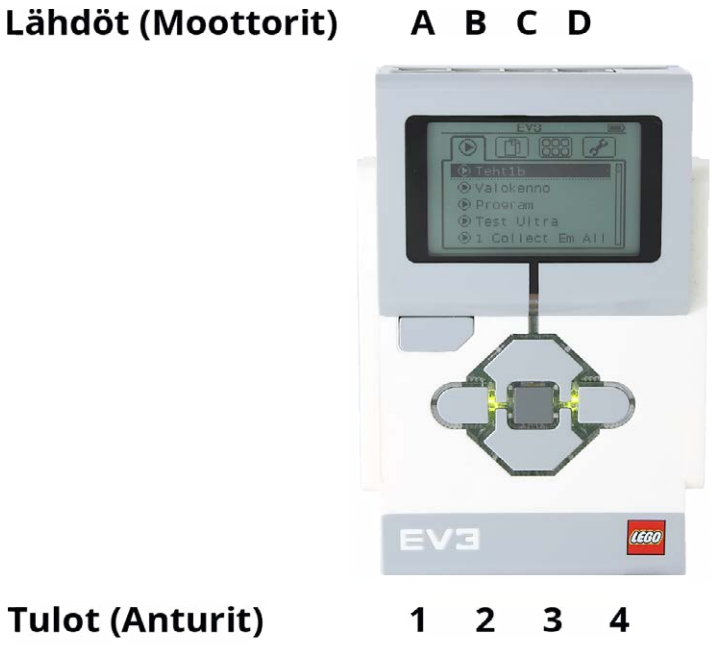
Toimilohko määrää MITÄ robotti tekee,

Tämä toimilohko ohjaa ohjaimen D-porttiin kytkettyä moottoria

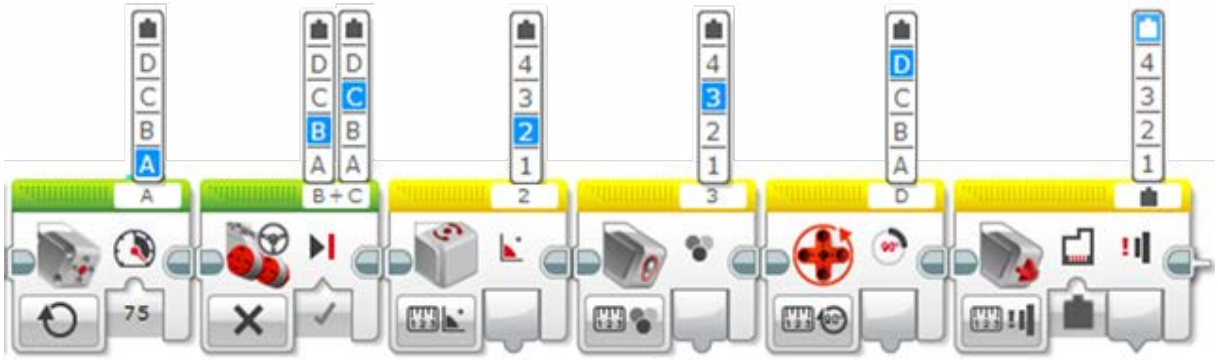
alareunan Parametrit määräävät, MITEN

(Yksi kierros myötäpäivään nopeudella 75%, lopuksi laitetaan jarru päälle)

Laitetunnuksen asettaminen



Moottorien ja anturien ollessa kytkettynä ohjaimeen, sekä ohjaimen ollessa kytkettynä tietokoneeseen ohjelmointiohjelma osaa ehdottaa toimilohkolle laitetunnusta. Mutta monestakin syystä tämä ei ole aukotonta. Siksi jokainen laitetunnus on silti syytä tarkastaa.



Laitetunnuksen asettaminen

Tietotyypit ja tietojohtimet

Parametritietojen ulkoiset kytkennät, eli toimilohkojen tulot ja lähdöt

Lähes kaikkien toimilohkojen kuvakkeiden alareunassa on ”laatikoita” parametrien syöttöä varten. Nämä ovat toimilohkon tuloja ja tuloiksi ne voi tunnistaa myös niiden **yläreunassa** olevista kolmion, neliön, tai puoliympyrän muotoisista ulokkeista (tulo=Input).

Monet toimilohkot kykenevät myös tuottamaan tietoa muiden toimilohkojen käytettäväksi, lähes samanlaisista laatikoista noudettaviksi. Nämä ovat toimilohkon lähtöjä ja lähdöiksi ne voi tunnistaa myös niiden **alareunassa** olevista kolmion, neliön, tai puoliympyrän muotoisista ulokkeista (lähtö=Output).

Nuo pienet kolmiot, neliöt ja puoliympyrät kertovat ohjelmoijalle, minkä tyyppistä tietoa niistä on saatavilla / niihin voidaan kytkeä.

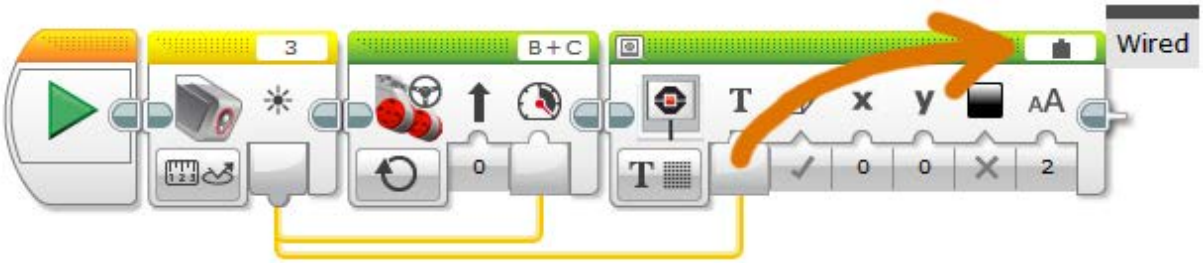
Type	Block Input	Block Output	Block Output Data Wire
Logic			
Numeric			
Text			
Numeric Array			
Logic Array			

Lähde: EV3 ohjelmointiohjelman Data Wires –Helppi sivu

Tietojohtimet piirretään kuten polutkin, lankarullaa hiirellä raahaamalla.

Tietojohtimen poistaminen käy tarttumalla sitä hiirellä sen loppupäästä ja raahaamalla tämä liitin takaisin sinne mistä johdin alkaa. Tietojohtimen väristä ei tarvitse murehtia. Ohjelma valitsee sen automaattisesti.

Samaa tietoa voi myös käyttää useassa paikassa. Toisinaan sen arvo kannattaa taltioida odottamaan käyttömuistiin. Mutta etenkin jos sitä tarvitaan vain muutamassa seuraavassa toimilohkossa, samasta lähdöstä voi vetää tietojohtimet suoraan näihin toimilohkoihin.



Moottorin nopeus määräytyy valokennon mittaaman heijastuvan valon mukaan. Sama tieto esitetään myös ohjaimen näytöllä.

Tietotyypien muunnokset

Vaikka lähdön ja tulon odotetaan edustavan samaa tietotyyppiä, tietyt poikkeukset ovat sallittuja. Niitä ei tarvitse opetella ulkoa, kokeilemalla pärjää. Ohjelmointiohjelma ei anna piirtää kytkentää jolla ei ole edellytyksiä toimia. Eli riittää kun muistaa että toimiviakin yhdistelmiä on.

Lähtö		**Tulo**		**Mitä tuloon kirjoittuu**
Tosi/epätosi (Logic)		Numeerinen		Tosi=1, Epätosi=0
Tosi/epätosi (Logic)		Teksti		Tosi=”1”, Epätosi=”0”
Tosi/epätosi (Logic)		Tosi/epätosi Jono		Yhden elementin mittainen jono
Tosi/epätosi (Logic)		Numeerinen Jono		Yhden elementin mittainen jono, 1 tai 0
Numeerinen		Teksti		Numero tekstinä, esim. ”6.12”
Numeerinen		Numeerinen Jono		Yhden elementin mittainen jono
Tosi/epätosi Jono		Numeerinen Jono		Saman mittainen jono, jonka elementtien arvoina 1 tai 0

Tietotyyppien muunnokset

Hyvät ohjelmointikäytännöt

Yhteistä kaikille hyville ohjelmointikäytännöille on niiden taipumus auttaa meitä kirjoittamaan helppolukuisempia ohjelmia ja välttämään virheitä.

Tee toimintakuvaus

Ajatuksella tehty toimintakuvaus on ohjelmoijan paras kaveri, käsikirjoitus ja muistilista

Ohjelmointitehtävissä kokonaisuus, tehtävä, on syytä purkaa ensin pieniin osiin ja laatia ohjelmasta toimintakuvaus vihkoon. Toimintakuvauksessa käytetään tuttuja, yksinkertaisia tekemistä täsmällisesti kuvaavia sanoja ja ilmaisuja, kuten: mene, käänny, etsi, nosta, työnnä, vedä, paina, tartu, irrota, käännä, pyöritä, ääni, poimi, kerää, kuljeta, etsi mustan viivan reuna, seuraa viivaa, miten pitkään, käänny, miten paljon, paikallaan vai kaartaen, laske tarttuja, nosta tarttuja jne.

Toistuvasti esiintyviin toimintoihin (silmukat/aliohjelmat), vaihtoehtoisiin tapahtumien kulkuihin (ehdollinen suoritus) sekä samanaikaisesti suoritettaviin toimintoihin (ohjelman haarautuminen) on syytä kiinnittää huomiota.

Yhdessä toimintakuvauksen kanssa on syytä kirjoittaa myös mekaniikan suunnittelua silmällä pitäen luettelo asioista, joita robotin pitää kyetä tekemään, sekä robotin rakennetta koskevista rajoitteista. Tästä pitäisi löytyä ainakin, millaisissa paikoista robotin pitää kyetä kulkemaan, mitä apuja sille on tarjolla suunnistamiseen, sekä tietysti kaikki sen tehtäviin liittyvä tieto.

Moottorin pyörimissuunnan asetus

Pyörimissuunnan kääntäminen käyttämällä miinusmerkkiä vain nopeustiedon yhteydessä on osoittautunut yhdeksi monista hyvistä ohjelmointikäytännöistä.

Miksi?

Moottorin pyörimissuunnan voi kääntää laittamalla (tai ottamalla pois) miinusmerkki joko nopeuden tai matkan asetusarvon eteen. Mutta jos moottorin toiminnoksi on valittu ON (matkaa ei ole määritelty) tai On For Seconds (aika ei kulje taaksepäin), pyörimissuuntaan ei voi vaikuttaa matkan asetusarvon kautta.

Graafinen ohjelmointikieli

Koska graafisia ohjelmointikieliä, kuten EV3:n LabView pohjaista ohjelmointikieltä käytettäessä jää pois mahdollisuus tehdä kaikenlaisia piste, pilkku yms. erikoismerkkien yhdistelmien käyttöön liittyviä virheitä (syntaksi virheet), kuvakkeiden käyttö ohjelmoinnissa on yksi tärkeimmistä hyvistä ohjelmointikäytännöistä. Sitä suositellaan jopa koneenrakenusta ohjaavissa tärkeimmissä kansainvälisissä koneturvallisuuden standardeissa. Mikään ohjelmointikieli ei kuitenkaan ole immuuni huolimattomasti laaditulle toimintakuvaukselle, tai inhimilliselle ajatusvirheelle (konteksti virheet)

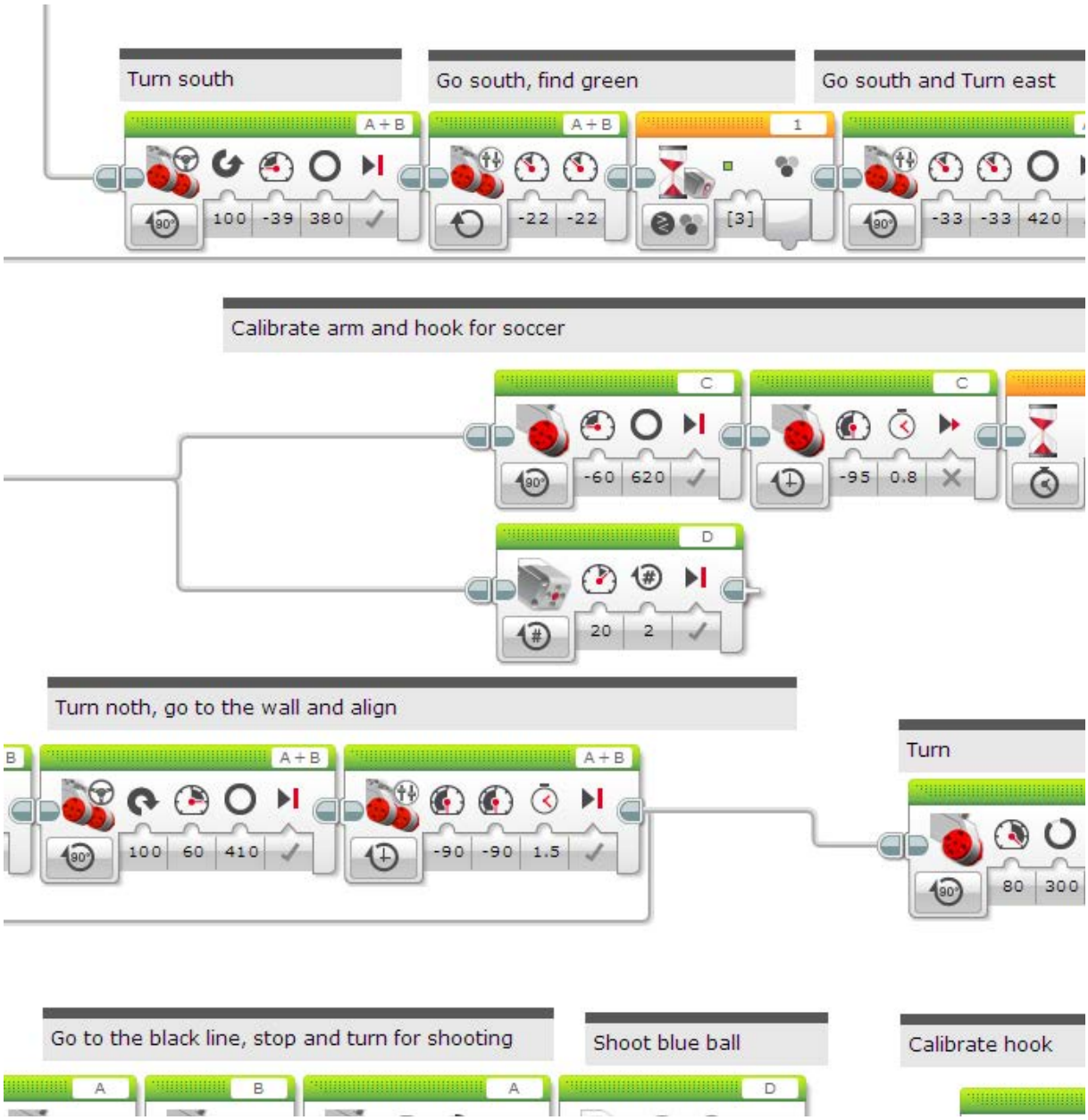
Kätevät näppäinkomennot

Opettele ainakin nämä näppäinkomennot

Kopioi	Ctrl + C
Liitä	Ctrl + V
Valitse kaikki	Ctrl + A
Leikkaa (mukaan)	Ctrl + X

Kommentoi

Kommentointi helpottaa ja nopeuttaa (dramaattisesti) yhtälailla ohjelman hienosäätöjen tekoa, kuin vianhakuakin. Ytimekäs, kattava ja kuvaava kommentointi on ohjelmoijan ylpeys ja edellytys sille että joku muukin voi lukea ohjelmaa ja päästä kärryille sen toiminnasta (ja hienouksista).



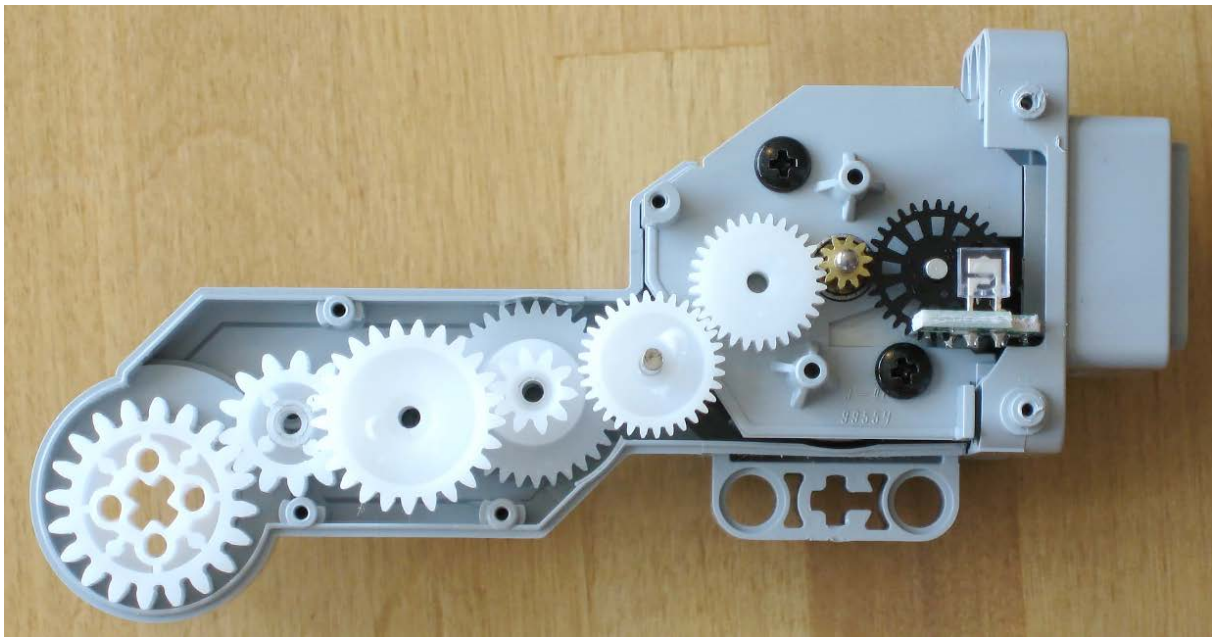
Tallenna

Tallenna usein, aloittaminen uudestaan alusta harmittaa. Tee myös varmuuskopioita hiukan eri nimillä ihan siltä varalta, että edellinen versio olikin parempi ja haluat palata siihen.

Servomoottori ja pulssianturi

EV3:n moottorit ovat servomoottoreita. Servomoottoreita on hyvin monen näköisiä ja kokoisia. Tunnusomaista kaikille servomoottoreille on, että ne kykenevät kertomaan ohjaimelle miten lujaa ne pyörivät, sekä miten monta kierrosta ja astetta ne ovat pyörineet (esimerkiksi pulssianturin tuottaman tiedon avulla).

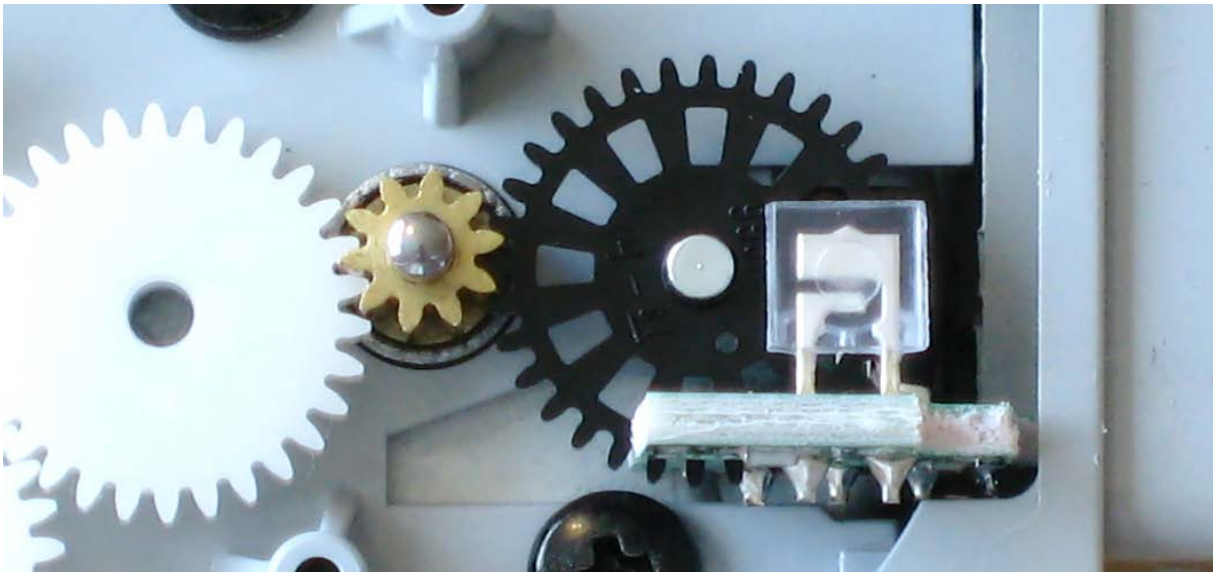
Niiden sisällä on pieni tavallinen sähköllä toimiva leikkikalumoottori ja pulssianturi. Hammaspyörien avulla moottorin pyörimisnopeutta pienennetään sopivaksi. Samalla sen vääntömomenttia, eli voima kasvaa. Nämä moottorit on varustettu ylikuorman suojauksella (lämpörele). Jos moottori lakkaa toimimasta ylikuormituksen seurauksena, anna sen jäähtyä rauhassa ja se toimii taas. Alla on kuva avatusta mootorista.



EV3:n pulssianturiin kuuluu kuvassa näkyvät

1. Musta hammaspyörä, jossa on monta isoa ”ikkunaa”
2. Sen edessä oleva läpinäkyvä lamppu (IR-led),
3. Mustan hammaspyörän takana oleva IR-valon vastaanotin
4. Signaalivahvistin (pintaliitoskomponentit piirilevyn alapinnassa)

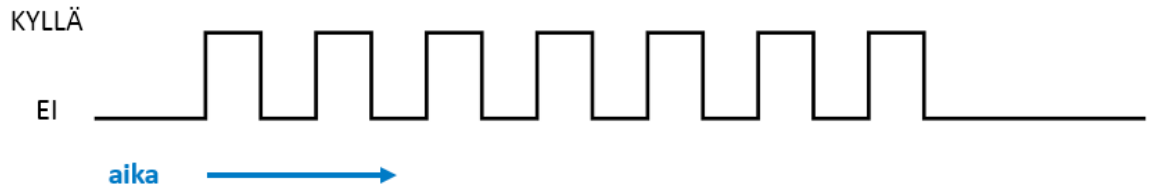
HUOM! Älä avaa itse moottoreita! Ne on rakennetu sillä tavalla ikävästi, että niitä ei pysty kukaan avaamaan rikkomatta ensin peruutamattomalla tavalla tiettyjä tärkeitä osia. Moottorin uudelleen kasaaminen on siten mahdotonta.



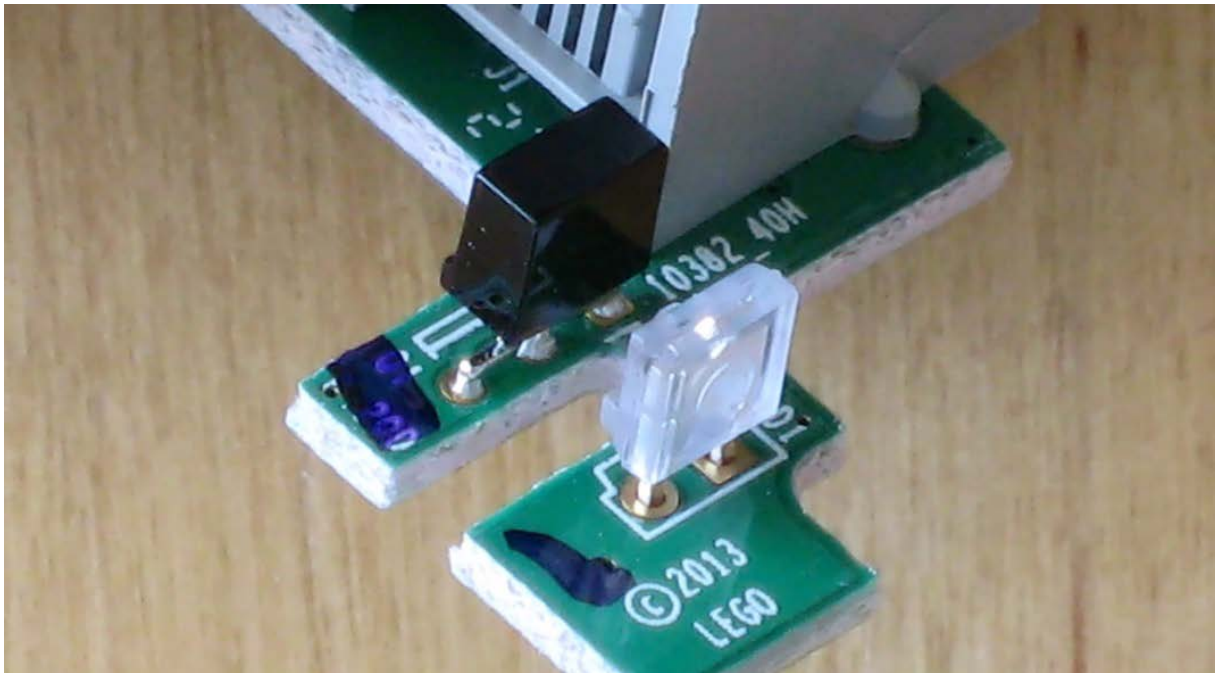
Pulssianturi

Pulssitieto kertoo ohjaimelle milloin mustan hammaspyörän läpi näkyy valoa. Tämän tiedon perusteella ohjain osaa laskea kuljetun matkan ja moottorin nopeuden

Valo näkyy?

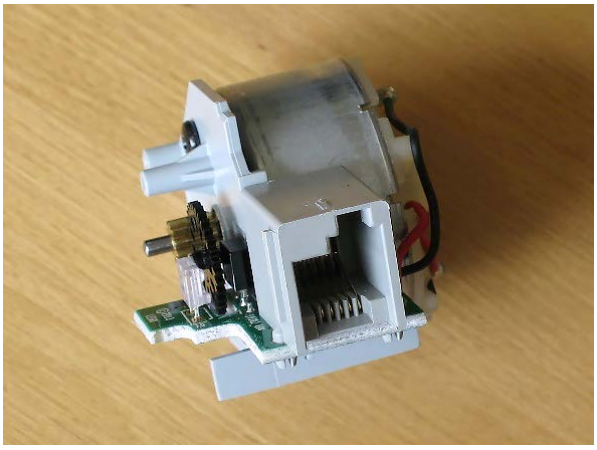


Moottorin pyöriessä ohjaimelle välittyvä signaali on jono lyhyitä pulsseja



IR-valon vastaanotin (tumma, kolme johdinta) ja lamppu, eli IR-ledi (läpinäkyvä osa, kaksi johdinta)

Terästapit ovat katkenneet moottoria avattaessa



Moottoripaketti ja liitin

Keskikokoisesta EV3:n servomoottorista löytyy kaikki samat osat, kuin isostakin. Vähän pienempinä vain. Pulssianturi on rakennettu liittimen ja moottorin väliin, harmaan suojuksen sisään. Vaihteisto on tässä mallissa koteloitu myös omaksi kokonaisuudeksi (se on rakenteeltaan ns. planeettavaihde).

Keskikokoinen EV3:n servomoottori avatuna

4. Liikkeen ohjaaminen, ohjelman silmukat ja haarat

Ensimmäisissä tehtävissä pärjättiin käyttämällä yhtä moottoritoimilohkon tyyppiä. Laajennetaan osaamista tutustumalla myös muihin yleisimpiin tapoihin ohjata moottoreita.
Opetellaan hyödyntämään silmukoita kun jokin käskysarja toistuu useita kertoja sekä tekemään ohjelmaan rinnan suoritettavia käskyjonojen haarautumia.

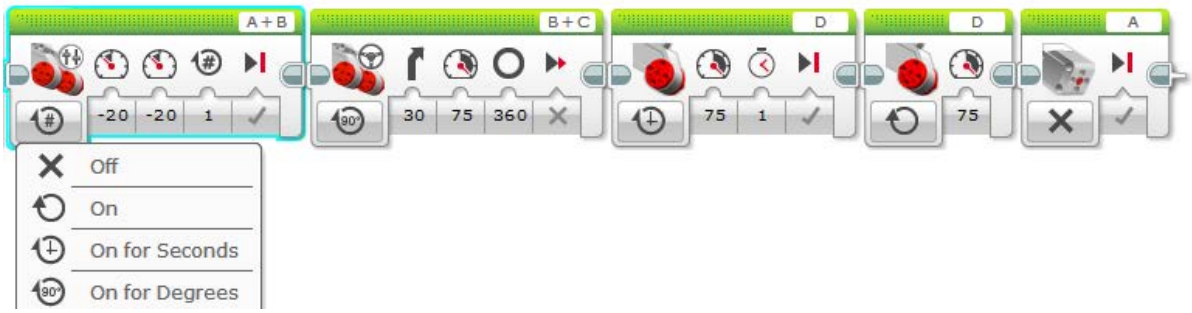
Etkot

Harjoitukset jatkuvat käyttäen ensimmäisessä harjoituksessa rakennettua moni-toimirobottia. Kerrataan moottori-, viive-, sekä silmukkalohkojen parametrointi.

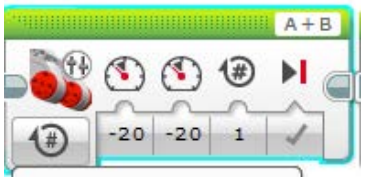
Ohjelmoinnissa käytettävät toimilohkot on esitelty kirjan lopussa. Tämän luvun tehtäviä silmällä pitäen on syytä kerrata niistä ainakin moottorien toimilohkot, sekä ajastin, silmukka ja ehdollinen If-Then –toimilohko.

Intro

Ensimmäisessä harjoituksessa kokeilimme kahden moottorin ohjaamiseen tarkoitettua Move Steering -toimilohkoa. Koska moottoritoimilohkoja tarvitaan tästä eteenpäin melkein aina, käydään niiden käytöä vielä läpi esimerkkien avulla. Seuraavassa kuvassa on esitetty viisi moottorilohkoa.

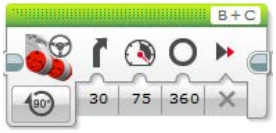


1. Ensimmäisenä on kahdella moottorilla eteen-, tai taaksepäin soveltuva Move Tank moottorilohko. Tässä sille annetaan seuraavat ohjeet: (moodi = **On for Rotations)**



   a. moottoreita A ja B pyörimään synkronoidusti
   b. taaksepäin 20% nopeudella (-20)
   c. yhden kokonaisen kierroksen (1) verran ja
   d. asettamaan lopuksi sähköisen jarrunsa päälle (✓),
   e. ennen ohjelman suorituksen siirtymistä eteenpäin.

2. Toisena on käännöksiin ja kaarroksiin soveltuva **Move Steering** moottorilohko. Tässä sille annetaan seuraavat ohjeet:



(moodi = **On for Degrees**)

a. robottia kaartamaan 30% jyrkästi (30)

b. käyttäen moottoreita B ja C synkronoidusti,

c. suhteellisella nopeudella 75% (75),

d. nimellisesti 360 asteen (360) verran ja

e. jäämään lopuksi vapaapyörintä –tilaan (✗),

f. ennen ohjelman suorituksen siirtymistä eteenpäin.

3. Kolmantena on yksittäisen ison moottorin ohjaamiseen käytettävä **Large Motor** moottorilohko. Tässä sille annetaan seuraavat ohjeet:

(moodi = **On**)

a. moottoria D pyörimään

b. eteenpäin 75% nopeudella (75)

c. yhden sekunnin (1) ajan ja

d. asettamaan lopuksi sähköisen jarrunsa päälle (✓),

e. ohjelman suoritus siirtyy välitömästi eteenpäin.

4. Neljäntenä on samanlainen **Large Motor** moottorilohko, kuin edellä. Nyt sille annetaan seuraavat ohjeet:

(moodi = **On for Rotations**)

a. moottoria D pyörimään

b. eteenpäin 75% nopeudella (75)

c. määräämättömän ajan verran.

d. Ohjelman suorituksen siirtymistä eteenpäin.

5. Viidentenä on keskikokoisen moottorin ohjaamiseen käytettävä Medium Motor moottorilohko (pienempi moottori). Tässä sille annetaan seuraavat ohjeet: (moodi = Off)

a. moottoria A pysähtymään välittömästi

b. asettamaan lopuksi sähköisen jarrunsa päälle (✗),

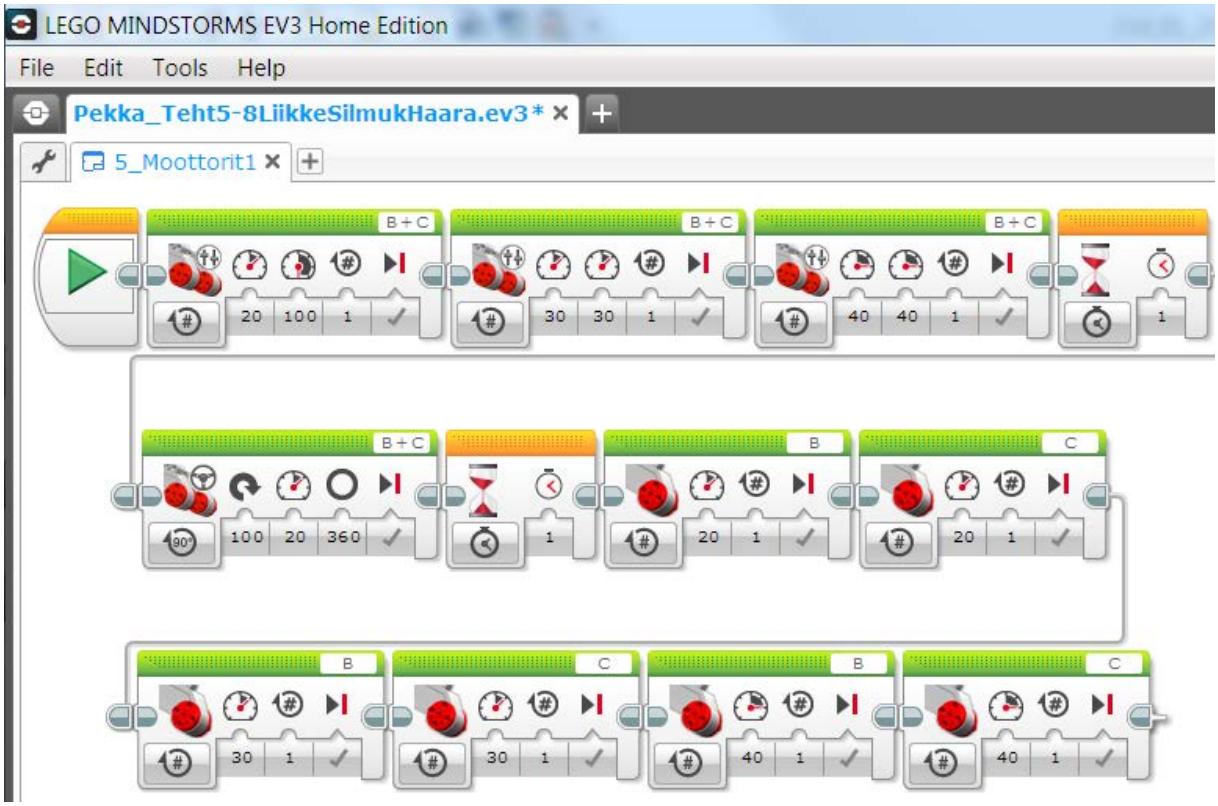
c. ohjelman suoritus siirtyy välitömästi eteenpäin.

Vinkki: Erillisten moottoritoimilohkojen käyttäminen on monasti kätevää, mutta se hidastaa aavistuksen itse ohjelman suorittamista. Tämä tosin tulee esille vasta äärimmäistä suoritusta edellyttävissä sovelluksissa, kuten kilparoboteissa. Kaikessa ohjelmointityössä kannattaa silti aina pyrkiä ”keveisiin”, nopeasti prosessorissa pyöriviin ratkaisuihin.

Lisää tietoa toimilohkoista löydät kirjan lopusta.

Tehtävä 5. Ladies and Gentlemens, Start your engines!

Tutustutaan moottorilohkojen toimintaan. Tehdään alla olvan mallin mukaan ohjelma, jossa robotti ajaa ensin asteittain kiihdyttäen noin 50cm, kääntyy puoli kierrosta ja palaa asteittain kiihdyttäen takaisin. Menomatka ja kääntyminen tehdään kahden moottorin moottorilohkoilla, paluu yhden moottorin moottorilohkoilla.

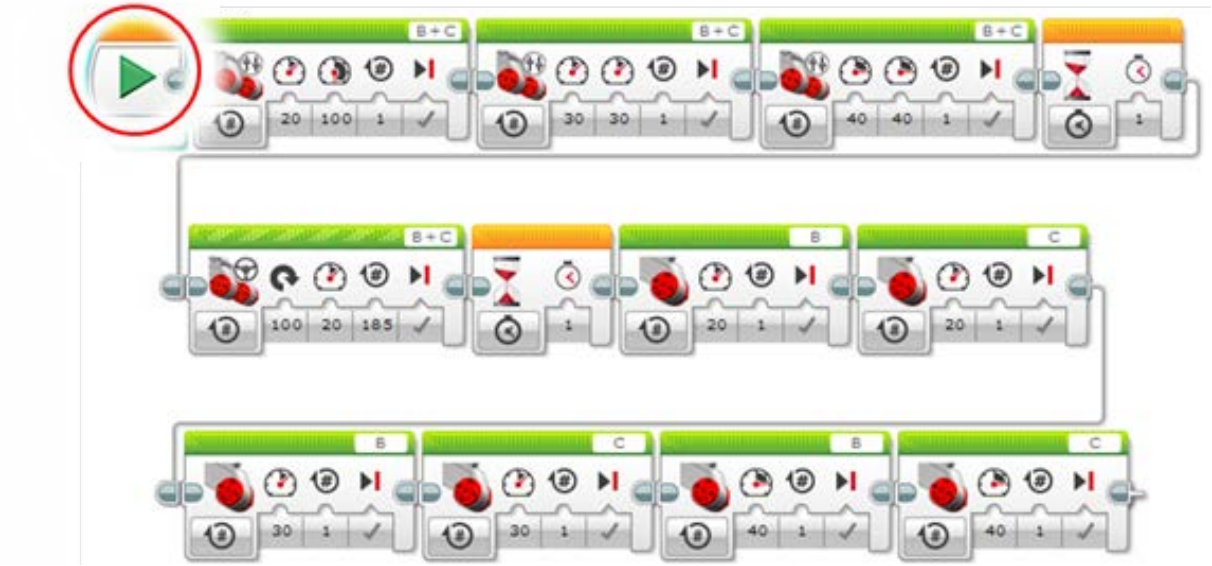


Laske robotti lattialle, lataa ohjelma ja kokeile sitä. Robotti ei tullut takaisin lähtöpisteeseensä. Ohjelmassa on nyt virhe (tarkoituksella).

EV3:n ohjelmointiohjelmistton liittyy mahdollisuus tarkastella ohjelman toimintaa reaaliaikaisesti. Tämä on osoittautunut käteväksi avuksi ohjelmien vikojen etsimisessä ja muutenkin ohjelmien kehittämisessä.

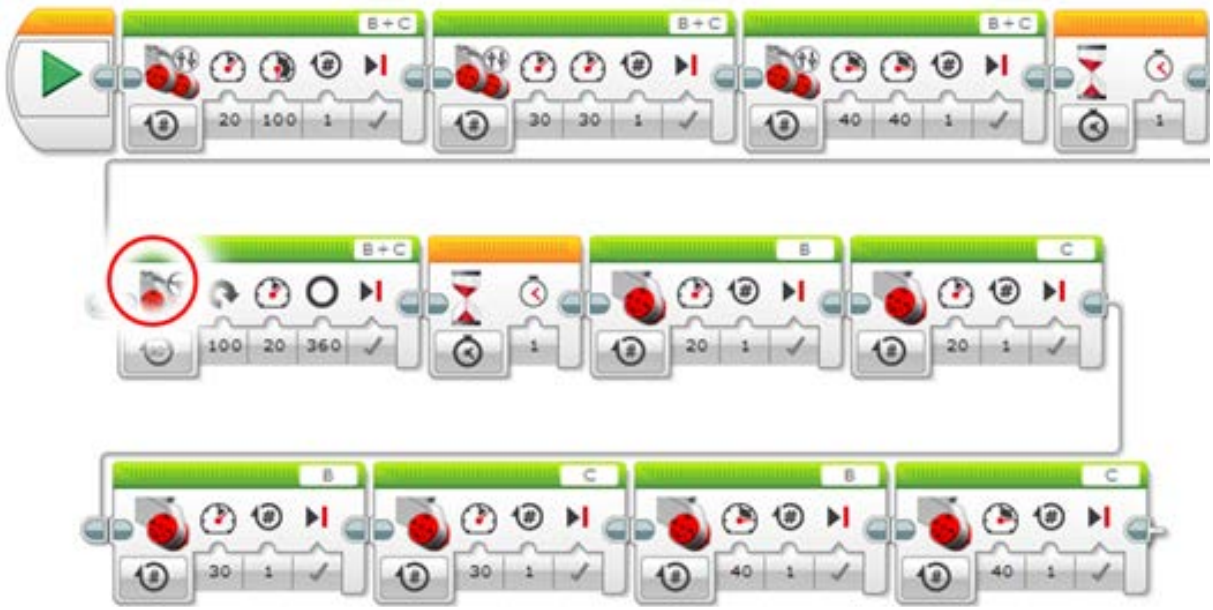
Pysäytetään ohjelman suoritus, kytketään ohjelmointikaapeli takaisin kiinni robottiin. Pidä robottia kädessäsi ja käynnistä nyt ohjelma alotustoimilohkon vihreästä kolmion muotoisesta painikkeesta. Ohjelman kulku näkyy nyt tietokoneen näytöltä. Ohjelmalohkon, jota kulloinkin suoritetaan, tunnistaa sen yläpalkissa juoksevasta raidoituksesta.

Ohjelman suoritus jumittaa nyt toisen rivin alussa. Robotti jää pyörimään paikallaan kohdassa jossa sen piti kääntyä puoli kierrosta.



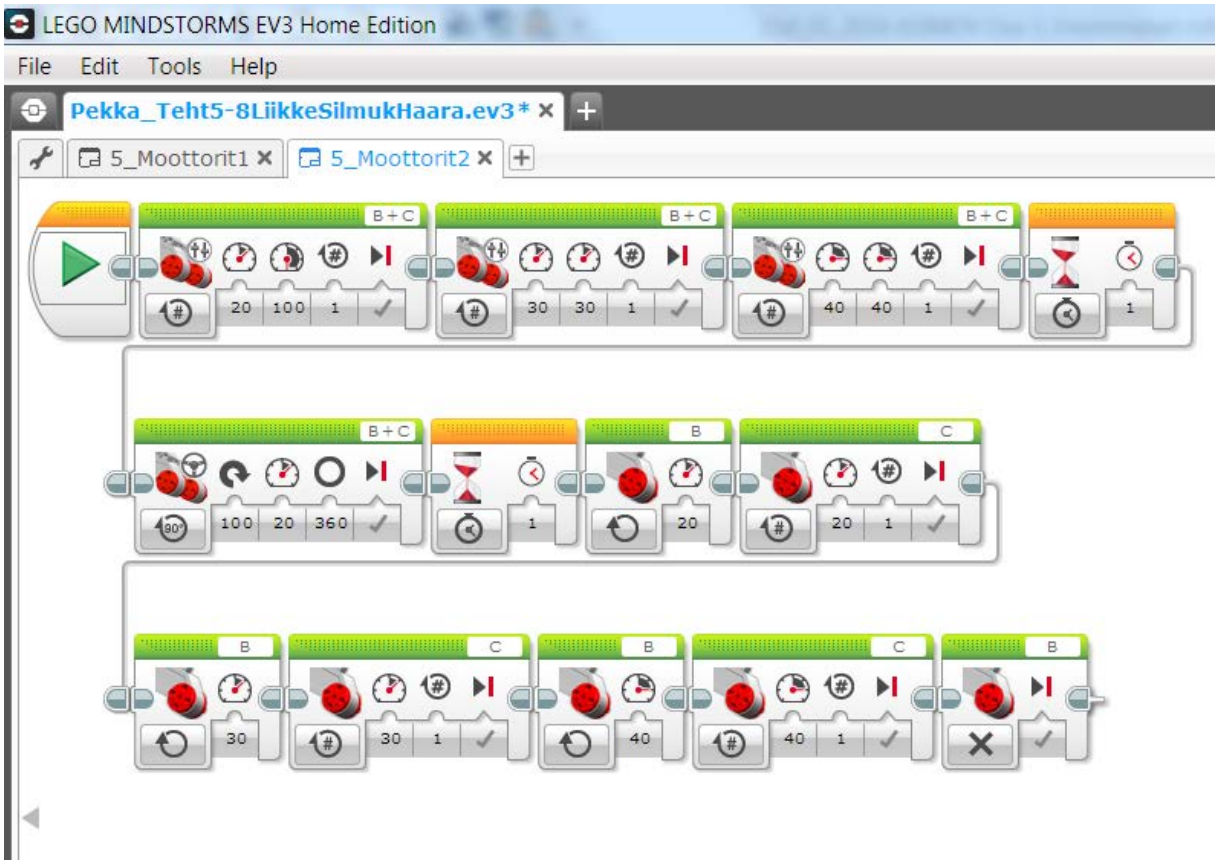
Huomataan että kääntymiseen käytetyn Move Steering toimilohkon toiminta-tavan asetus pitää vielä vaihtaa kierroksista asteiksi (On for rotations -> On for degrees).

Korjataan asetus oikeaksi (On for degrees) ja kokeillaan uudestaan.



Nyt robotti kääntyi, mutta kiemurtelee paluumatkalla, mikä ei ollut tarkoituksena. Kun seurataan ohjelman kulkua, voidaan havaita suorittavan aina B-moottoria koskevan komennon ensin loppuun asti, ennen C-moottorin vastaavaa komentoa.

Muutetaan seuraavaksi ohjelmaa siten, että se vain käynnistää B-moottorin ja antaa välittömästi C-moottorille käskyn, joka pitää sisällään tiedon myös jakson kestosta. Lopuksi pitää muistaa pysäyttää B-moottori erikseen.



Kokeillaan jälleen ohjelman toimintaa.

Kysymys 1: Seuraa robotin toimintaa tarkasti.

Miten sen toiminta menomatkalla poikkeaa paluumatkasta?

Robotin tarkkuus

Robotin servomoottorit lähettävät ohjaimelle pulssitietoa aina niiden akselin pyöriessä. Ohjain laskee nämä pulssit ja osaa pysäyttää robotin juuri oikeaan kohtaan. Kuulostaa tavattoman helpolta, mutta teknisiin laitteisiin liittyy aina virhelähteitä. Joihinkin voidaan vaikuttaa, toisiin varautua ja aina jää silti jotakin jäljelle. Kaikkia virhelähteitä ei normaaliolosuhteissa yleensä voida eliminoida.

Monitoimirobottimme tarkkuuteen vaikuttaa moni tekijä, tässä joitakin tunnettuja

- Moottorien pulssilaskennassa voi etenkin suurilla nopeuksilla, kiihtyvyyksillä ja massoilla syntyä pieniä virheitä.
- Kovissa kiihdytyksissä ja jarruruksissa pyörät liukuvat, ja on käynyt niinkin että rengas on pyörähtänyt vanteella.
- Renkaat; halkaisija, muoto, kuvio, kitka, litistyminen...
- Alustan ominaisuudet, kova/pehmeä, karhea/liukas, puhdas/pölyinen....
- Nopeuden pitäminen alueella 20-75% auttaa asiaan, pienempi parempi.
- Akkujen/paristojen jännite, käytä aina täyteen ladattuja akkuja
- Robotin paino

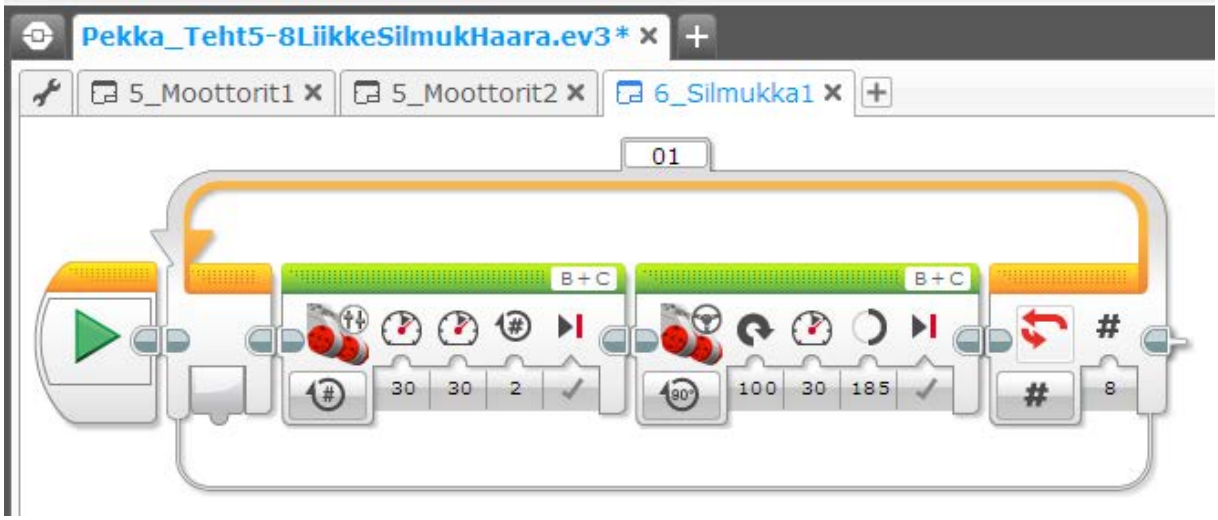
Käännöksien tarkkuuteen vaikuttavat myös esimerkiksi

- Raideleveys, suurempi raideleveys -> tarkempi
- Renkaiden leveys, kapeampi rengas -> tarkempi
- Renkaiden akselien tukeminen molemmista päistä -> tarkempi

Tehtävä 6. Ohjelman osia toistetaan silmukoilla

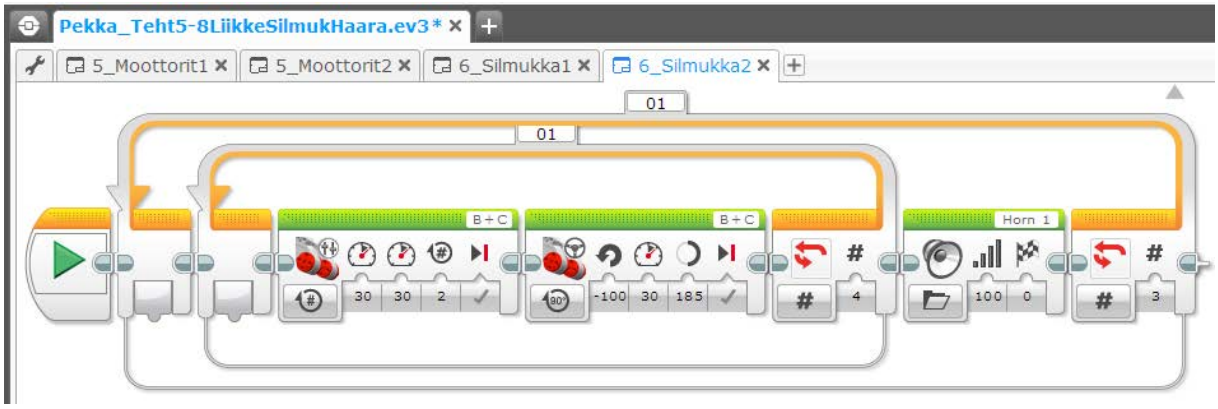
Ohjelmoitaessa kohdataan usein tilanteita, joissa jokin käsky, tai useamman käskyn muodostama kokonaisuus halutaan toistaa. Tällöin ohjelmaan lisätään silmukka. EV3:a ohjelmoitaessa toistettava ohjelmaosuus joko raahataan hiirellä, tai rakennetaan suoraan silmukkatoimilohkon sisään.

Ohjelmoi seuraavaksi robotti kulkemaan neliön (35cmx35cm) muotoinen reitti myötäpäivään, kahdesti.



Tutustu samalla silmukka-toimilohkon parametrointiin. Ohjelmoinnissa käytettävät toimilohkot on esitelty kirjan lopussa.

Kehitetään ohjelmaa edelleen. Ohjelmoi robotti kulkemaan neliön (35cmx35cm) muotoinen reitti vastapäivään, kolmasti, mutta niin, että se antaa lisäksi äänimerkin jokaisen kierroksen päätteeksi.



Vinkki: Horn 1 –äänen löydät klikkaamalla äänitoimilohkon tyhjää nimikenttää ja valitsemalla soitettavien äänien valikosta ”LEGO Sound Filles” → ”Mechanical” ja sieltä ”Horn 1”-tiedoston.

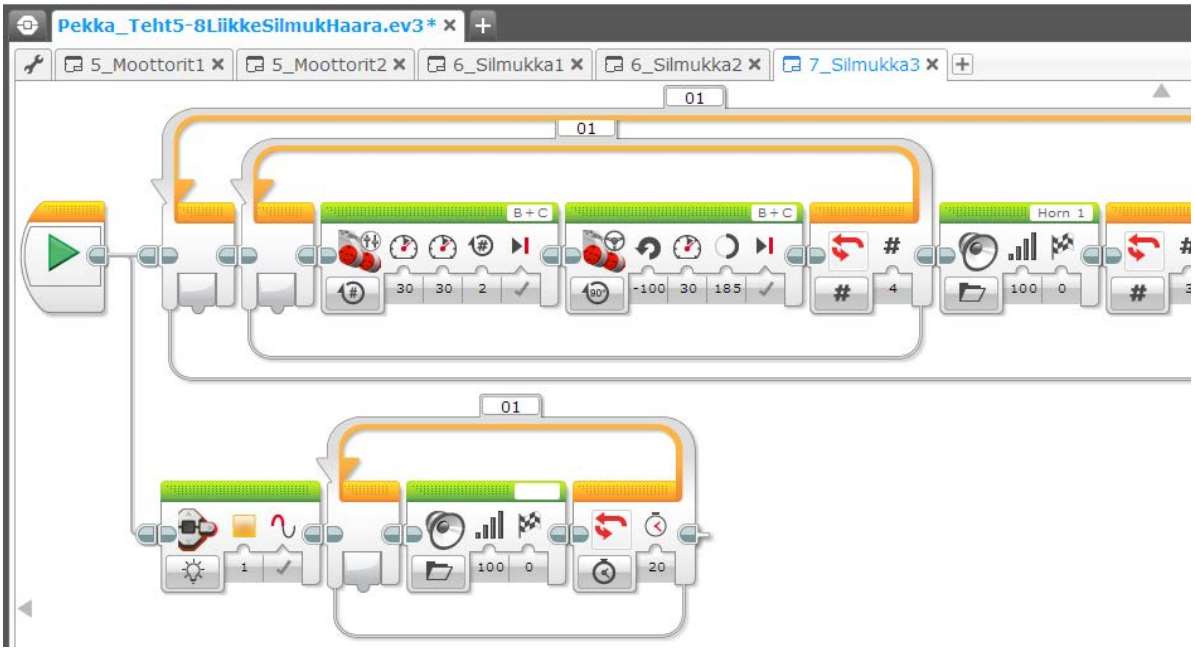
Tehtävä 7. Ohjelman osien suorittaminen rinnatusten

Osa ohjelmoitavista laitteista, kuten EV3, kykenee suorittamaan useamman rinnakkain kulkevan polun käskyjä yhtä aikaa.

Nyt, vielä kun robottiin on kytketty vasta kaksi moottoria, tälle toiminnolle on vähemmän käyttöä. Mutta viimeistään kilparobotteihin tutustuessamme tälle ominaisuudelle tulee käyttöä.

Kokeillaan ohjelman osien rinnakkaista suorittamista lisäämällä äsken tehtyyn ohjelmaan hiukan moottorin pärinää. Malli alla, tarkenna lopuksi pärinä kestämään muutaman sekunnin verran ohjelman päähaaran suorittamista pidempään. Ohjelman suoritus jatkuu kunnes ohjain on saanut suoritettua viimeisenkin polun kaikki käskyt loppuun asti.

Kun teet ohjelmaan haaroitusta, huomaa että kaikki polut (eli toimilohkoja yhdistävät viivat) alkavat toimilohkon ”liitimestä” ja päättyvät toimilohkon ”liittimeen”.



Vinkki: Motor idle –äänen löydät klikkaamalla ääni-toimilohkon tyhjää nimikenttää ja valitsemalla soitettavien äänien valikosta ”LEGO Sound Filles” → ”Mechanical” ja sieltä ”Motor idle”-tiedoston.

Tehtävä 8. Tanssiaiset robottien tapaan

Keksikää robotillenne hauskoja tanssiliikkeitä. Miettikää samalla miten silmukoita voisi hyödyntää koreografian ohjelmoinnissa, tansseissahan samat liikekuviot toistuvat usein montakin kertaa. Lopuksi voidaan järjestää vielä pienet robottien tanssiaiset ja kuvata kurvailut videolle.

Käsitöissä voidaan valmistaa robotille vaikka auton näköinen kuori tai kuljetettava nukke, pehmo tai muu hauska hahmo kyytiin. Varo kuitenkin, ettei kuormasta tule liian painavaa.

(Nukke: Hilkka Erwe)

Jatkot

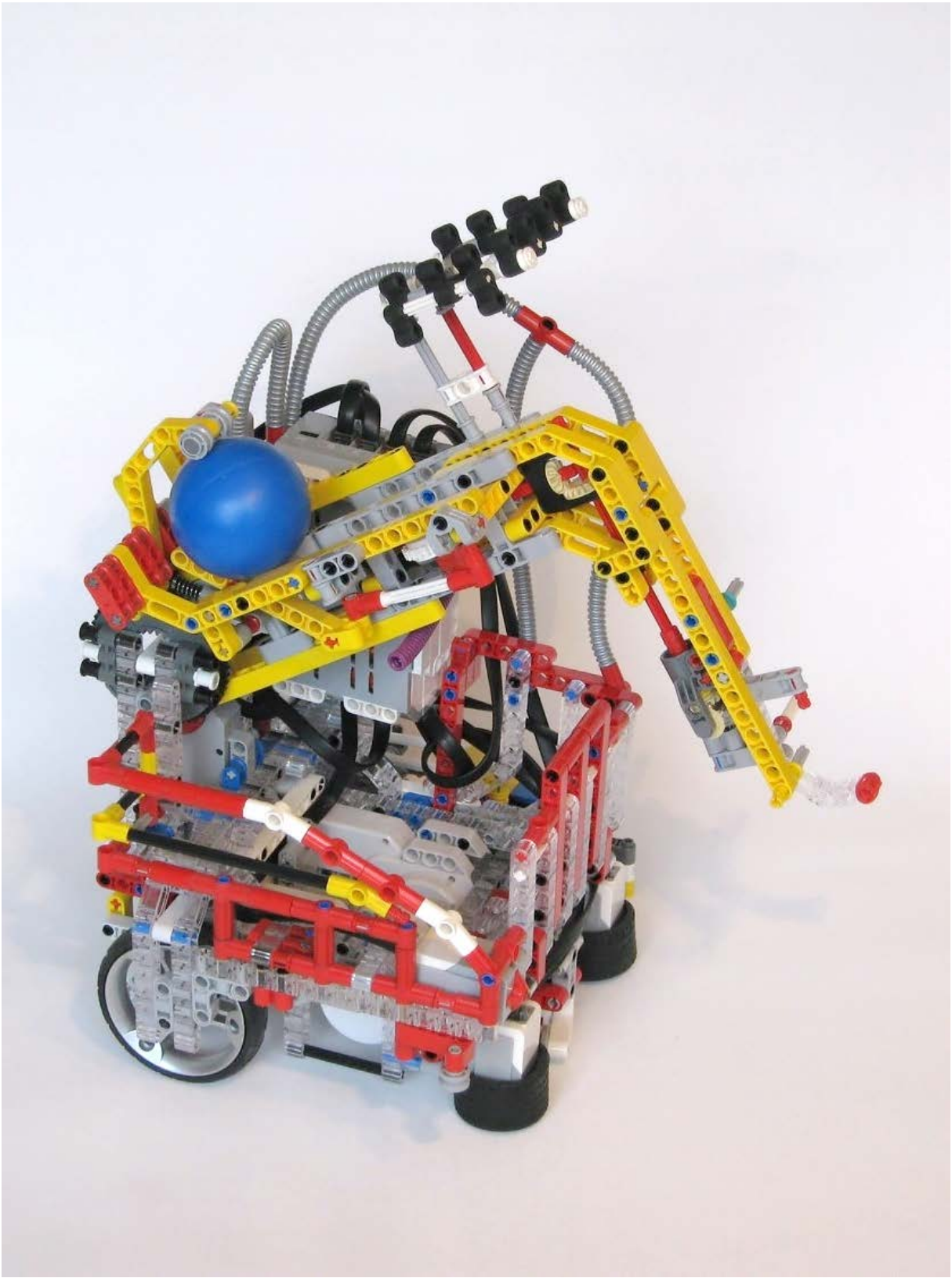
Etsi esimerkiksi Wikipediasta lisää tietoa hyvistä ohjelmointikäytännöistä.

Seuraavalla harjoituskerralla rakennetaan robotti, joka osaa seurata viivaa. Viivan seuraamiseen voit tutustua jo ennakolta vaikka katsomalla netistä videoita hakulauseella ”EV3 line follower”.

Vastaukset

Tehtävä 4

Menomatkalla robotti pysähtyy joka kerta suoritettuaan kahden moottorin toimilohkolla määritellyn tehtävän. Paluumatkalla robotti kulkee pysähtelemättä, jouhevasti. Näitä molempia ominaisuuksia voidaan hyödyntää robotin ohjelmoinnissa, tietysti tilanteesta riippuen.



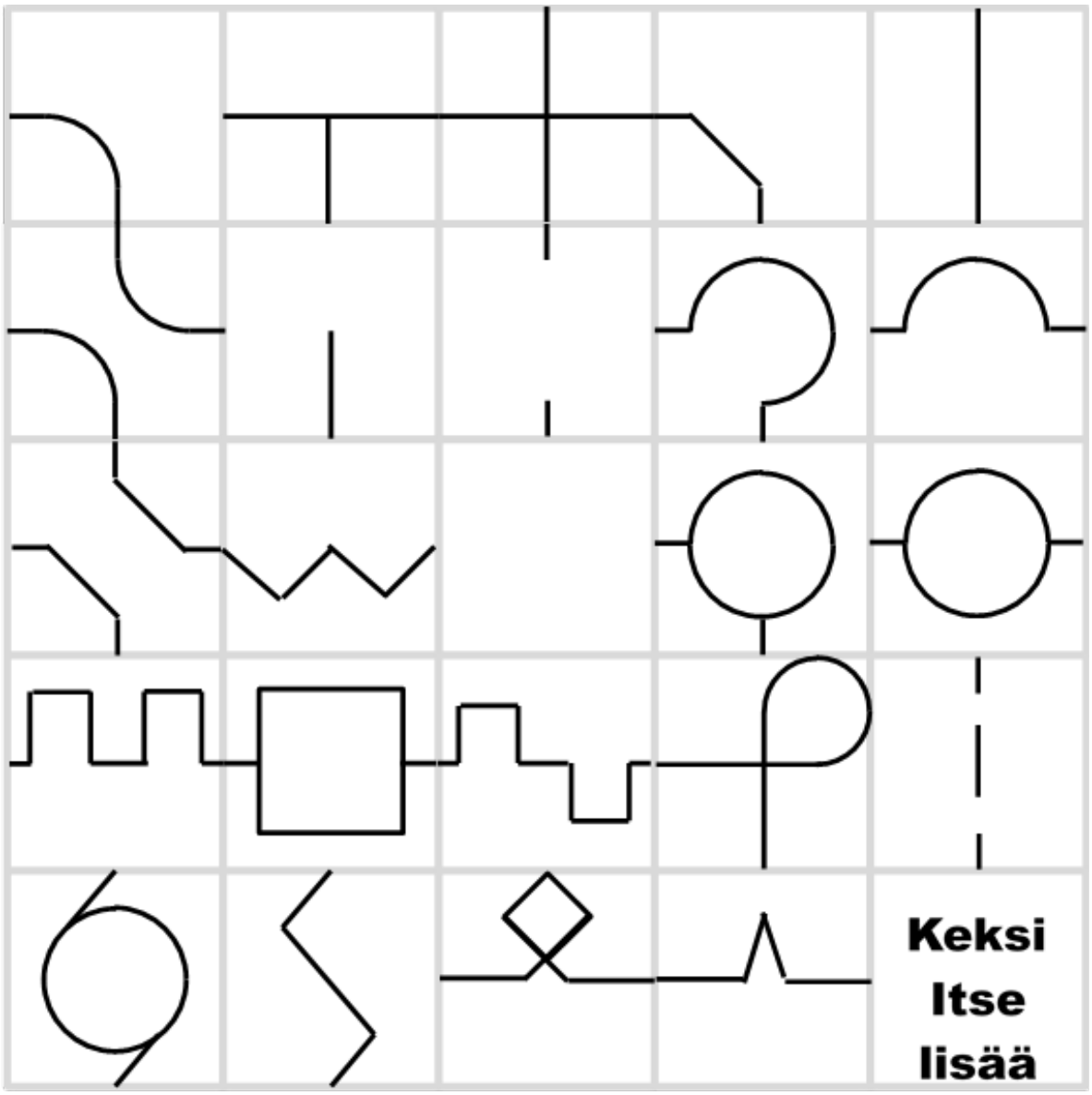
Kilparobotti

5. Valokenno, paikoitus, viiva ja ehtolauseke

Opitaan ohjelmoimaan ehdollisesti suoritettavia ohjelman osuuksia, sekä ohjelmoimaan peruslaskutoimituksia. Ymmärretään ehtolauseiden merkitys ja käyttö.
Aloitetaan antureihin tutustuminen selvittämällä mitä valokenno kykenee tunnistamaan.

Etkot

- Viivan seuraamisen harjoitteluun tarvitaan harjoitusalusta.
- Jos käytettävissä on vaalea sileä lattiapinta, harjoitusradan voi tehdä käyttäen mustaa 1-2 cm leveää PVC-teipiä (sähkömiesten yleisesti käyttämä eristysteippi, löydät sitä mm. rautakaupasta sähkötarvike-osastolta).
- Vastaavan voi tehdä maalaamalla, piirtämällä, teippaamalla mille tahansa sileälle vaalealle alustalle (levy/kartonki/paperi/jne.).
- Harjoitusalustasta on syytä tehdä nyt ensimmäistä vaativampi.
- Seuraavalle sivulle on kerätty esimerkkejä muodoista joita voi sisällyttää rataan. Ruudukon mittakaava on 30cm x 30cm.
- Lisää vinkkejä on tarjolla esimerkiksi Youtubessa hakusanoilla ”robocup junior rescue”



Yksi tapa tehdä helposti muunneltava rata on kopioida oheisia kuvia

30cm x 30 cm kartongin tai kevyn paloille sopiva määrä. Radalle voidaan lisätä myös esim. tiiliskivi kierrettäväksi esteeksi (pystyasennossa).

Tehtävä 9. Tutustutaan tekstien esittämiseen näyttöruudulla

Liitetään valokenno robottiin (Portti 1) ja tehdään apuohjelma, joka näyttää ohjaimen näyttöruudulla mitä valokenno ”näkee”. Mitataan millä etäisyydellä EV3:n valokennon erottelukyky on parhaimmillaan ja kokeillaan eri värien tulkintaa.

Tehdään ohjelma, joka

- Tyhjentää näytön ja kirjoittaa ensimmäiselle riville ”VALOKENNO 1”,
- toiselle riville ”Vari” + tätä vastaavan mittaustiedon,
- kolmannelle riville ”Valo” + tätä vastaavan mittaustiedon,
- ja neljännelle riville ”Heijast.” + tätä vastaavan mittaustiedon.
- Mittauksesta siirrytään aina seuraavaan 0.5s välein, päättymätön silmukka.

Valokenno

Näkyvän valon ja värin mittaaminen, tekniikka

EV3:n valokennossa on yhdistetty kolme erilaista toimintoa.

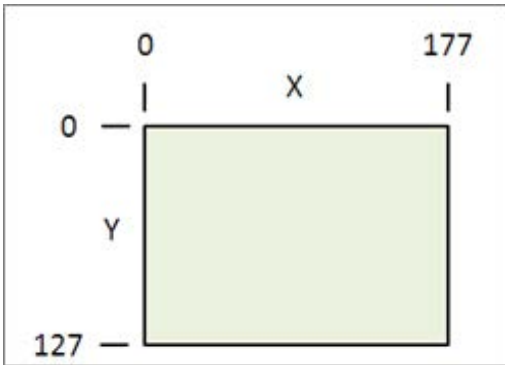
- Valon määrän mittaus
- Heijastuvan valon mittaus
- Värin tunnistus

Tämä valokenno on varustettu valoa vastaanottavan kennon lisäksi vihreällä, punaisella ja sinisellä ledillä, joita käytetään värien tunnistuksessa ja apuvalona heijastuvan valon määrää mitattaessa.

Tunnistettavat värit ovat musta, sininen, vihreä, keltainen, punainen, valkoinen ja ruskea. Muut värit tulkitaan luokkaan ”No Color”, tai lähimpään edellä mainituista. Tarkista tarvittaessa miten kenno tulkitsee värin. Esimerkiksi oranssi saattaa tulla tulkituksi joko keltaiseksi tai punaiseksi, ruskeaksi, jopa mustaksi, sävystä ja etäisyydestä riippuen.

Näyttöruutu

EV3:n näyttöruutu on 178x128 pisteen kokoinen. Tekstin aloituskohta, eli sen ensimmäisen merkin vasemman yläkulman paikka määritellään tämän vaaka- (X) ja pysty- (Y) suuntaisena etäisyytenä näyttöruudun vasemmasta ylänurkasta.



Näyttöruudun koordinaatisto

Tämä koordinaatisto on tavallisesta poikkeava. Yleensä XY-koordinaatistot esitetään siten että lukemat alkavat nollasta ja kasvavat oikealle, sekä, toisin kuin tässä, ylös päin. Mutta on ohjelmoitaessa tullut kummallisempiakin poikkeuksia vastaan (kirjoittajan kommentti).

EV3:ssa tekstit kohdistetaan näytölle määrittelemällä niiden vasemman ylänurkan paikka. Edellisen sivun kuvassa rivit (Y) oli määritelty alkamaan korkeudelta 25, 50, 75 ja 100 pistettä. Kaikki tekstit alkavat ruudun vasemmasta reunasta, eli kaikkien tekstien X parametrin arvo =0. Mittaustietojen sarake alkaa vastaavasti kohdasta X=130.

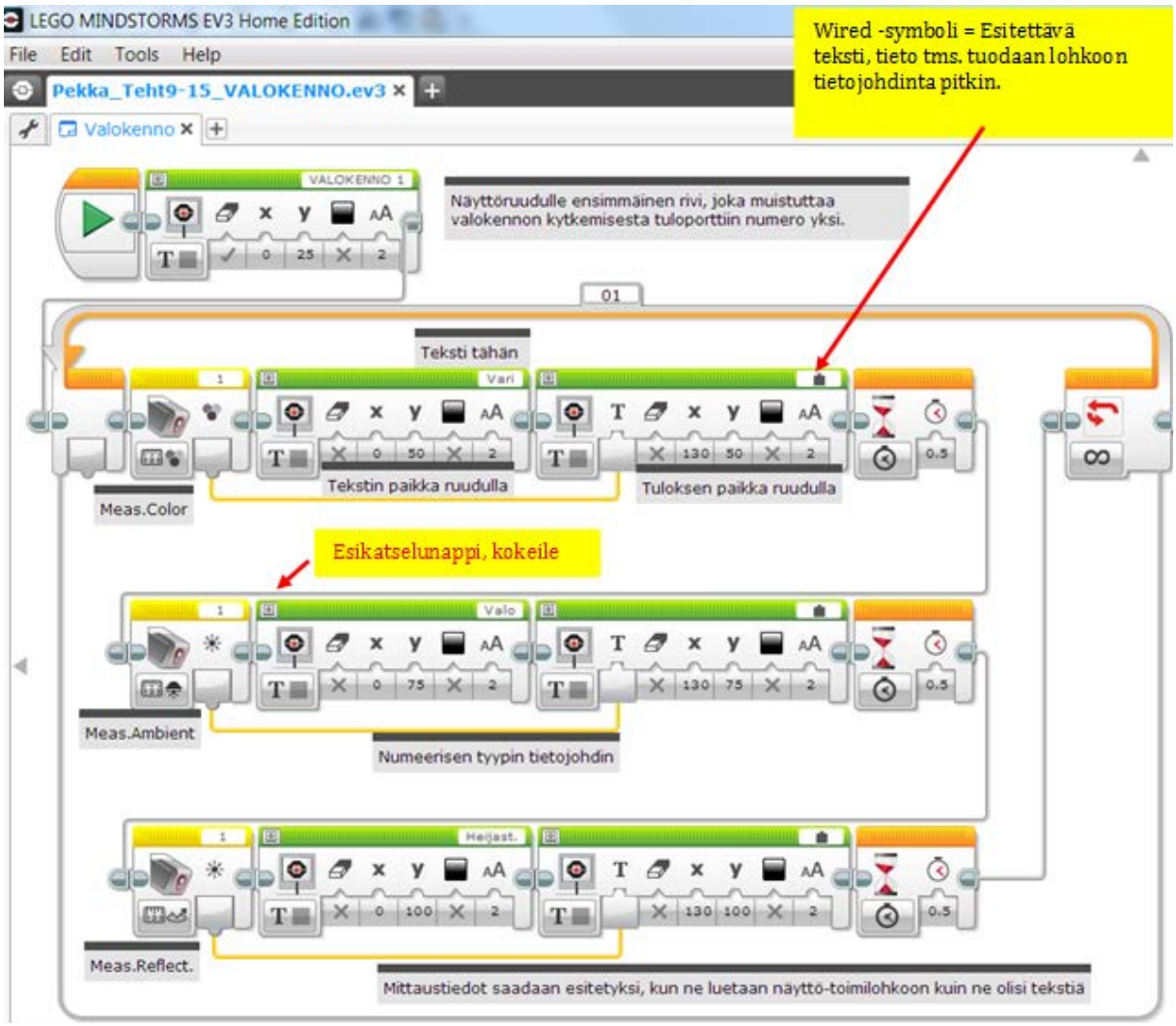
Jokainen kirjain, tai merkki vie tietyn tilan näyttöruudulta. Kirjaimia EV3:ssa on kolmea eri kokoa, 8x8, 8x9 ja 16x16 pikseliä. Pikseli on pienin näytöllä ohjattavissa oleva piste.

Montako suurta kirjainta, merkkiä tai numeroa mahtuu yhdelle riville, 178 : 16 = ?

Entä pienempiä 8 pikseliä leveitä kirjaimia?

Tietokoneen näyttöruudun toimintaperiaate on aivan sama. Niiden ohjelmat ovat vain paljon kehittyneempiä. Kirjaimet löytävät ruudulla kukin omalle paikalleen kuin itsestään, koska taustalla toimiva ohjelma osaa huolehtia tästä.

Kirjoitetaan ensin ohjelma mallin mukaan.



Vinkki: Tee ensin tarkasti yksi rivi. Kopioi se kolmeksi ja ketjuta nämä. Siirrä aikaansaannos silmukkalohkon sisään ja viimeistele ohjelma.

Vinkki2: Tätä ohjelmanpätkää voit tarvita aina kun käytät valokennoa. Jos joku ohjelma kannattaa pitää aina valmiiksi ladattuna ohjaimen muistissa, se on tämä.

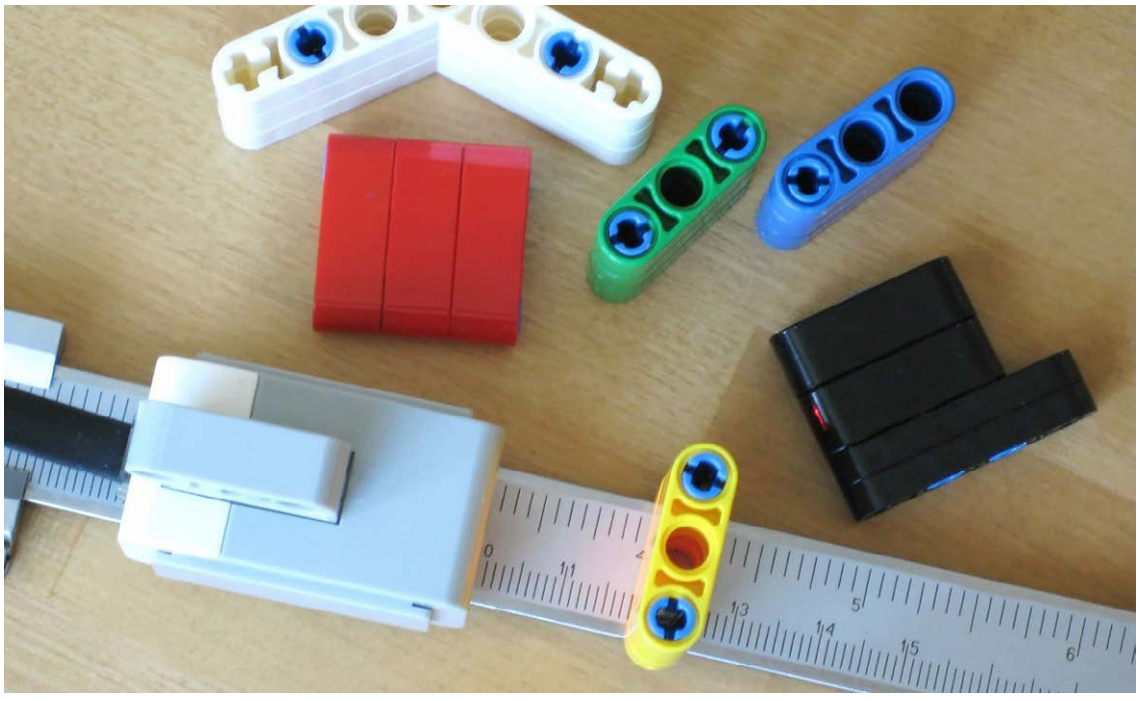
Uuden kommenttikentän lisääminen ohjelmakaavioon: Merkitse ensin paikka klikkaamalla kohtaa johon haluat uuden kommentin. Klikkaa sitten nuolen osoittamaa painiketta (kuva).

Kun ohjelma on saatu toimimaan, voidaan ryhtyä selvittämään valokennon ominaisuuksia kokeellisesti.

Tehtävä 10. Värien tunnistaminen

Tehtävään tarvitaan viivotin mittaamiseen ja vihko tuloksien kirjaamista varten. Robottisarjan toistaiseksi vapaana olevista sinisistä, punaisista, vihreistä, keltaisista, valkoisista ja mustista osista saa rakennettua kätevät, kolme nuppia leveät mittapalikat (6/7 väriä, yksi jää puuttumaan).

- Kirjaa vihkoon sivun reunaan allekkain numerot 0, 1, 2, 3, 4, 5, 6, ja 7.
- Selvitä mittapalikoiden avulla mitä väriä kukin numero vastaa, kirjoita värit numeroiden perään.
- Kokeile ja mittaa mikä on lyhin ja pisin etäisyys jolloin kenno vielä tunnistaa kunkin värin oikein. Kiinnitä huomiota myös kennon ja palikoiden asentoihin. Kirjaa tulokset vihkoon.
- Keskustelkaa tuloksista.
  - ? Oliko värien kesken eroja
  - ? Saivatko kaikki samansuuntaisia tuloksia
  - ? Mitä värejä mahtaisivat edustaa kaksi tyhjäksi jäänyttä riviä



Vinkki: Mittaaminen on helppoa, kun asettaa kennon ja mittapalikan viivottimen tms. päälle.

Tehtävä 11. Kohteesta heijastuvan valon määrä

Tutkitaan kohteesta heijastuvan valon määrän mittaamista.

- Apuvälineiksi tarvitaan viivoitin, sekä mattapintainen paperiarkki, harjoitusalusta tms. jolle on tulostettu tai tussilla maalattu, vähintään 3cm leveä, mahdollisimman musta viiva.
- Kokeile ja mittaa, millä etäisyydellä ja missä asennossa (tarkasteltavaan pintaan nähden) valokennon on oltava, jotta se näkisi jyrkimmän rajan, suurimman mahdollisen eron mustan ja valkoisen välillä. Kirjaa havainnot vihkoon.
- Keskustelkaa tuloksista.
  - ? Oliko valokennon asennolla väliä
  - ? Paras tunnistusetäisyys
  - ? Saivatko kaikki samansuuntaisia tuloksia

Tehtävä 12. Ympäristön valoisuuden mittaaminen

Mitataan valoisuutta, muista valon lähteistä peräisin olevan valon määrää.

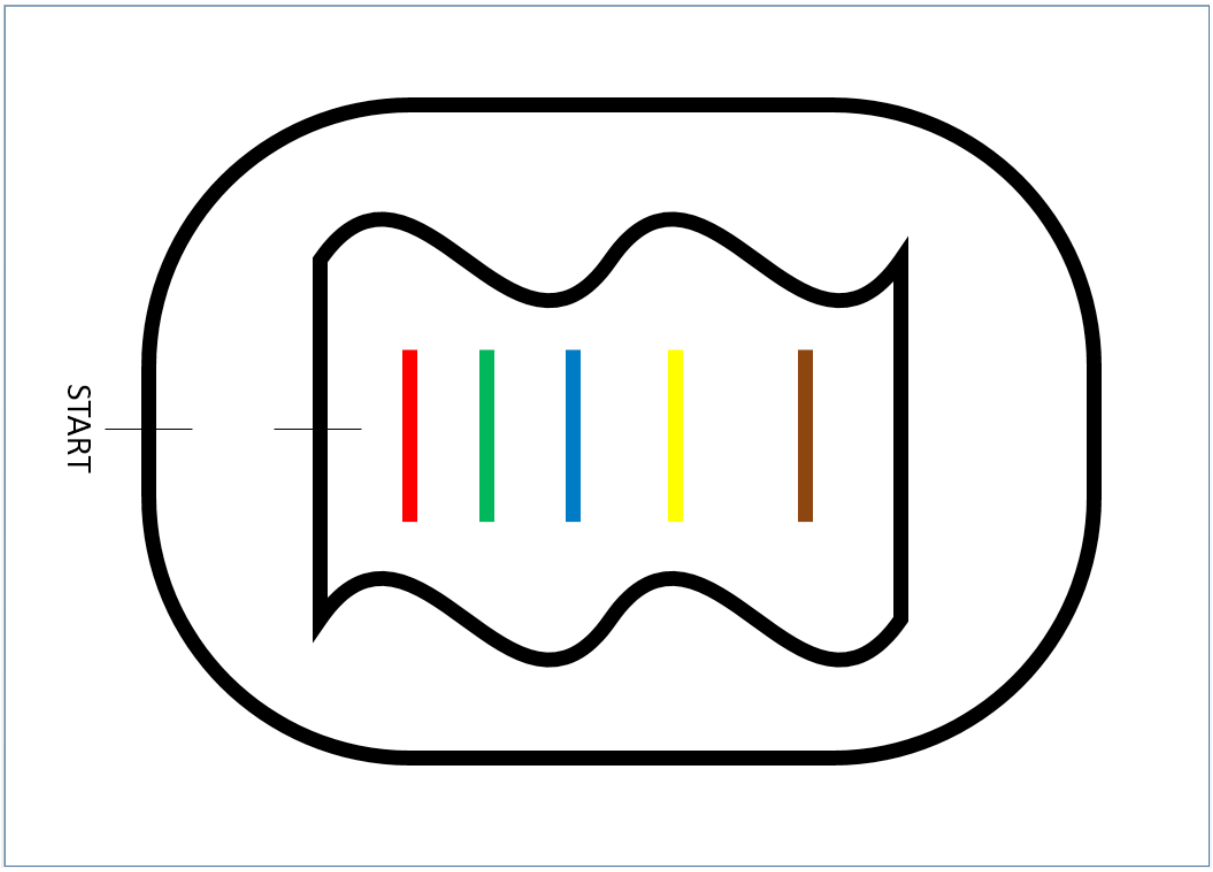
- Kokeilkaa millaisia lukemia saatte ikkunasta tulevasta valosta, tietokoneen näytön valosta, jne.
- Koulussa, opettajan luvalla, voidaan etsiä luokan valoisin ja pimein kohta.
- Sääntönä voidaan pitää, että mittausta suoritettaessa valokennon edessä pitää olla 30cm vapaata tilaa (viivoittimen pituuden verran). Tavaroiden ja ihmisten liikuttelua tulee välttää tätä koetta tehtäessä. Etenkin, jos siitä voi seurata havaittavissa oleva muutos mittauksen tulokseen.

Nyt on tutkittu valokennon toimintaa. Miettikää yhdessä miten valokennoa voisi hyödyntää robotin toiminnassa.

Tehtävä 13. Testataan värien tunnistamista

Paikoitus

Tehdään yksinkertainen testirata käyttäen mustaa ja värillisiä PVC-teippejä (tunnetaan myös sähkömiehen teippinä, jota saa ostaa mm. kaikista rautakaupoista), tai käytetään soveltuvaa valmista pohjaa. Ala esitetyn mallin mukainen testirata löytyy tämän kirjan liitteistä tulostettavana isona .pdf –tiedostona, 100cm x 140cm. Testiradan tulostaminen edellyttää joko suurkuvatulostusta, tai mosaiikki-tulostusta.

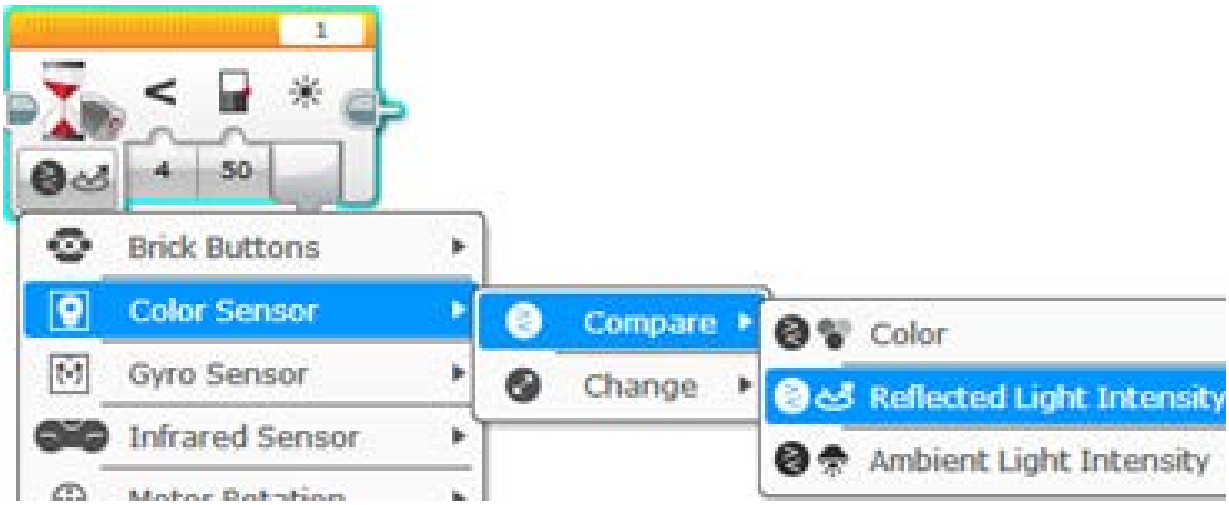


Vinkki: Testiradan ei tarvitse olla tämän näköinen, periaate on tärkeämpi.

Laitetaan valokenno takaisin paikalleen ja kokeillaan ensin, valokennon testausohjelmaa käyttäen, mitkä värilliset viivat robotti onnistuu tunnistamaan. Teippien ja värien värisävyissä on eroja eikä robotti siksi välttämättä tunnista kaikkia värejä oikein (tähänkin on keinonsa, mutta ei mennä niihin nyt).

Jos värin tunnistamisessa on vaikeuksia, voi pysähtymistä kokeilla myös heijastuvan valon määrän perusteella. Kokeilkaa mitkä värilliset viivat robotti kykenee tunnistamaan heijastuvan valon määrää mittaamalla..

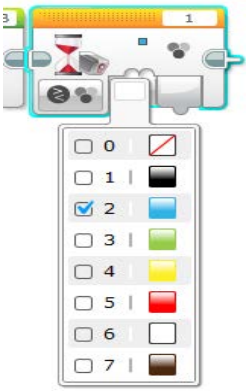
Heijastuvan valon määrän mittaamisessa valokenno lähettää valoa ja mittaa paljonko siitä heijastui takaisin.



Tämä viivetoimilohko odottaa kunnes heijastuneen valon määrä on laskenut alle 50% rajan.

Ohjelmoidaan robotti kulkemaan ”Start” –tekstin luota suoraan siniselle viivalle asti ja peruuttamaan sieltä sitten takaisin lähtöpaikkaansa. Suoritusta voidaan havainnollistaa vapaavalintaisten äänimerkkien avulla. Tarkasta kuitenkin ensin, että valokenno tunnistaa värin. Teippien ja tulostimien väreissä on pieniä eroja. Jos sininen ei jostain syystä toimi, voit valita minkä tahansa sellaisen värin jonka kenno tunnistaa.

Ohjelmointia voidaan aina tehdä lukuisilla ei tavoilla. Tässäkään tehtävässä ei ole olemassa yhtä oikeaa ratkaisua. Nyt on hyvä tilaisuus tehdä ihan oma ohjelma.



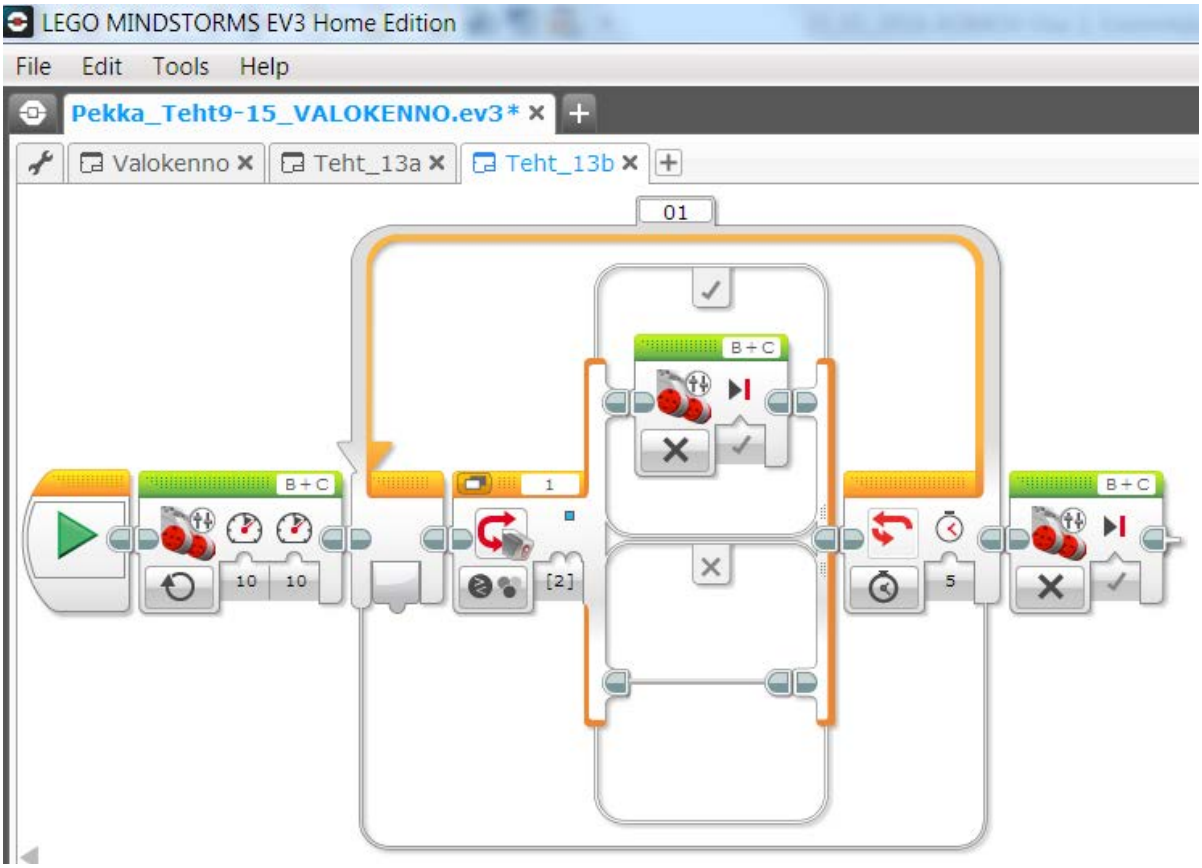
Esimerkiksi viivetoimilohko on hyvin monikäyttöinen. Se toimii kuin portilla seisova vartija. Matka voi jatkua vasta kun vartija on tarkastanut vaaditun asian, eli kun ohjelmaan kirjoitettu ehto on kunnossa.

Sillä voi esimerkiksi pidätellä ohjelman suoritusta ohjelmoijan määrittelemän ajan verran, tai ehdoksi voidaan määritellä ajan sijaan vaikka sellainen, että valokennon pitää kertoa tunnistaneensa vaikka jotakin sinistä.

Valitaan viivetoimilohkojen toiminnot. Ensimmäinen menee oletusasetuksilla ja toisen ehdoksi tulee valokennon mittaamaa tieto (Color Sensor), ja vertaa (Compare) väriä (Color).

Ohjelman pitäisi nyt pysäyttää robotti mustan viivan kohdalla ja siinä se myös kaikella todennäköisyydellä onnistuu. Mutta mitä tapahtuu jos etsitäänkin vaikkapa sinisen väristä viivaa, eikä se löydäkään mitään sen omasta mielestä sinisen väristä missään vaiheessa?

Muutetaan ohjelmaa niin, että robotti pysähtyy joka tapauksessa viiden sekunnin kuluttua.



Kokeillaan vielä tämän tehtävän lopuksi, mitkä kaikki värit robotti onnistuu tunnistamaan oikein.

Tehtävä 14. Viivanseurannan ja ehtolausekkeen ohjelmoinnin perusteet

Viivanseuranta

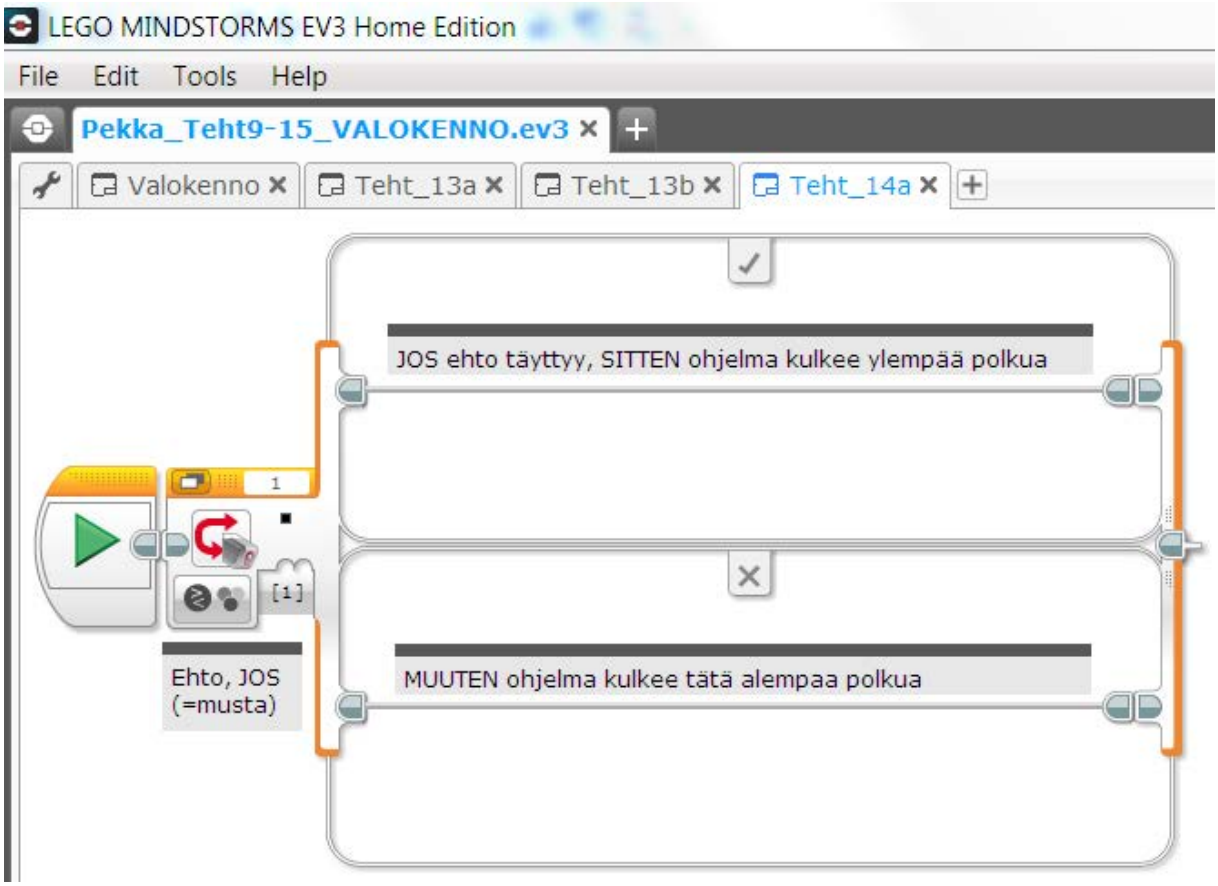
Viivanseurannassa robotti koittaa joko kulkea mustan viivan päällä, tai jompaa kumpaa sen reunaa pitkin. Käytännössä, molemmissa edellä mainituissa tapauksissa, robotti myös kaartaa koko ajan loivasti joko oikealle, tai vasemmalle.

Selkeästi suosituin tapa seurata viivaa on viivan reunan seuraaminen. Tähän on parikin syytä. Tehtävään voi liittyä viivojen risteys, joista pitää kääntyä tiettyyn suuntaan. Robotti saadaan kääntymään haluttuun suuntaan helposti seuraamalla tämän puoleista viivan reunaa. Toinen liittyy tehtäviin joissa robotin pitää viivaa pitkin kuljettuaan osua hyvin tarkasti johonkin kohteeseen. Reunaa seuraamalla robotti kiemurtelee käytännössä vähemmän ja näin sen osumatarkkuus paranee.

- Hienommat viivanseuraaja-ohjelmat osaavat tunnistaa milloin tullaan kaarteeseen tai risteykseen, sekä sopeuttaa reaktioitaan aina tilanteen mukaan Osaavatpa ne jopa ajaa viivaan jätetyn aukonkin yli ja löytää taas takaisin radalle.
- Kuvasta ilmenee miten robotti mustan viivan nähdessään kaartaa kohti valittua reunaa, ja valkoisella taas takaisin kohti mustaa.
- Seurattavan reunan valinta tehdään valitsemalla kumpi moottori pyörii nopeammin kun ollaan mustalla (ja se toinen pyörii sitten nopeammin valkoisella).

Ehdollinen suoritus, If-Then -toimilohko

Aina kun toimintakuvauksessa voidaan käyttää sanaa ”JOS” (If) on kyse jonkun käskyn ehdollisesta suorittamisesta (JOS-SITTEN, eli IF-THEN). Oikeastaan tätä pitäisi jatkaa vielä sanalla MUUTEN (ELSE), joka kertoo siitä että on olemassa toinenkin vaihtoehto. Mutta koska se on ilmeistä muutenkin, käytetään ehdollisesta suorittamisesta yleensä lyhintä nimeä, IF-THEN.

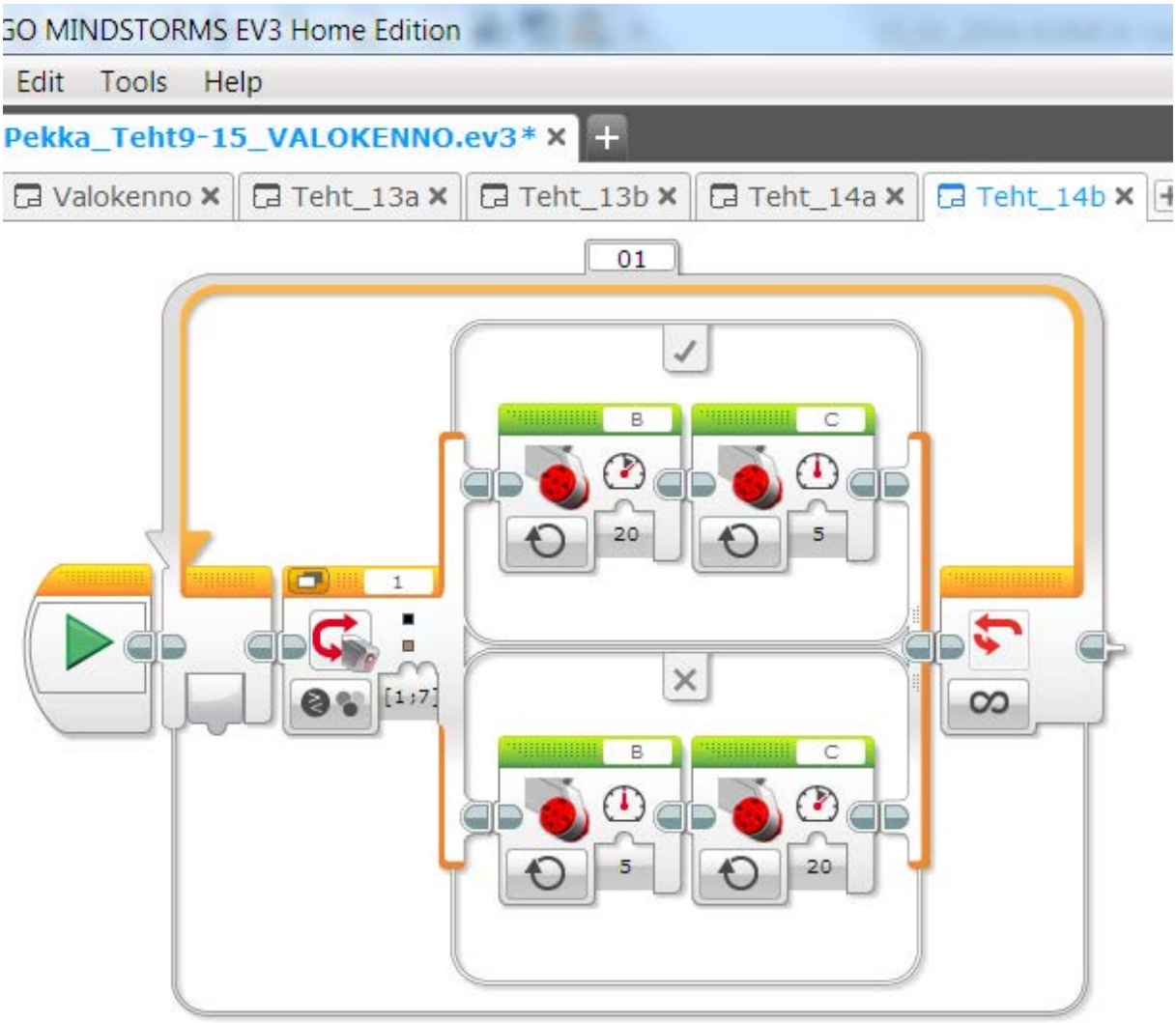


IF-THEN -toimilohkon tarkempi kuvaus löytyy kirjan lopusta.

Perusmallin ohjelma viivan seuraamiseen on helppo tehdä käyttäen ehdollista ohjelman suoritusta (Katso tehtävä 4).

Viivan seurantaa käytetään jo nyt monessa paikassa, kuten vihivaunuissa eli tavaroita kuljettavissa roboteissa. Tätä kirjaa kirjoittaessa ovat tälle rintamalle tulleet juuri mukaan mm. maantien reunamerkintöjä tunnistavat autojen turvajärjestelmät ja erilaisia robottiautoja.

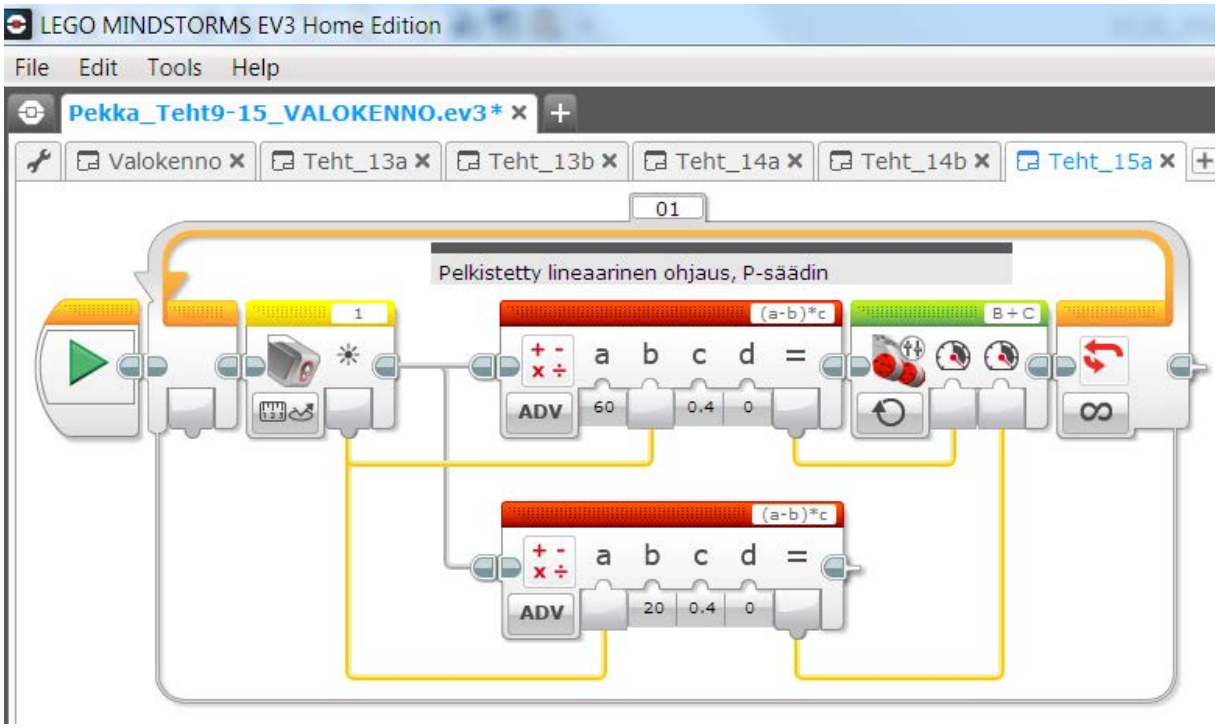
Seuraavaksi tehdään viivan oikeaa reunaa seuraavaa ohjelmaa. Tavoitteena on ohjelmoida robotti kulkemaan rataa, vaikka siinä olisi tiukenpiakin käännöksiä, molempiin suuntiin. Robotti seuraa viivaa helpommin pienemmällä nopeudella.



Ohjelman hienosäätö tehdään muuttamalla moottorilohkojen nopeusasetuksia. Mitä pienempi moottorien nopeusero, sitä loivemmin robotti kääntyy, ja päinvastoin. Etsikää sopivat arvot kokeilemalla.

Tehtävä 15. Kehittyneempi viivanseuraaja, peruslaskutoimituksien ohjelmointi

Tässä tehtävässä kokeillaan vielä aavistuksen kehittyneempää viivanseuraajaa (P-säädin).



Tämä robotti kulkee suoraan, jos heijastuvan valon määrä on 40% (harmaa). Nopeus on määritelty kertoimella ”c”. Kasvattamalla sen arvoa nopeus kasvaa.

Yllä olevassa ns. lineaarisessa mallissa on määritelty, että haluamme kulkea mustan ja valkoisen puolivälissä, alueella jonka valokenno tulkitsee harmaaksi. Mitä enemmän heijastuvan valon määrä poikkeaa tästä, sitä suurempaa korjausliikettä pyydetään moottoreilta.

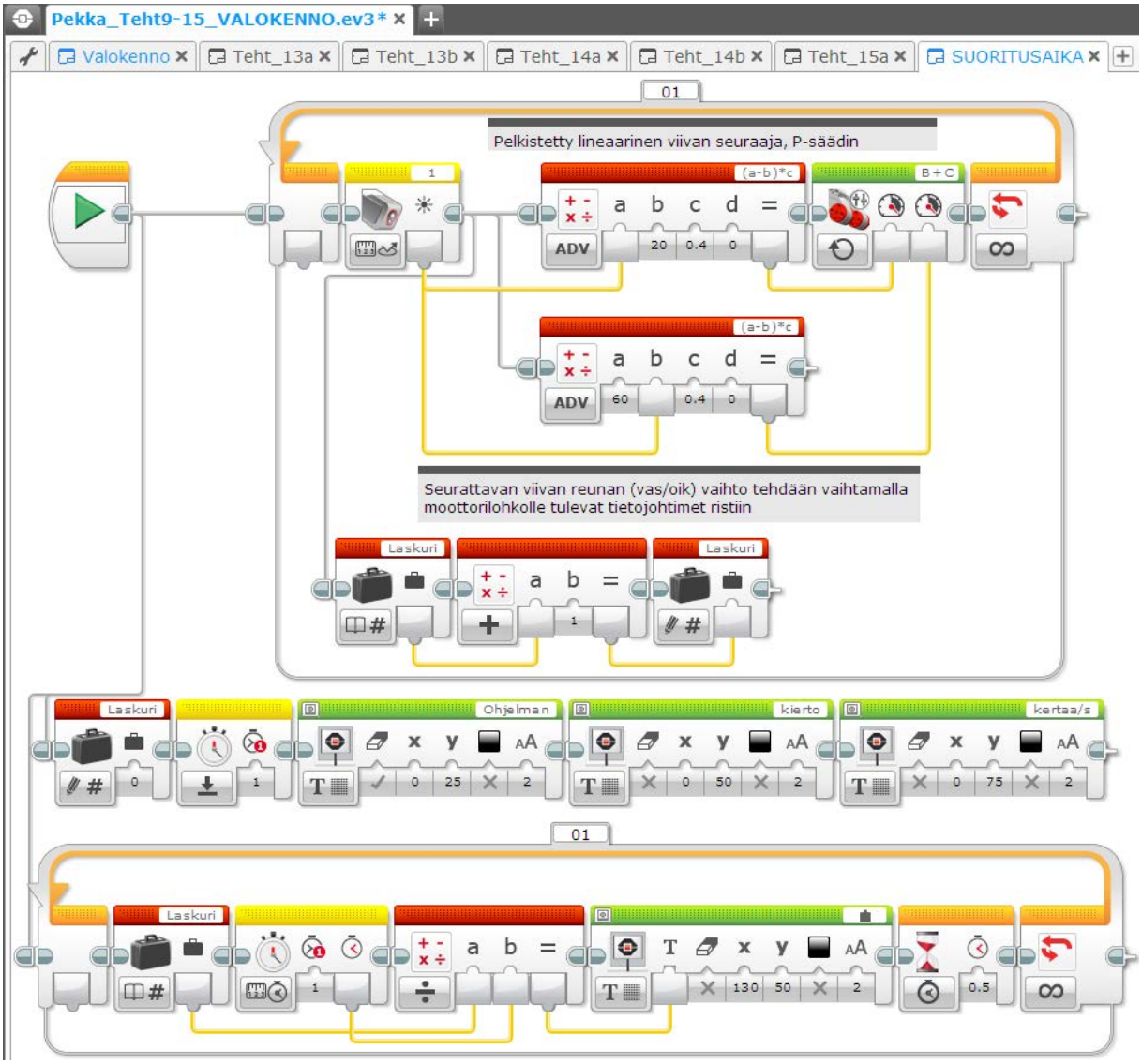
Viivanseuraajia ohjelmoidaan moniin tarpeisiin. Monasti yleismalli riittää, mutta aina toisinaan tulee vastaan tilanteita kun pitäisi päästä nopeammin, tarkemmin, kapeampaa viivaa pitkin, tai vaikka tiukemmista mutkista. Tällöin ohjelmointi muuttuu mielenkiintoiseksi. Ohjelman kiertoa voidaan yrittää nopeuttaa ja moottorien ohjaukseen haetaan yhä monimutkaisempia matemaattisesti mallinnettuja ratkaisuja, mm. PID-säätäjiä ym. Tai sitten lisätään valokennojen määrää.

Ohjelman suoritusajan mittaaminen

Myös ohjelman kiertoaikaa voi mitata. Alla olevaan ohjelmaan on lisätty yksinkertainen suorituskertalaskuri, jonka tulos jaetaan ajalla jonka ohjelma on ollut käynnissä. Lopuksi tulos esitetään ohjainyksikön näyttöruudulla.

Ohjainyksikkö suoritti tämän ohjelman 32 kertaa sekunnissa, mikä ei ole kovin paljon. Kun piiriä muutettiin hiukan niin, että molemmille moottoreille oli oma erillinen moottorilohko, nopeus laski 28 kertaan sekunnissa.

Pohdi millaisia vaikutuksia ohjelman suorittamisen nopeudella voisi olla esimerkiksi viivan seuraamisessa?



Muistitoimilohkoon tutustutaan 6. luvussa.

Vastaukset

Harjoitus 10

nro	Toiminto	min	max
0	Ei väriä (ei kohdetta)		
1	Musta	4	17
2	Sininen	5	15
3	Vihreä	3	18
4	Keltainen	1	18
5	Punainen	2	15
6	Valkoinen	2	21
7	Ruskea	-	-

Harjoitus 11

Valokenno tunnistaa parhaiten mustan viivan reunan, kun se on asennettu kohtisuoraan tarkasteltavaa pintaa vastaan ja n.8mm etäisyydelle siitä.

Virhelähteitä

Näihin mittauksiin, sekä valokennon toimintaan yleisesti liittyviä virhelähteitä:

- **Satunnaisia**, kuten kennon ja mitattavan pinnan pienet asentojen poik keamat. Kenno ja mitattava pinta eivät ole aina optimaalisesti keske nään kohtisuorassa asennossa.
- **Systemaattisia**, kuten kennon asentaminen väärälle etäisyydelle tar kasteltavasta pinnasta, tai alentuneesta käyttöjännitteestä johtuva apu valon heikkeneminen (akku on lähes tyhjä).
- **Ulkopuolisia**, kuten sivusta tuleva kirkas valo, tai epäselvä ratamerkin tä.
- **Sisäisiä**, kuten mittaustiedon tarkasteluväli. Ohjain kykenee lukemaan valokennon mittaustiedon 1000 kertaa sekunnissa. Mutta riippuen siitä, miten kennoa luetaan ohjelmassa, isokin kohde saattaa jäädä huomaa matta.

Darth Vader + SegWay kauko-ohjauksella

6. Etäisyyden mittaaminen, muuttujat ja muisti

Tutustutaan muutamaan tapaan mitata etäisyyttä / matkaa sekä tässä yhteydessä käytettäviin antureihin. Opitaan mitä ovat muuttujat ja muistipaikat ja miksi ja miten niitä käytetään ohjelmoinnissa.

Opittua ryhdytään nyt soveltamaan myös arkipäiväisien pulmien ratkomisessa.

Etkot

Tässä harjoituksessa tutustutaan etäisyyden mittaamiseen ultraääni- ja/tai infrapuna anturia käyttäen. EV3:n Education –versio sisältää ultraäänianturin, mikä onkin hyvä ja yleiskäyttöinen anturi. Samalla esitellään myös EV3:n kuluttajaversioon sisältyvä, erikseenkin ostettavissa oleva infrapuna-anturi. Tällä on omat erikois piirteensä, etunsa, sekä käyttökohteensa (esimerkiksi robottijalkapallo). Näihin palataan myöhemmin.

Etäisyyden mittaamisen harjoitteluun tarvitaan harjoitteluympäristö.

- Varaudutaan rakentamaan opetustilan lattialle taskupysäköinnin harjoitusrata, yksinkertainenkin käy. Pysäköityjen” autojen” tulisi olla vähintään 10 cm korkeita, pinnaltaan suunnilleen tasaisia, eli ääntä/valoa hyvin heijastavavia materiaaleja. Pienet pahvilaatikot, laudan kappaleet, kirjapinot jne. ovat loistavia. Kovin epätasaiset pinnat ovat vastaavasti hankalia.
- Tarpeita varataan siten, että niistä saa tehtyä 5+(tyhjä)+2 pysäköidyn ”auton” jonon.

Lisäksi

- Mustaa teippiä ratamerkintöjen tekemiseen (vaalea lattia), tai valkoista teippiä (tumma lattia)

Etäisyyden mittaaminen, tekniikoita ja tuttuja sovellutuksia

Etäisyyden mittausta käytetään monissa arkipäivän sovellutuksissa. Etäisyyttä voidaan mitata monellakin tavalla. Yleisimmin käytetyissä menetelmissä mittalaite lähettää suunnattuja, joko ääni- valo-, tai radiotaajuisia signaaleja. Seuraavaksi se mittaa heijastuisiko osa kohteeseen osuneesta signaalista takaisin sen vastaanottimeen. Signaalit kulkevat aina niille ominaisella vakionopeudella. Valo nopeammin ja ääni hitaammin. Kun tiedetään signaalin edestakaiseen matkaansa käyttämä aika, etäisyys kohteeseen saadaan selville laskemalla,

Koska signaaleissa käytettävät äänet ja valot ovat sellaisia, joita ihmisen korva ja silmä eivät saata havaita, tilanteet joissa etäisyyttä mitataan jäävät meiltä usein huomaamatta.

Ultraääni

Ultraääniantureita käytetään lukuisissa teollisuuden prosessien mittauksissa, lääketieteellisissä tutkimuksissa ja esimerkiksi autojen peruutustutkissa. EV3:n ultraäänianturissa on erilliset kaiutin ja mikrofoni. Näin ei kuitenkaan tarvitse olla. Yksinkertaisintakin kaiutinta voidaan nimittäin käyttää vuoroin signaalia lähettävänä kaiuttimena ja vuoroin heijastunutta signaalia vastaanottavana mikrofonina. Autojen peruutustutkien anturit toimivat juuri näin. Tällä tavalla niistä on saatu sopivan pieniä ja ne voidaan jopa maalata auton värisiksi, niiden toiminnan siitä juuri kärsimättä.

Infrapuna

Infrapunavaloa (IR, Infra Red) käytetään lähes kaikissa kodin kaukosäätimissä. Lyhyen kantaman takia sen käyttö etäisyyden mittaamiseen on rajallista, rajoittuen käytännössä lähinnä teollisuudessa käytettäviin kohteesta tunnistaviin lähestymiskytkimiin. Näitä valmistetaan myös kuumiin, likaisiin ja täriseviin, siis teollisuudessa tavattaviin haastaviin olosuhteisiin. Useimpien kohteesta tunnistavien mallien tunnistusetäisyyttä voidaan myös säätää.

EV3:n infrapuna-anturi on suunniteltu monikäyttöiseksi, osin etäisyyden mittaamisen kustannuksella. Sen yläreunassa on keskellä pieni näkymätöntä valoa lähettävä ledi, joka valaisee ympäristöään varsin laajasti. Vastaanottimia sillä sen sijaan on kaksi, oikea ja vasen. Kun käytetään muita IR-valonlähteitä, se kykenee tunnistamaan etäisyyden lisäksi suunnan josta valo tulee. Verrattomia ominaisuuksia esimerkiksi robottijalkapallossa. Näiden lisäksi sitä voidaan käyttää myös EV3:n omalla IR-kauko-ohjaimella annettujen komentojen vastaanottamiseen.

Laser

Laser mittalaitteiden etuina on niiden tarkkuus, pitkä kantama ja tarkka kohdistus mitattavaan kohteeseen. Siksi niitä näkee (käytetään) hyvin yleisesti erilaisilla työmailla, kuten maanrakennustöissä.

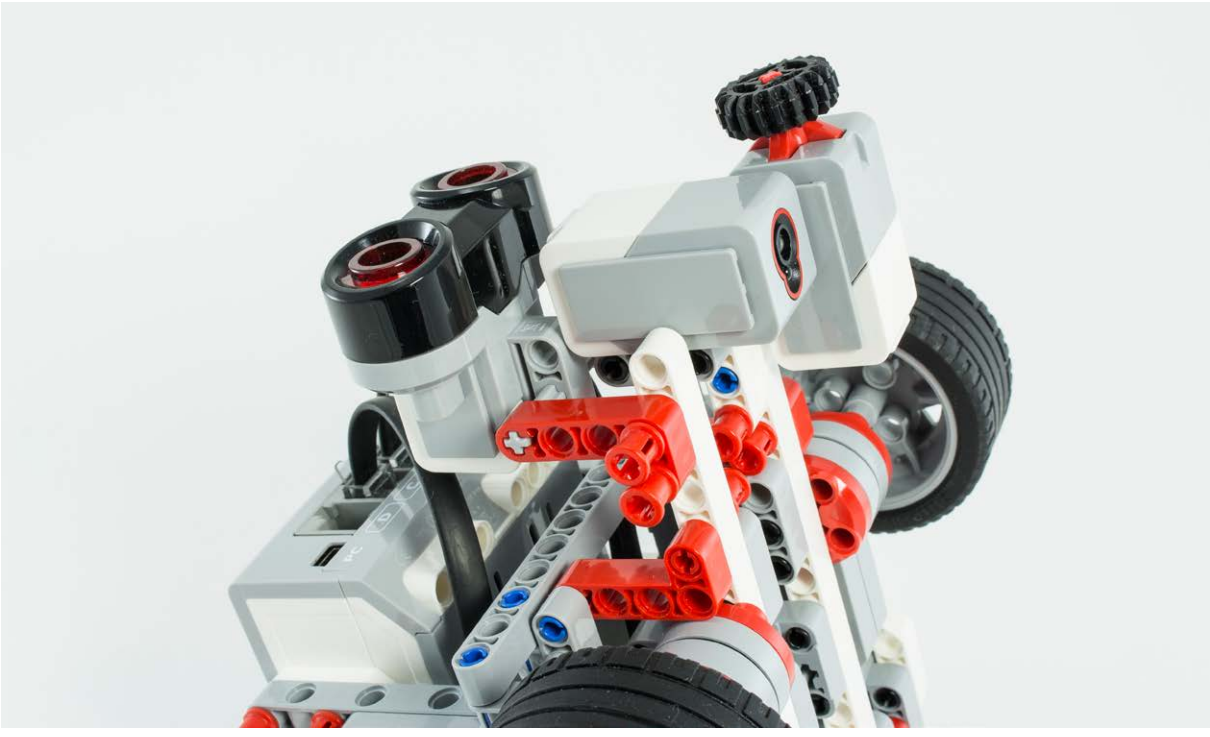
Mikroaallot

Hyvin korkeita radiotaajuuksia, sähkömagneettista värähtelyä, käytetään esimerkiksi ilmailu- ja säätutkissa mm. tarkasteltavien kohteiden etäisyyden mittaamiseen.

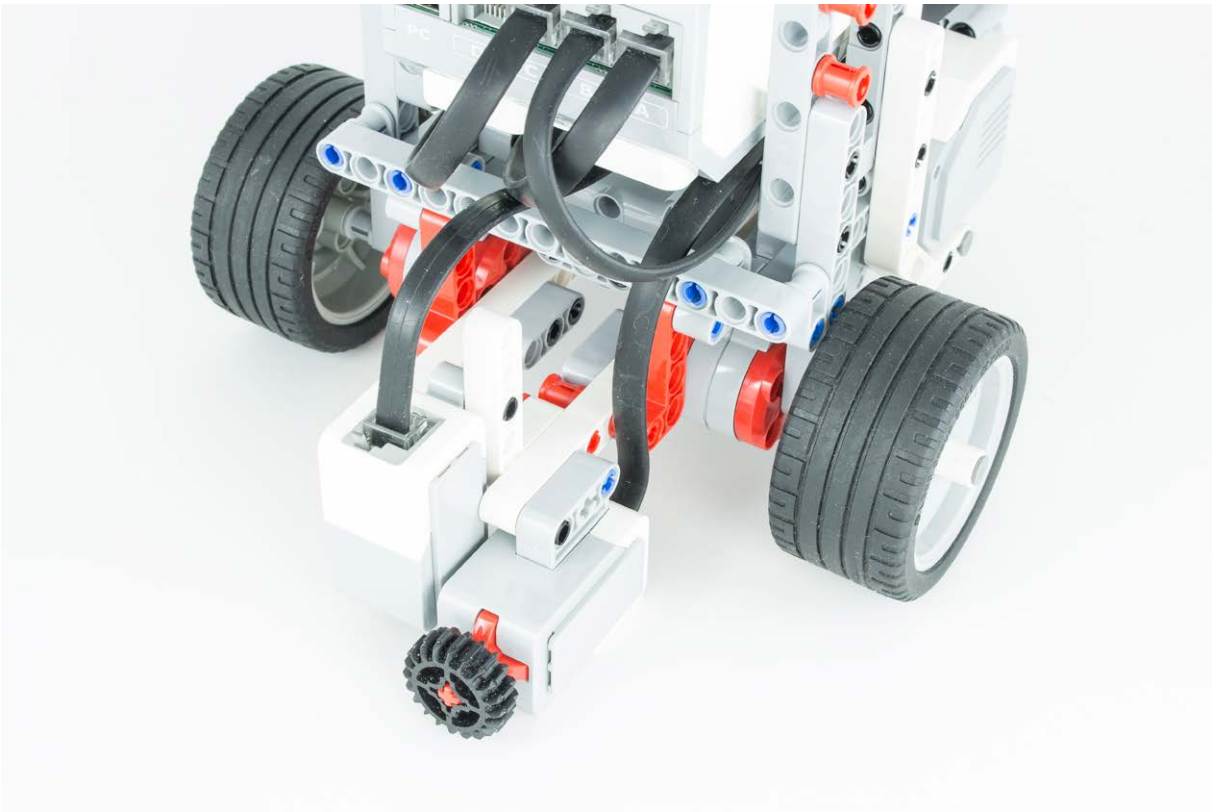
Menetelmiä on toki muitakin, kuten vaikka vesillä liikuttaessa käytetty kaikuluotaus, tässä siis vain yleisimmät. Mitä menetelmää sitten milloinkin käytetään, siihen vaikuttavat tietysti menetelmän soveltuvuus, saatavuus ja usein myös hinta.

Tehtävä 16. Ultraääni- ja infrapuna-anturin toiminta, sekä ohjelman osien kopiointi

Aloitetaan varustelemalla ensin anturi sen kiinnittämiseen tarvittavilla osilla, ja kiinnitetään se robottiin. Sama kiinnitys toimii molempien anturien kanssa.



Alla olevasta kuvasta näkyy kaapelien reitit, miten valokennon kaapeli mahtuu juuri ja juuri ultraäänianturin ja sen kiinnittämisessä käytettävän harmaan akselin väliin.

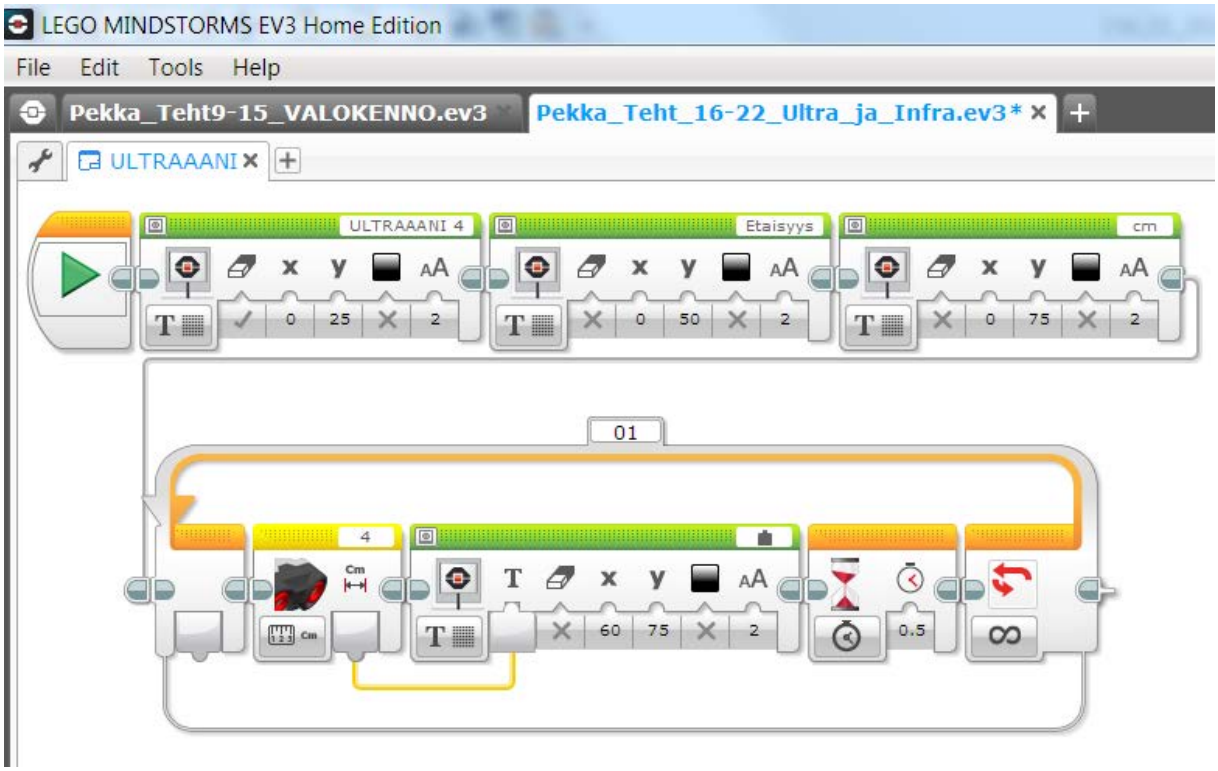


Liitetään ultraäänianturi, tai infrapuna-anturi laitekaapelilla robottiin (Portti 4) ja tehdään apuohjelma, joka näyttää ohjaimen näyttöruudulla anturin mittaaman etäisyyden. Selvitetään kokeellisesti millä alueella anturi kykenee mittaamaan etäisyyttä oikein.

Tehdään ohjelma, joka

- Tyhjentää näytön ja kirjoittaa ensimmäiselle riville ”Ultraaani 4”, tai ”Infrapuna 4” (huom. EV3 ei osaa mm ä ja ö –kirjaimia).
- Toiselle riville ”Etaisyys”.
- Kolmannelle riville ”cm” (tai infrapuna anturiile ”mm” + ja anturin antaman mittaustiedon.
- Mittaustieto päivittyy 0.5s välein, päättymätön silmukka.
- Kuten ohjelmoinnissa yleensä, hyödyntää jo aikaisemmin toimivaksi havaittua koodia. Ultraäänianturin testiohjelma voidaan kopioida lähes sellaisenaa infrapuna-anturin testiohjelmaksi.

Vinkki: Ohjelman suoritus nopeutuu kun silmukka -toimilohkon ulkopuolelle jätetään kaikki ne toimilohkot, joita ei tarvitse suorittaa kuin kerran. Tällä kertaa sellaisia ovat kaikki kirjaimista muodostuvaa tekstiä näyttöruudulle tuottavat toimilohkot.

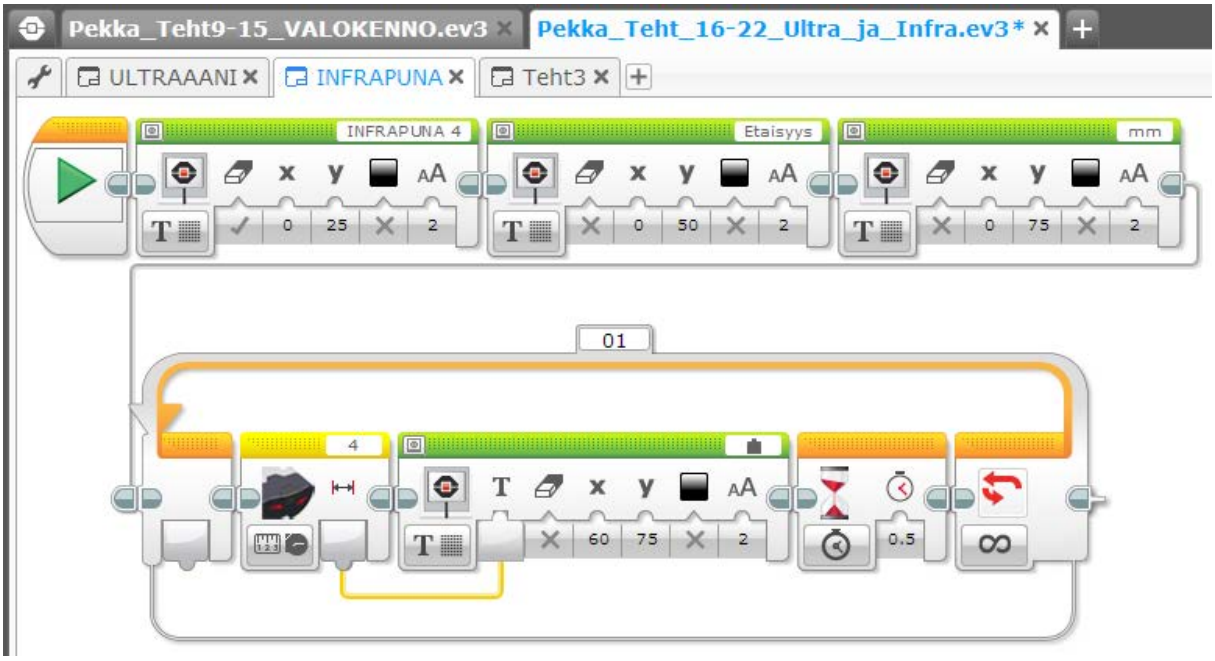


Ohjelman osan kopiointi, esimerkiksi projektista toiseen projektiin

Avataan uusi ohjelma, tai projekti ja kopioidaan toisesta ohjelmasta, tai toisen projektin ohjelmasta osa siihen

- Osoita hiirellä kopioitava kokonaisuus (ikkunoi)
- Katso että kaikki tarvittavat osat saivat siniset kehykset
- Lisää tai poista valintoja hiirellä klikkaamalla, pitäen samalla Ctrl-näppäintä alhaalla
- Valitse sitten Edit -> Copy (tai vie hiiren nuoli kopioitavan toimilohkon yläpakin päälle ja paina Ctrl ja C –näppäimiä yhtä aikaa). Nyt kopioitavat lohkot ovat tallessatietokoneen leikepöydällä.
- Sirrytään siihen projektiin ja ohjelmaan, johon kopioitu ohjelman osa halutaan viedä
- Klikkaa ohjelmointiohjelman työpöydältä kohtaa johon haluat tuoda kopioidunohjelman osan ja valitse Edit -> Paste (tai paina Ctrl ja
- V –näppäimiä yhtä aikaa
- Tietokone kopioi nyt lohkot välimuististaan ohjelman työpöydälle
- Kopioituja lohkoja voi siirrellä, yhdistellä ja muokata nyt ihan normaalisti.

Avataan uusi ohjelma ja kopioidaan ultraäänianturin testiohjelma siihen. Aloitustoimilohkoja tarvitaan aina yksi, mutta vain yksi. Päivitetään tekstit ja vaihdetaan anturin toimilohko. Ohjelma on valmis kokeiltavaksi.



Sama ohjelma tehtynä nyt infrapuna-anturille.

Ryhdytään seuraavaksi selvittämään kokeilemalla millaiseen etäisyyden mittaamiseen nämä anturit soveltuvat.

Näyttöruudun näkymät

Tehtävä 17, Ultraäänianturilla mittaaminen

Kokeile ensin laitteen toimintaa mittaamalla etäisyyksiä erilaisiin ja eri etäisyyksillä oleviin kohteisiin.

Kiinnitä huomiota (kirjaa havainnot vihkoon!)

- Siihen, näyttääkö mittaus edes suunnilleen oikein.
- Vaikutavatko lähettyvillä mahdollisesti olevat muut esineet, tai pinnat mittauksen luotettavuuteen? Miksi?
- Mittasitko kohtisuorassa, vai vinossa olevaa pintaa?
- Mikä vaikuttaa lyhimmältä etäisyydeltä, jolloin mittaus vielä tuntuu luotettavalta?
- Entä mikä oli suurin lukema jonka onnistuit saamaan?
- Mutta mikä vaikutti pisimmältä etäisyydeltä, jolta vielä yrittäisit tunnistaa esimerkiksi kahvikupin tms., kun tämän pitäisi tapahtua luotettavasti?

Tehtävä 18. Infrapuna-anturilla mittaaminen

Kokeile ensin laitteen toimintaa mittaamalla etäisyyksiä erilaisiin ja eri etäisyyksillä oleviin kohteisiin. Huomioi myös kohteen väri ja kiilto.

Kiinnitä huomiota (kirjaa havainnot vihkoon!)

- Siihen, missä olosuhteissa mittaus toimii suunnilleen oikein.
- Vaikutavatko lähettyvillä mahdollisesti olevat muut esineet, tai pinnat mittauksen luotettavuuteen? Miksi?
- Mittasitko kohtisuorassa, vai vinossa olevaa pintaa?
- Vaikuttaako muu valaistus / ulkoa tuleva valo mittauksen luotettavuuteen?
- Mikä on lyhin ja pisin etäisyys, joita EV3:n infrapuna-anturilla voi mitata ja mitä lukemaa se näytti mitta-alueen ylittyessä?
- Mikä vaikutti pisimmältä etäisyydeltä, jolta vielä yrittäisit tunnistaa esimerkiksi kahvikupin tms., kun tämän pitäisi tapahtua luotettavasti?

Muisti

Massamuisti ja käyttömuisti, eli välimuisti

Ohjelmoitavissa laitteissa, kuten tietokoneissa on pääsääntöisesti kahta erilaista muistia. Nimensä mukaisesti massamuistit ovat isoja tietovarastoja.

Massamuistina käytettävän tavallisen-, tai SSD-kovalevyn sisältö ei katoa, vaikka laite sammutetaan. Niitä käytetään mm. ohjelmien ja tiedostojen tallentamiseen.

Käyttömuisti sen sijaan pyyhkiytyy tyhjäksi, kun laite sammutetaan. Sen luku- ja kirjoitus nopeus on suuri, josta syystä sitä käytetään lyhytaikaiseen tiedon tallentiointiin. Sovellustason, kuten pelien tai vaikka robottien ohjelmissa käytetään juuri tätä lyhytikäistä käyttömuistia.

Tietokoneen kovalevy ja käyttömuisti, jota kutsutaan myös muistikammaksi

Muistipaikka on kuin pahvilaatikko varastossa

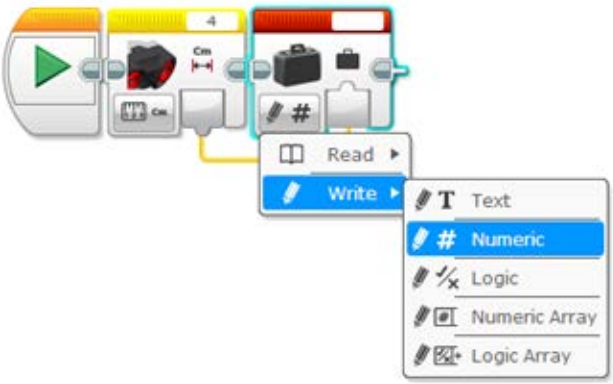
”Muistia ja sen yksittäisiä muistipaikkoja voidaan ajatella vaikka suunnattoman suurena joukkona tyhjiä pahvilaatikoita isossa varastohallissa. Jokaisen laatikon kylkeen on kirjoitettu sen on oma nimi tai numero, jonka perusteella kone tunnistaa sen.” Monissa tapauksissa, kuten EV3:n ohjelmoinnissa, käyttäjälle on annettu mahdollisuus kiinnittää laatikoihin vielä lisäksi ihan omat, itse keksityt ”nimilaput”.

Muistipaikan tyyppi

Aivan kuten pahvilaatikon koko määräytyy siihen laitettavan tavaran perusteella, niin myös muistipaikan koko (tyyppi) määritellään siihen tulevan tiedon mukaan. Yksinkertainen Tosi/Epätosi –tieto tarvitsee vain vähän tilaa (Logic True/False). Numerot ja tekstit tarvitsevat hiukan isomman muistipaikan ja jono, eli vektori-muotoinen data vielä isomman (Array).

Miksi sitten on olemassa eri kokoisia muistipaikkoja? Ohjelmoitavalla laitteella on yleensä käytössään vain rajallinen määrä muistia. Muisti on kuin jo aiemmin mainittu varastohalli. Jättimäisiä laatikoita mahtuisi tähän halliin vain muutamia. Pienenpieniä laatikoita sen sijaan mahtuisi miljoonia, mutta ne eivät sovellu kaikkiin tarpeisiin. Käyttämällä sopivan kokoisia laatikoita, eli muistipaikkoja, kaikki tarvittava tieto mahtuu muistiin.

HUOM! Muistipaikan tyyppi pitää muistaa valita ennen sen nimeämistä.

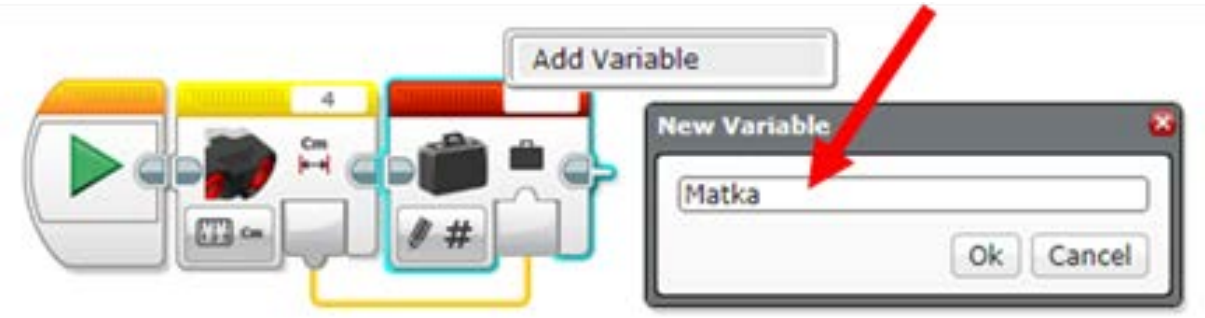


Kuvassa ultraäänianturin tuottama numeerinen (Numeric) mittaustieto kirjoitetaan (Write) muistipaikkaan, jolle on annettu nimeksi ”Matka”.

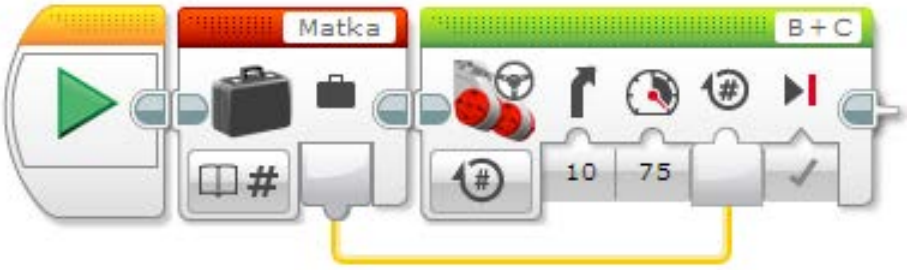
Muistipaikan nimeäminen

Lyhyiden tallennettua tietoa osuvasti kuvaavan nimien käyttäminen helpottaa ohjelmointityötä ihan hurjasti.

EV3:n ohjelmoinnissa muuttujia voi tallentaa muistipaikkoihin, joiden symbolina on salkku. Muistipaikan nimi tulee toimilohkon yläreunaan, valkoiseen laatikkoon. Huom! muista valita muistipaikan tyyppi ennen sen nimeämistä! Tyypin vaihtaminen onnistuu jälkeenpäin vain tuhoamalla ensin muistpaikka muistipaikkojen hallinnoinnin kautta.

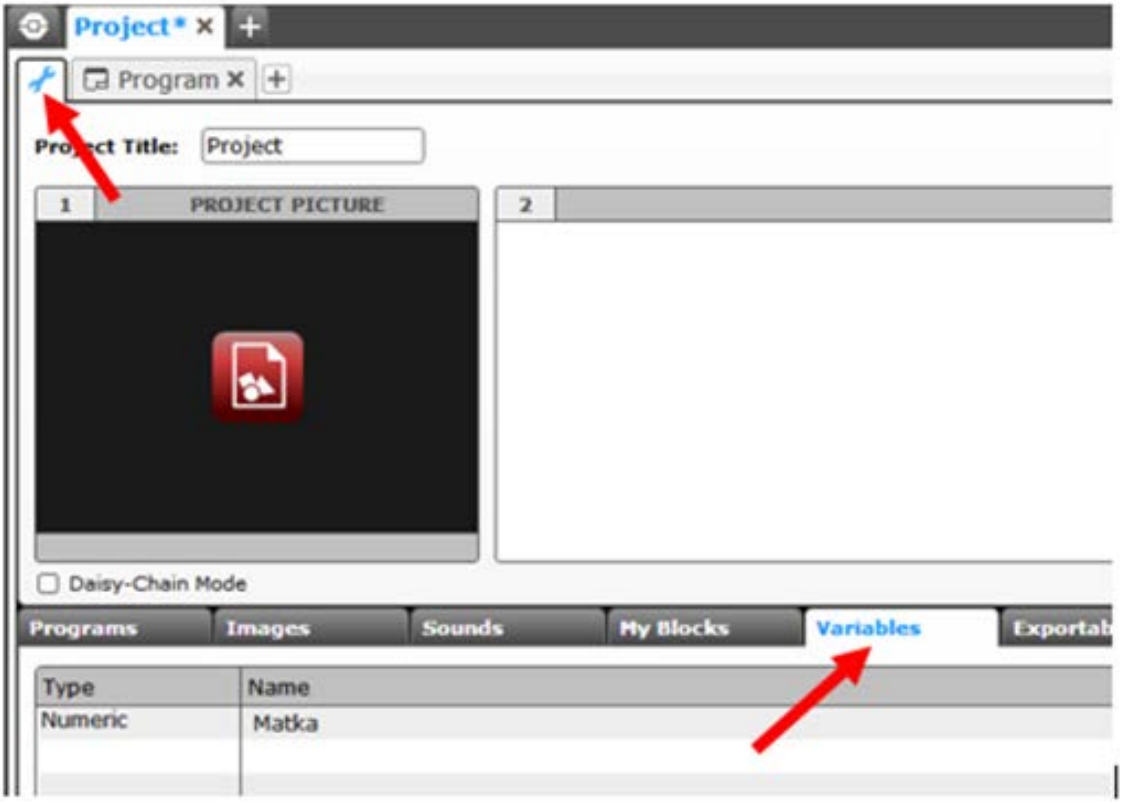


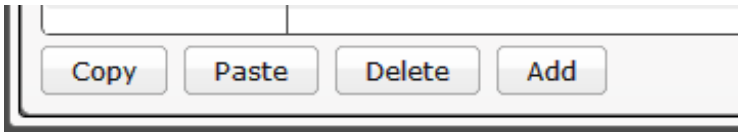
Kun aikanaan muistipaikasta ”Matka” halutaan lukea sinne taltioitu tieto, valitaan ensin typpi, eli ”Read”, ”Numeric” ja ohjelma lohkon nimi-ikkunaa klikkaamalla avautuvasta luettelosta luettavan muistipaikan nimi, ”Matka”.



Muistipaikkojen hallinnointi

Muistipaikkojen hallinnointi EV3:n ohjelmoinnissa tehdään projektin hallinnan kautta. Klikkaa ensin kiintoavaimen kuvaa ja valitse sitten ”Variables” näkymä.





Muistipaikan tyypin muuttaminen ei onnistu enää sen nimeämisen jälkeen. Ainoa keino vaihtaa muistipaikan tyyppi toiseksi on tuhota se ensin täällä ja perustaa se kokonaan uudestaan.

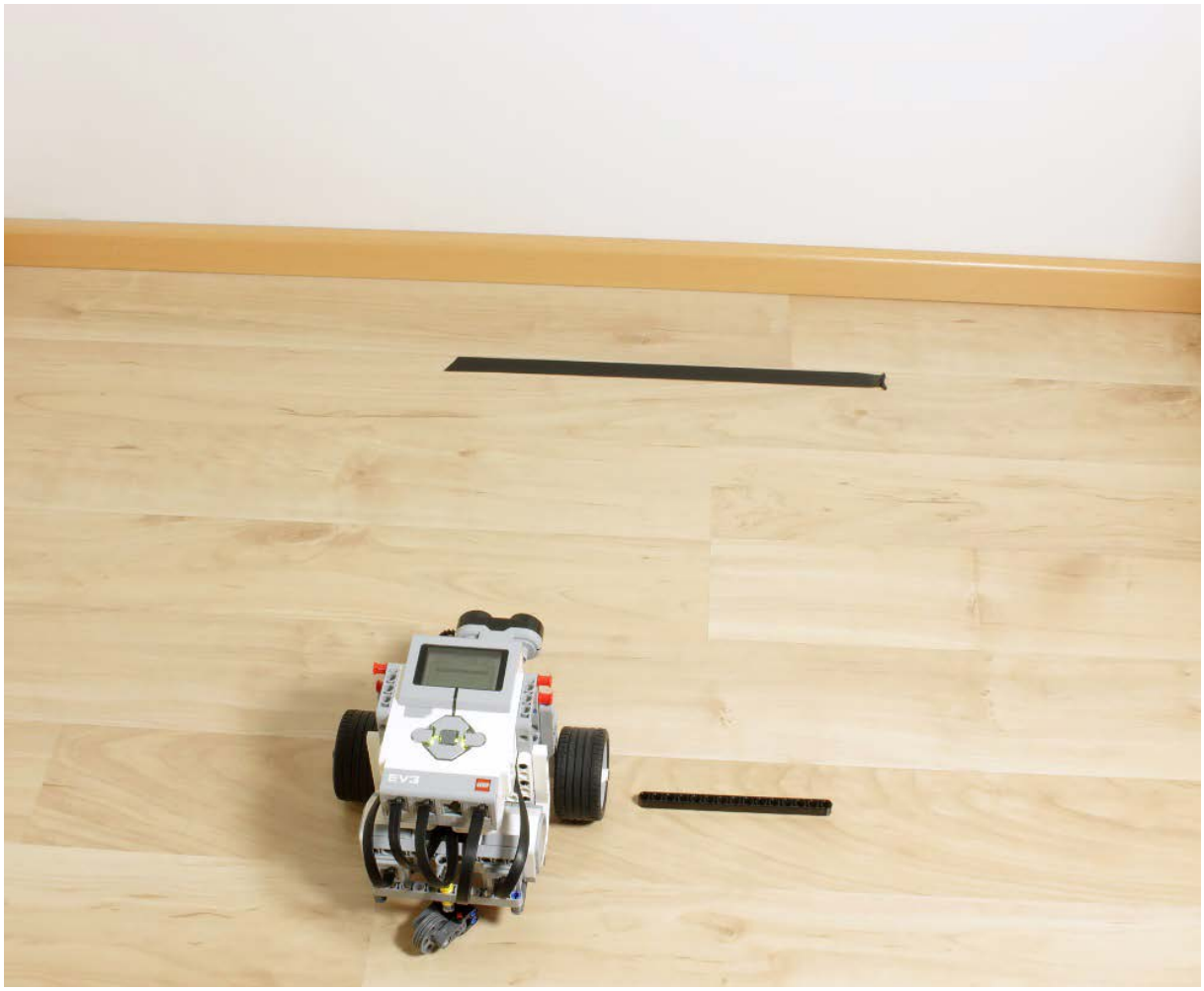
Jos ja kun muistipaikka halutaan tuhota, sen voi tehtä täällä. Huomaa, että kaikki täällä tehdyt muutokset eivät kuitenkaan päivity ohjelman puolelle (kuten kokenut ohjelmoija saattaisi toivoa). Pääsääntönä kun muistaa, että ohjelman puolella lisätään ja täällä voi poistaa, sillä pärjää.

Tehtävä 19. Yksi meno-paluu, kiitos

Tämä on helppo ja hauska kilvoittelu. Tehtävänä on ohjelmoida robotti kulkemaan kohti seinää ja pysähtymään 10 cm ennen sitä. Tässä robotti odottaa sekunnin antaen samalla äänimerkin. Lopuksi robotti peruuttaa tarkasti takaisin siihen kohtaan mistä se lähti ja antaa vielä äänimerkin. Pienin sallittu nopeus on tässä tehtävässä 10%.

Muuttuva etäisyys

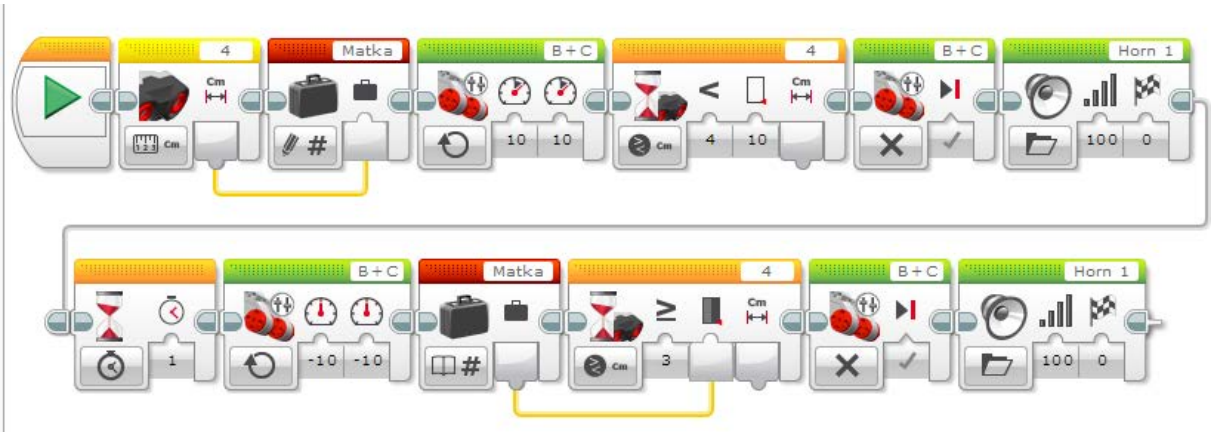
Suorituspaikaksi kannattaa valita sellainen, joka sopii hyvin etäisyden mittaamisessa käytettävälle anturille. Seinän lähelle merkitään teipillä 10 cm viiva. Hauskuutta kilpaan saadaan sillä, että ***lähtöpaikka voi vaihdella*** vaikka joka yrityksellä, sopiva etäisyys on 40-60 cm seinästä. Lähtöpaikka merkitään aina ennen suoritusta, vaikka asettamalla jokin pieni esine tai viivotin sen merkiksi. Kun sitten robotti palaa, nähdään miten tarkasti se pysähtyy juuri samaan paikkaan, mistä se lähti äsken liikkeelle. Kenen robotti on tarkin?



Robotin ohjelmoiminen kulkemaan suoraan kohti seinää ja pysähtymään käy vielä helposti. Jotta robotti osaisi palata tarkasti omaan lähtöpaikkaansa, sillä pitää olla muistissa tieto siitä miten kaukana tämä oli.

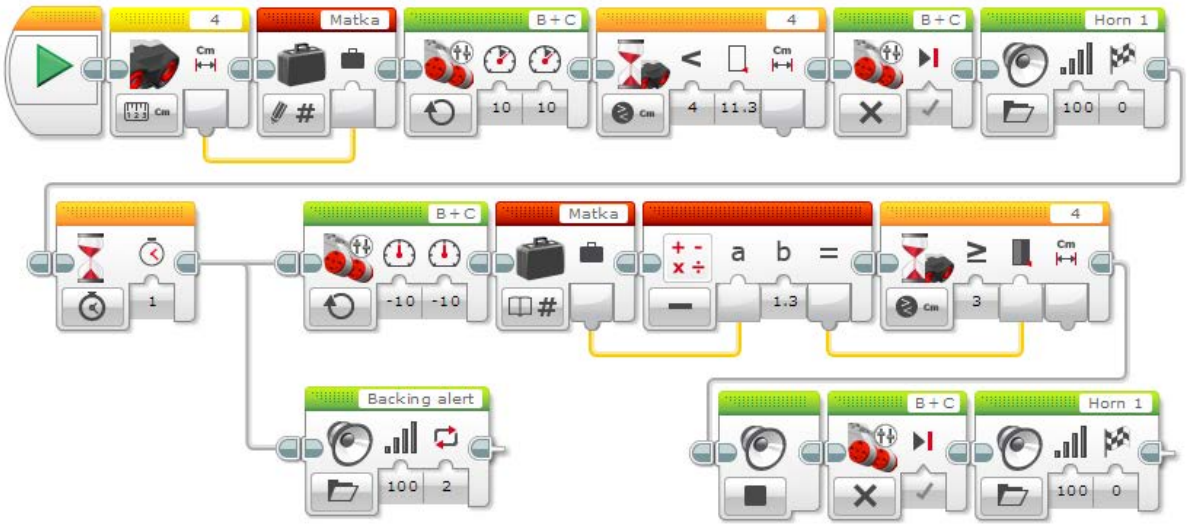
Robotin pitää siis mitata tarkka etäisyys seinästä ja tallentaa tämä tieto muistiinsa ennen liikkeelle lähtöään. Paluumatkalla verrataan mittaustietoa tallennettuun tietoon ja pysähdytään kun mittaus näyttää samaa kuin lähtiessä (on suurempi tai yhtäsuuri kuin).

Tehtävän 19 voi ratkaista esimerkiksi tällaisella ohjelmalla.



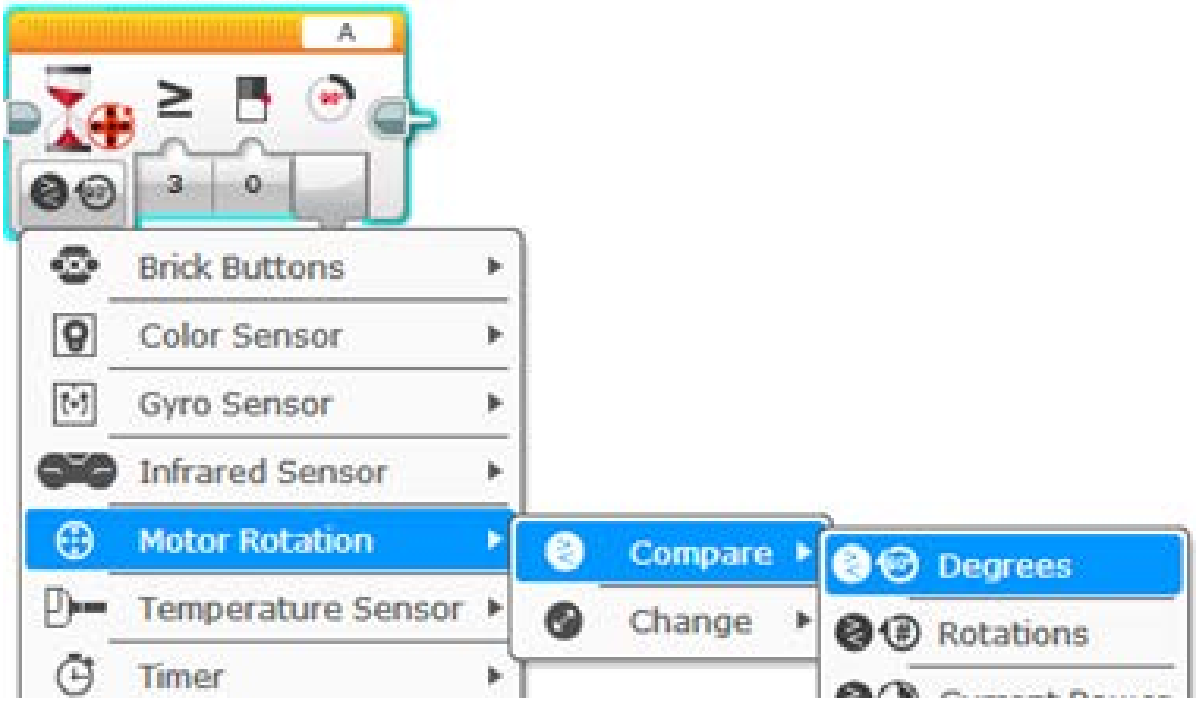
Siinä ei kuitenkaan ole otettu huomioon sitä, että moottorien pysähtyminen ottaa oman pienen hetkensä, jonka kuluessa robotti ehtii liikkua vielä noin senttin matkan.

Alla tuunattu koodi, jossa jarruttaminen aloitetaan 1.3cm aikaisemmin, molempiin suuntiin. Paluumatkan osalta tämä muutos edellyttää jarrutusmatkan vähentämistä muistiin taltioidusta lähtöpaikan tiedosta.

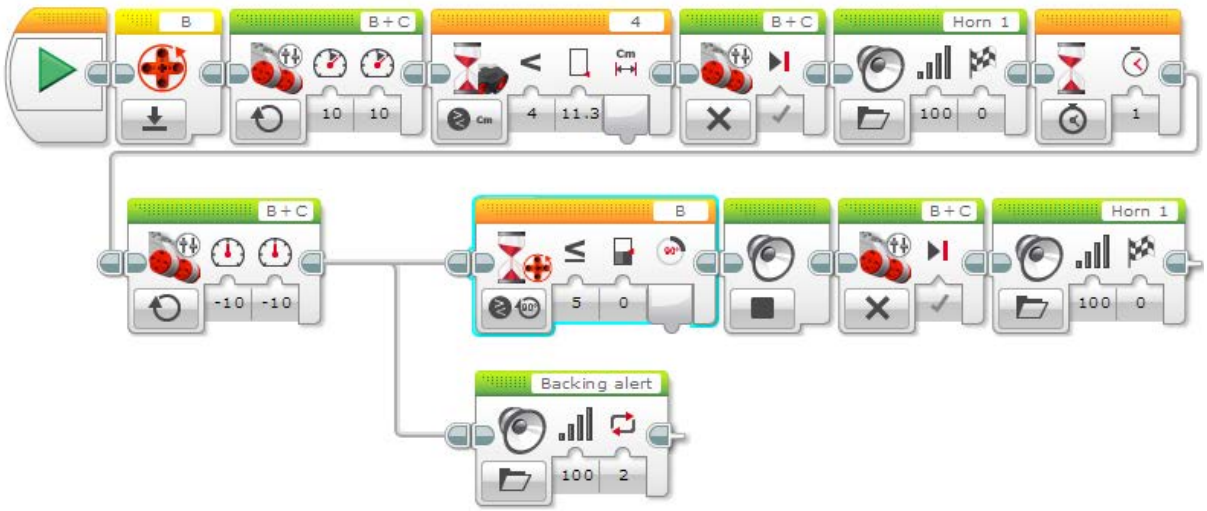


Tehtävä 20. Kuljetun matkan mittaaminen

Tehtävän 19 voi ratkoa myös hyödyntämällä moottorien pulssianturien tuottamaa tietoa kuljetusta matkasta. Pulssianturi nollataan heti ohjelman alussa, ennen robotin liikkeelle lähtöä. Pysähtyminen 10cm etäisyydelle seinästä tehdään ultraäänianturin mittaaman tiedon pohjalta, kuten aiemminkin. Kotiasemaan palataan ohjelmoimalla ajastin-lohko (Wait) odottamaan pulssilaskurin lukeman paluuta nollaan (=lähtöpaikka).



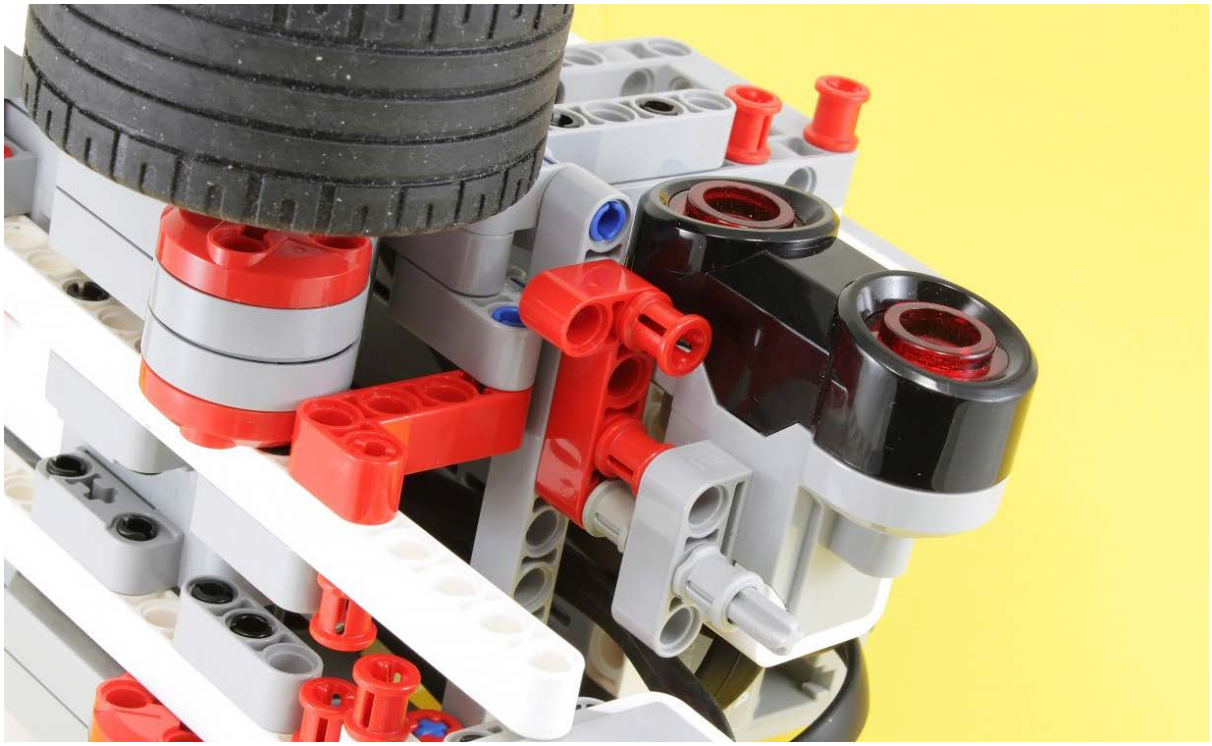
Moottorin pulssianturia hyödyntävä ratkaisu



Tehtävä 21. Opetetaan robotti tekemään taskupysäköinti

Useat autojen valmistajat ovat alkaneet varustaa tuotteitaan taskupysäköintiä avustavilla laitteilla. Tässä tehtävässä rakennetaan robotti joka osaa tehdä taskupysäköinnin.

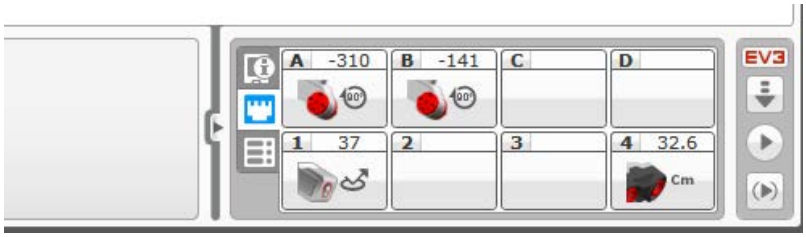
Muokataan ensin ultraäänianturin kiinnitystä niin että saadaan se ”katsomaan” oikealle ja kiinnitetään se sitten alemman kuvan osoittamaan paikkaan.



Seuraavaksi rakennetaan sopiva testirata merkitsemällä lattiaan mustalla teipillä suora linja (1.5m) ja asettamalla siitä n.10 cm etäisyydelle muutamia kirjapinoja.

Ohjelmoidaan robotti kulkemaan viivaa pitkin ja etsimään kirjapinojen rivistä sopivaa, kulkusuunnassa vähintään 25cm pitkää tyhjää tilaa. Ohjelman teossa kannattaa hyödyntää edellisessä harjoituksessa tehtyä viivanseuranta-ohjelmaa, version voit valita vapaasti. Rauhallisesti ja suoraan etenevä robotti löytää varmasti paremmin tiensä ruutuun. Tähän voidaan käyttää moottorin pulssi-anturin tuottamaa tietoa kuljetusta matkasta. Ohjelmointikaapelin ollessa kytkettynä antureiden tuottamat tiedot näkyvät myös ohjelmointiohjelman laitemonitorin portti-näkymässä.

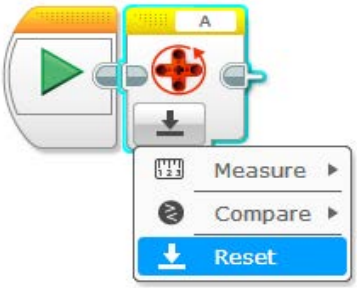
Laitemonitori



Moottorin pyörimisen mittaaminen, pulssitieto

Pyöritä kokeeksi robotin oikeaa pyörää käsin (moottori C). Lukema kasvaa kun pyörää pyöritetään myötäpäivään, ja vastaavasti se pienenee kun pyörää pyöritetään vastapäivään. Yksi kokonainen pyörähdys on aina 360°, mikä on robotin kulkemana matkana 176mm. Tästä saadaan laskemalla etsittävän pysäköintipaikan pituudeksi robotin pyörän pyörähdyksinä ilmaistuna 512°, mikä on hiukan yli puolitoista kierrosta.

Moottorin pyörähdyksistä kertova muistipaikka on yksi monista ohjausjärjestelmän sisään kirjotetuista vakiomuistipaikoista, joita käyttäjän ei tarvitse itse ensin määritellä ohjelmaan. Sen voi nollata kirjoittamalla nollaus –käskyn ohjelmaan (Reset).



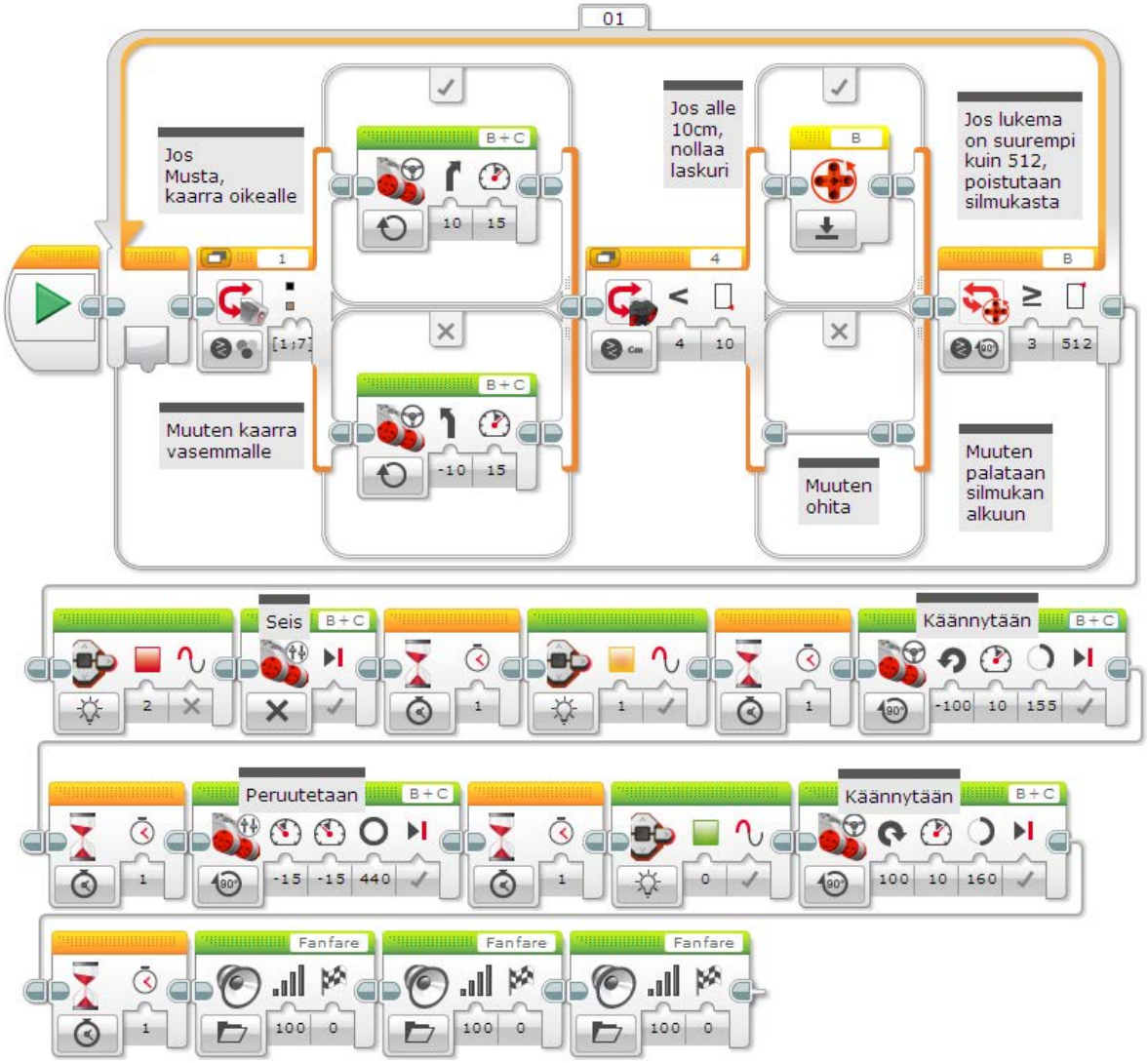
Tee tämän toimintakuvauksen mukainen ohjelma ja kokeile sitä

1. Jos kenno näkee mustaa, robotti kaartaa oikealle, muuten vasemmalle
2. Jos robotin oikealla puolella on jotakin lähempänä kuin 10cm, pulssilaskenta nollataan.
3. Jos pulssilaskenta näyttää suurempaa lukemaa, kuin 512 (eli 25cm), viivan seuraaminen keskeyttetään ja molemmat moottorit pysäytetään.
4. Robotti kääntyy, peruuttaa ruutuun ja oikaisee asentonsa ”kadun” suuntaiseksi.
5. Lopuksi robotti soittaa vapaavalintaisen fanfaarin.

Voisiko robottin merkkivaloa käyttää jarruvalona?

Entä vilkkuna?

Esimerkkiratkaisu tehtävään 21



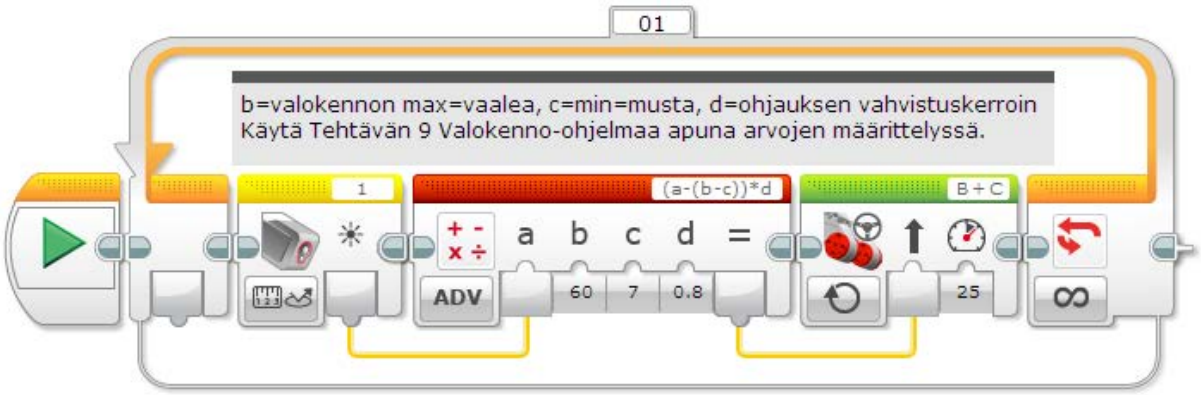
Tehtävä 22. Turvaväli & hätäjarrutus, 2x P-säädin, säestyksellä

Autojen turvajärjestelmiin tekee tuloaan myös etäisyyttä edellä kulkevaan autoon, eli turvaväliä tarkkailevat ja ylläpitävät järjestelmät. Tässä tehtävässä on tavoitteena saada robotit kulkemaan rataa jonossa, samalla tavalla, turvavälit säilyttäen.

Tehtävään tarvitaan useita robotteja, sekä pidempi, mustalla teipillä merkity rata. Kaarteiden on syytä olla hyvin loivia, kääntösäde min 50cm.

Ohjelma

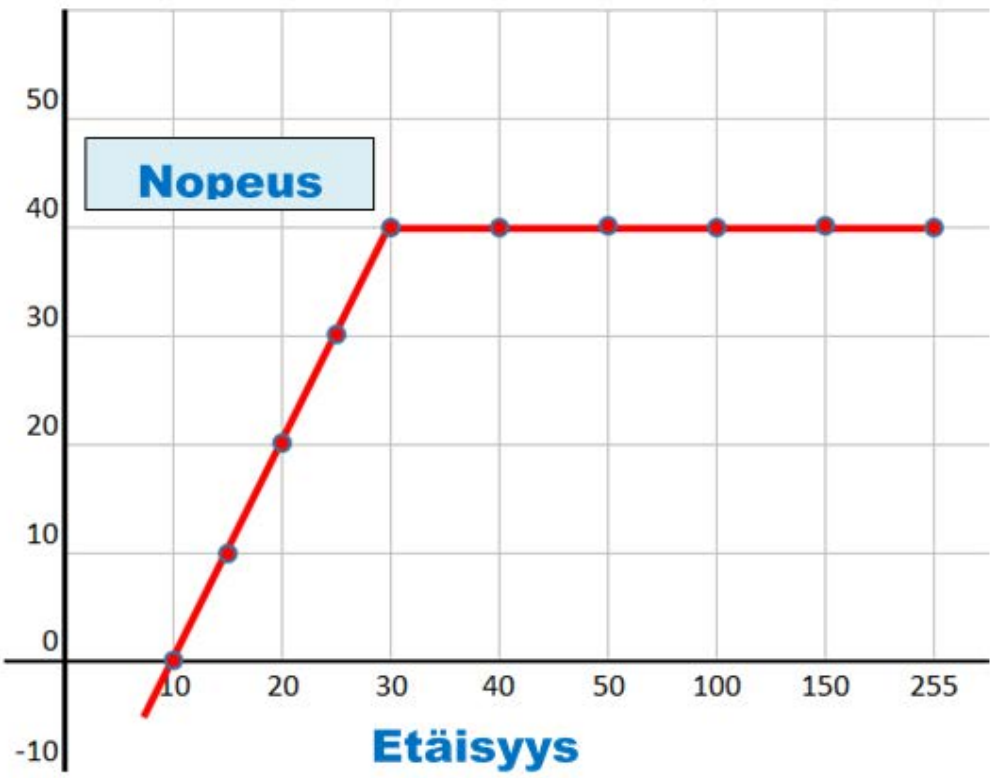
1. Siirretään ultraäänianturi takaisin alkuperäiselle paikalleen, katsomaan eteenpäin.
2. Tehdään viivanseuraaja, Move Steering –moottoritoimilohko ja ohjauksen P-säätö.



Ohjelmoinnissa ei ole yhtä oikeaa tapaa tehdä asioita (tässä on tämän kirjan viides erilainen viivanseurantaohjelma).

3. Seuraavaksi tehdään nopeudensäätö, P-säädin, nopeusalue 0-40%. Mitataan ultraäänianturilla etäisyyttä edellä kulkevaan ajoneuvoon ja pidetään välimatka vähintään 10 cm suuruisena (mutta enintään 30cm). Mittauksen max on 255cm.

Seuraava kuva havainnollistaa haluttua nopeuden ja etäisyyden suhdetta.



- Kuvasta voidaan nähdä, miten nopeuden halutaan pysyvän aina samana (40%), JOS etäisyys on yli 30cm. Tässä on siis mahdollista käyttää If-Then –toimilohkoa.

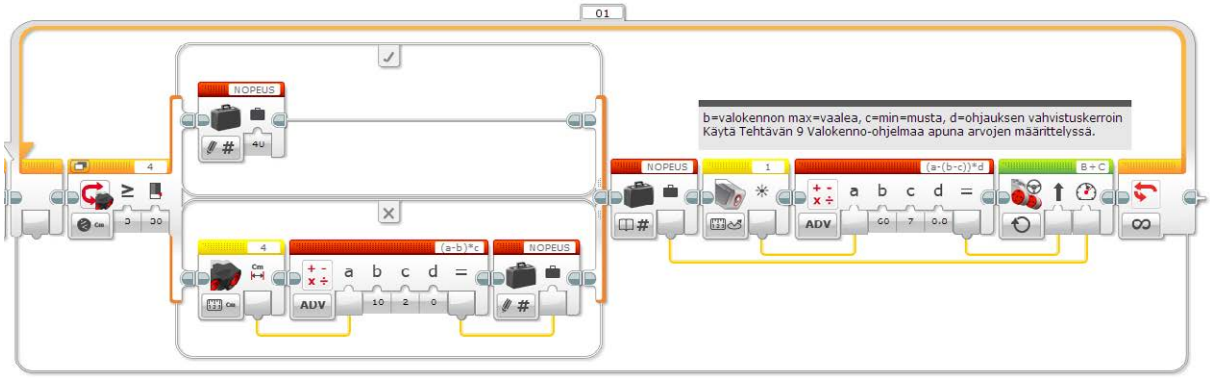
Lisäksi tarvitaan laskentaa

- Jos etäisyyden mittauksesta vähennetään 10cm, saadaan etäisyyden ja nopeuden nollakohdat kohtaamaan.
- Nopeuden muutos etäisyydellä 10 - 30cm on suoraviivainen, lineaarinen. Kun etäisyys muuttuu 10 yksikköä, nopeuden pitää muuttua 20 yksikköä. Lineaarinen muunnos tehdään kertolaskulla.
- Tehdään nopeudensäädön osuus ohjelmasta. Ylemmällä polulla asetetaan muuttujan (muistipaikan) NOPEUS arvoksi aina 40%, kun etäisyys on suurempi tai yhtäsuuri kuin 30cm.

Nopeudensäätöpiiri, laskukaava: Nopeus = (Etäisyys – 10cm) x 2

Alemmalla polulla luetaan ensin etäisyyden arvo anturilta, lasketaan nopeus ja tallennetaan sen arvo muuttujan NOPEUS –arvoksi.

Liitetään nopeudensäätöpiiri ohjauksensäätöpiirin eteen ja lisätään niiden väliin muutujan NOPEUS arvon lukeminen, ja kytketään tämä tieto moottoritoimilohkon nopeustiedoksi.



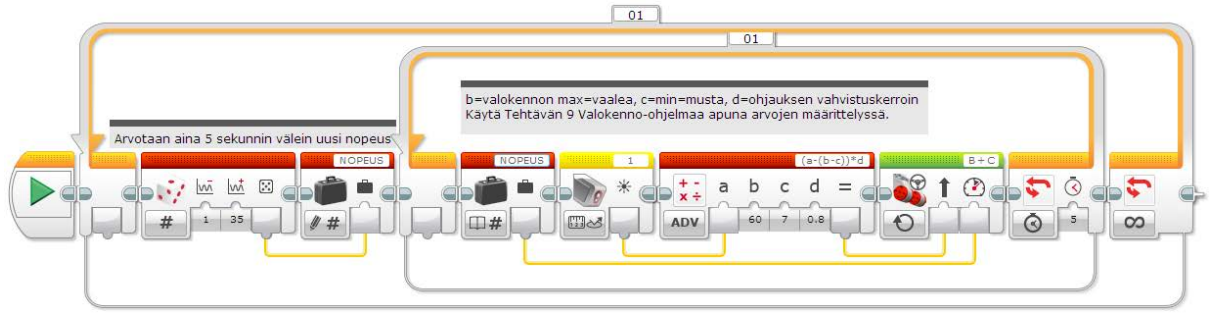
Kokeillaan robottia

Miten robotti reagoi edessä oleviin esteisiin? Pysähtyykö se 10cm etäisyydelle esteestä? Saadaanko robotti seuraamaan kättä, tai jalkaa? Entä osaako se peruuttaa?

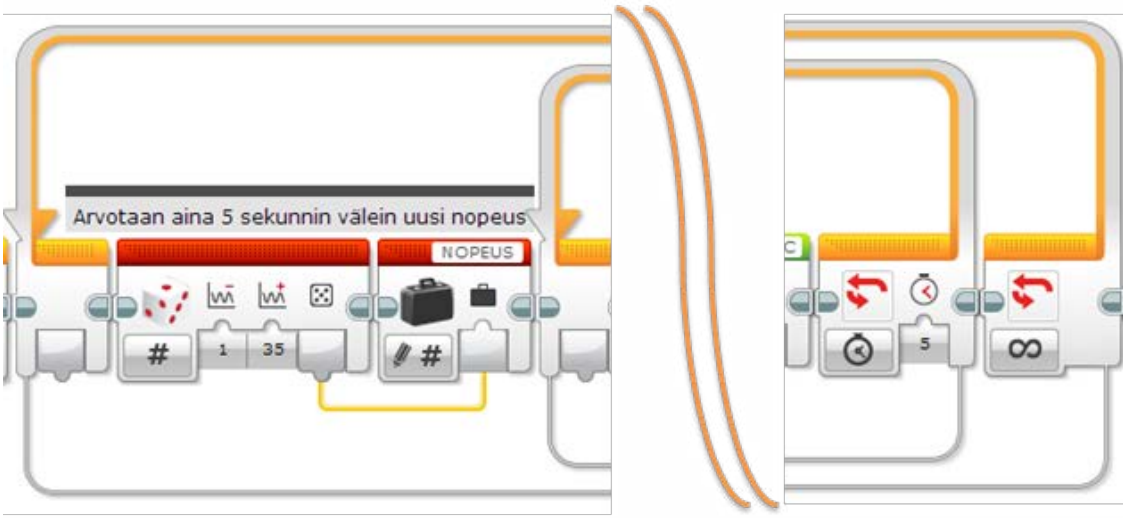
Jos robotti meinaa karata reitiltä, lisätään ohjauksen säätöpiirin vahvistusta (vahvistuskerroin).

Ryhmä tarvitsee johtajan

Mikäli käytettävissä on useampia robotteja, ne voi laittaa radalle samanaikaisesti. Ryhmän vetäjäksi voidaan ohjelmoida yksi robotti, jonka nopeusohje tehdään satunnaisluvun pohjalta (ohje alla)



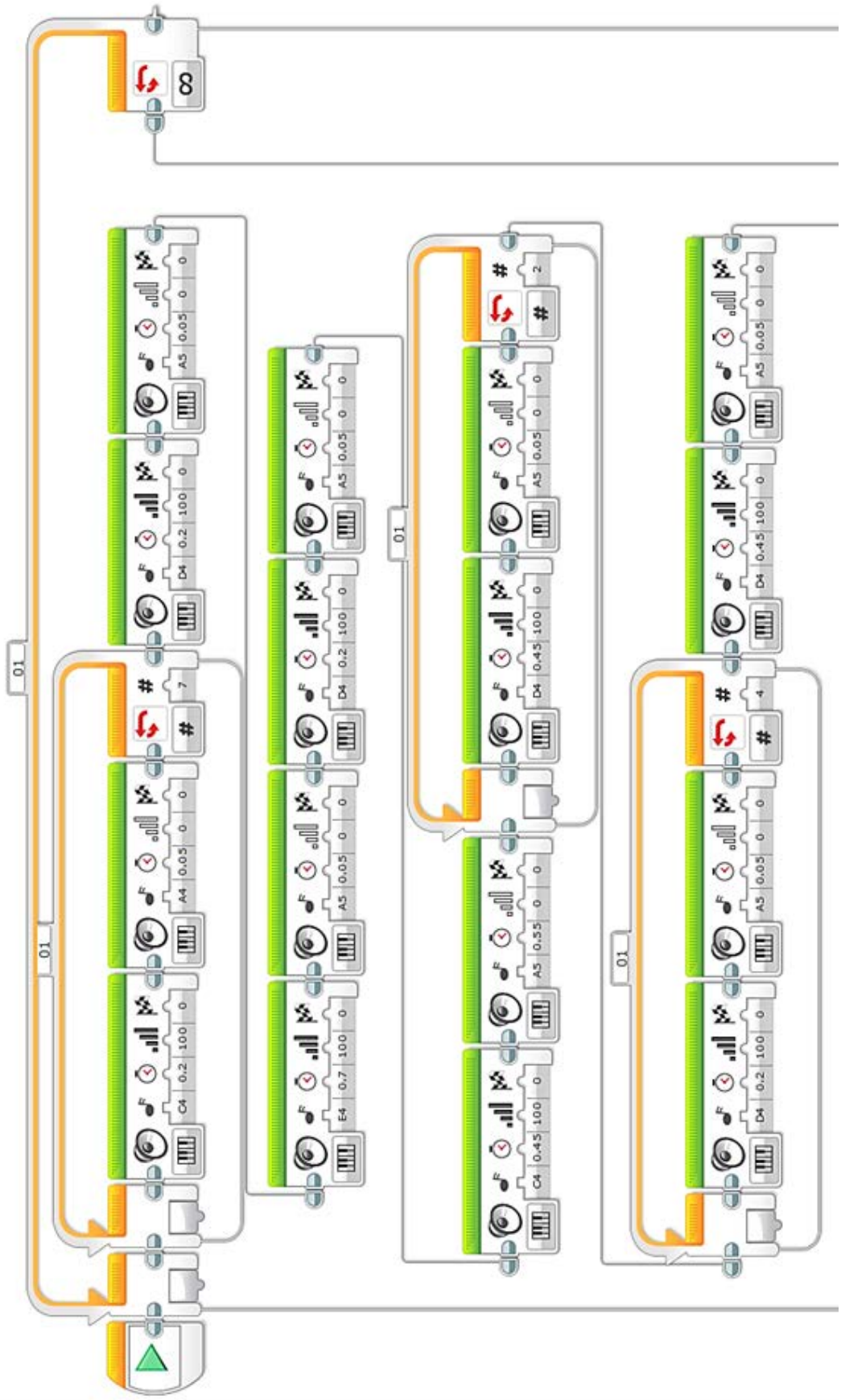
Satunnaisluvun pienimmäksi arvoksi annetaan 1, ja suurimmaksi 35. Sisemmän, ohjauksen säätöpiirin silmukan päättymisen ehdoksi määritellään 5 sekuntia -> nopeusohje vaihtuu uuteen aina viiden sekunnin välein.



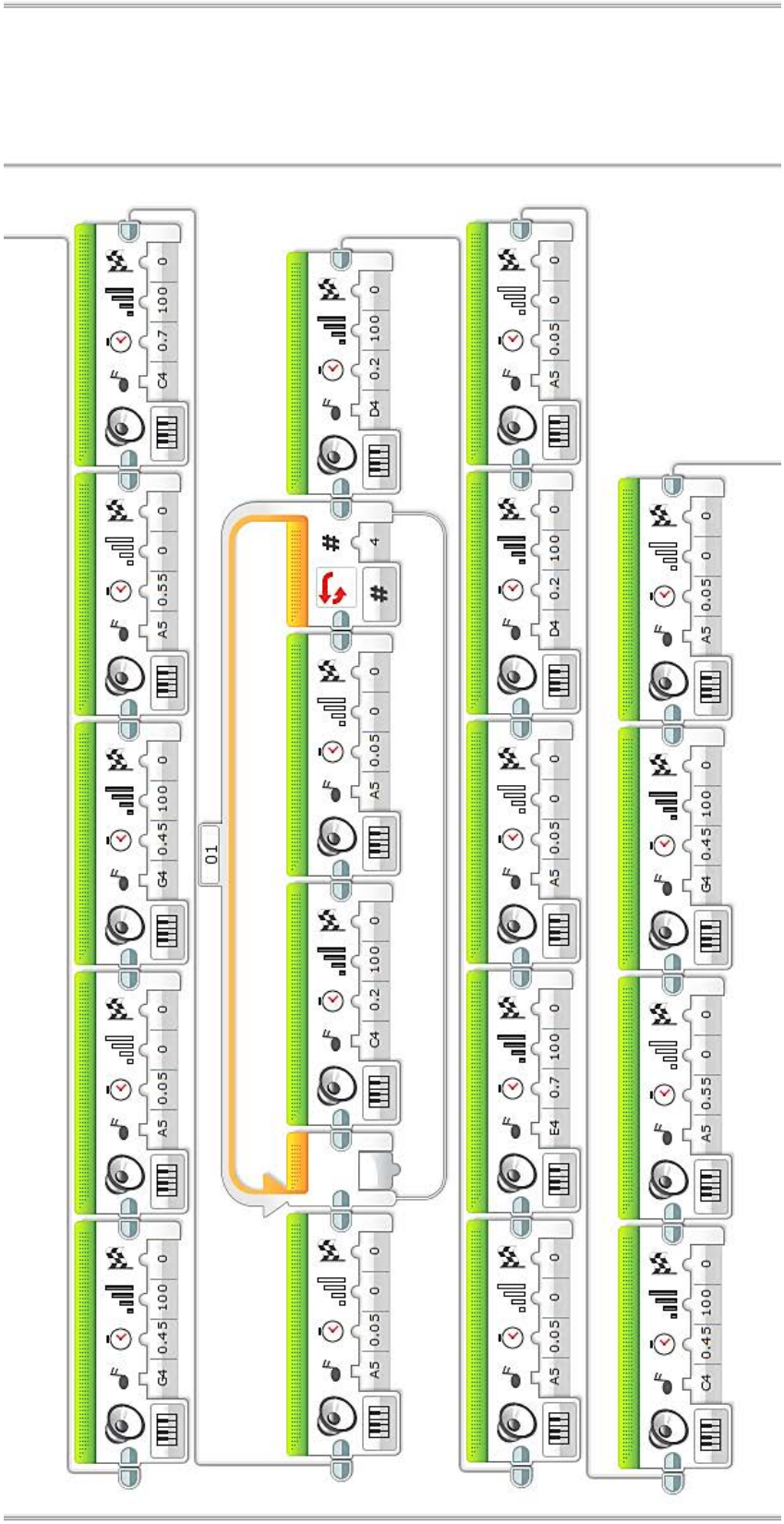
Elefanttimarssi

Robottien paraati on miltei valmis. Ohjelmoidaan vielä yksi robotti soittamaan Elefanttimarssia kulkiessaan. Ohjelmasta on paremmat kuvat seuraavilla sivuilla. Kun kaikki robotit on saatu toimimaan, voidaan pitää suuri juhlaparaati.

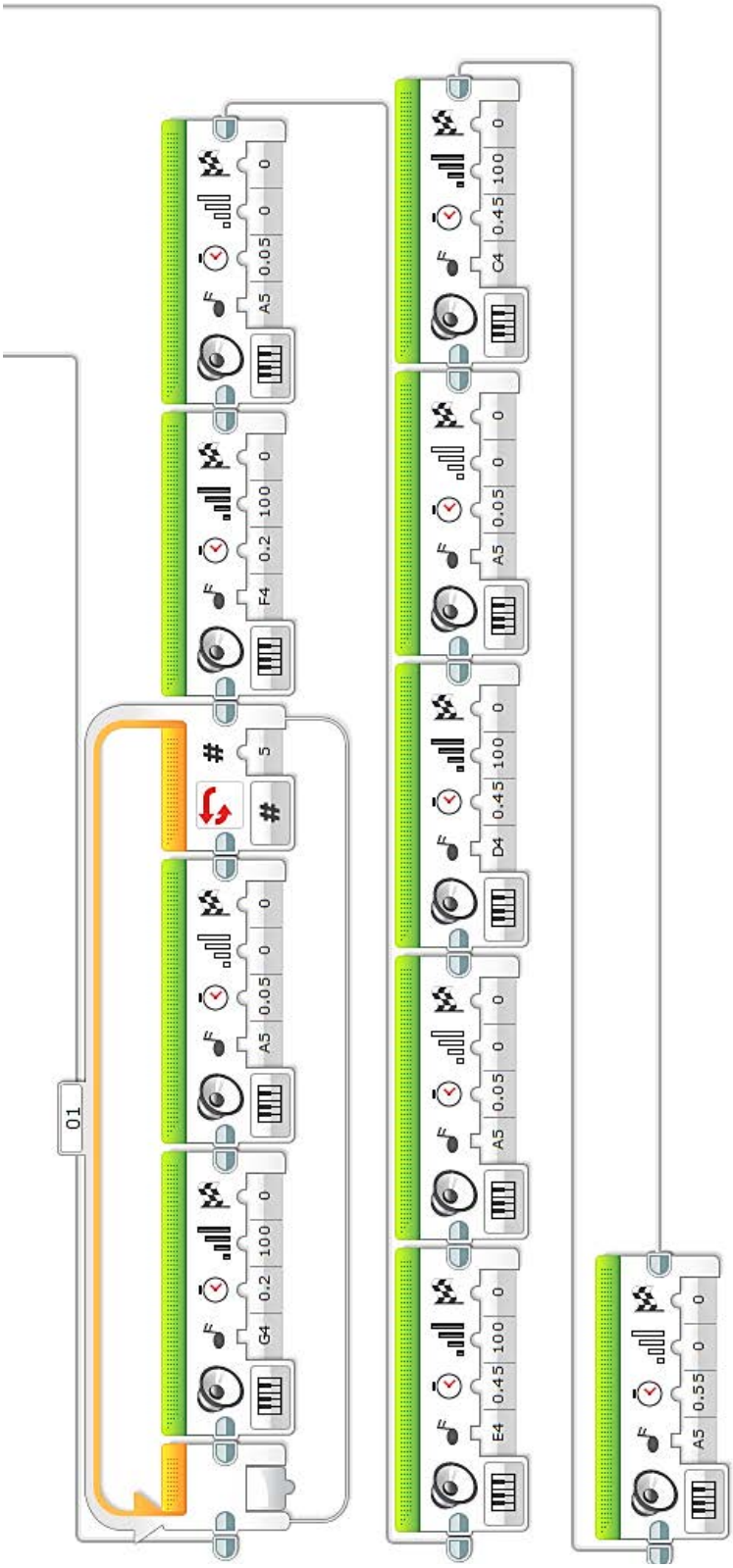
Vinkki: Tee ensin työpöydälle ihan koko ohjelmanpätkä, joka tulee silmukan sisään. Kytke rivit yhteen poluilla. Zoomaa lopuksi näkymä niin pieneksi, että tämä koko ryhmä mahtuu näkösälle. Valitse (maalaa) koko ryhmä hiirellä ja vie ne sitten yhdellä kertaa silmukan sisään.



Elefanttimarssi, sivu 1



Elefanttimarssi, sivu 2



Elefanttimarssi, sivu 3

Miten nuotein kirjoitettu musiikki sovitetaan robotin ohjelmaksi

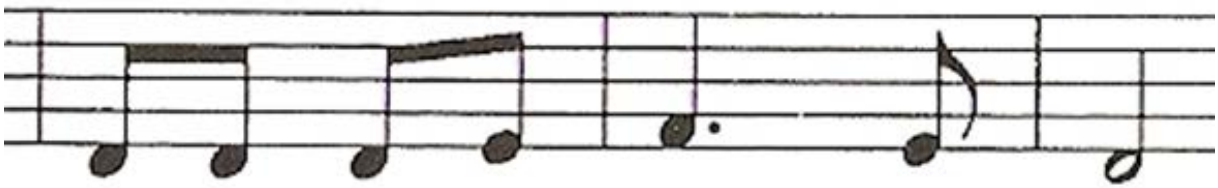
Musiikkia on sovitettu kautta aikain eri soittimille, kuten pianolle, jousille, puhaltimille jne. Seuraavaksi tutustutaan musiikin sovittamiseen robotille.

Aluksi pitää etsiä sopivat nuotit.

Seuraavaksi nuotit pitäisi muuttaa ohjelmaksi. Miten saadaan selville mitä ääntä ohjelman pitäisi soittaa, ja miten pitkään?

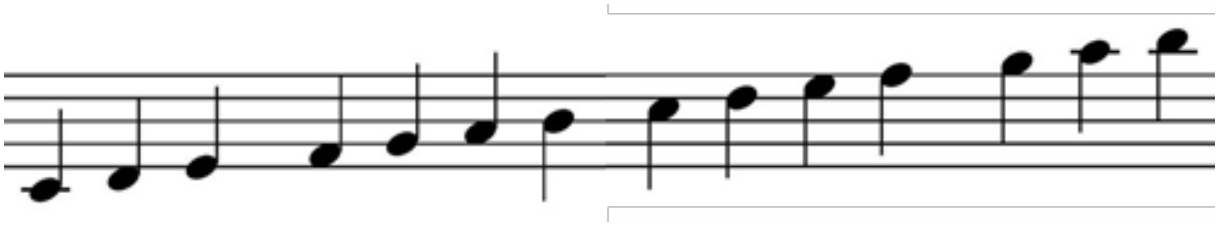
Nuotti kertoo soitettavan sävelen korkeuden ja keston. Tahti on nuottiviivastolla kahden pystyviivan väliin jäävä alue. Sen kesto, eli siihen merkittyjen nuottien yhteiskesto riippuu tahtilajista. Yllä olevassa esimerkissä tahtilaji on 2/4, eli yksi tahti kestää puolinuotin verran (eli kaksi iskua).

Ohjelmoijaa kiinnostaa kuitenkin lähinnä, paljonko se yksi tahti sitten tässä tapauksessa kestää sekunteina? Se riippuu temposta jolla kappaletta soitetaan. Esimerkin kappale kuulosti hyvältä kun sitä soitti nopeudella 120 iskua minuutissa (kirjoittaja on insinööri, ei muusikko). Näin ollen yhden tahdin kestoksi tulee yksi sekunti.



Yllä	4kpl	1/8 nuottia	3/8	1/8	1/2
Sekunteina		0.25s	0.75s	0.25s	1.00s
(ohjelmaan)		(0.20s)	(0.70s)	(0.20s)	(0.95s)

Sävelkorkeuden ohjelmointi edellyttää pientä sävelasteikkoon tutustumista EV3:n sävelasteikko



C4 D4 E4 F4 G4 A4 B4 C5 D5 E5 F5 G5 A5 B5 . . . B6

Korotusmerkit

Esimerkin alkuun oli lisäksi merkitty kaksi korotusmerkkiä, kohtiin C5 ja F5. Korotusmerkki koskee kaikkia oktaaveja, eli myös nuotit C4, F4, C6 ja F6 pitää korottaa.

Ohjelmointi

- Jotta nuottien väliin saadaan pieni väli, lyhennetään niitä 0.05s ja ohjelmoidaan niiden väliin aina 0.05s pituinen tauko.
- Korotettu C4 on EV3:n koodissa C#4 ja korotettu F4 on F#4.
- Käytetään silmukkaa, jos sama nuotti toistuu.

Esimerkki, 3/8 nuotti (0.05s tauko, volyme=0) 1/8 nuotti (0.05s tauko, volyme=0)

Jatkot

- Mieti tuleeko sinulle mieleen muita kohteita, tai laitteita joissa käytetään tai voisi käyttää etäisyydenmittausta? Käyttökohteen saa keksiä myös itse.
- Mitä kahta erilaista muistityppiä ohjelmoitavilla laitteilla on yleensä käytössään?
- Kumpaa tyyppiä mahtoi olla se muistipaikka, jossa moottorin pulssianturin lukema oli tallennettuna?
- Miksi ohjelmoitaessa käytetään eri kokoisia muitipaikkoja erilaisille tiedoille?

Vastaukset

Tehtävä 17, Ultraääni

- Hyvissä mittausolosuhteissa ultraäänianturi on välttävän tarkka, etenkin lyhyillä etäisyyksillä.
- Ääni ei kuitenkaan ole kovin helposti suunattavissa kapeaksi keilaksi, se pikemminkin yrittää levitä joka suuntaan. Tällöin menetelmä on todella herkkä ihan kaikelle lähettyvillä olevalle tavaralle. Joissakin käytännön sovellutuksissa, kuten auton peruutustutkassa, tämän menetelmän taipumus mittaavan keilan suureen kulmaan, mittausalueeseen on kuitenkin aivan ylivoimainen etu.
- Ultraäänianturi tunnistaa hyvin kohtisuoria ja hiukan kaarevia pintoja. Mittaukseen nähden vinossa oleva pinta, kuten robotin alla oleva pöytä sen sijaan voi jäädä kokonaan havaitsematta (kun mitään ei kimpoa suoraan takaisin).
- Lyhin etäisyys jolla mittaus toimii vielä luotettavasti, kohtisuora mittaus, on noin 5cm.
- Suurin lukema puolestaan on n. 2.5m
- Kahvikupin tunnistaminen, 20-30 cm on aika realistinen arvio.

Tehtävä 18, Infrapuna

- Hyvissä mittausolosuhteissa EV3: infrapuna-anturi on välttävän tarkka. Tämä riippuu kuitenkin suuresti tarkasteltavan kohteen väristä ja kiillosta, sekä vallitsevista valaistusolosuhteista.
- Valo leviää laajana keilana, mikä lisää anturin herkkyyttä kaikelle lähettyvillä olevalle tavaralle. Joskus se on toivottavaa, toisinaan ei. Tähän voidaan vaikuttaa käyttämällä anturin yhteydessä erilaisia linssejä ja valokuituja. Kuitukennojen etuja ovat mm. kuidun mahtuminen hyvin pieniin paikkoihin, sekä mahdollisuus tuoda herkkä elektroniikka etäämmälle tarkasteltavasta kohteesta (esim. 300 asteisen uunin sisältä)
- Lyhin etäisyys jolla EV3:n infrapuna-anturin mittaus toimii vielä luotettavasti, kohtisuora mittaus ja valkoinen paperi, on noin 3.5 cm.
- Suurin lukema puolestaan on 1 m
- Kahvikupin tunnistaminen, 20 cm on aika realistinen arvio.

Joitakin EV3-yhteensopivia laitteita

7. Sokkeloinen seikkailu

Harjoituksen tavoitteena on kerrata ja soveltaa lähes kaikkea tähän asti opittua, ymmärtää toimintakuvauksen merkitys ohjelmoinnissa tärkeänä työvaiheena, sekä pohtia ohjelmointiin oikeassa elämässä liittyviä haasteita.

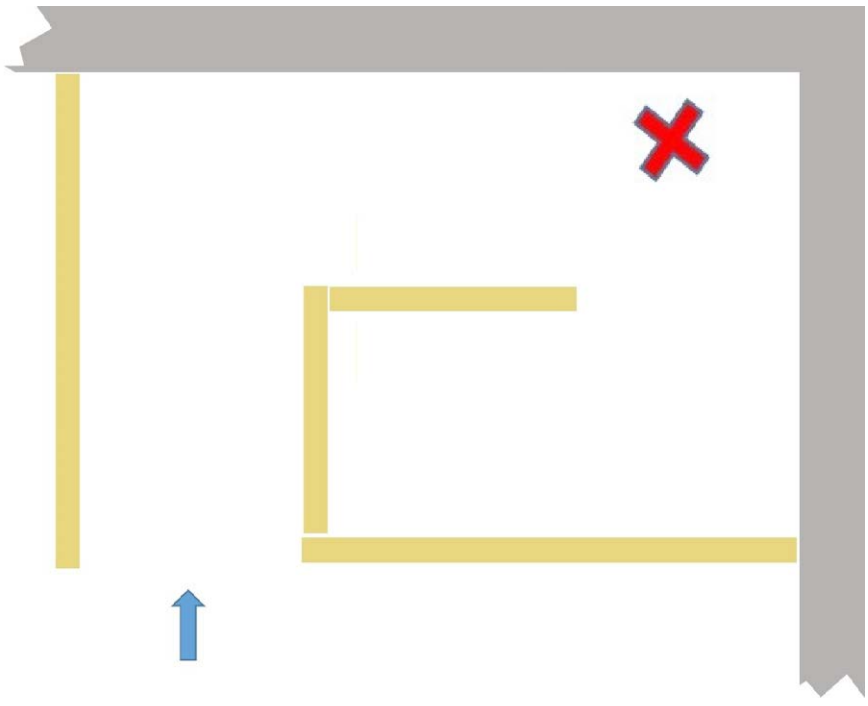
Etkot

Rakennetaan kuvan mukainen yksinkertainen labyrintti. Ensimmäinen harjoitus voidaan yhdistää myös kuvaamataidon opetukseen. Tarvikkeet:

- 2x4”, tai ”2x5” höylättyä lankkua, kertopuuta tms, 50cm pitkiä paloja 2kpl ja 100cm pitkiä paloja 2kpl.

Kirjoittajan huomautus: ”Kahden viikon takkuamisen jälkeen ostin riittävän

korkeat ja sileät reunalankut (kertopuuta). Robotti toimi välittömästi.”



Laudat asetetaan lattialle kyljelleen kuvan osoittamalla tavalla.

Kun vaara on suuri

Ohjelmoitavia robotteja käytetään usein kun olosuhteet ovat ihmiselle liian vaaralliset, tai kohtuuttoman rasittavat. Tehtävät ovat tyypillisesti tutkimiseen, työskentelyyn, raivaamiseen tai pelastamiseen liityviä ja niitä tehdään

- suurissa syvyyksissä (paine)
- sortuneissa rakennuksissa, tai kaivoksissa
- tilanteissa joihin liittyy räjähdysvaara
- radioaktiivisesti tai muuten saastuneissa tiloissa
- jne.

Liikkumisen ja suunnistamisen ohjelmoiminen tällaisiin olosuhteisiin on monin verroin haasteellisempaa kuin yleensä. Yllätyksellisyys ja mahdottomienkin tilanteiden ennakoimisen vaikeus korostuvat tällaisissa ohjelmointitehtävissä.

Laitteiden testaaminen on myös ihan oma haasteensa. Todellisten olosuhteiden jäljitteleminen ei välttämättä ole mahdollista, tai edes turvallista. Pelastusrobottien rakentajat ovat myös joutuneet toteamaan, että olosuhteiden jäljittelyn ”onnistuessa” liian hyvin, se kallis ja ainoa koekappale saattaa tuhoutua.

Todellisessa elämässä, oikeita koneita ohjelmoitaessa, pyritään varautumaan ennakoitavissa oleviin poikkeamiin, virheisiin ja ulkopuolisiin häiriötekijöihin. Tyypillisesti tämä varautuminen moninkertaistaa ohjelman koon. Vaativimmissa kohteissa on varauduttava mahdottomiinkin tilanteisiin. Koneen toiminnan kannalta välttämättömän ohjelman osuus koko ohjelmistosta saattaa tällöin jäädä alle kymmenen prosentin. Vahingot, joita tällä voidaan välttää, saattavat kuitenkin olla niin merkittäviä, että ohjelmointi kannattaa tehdä todella huolella ja siihen kannattaa käyttää rahaa.

Pelastusrobotit työssä

Vuonna 2011 sattui Fukushimassa, Japanissa vakava ydinvoimala-onnettomuus. Sen seurauksena mm. voimalan rakennuksiin ja sen ympärille levisi vaarallista, voimakkaasti radioaktiivista ainesta.

Tapahtumasta aiheutui suuria vahinkoja, mutta myös jotakin hyvää. Nimittäin ennennäkemätön into kehittää uusia ja parempia pelastus, sekä raivaus tehtäviin soveltuvia robotteja. Ei yksin Japanissa, vaan kaikkialla maailmassa.

Ensin kartoitetaan tilanne

Ensimmäiset robotit kartoittivat tilannetta voimalassa, suorittivat erilaisia mittauksia ja keräsivät näytteitä. Robotti tarvitsee kaikkia mahdollisia keinoja joilla se voi hahmottaa ympäristöään voidakseen kulkea ja suunnistaa törmäilemättä esteisiin, tai välttymään esimerkiksi putoamasta kuoppaan. Tiedustelijarobottien pitää kyetä ylittämään rojukasoja, kulkemaan portaissa ja avaamaan ovia. Tiedustelijarobotin kuljetuskapasiteetin ei kuitenkaan tarvitse olla kovin iso. Mittalaitteet, kamerat yms. eivät paina kovin paljoa.

Toimenpiteet suunnitellaan saadun tiedon pohjalta

Tiedustelijoita seurasi joukko pieniä maansiirtokoneita muistuttavia raivaajarobotteja, joita ohjattiin etäältä. Näimä avasivat kulkureittejä korjaus ja puhdistusroboteille. Työ jatkuu vielä vuosia ja sen edistymistä voi seurata netissä (lähde: Tepco)

Netistä löytyy myös suuri joukko kuvia erilaisista pelastusroboteista. Kuvia voi hakea esimerkiksi hakulauseella ”rescue robot” ”ASTACO-SoRa” ”Atox Raccoon”.

Tehtävä 22. Sankarihommissa

- Mietitään yhdessä mahdollisia tilanteita, missä pelastusrobotista voisi olla apua.
- Ideoidaan omia pelastusrobotteja, miten ne voisivat liikkua ja suunnistaa, sekä millaisia asioita niiden olisi hyvä osata?
- Millaisia asioita niiden ohjelmoinnissa kannattaa ottaa huomioon?
- Piirretään lopuksi kuva itse keksitystä pelastusrobotista

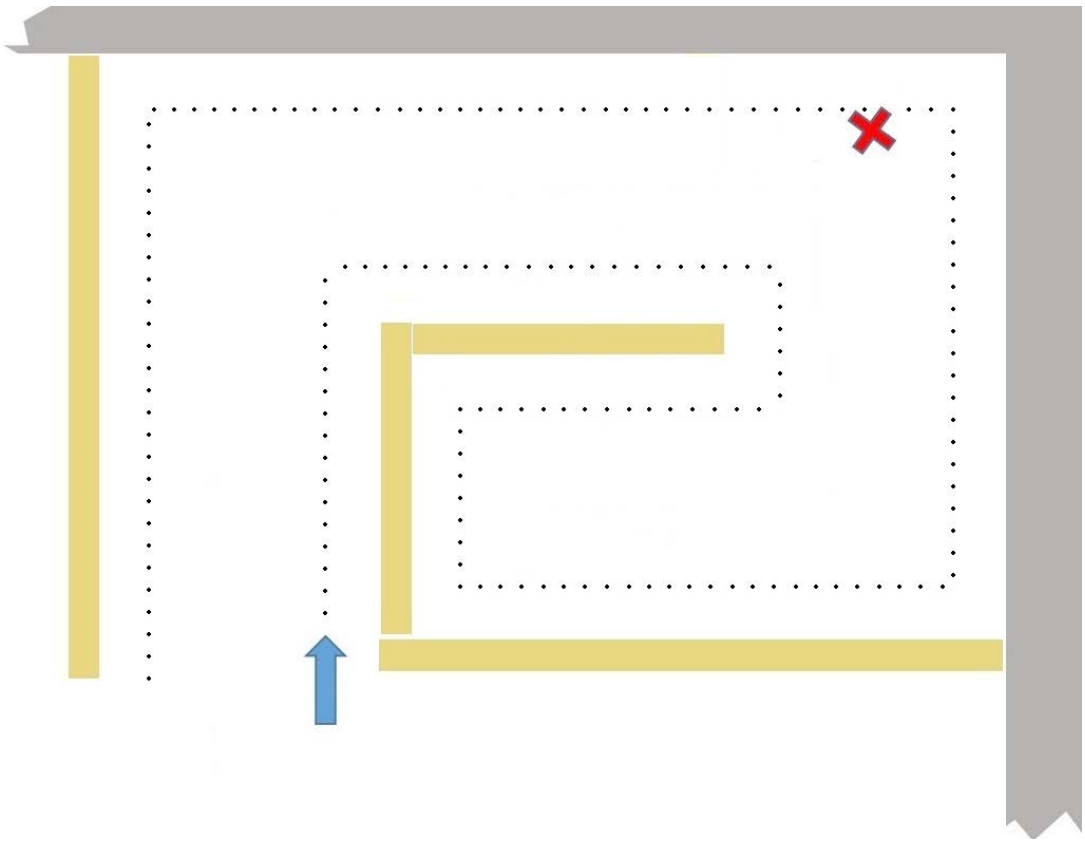
Tehtävä 23. Selätetään labyrintti älyllä ja opituilla taidoilla

Tutustutaan opetustilaan rakennettuun labyrinttiin. Tehtävänä on ohjelmoida robotti kiertämään koko labytintti läpi ja löytämään tiensä sieltä ulos. Kuvitellaan lisäksi, että näemme ovelta vain käytävän alun ja emme voi tietää mihin suuntaan, tai miten pitkälle käytävä jatkuu kulman takana. Tehtävään käytetään aikaisemmista harjoituksista tuttua monitoimirobottia. Tehtävää helpottavana lähtötietona kerrottakoon kaikkien kulmien olevan n. 90 astetta.

Vaihtoehtoja labyrintissa suunnistamiseen on monia, esim.

Ennalta ohjelmoitu reitti, näitä olemme jo harjoitelleet. Tämän heikkoutena on sen herkkyys pienimmillekin ohjelmoinnin yms. virheille, mukaanlukien radan poikkeamat.

Seuraavana vaihtoehtona on lähteä kulkemaan seuraamalla seinää. Tämä on varmatoimisempi kuin ensimmäinen vaihtoehto, mutta silti nopea. Lähdetään tekemään seinää seuraavan robotin ohjelmaa.

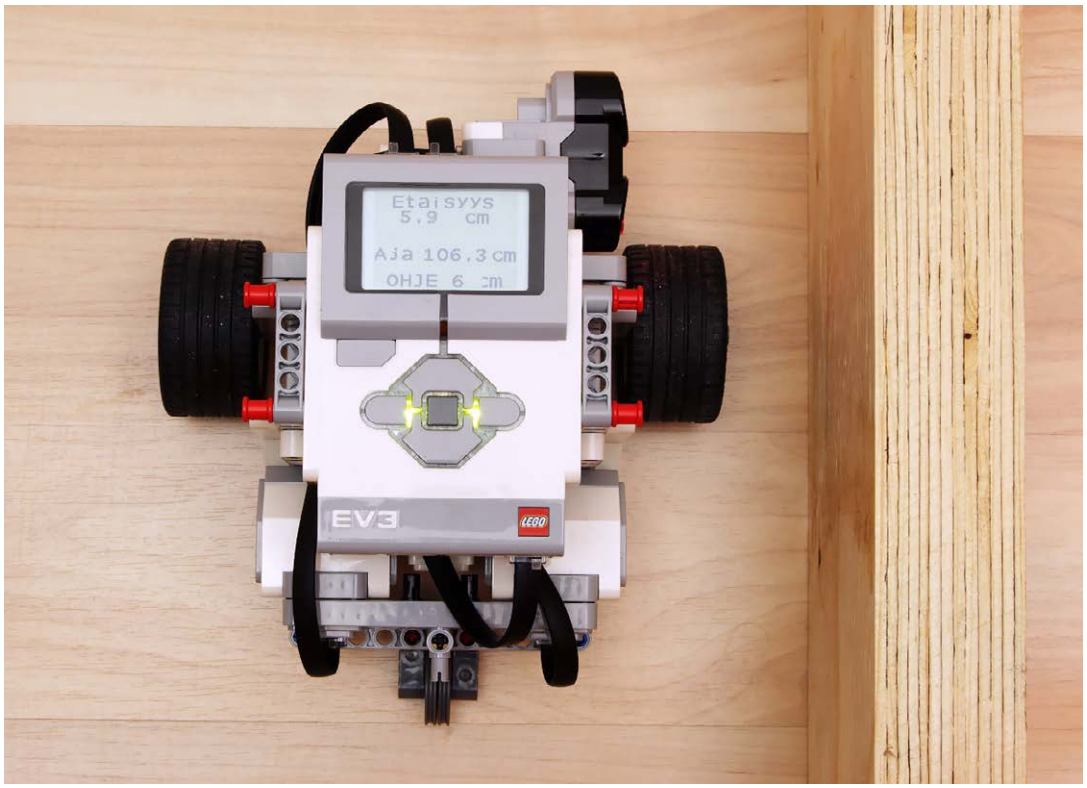


Jos labyrinttiin olisi kätketty etsittävä kohde, sekin todennäköisesti löytyisi.

Yhdellä ultraäänianturilla

Antureita voi kiinnittää tähänkin robottiin lähes lukemattomilla tavoilla. Päätöksen siitä mihin ja miten anturit kiinnitetään ratkaisee yleensä käyttötarkoitus.

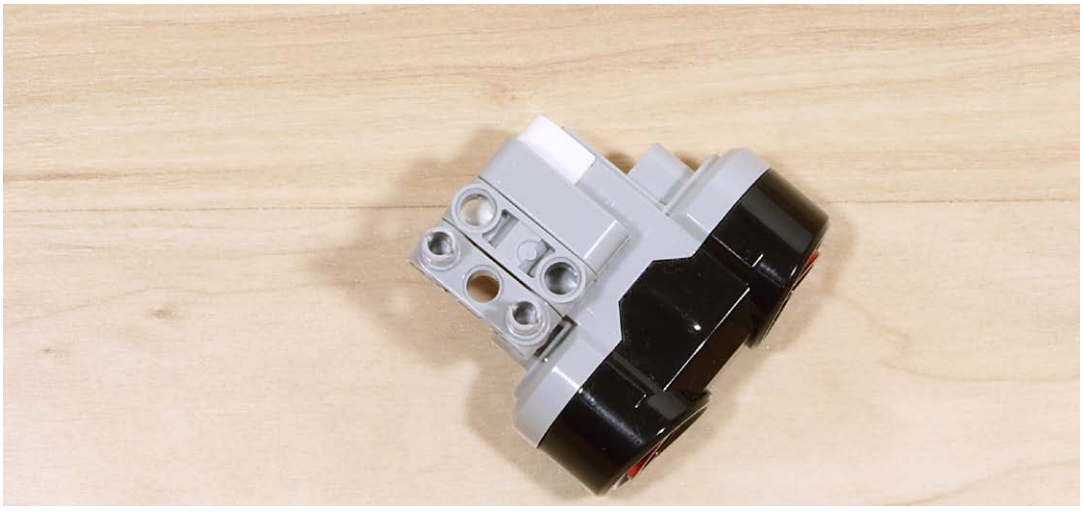
Tässä esimerkkisuorituksessa on robotin liikkumisen strategiaksi valittu sen oikealla puolella olevan seinän seuraaminen (vasenkin kävisi). Tämän voi ratkaista useammalla tavalla. Seinä voidaan tunistaa esimerkiksi kosketusanturilla, tai kosketuksettomilla etäisyyden mittaamisen menetelmillä. Antureita voi olla yksi tai useampia, eivätkä ne välttämättä ole kaikki samaa tyyppiä.



Kuvan esittämä, kuten myös pysäköintitehtävän ”sivulle katsova” robotti soveltuvat hyvin kulkemaan labyrintissa. Yhdellä ultraäänianturilla pärjää, mutta ohjelma on vähän monimutkaisempi (älykkäämpi).

Pitkä takaosa on tietysti otettava huomioon seinän lähellä tehtävissä käännöksiä. Mutta toisaalta mahdollisuus kiinnittää anturi lähelle akseli-linjaa helpottaa tässä tapauksessa merkittävästi ohjelmointia.

Kiinnitetään ultraäänianturi kahdella liittimellä suoraan ohjainpalikkaan (katso kuva edelliseltä sivulta). Huomaa miten ohjain on siirretty keskimmäiseen positioon -> saadaan lisää painoa vetäville pyörille -> tarkkuutta käännöksiin.

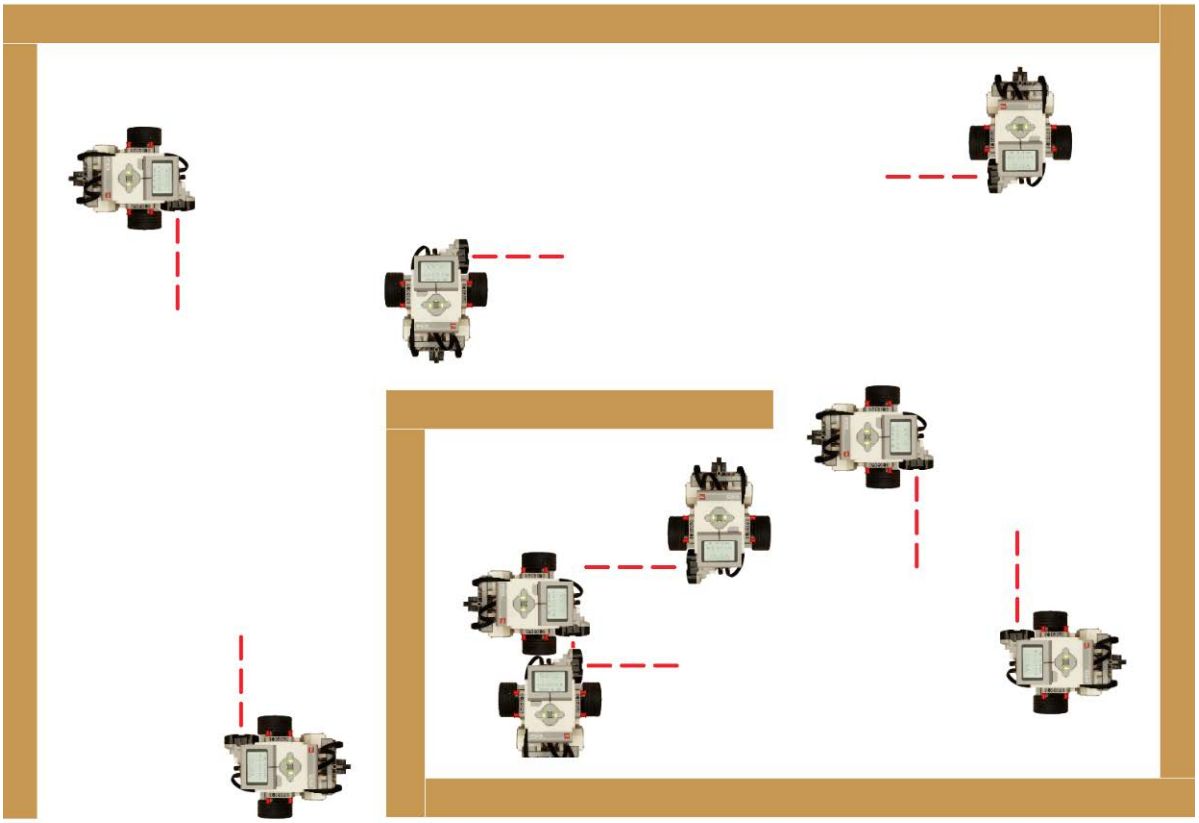


Kun anturi on saatu paikoilleen, ryhdytään miettimään miten robotti kulkisi labyrintissa. Anturin mittaustiedon perusteella sen on helppo kulkea sopivalla etäisyydellä seinästä. Tämän voi ohjelmoida kuten viivanseurannan. Lisäksi kohdat joissa sen tulee kääntyä oikealle on helppo tunnistaa (mittauksen lukema kasvaa yllättäen >30cm) (ulkonurkka).

Mutta miten robotti voisi selvittää, milloin sen pitää pysähtyä ja kääntyä vasemmalle? (sisänurkka). Robotin pitää mitata aina käännyttyään paljonko sillä on edessä tilaa kulkea.

Kuten oikeassa ohjelmointityössä, tässäkin tehtävä kannattaa ensin purkaa pieniin osiin, joita on sitten helpompi ratkoa yksitellen.

Alla on kuva reitistä, jota lähdetään nyt yrittämään. Siihen on myös luonnosteltu kohdat, joissa robotin pitää mitata paljonko sillä on edessään vapaata tilaa edetä.



Piirtäminen auttaa usein ohjelmointiin liittyvien pulmien ratkomisessa. Vielä tehokkaammaksi on osoittautunut käytännön kenttätyö. Katsotaan millainen se kone, laite tai robotti on ihan oikeasti ja miten se istuu sen oikeaan käyttöympäristöön. Eli, viedään robotti labyrinttiin ja pyöritellään sitä sielä ihan käsin.

Apuna kannattaa käyttää edellisessä harjoituksessa tehtyä ohjelmaa, joka näyttää ultraäänianturin mittaaman etäisyyden ohjaimen näyttöruudulla.

- Mieti miten robotti voisi toimia sisänurkissa, mistä ja miten se voi tietää milloin sen pitää pysähtyä.
- Entä ulkonurkissa?
- Mistä ja miten robotti voi päätellä, mihin suuntaan sen tulee seuraavaksi kääntyä?
- Miten robotin tulee toimia seinän päätyä kiertäessään?
- Millaisia käännöksiä robotin tulisi taitaa?
- Ovatko kaikki käännökset samanlaisia?

Muista että robotti ei tiedä etukäteen reittiä.Tehtävän ratkaiseminen edellyttää myös joitakin peruslaskutoimituksia.

Harjoituksen labyrinttiin sisältyy

- käännös oikealle,
- tuplakäännös oikealle,
- tuplakäännös vasemmalle,
- kolme tavallista käännöstä vasemmalle
- ja eri mittaisia suoraan ajettavia osuuksia.

Luokkaan voidaan tehdä samanlainen, mutta isompi labyrintti pöydistä. Kokeilkaa kävelemällä, löytäisittekö ulos labyrintista näillä ohjeilla

A. Jos oikealla puolellasi on labyrintin ”seinä”, käänny vasemmalle.

B. Arvioi, paljonko oikealla puolellasi on nyt matkaa lähimpään ”seinään”.

C. Käänny oikealle.

D. Nollaa matkamittari (laskuri) ja lähde seuraamaan ”seinää”.

E. Jos kuljettava matka tulee täyteen, pysähdy ja käänny vasemmalle, tai

F. jos seurattava seinä loppuu, käänny oikealle kulman ympäri.

G. Aloita käännöksen jälkeen uudestaan kohdasta A.

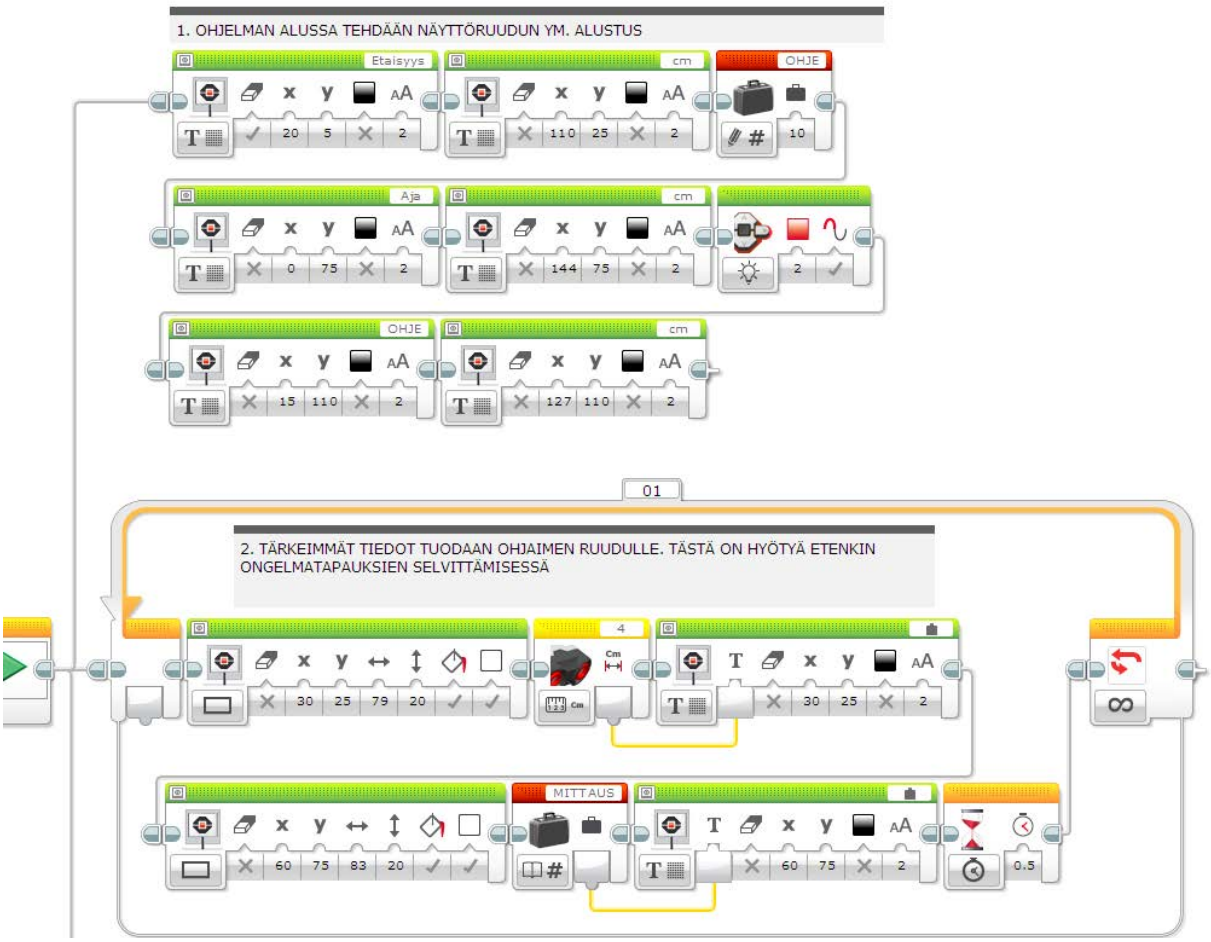
Oikean robotin ohjelman toimintakuvaus poikkeaa tästä vain vähän. Periaate jolla robotti toimii on aivan sama. Koneille pitää kuitenkin kertoa joitakin asioita vähän tarkemmin, kuin meille ihmisille.

Kun esimerkiksi joku pyytää meitä kääntymään vasemmalle, me oletamme pyynnön tarkoittavan 90 asteen käännöstä paikallaan. Koneelle meidän pitää kertoa kaikki sen tekemisiin vaikuttavat tiedot pienimpiä yksityiskohtia myöten ja vieläpä hyvin tarkasti.

Ihmiset osaavat myös arvioida omien havaintojensa luotettavuutta. Koneet, kuten robotit, sen sijaan joutuvat luottamaan anturiensa tuottamaan tietoon sokeasti, ellei tähän ole puututtu ohjelmoinnin keinoin.

Labyrintti-robotin toimintakuvaus

1. Alkurutiinit, ohjelmoitiinpa mitä tahansa, ylensä ohjelman alkuun joudutaan kirjoittamaan jotakin, että ohjelma lähtee jouhevasti liikkeelle. Muuttujille määritellään arvoja, antureita ja laskureita nollataan. Samalla voidaan tyhjentää näyttöruutu ja kirjoittaa sille sellaiset tekstit, joita ei tulla muuttamaan.

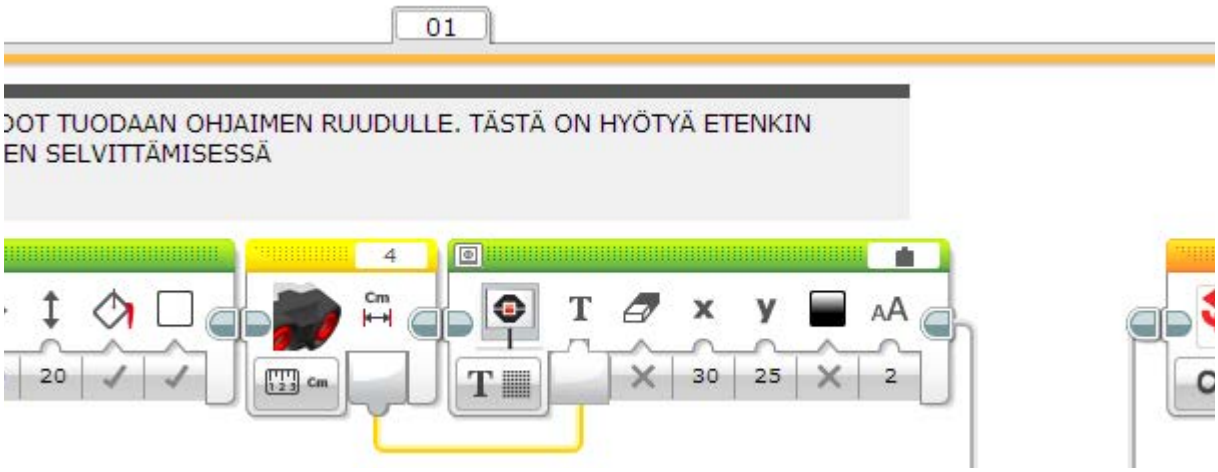


2. Täydennetään näyttöruudun ohjausta muuttuvilla tiedoilla.

Joukossa on myös näytönohjaustoimilohkoja, joissa on maaliämpärin näköinen kuva. Mitä luulisit niiden tekevän?

Näytönohjaustoimilohkoista löytyy lisää titoa kirjan lopusta.

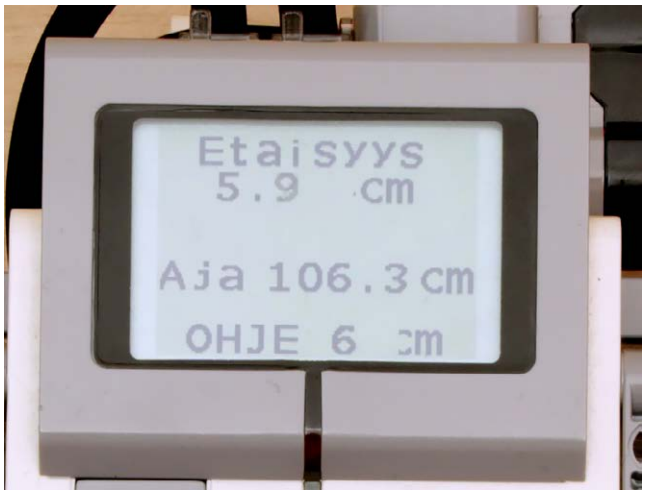
Kun olet kirjoittanut kuvassa näkyvän osan ohjelmaa, voit ladata sen robottiin ja katsoa että mittaukset tulevat oikein ruudulle. Ota nyt kokeeksi nuolella merkitty näytönohjaustoimilohko pois ohjelmasta ja kokeile sitä sitten uudestaan. Joko keksit tämän toimilohkon merkityksen?



Kun robotti mittaa matkan, jota sen pitäisi seuraavaksi lähteä ajamaan, mittaako se sen oikein? Tätä voi arvioida helposti, kun tuo matka kerrotaan näyttöruudulla. Ohjelma kirjoittaa näytölle ”mittauksen” sijaan lyhyemmän sanan ”AJA”, jotta koko asia saadaan mahdutettua näytöllä yhdelle riville.

Vastaus edellisen sivun kysymykseen

Näytölle pitäisi tulla nyt näkyviin neljä riviä. Koska näytönohjaustoimilohko kirjoittaa ruudulle aina vain vanhan päälle ja etäisyyden mittaustiedon pituus voi vaihdella parista numerosta neljään, on meidän joka silmukan kierrolla ensin tehtävä sille tyhjä suorakaiteen muotoinen tila. Muuten ruudulle voi jäädä jo aikaisemmin kirjoitettuja numeroita sotkemaan tuota lukemaa.



Seuraavaksi opetellaan, miten koneelle, kuten robotille voidaan kertoa jokin sen tarkoitetun toiminnan kannalta tärkeä tieto.

Kuvassa näytön alimmalla rivillä näkyy ohjearvo robotille, miten kaukana sen

odotetaan kulkevan labyrintin seinästä (etäisyys ultraäänianturista).

Ohjearvoa halutaan muuttaa ruudun alla olevista YLÖS ja ALAS -painikkeista, yksi sentti kerrallaan. Suurin etäisyys on 8cm ja pienin 5cm.

Harjoitellaan samalla tietotyypin muunnosta

Tämän kirjan alkupuolella esiteltiin erilaiset tietotyypit, tietojohtimet, sekä miten näitä voi käyttää. Tässä harjoituksessa voidaan hyödyntää yhtä tietotyyppien yhteensopivuutta.

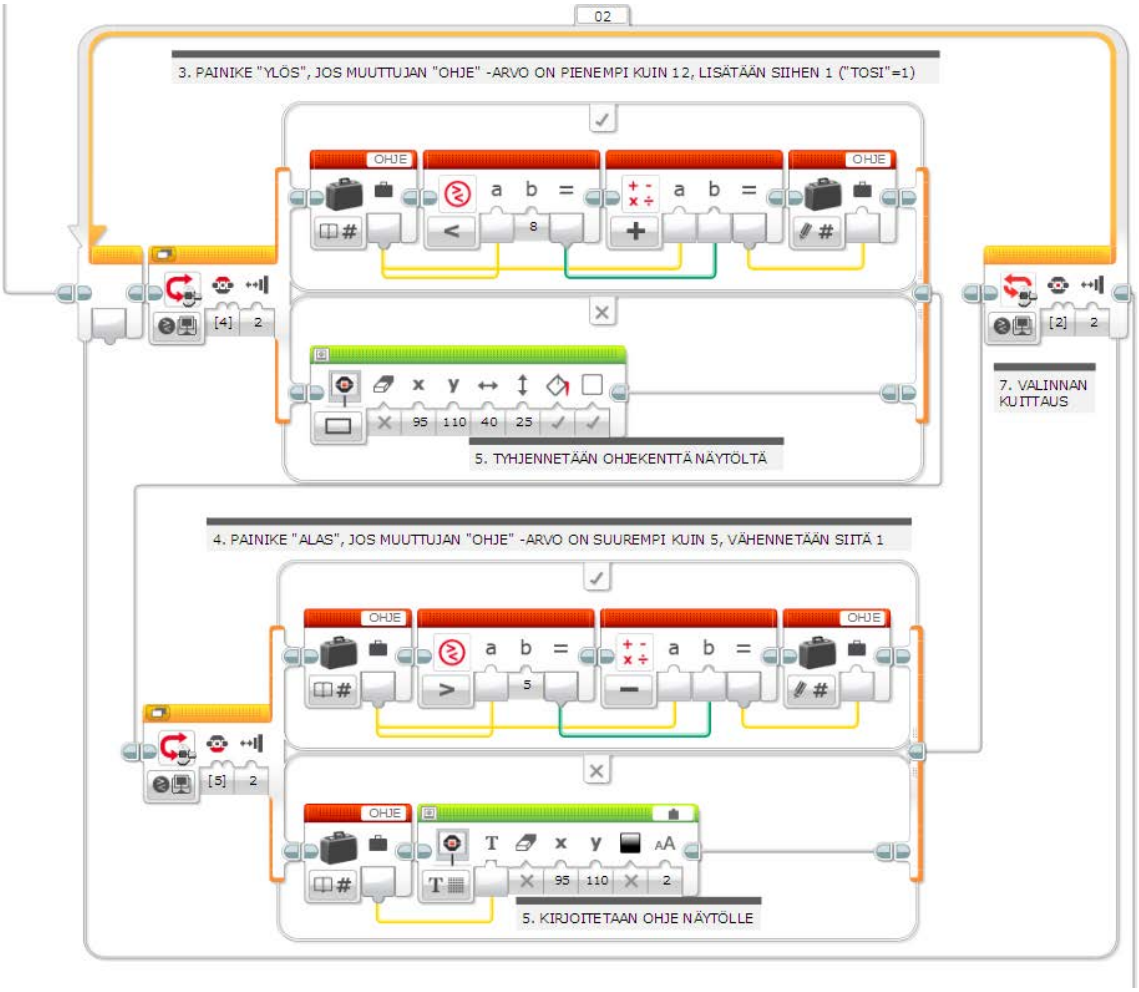
Lähtö	Tulo	Mitä tuloon kirjoittuu
Tosi/epätosi (Logic)	Numeerinen	Tosi=1, Epätosi=0

Tietotyyppiä Logic voi käyttää ohjelmassa myös kuin numeroa

Aseteltava ohjearvo, rajoitettu minimi ja maksimi

Alla olevassa kuvassa IF-THEN –toimilohko on parametroitu seuraamaan YLÖS –painikkeen mahdollisia painalluksia.

- 3. Painalluksesta (YLÖS)
- Luetaan numero muistipaikasta OHJE.
- Verrataan onko tämä numero pienempi kuin 8, eli maksimi.
- **Jos** luku oli alle 8, vertailun tulos on TOSI, eli 1
- Mutta jos OHJE on jo 8, vertailun tulos on EPÄTOSI, eli nolla
- Viedään OHJE, sekä vertailun tulos laskentatoimilohkoon.
- Suoritetaan yhteenlasku, OHJE +1, tai OHJE +0
- Talletetaan laskun tulos muistipaikkaan OHJE



Näin kirjoitettuna, tietotyypin muunnosta hyödyntäen, tämän ohjelman osan pituus jää alle puoleen tavallisesta. voit käyttää tätä aina kun haluat antaa jonkin aseteltavan lähtötiedon painikkeilla. Huom. Silmukan nimeksi on vaihdettu 02.

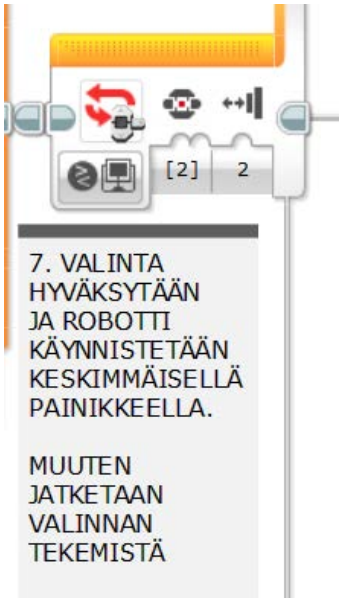
Käyttäjän informointi ohjaimen merkkivaloilla

Olet varmasti jo huomannut EV3-ojaimen painikkeita ympäröivän valon, joka palaa tai vilkkuu milloin punaisena, milloin vihreänä. Ohjain kertoo sillä käyttäjälle monenlaisista asioista. Näistä kerrotaan tarkemmin EV3:n käyttöohjeessa sivulla 6.

Tätä merkkivaloa voidaan myös ohjelmoida. Sen avulla voi välittää käyttäjälle tietoa esimerkiksi ohjelman kulun etenemisestä, tai vaikka valmiudesta ryhtyä suorittamaan jotakin tehtävää.

Täydennetään kokeeksi labyrintti-ohjelmaa muutamalla robotin käyttöä helpottavalla merkkivalolla. Eli lisätään ohjelmaan merkkivalo-toimilohkoja kertomaan käyttäjälle milloin se

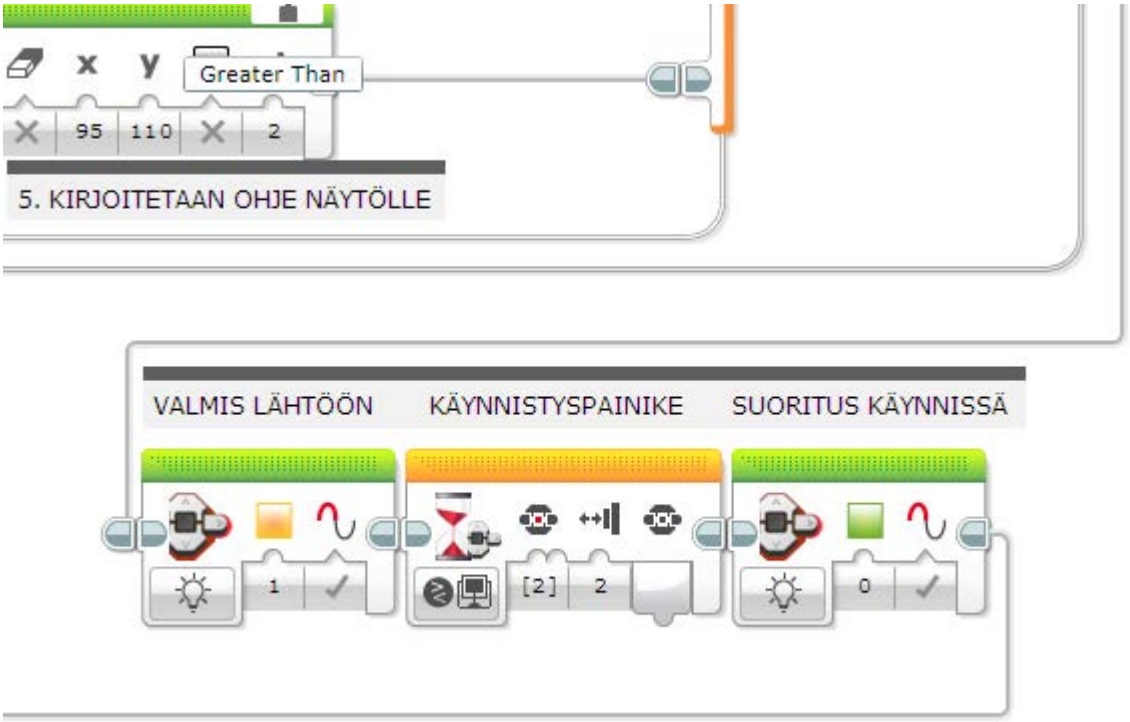
1. odottaa käyttäjän asettavan ja hyväksyvän OHJE-arvon (punainen vilkkuva valo)
2. odottaa lähtökäskyä (keltainen vilkkuva valo)
3. suorittaa tehtävää (vihreä vilkkuva valo)
4. Tästä määritellään silmukasta ulostulo ja samalla robotin varsinaisen labyrintti-ajelun aloittaminen tehtäväksi ohjaimen keskimmäisen painikkeen painalluksesta.



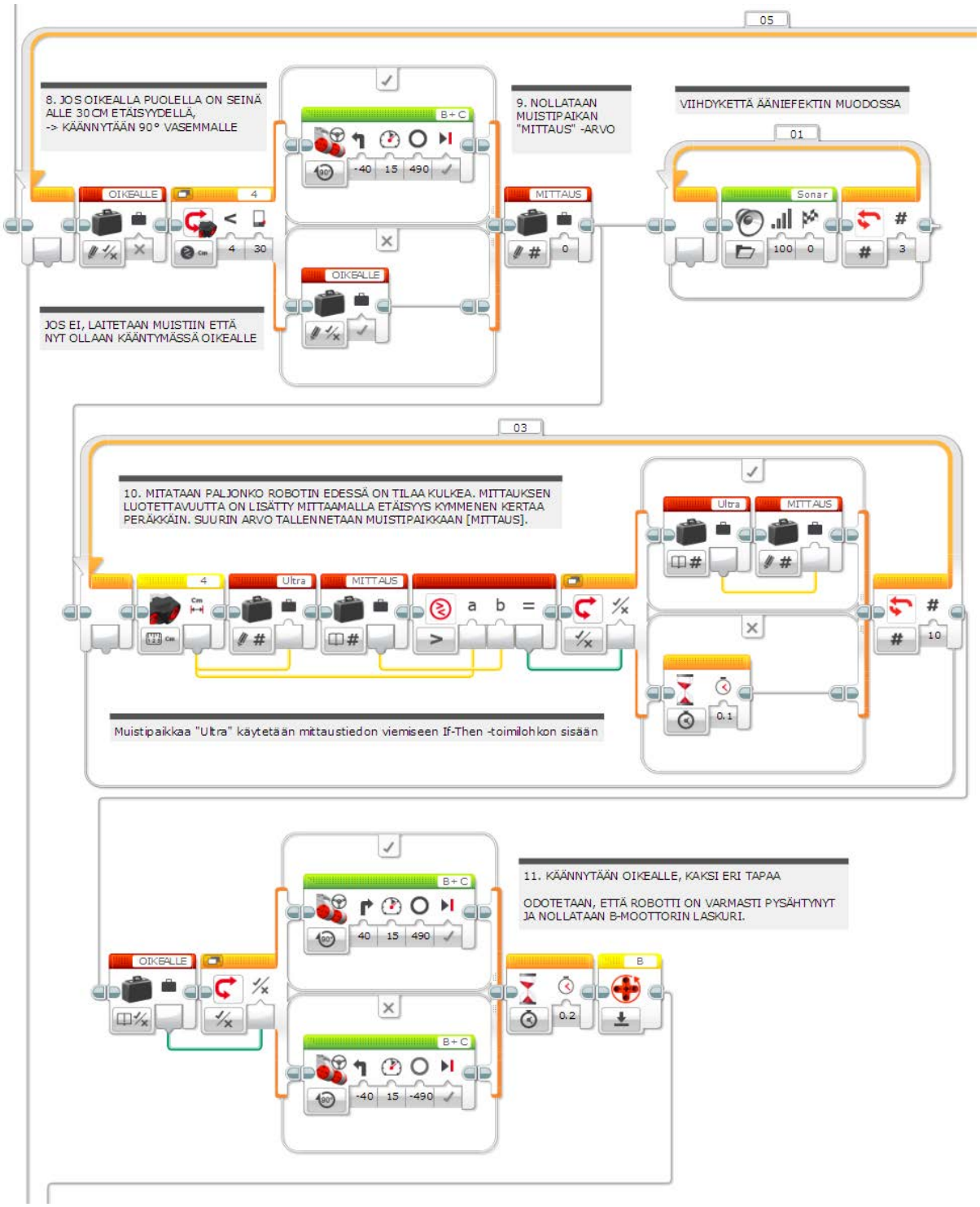
Nyt ohjelmaa voidaan taas kokeilla. Koska robotti ei ole vielä karkamassa mihinkään, voidaan ohjelmointikaapeli pitää kiinni robotissa. Käynnistä ohjelma tietokoneen näytöltä, joko ohjelman aloitus-toimilohkon vihreää kolmiota klikaten, tai näytön oikeasta alanurkasta laitemonitorin Download & PLAY-symbolia klikaten (lataa & käynnistä ohjelma).

Kokeile OHJE-muuttujan arvon muuttamista. Toimiiko se kuten piti? Miten tämä näkyy tietokoneen ruudulla? Viemällä hiiren osoittimen ohjelmaan piirrettyjen tietojohtimien päälle voit nähdä niissä kulkevat tiedot tietokoneen ruudulla.

5. Lisätään etäisyyden ohjearvon asettamisen perään käynnistyspainike ja käyttäjää opastavien valosignaalien käskyt.

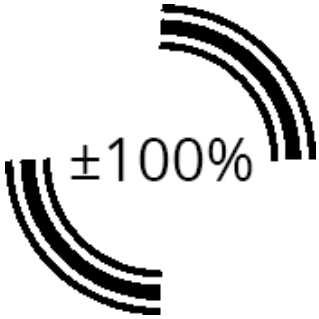


6. Ohjelman viimeinen osuus, suuri silmukka numero 05 johon on ohjelmoitu mittaus ja robotin liikkuminen, on kätevämpää käsitellä useammassa osassa. Tässä osassa robotti mittaa paljonko sillä on edessään tilaa kulkea. Silmukka 05 on päättymätön.


05
8. JOS OIKEALLA PUOLELLA ON SEINÄ ALLE 30 CM ETÄISYYDELLÄ, -> KÄÄNNYTÄÄN 90° VASEMMALLE
B+C
-40 15 490
OIKEALLE
4
4 30
OIKEALLE
JOS EI, LAITETAAN MUISTIIN ETTÄ NYT OLLAAN KÄÄNTYMÄSSÄ OIKEALLE
9. NOLLATAAN MUISTIPAIKAN "MITTAUS" -ARVO
MITTAUS
0
VIIHDYKETTÄ ÄÄNIEFEKTIN MUODOSSA
01
Sonar
100 0
3
03
10. MITATAAN PALJONKO ROBOTIN EDESSÄ ON TILAA KULKEA. MITTAUKSEN LUOTETTAVUUTTA ON LISÄTTY MITTAAMALLA ETÄISYYS KYMMENEN KERTAA PERÄKKÄIN. SUURIN ARVO TALLENNETAAN MUISTIPAIKKAAN [MITTAUS].
4
Ultra
MITTAUS
a b =
>
Ultra
MITTAUS
0.1
10
Muistipaikkaa "Ultra" käytetään mittaustiedon viemiseen If-Then -toimilohkon sisään
11. KÄÄNNYTÄÄN OIKEALLE, KAKSI ERI TAPAA
ODOTETAAN, ETTÄ ROBOTTI ON VARMASTI PYSÄHTYNYT JA NOLLATAAN B-MOOTTORIN LASKURI.
B+C
40 15 490
OIKEALLE
0.2
B
B+C
-40 15 -490

Muutama sana käännöksistä

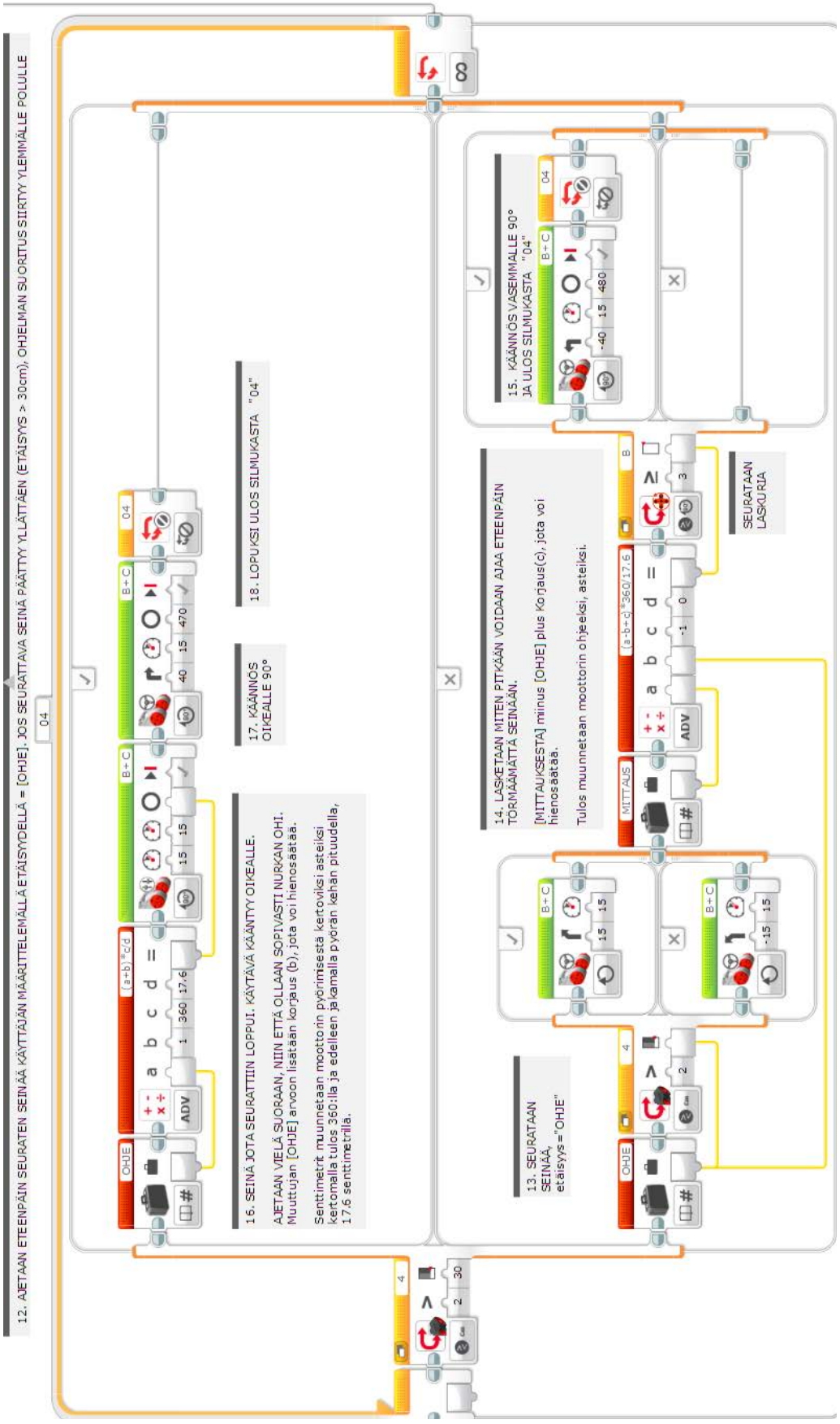
Olemme oppineet, että kun käytetään Move Steering toimilohkoa ja määritellään ”rattia” käännettäväksi ±100%, robotti pyörii paikallaan(eli renkaanjäljet muodostavat vain yhden ympyrän).

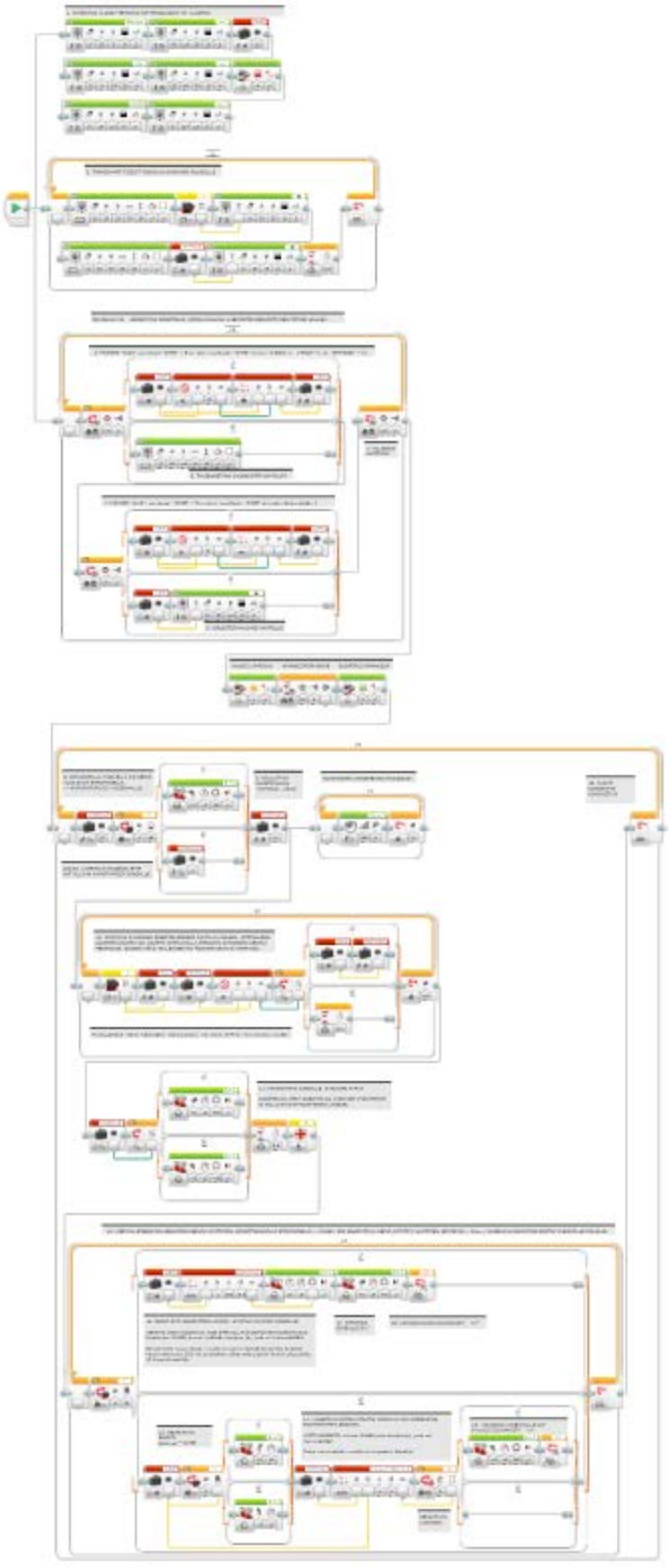


Vastaavasti ±50% käännös tarkoittaa sitä että toinen pyörä pyörii ja toinen pysyy paikoillaan. Toisen pyörän pyörimättömyyden on havaittu voivan aiheuttaa yllättäviä häiriöitä, **jos käytetään leveitä ja hyvän pidon omaavia renkaita** (kuten monitoimirobo-tissamme käytetään). Renkaan leveys vastustaa kääntymistä ja siitä seuraa väkisinkin vääntömomenttia myös pysähtyneenä olevaan renkaaseen. Tämän seurauksena robotti saattaa nykiä oudosti käännök-sien päätteeksi.

Tämä ongelma voidaan kiertää varsin helposti kääntämällä ”rattia” hiukan enemmän, tai vähemmän (esimerkiksi ±40% tai ±60%). Kun molemmat pyörät pyörivät, ne myös liukuvat ja niiden välille ei pääse kehitymään merkittävää vääntömomenttia.

Tämän tehtävän esimerkkisuorituksessa päädyttiin käyttämään käännöksissä ohjeenä ±40%.


12. AJETAAN ETEENPÄIN SEURATEN SEINÄÄ KÄYTTÄJÄN MÄÄRITTELEMÄLLÄ ETÄISYYDELLÄ = [OHJE]. JOS SEURÄTTAVA SEINÄ PÄÄTTYY YLÄTTÄEN (ETÄISYYS > 30cm), OHJELMAN SUORITUS SIIRTYY YLEMMÄLLE POLULLE
04
16. SEINÄÄ JOTA SEURATTIIN LOPPUU. KÄYTÄVÄÄ KÄÄNTYY OIKEALLE.
AJETAAN VIELÄ SUORAAN, NIIN ETTÄ OLLAAN SOPIVASTI NURKAN OHI.
Muuttujan [OHJE] arvoon lisätään korjaus (b), jota voi hienosäätää.
Senttimetrit muunnetaan moottorin pyörimisestä kertoviksi asteiksi kertomalla tulos 360:lla ja edelleen jakamalla pyörän kehän pituudella, 17.6 senttimetrillä.
17. KÄÄNNÖS OIKEALLE 90°
18. LOPUKSI ULOS SILMUKASTA "04"
14. LASKETAAN MITEN PITKÄÄN VOIDAAN AJAA ETEENPÄIN TÖRMÄÄMÄTTÄ SEINÄÄN.
[MITTAUKSESTA] miinus [OHJE] plus Korjaus(c), jota voi hienosäätää.
Tulos muunnetaan moottorin ohjeeksi, asteiksi.
15. KÄÄNNÖS VASEMMALLE 90° JA ULOS SILMUKASTA "04"
13. SEURATAAN SEINÄÄ, etäisyys="OHJE"
SEURATAAN LASKURIA
OHJE
ADV
MITTAUS
B+C
8



Esimerkkiohjelmamme näyttää nyt tältä.Ylimmäksi siihen on koottu ohjelmaa alustettaessa vain kerran suoritettavat toimilohkot (käskyt).

Seuraavaksi tuotetaan reaaliaikaista mittaustietoa ohjaimen näytölle (silmukka 01).

Silmukassa 02 annetaan robotille ohje, millä etäisyydellä seinästä sen tulisi kulkea. Muuttujan arvolle on määritelty minimi ja maksimi, joista ohjelma ei anna poiketa. Valittu arvo esitetään näytöllä.

Sokkelossa ajon käynnistyspainike ja valomerkit.

Käännytään tarvittaessa vasemmalle ennen vapaan tilan mittaamista.

Mitataan vapaa tila.

Käännytään oikealle.

Silmukassa 04 seurataan seinää tilan puitteissa (alempi polku),

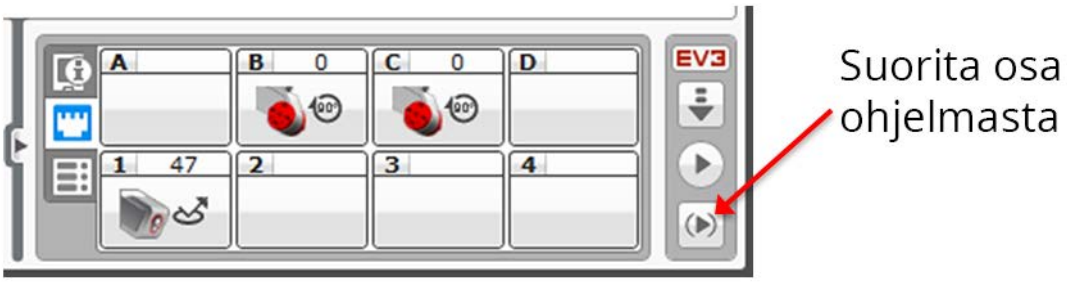
ellei käytävä sitten käänny oikealle (ylempi polkku).

Testaus ja säätäminen

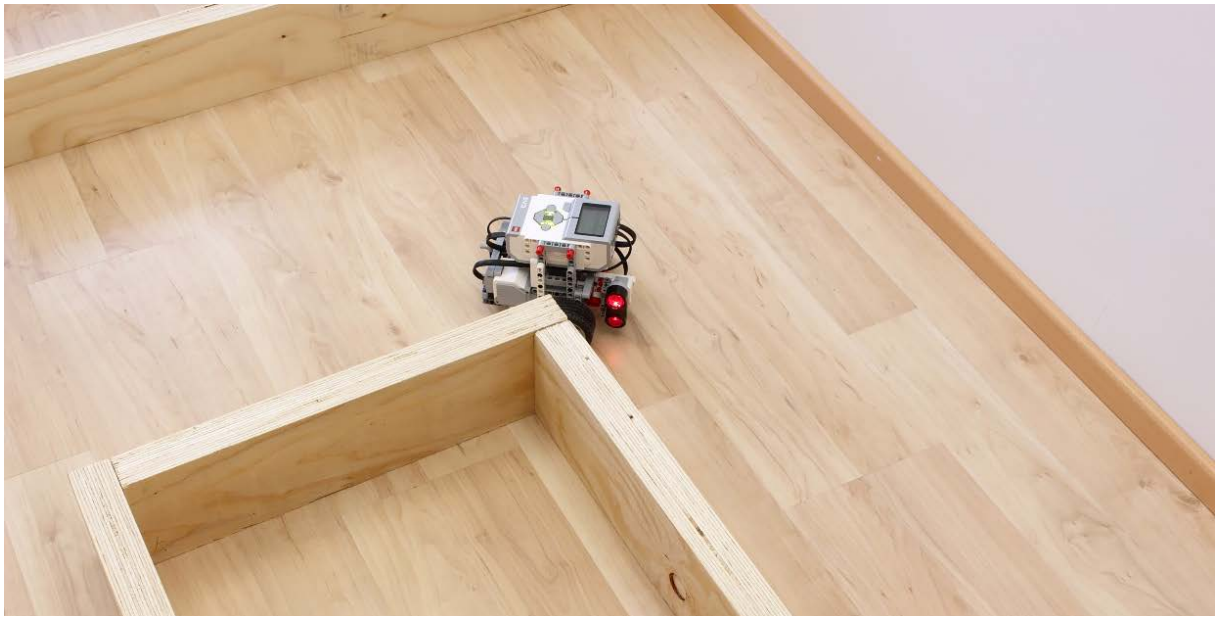
Esimerkkiohjelma on laadittu ja testattu etäisyyden ohjearvolla 6cm.

Ohjelman toiminnan testaaminen on mittaamiseen asti varsin helppoa. Kunnollisten reunuksien merkitys mittauksen luotettavuudelle on suuri. Siinä vaiheessa tulevat vastaan ensimmäiset liikkeet. Niiden testaamiseen ja hienosäätämiseen on omat kikkansa.

Esimerkiksi mahdollisuus suorittaa ohjelmaa vaikka vain yksi toimilohko kerrallan on laitemonitorin porttinäkymän ohella osoittautunut hyvin käteväksi työkaluksi kaikkien liikkeiden ja käännöksien hienosäätämisessä. Lähes välttämättömäksi nopeiden ja yhdistettyjen liikkeiden säätämisessä on osoittautunut myös mahdollisuus tutkia suoritusta hidastettuna videolta (etenkin kilparobotit).



Riittävän paksut reunukset pysyvät hyvin pystyssä ja robotti ”näkee” myös sen päädyn aina kunnolla. Kun mittaus on saatu toimimaan luotettavasti, voidaan siirtyä kokeilemaan kääntymistä kulman ympäri oikealle. Labyrintin ohella oikealle kääntymistä voi harjoitella vaikka pahvilaatikkoa kiertämällä.



Tässä harjoituksessa on kerrattu

- Ohjelman suorittamiseen vaikuttavien haarautumisien, silmukoiden ja
- Ehdollisien ohjelman osien ohjelmoiminen.
- Muuttujien ja muistipaikkojen käyttöä ohjelmissa, sekä erilaisien
- tietotyyppien ja niiden muunnoksien käyttöä
- Moottori, anturi, ääni, ja laskenta –toimilohkojen käyttöä.
- Käyttöliittymän, eli näyttöruudun ja ohjaimen painikkeiden ohjelmointia.
- Matemaattisten kaavojen käyttöä ohjelmoinnissa ja ohjelmassa.
- Kirjoitettavan ohjelman toimintakuvauksen laatimista.

Uutena asiana tuli

- Looginen vertailu ja siihen liittyvä toimilohko
- Ehdollinen poistuminen silmukasta

Kokeilkaa lopuksi kiertää labyrinttia eri etäisyksillä seinästä

Tehtävä 24. Indian Jonesin jalanjäljillä, kilpailu

Kun robotti on saatu kulkemaan labyrintissa voidaan tehtävään lisätä uusia piirteitä.

Kuvaamataidon tunnilla voidaan esimerkiksi maalata inka temppelin kulisseja, jotka sitten kiinnitetään labyrintin seiniin. Jos labyrintti tuntuu pieneltä, sitä sopii toki kasvattaa. Kulkutien tulisi kuitenkin pysyä vähintään yhtä leveänä ja suunnilleen suorakulmaisena. Myös roboteille voi tehdä rooliasuja, vaikka ruskeaksi maalatun hatun paperimassasta jne.

Lavastuksessa saa käyttää muutenkin mielikuvitusta. Sopivalla valaistuksella, muutamalla hämähäkillä ja ansalla ryyditettynä rata muuttuukin jo aivan toiseksi.

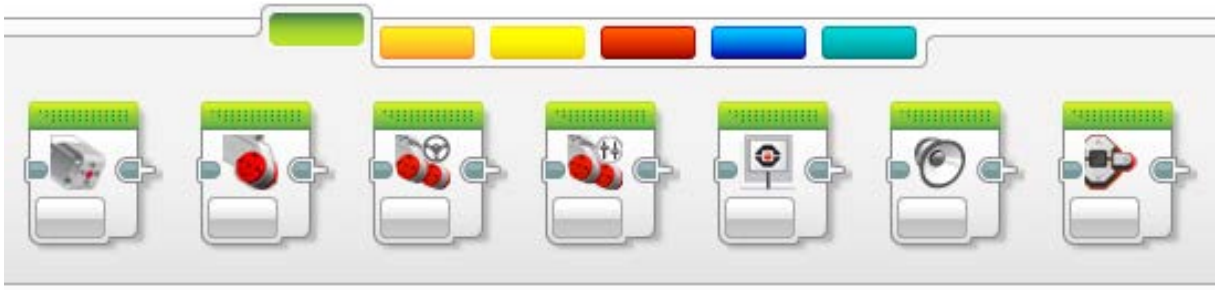
Tehtävä

Merkitään radalle kadonneita aarteita värillisillä kartongista leikatuilla korteilla (eri värejä). Tehtävänä on varustaa robotti valokennolla, sekä ohjelman lisäyksellä, joka kykenee tunnistamaan niistä mahdollisimman monta. Robotin tulee antaa äänimerkki aina kun se löytää aarteen ja pitää lukua löydetyistä aarteista.

Robottien seikkailuista voidaan kuvata ja leikata video muistoksi, sekä laittaa se vaikka koulun verkkosivuille.

8. Ohjattavien laitteiden toimilohkot

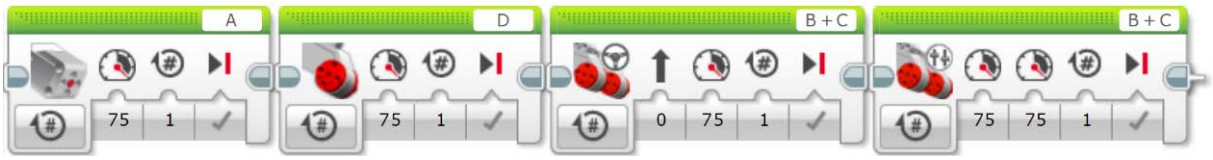
Graafisissa ohjelmointikielissä käskyt esitetään helposti omaksuttavien kuvakkeiden eli toimilohkojen muodossa. Seuraavaksi esiteltävistä toimilohkokonaisuuksista löytyvät toimilohkojen yksityiskohtaiset suomenkieliset kuvaukset, sekä niiden käyttöä koskevat tiedot.



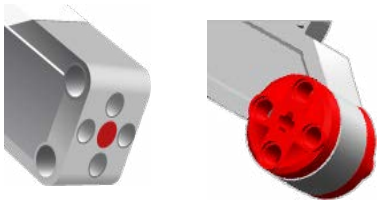
- **KESKIKOKOINEN MOOTTORI**
- **ISO MOOTTORI**
- **KAKSI MOOTTORIA, ohjataan kuten autoa (Move Steering)**
- **KAKSI MOOTTORIA, ohjataan kuten työkonetta (Move Tank)**
- **OHJAIMEN NÄYTTÖRUUTU**
- **ÄÄNI**
- **OHJAIMEN MERKKIVALOT**

Moottoritoimilohkot (Medium & Large Motor, Move Steering & Tank)

EV3:n moottorien ohjaamisessa voidaan käyttää neljää, toisistaan hiukan poikkeavaa toimilohkoa.



Medium ja Large Motor



Näillä kahdella muuten samanlaisella toimilohkoilla annetaan komentoja aina yhdelle moottorille kerrallaan, tekivätpä nämä moottorit mitä tahansa.

Seuraavat kaksi toimilohkoa taas ovat käteviä, jos robotti etenee kahdella moottorilla, ja sen ohjaus perustuu näiden nopeuseroon, eli pyörimiseen eri nopeuksilla.

Move Steering (ohjauspyörän kuva)

Tätä toimilohkoa käytetään, kun halutaan ohjata robottia kuten autoa, kääntämällä rattia enemmän, tai vähemmän.

Move Tank (liiku kuin tankki)

Tätä toimilohkoa käytetään, kun halutaan ohjata robottia kuten tankkia, tai työkoneita. Kun halutaan ajaa suoraan, molemmille moottoreille annetaan sama nopeusohje. Ja käännyttäessä nopeusohjeiden ero määrää miten jyrkästi robotti kääntyy.

Moottoritoimilohkojen toiminnot

ON, käynnistä moottori, moottori käy kunnes toisin käsketään

OFF, pysäytä moottori

On for Seconds, käynnistä moottori ja pysäytä se tietyn ajan kuluttua. Haluttu matka kannattaa ilmoittaa sekunneissa, jos esimerkiksi ei ole varmaa, että robotti voi ajaa esteettömästi pyydetyn matkan loppuun asti. Vaikka tielle olisi sattunut este, ajan tultua täyteen moottori pysähtyy ja ohjelman suoritus jatkuu normaalisti. Aika ei ole matkan määreenä yhtä tarkka, kuin muut menetelmät (mm. akun lataus ja kuormitus vaikuttavat).

On for Degrees, käynnistä moottori ja pysäytä se, kun se on kääntynyt tietyn tarkan määrän. Määrä ilmoitetaan astelukuna. Yksi aste on 1/360 osa täydestä ympyrästä, eli hyvin pieni osa siitä. Tämä on kätevin tapa ilmoittaa haluttu matka, jos se on pieni.

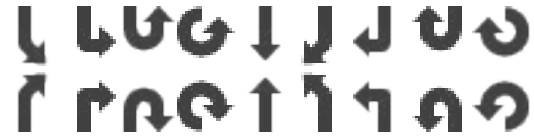
On for Rotations, käynnistä moottori ja pysäytä se, kun se on pyörinyt tietyn määrän kokonaisia kierroksia. Tässä, kuten lähes aina, voidaan käyttää myös desimaali arvoja (desimaalierottimena käytetään pistettä). Eli esimerkiksi ”kaksi ja puoli kierrosta” kirjoitetaan numeroina 2.5

Moottoritoimilohkojen toiminnot

Moottoritoimilohkojen parametrit

Nopeus (-100...+100)

- Iso max 175rpm (kierrosta/min)
- Keskikokoinen max 260rpm



Ohjaus, eli paljonko rattia käännetään (-100...+100)

±100 = robotti pyörii paikallaan

± 50 = yksi pyörä pyörii jne paikallaan

Matka asteina, On for Degrees

1 moottorin kierros on 360 astetta

Matka kierroksina, On for Rotations

Matka sekunteina, On for Seconds

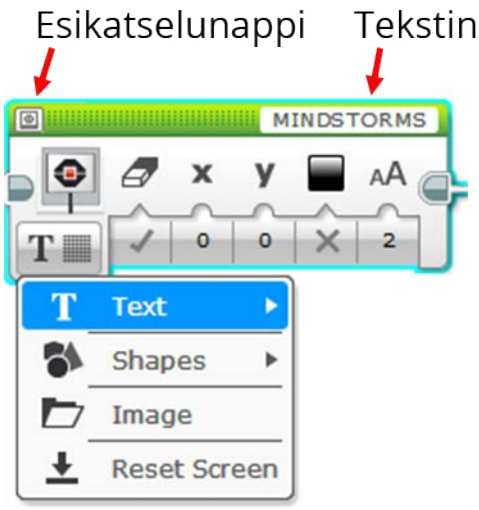
Kun moottori pysäytetään, laitetaan sähköinen ”käsijarru” päälle

Kun moottori pysäytetään, se saa jäädä pyörimään vapaasti

Moottorin laitetunnus, ulkoinen tulotieto (Wired)

HUOM: Molempien moottoreiden tulee pyöriä pyydetty matka täyteen, ennen kuin ohjelma jatkaa suoritustaan.

Näytönohjaus-toimilohko (Display)



Näytönohjaus-toimilohkoa käyttäen voidaan esittää tekstejä, kuvia, mittaustietoja jne EV3:n 178x128 pikselin kokoisella näyttöllä.

Toiminnot

Text

Näytölle voidaan kirjoittaa tekstiä vapaa valintaisella sijoittelulla Pixels moodissa. Grid moodissa tekstiä varten on 11 ennalta määriteltyä riviä.

Shapes

Näytölle voidaan tätä toimilohkoa käyttäen myös piirtää yksinkertaisia geo-metrisiä muotoja, kuten viivoja, suorakaiteita, ympyröitä, sekä yksittäisiä pisteitä.

Image

Monimutkaisemmat kuviot kannattaa tehdä kuvatiedostoiksi ohjelmointi-työkalun Image Editorilla, josta on kerrottu tarkemmin ohjelmointiohjelman Help:issä. Image Editoriin tutustutaan myöhemmin **tässä kirjasarjassa**.

Reset Screen

Reset Screen toiminto palauttaa näytölle vakionäkymän.

Näytönohjaus-toimilohkon parametrit

Clear Screen, tyhjennetäänkö näyttö ensin (tosi/epätosi)

Tekstin tms. paikka vaaka ja pystysuunnassa

Color, Taustan ja tekstin väri, normaali=False, Negatiivi=True

Font, Kirjaimien koko

Column, Sarake Row, Rivi (tekstin paikka)

Viivan alku- ja loppupisteiden koordinaatit

Radius, Ympyrän säde

Fill, Täytetäänkö ympyrä tai neliö

Width & Height, Suorakaiteen leveys ja korkeus

Esitettävän kuvatiedoston nimi, ulkoinen tulotieto (Wired)

Ääni-toimilohko (Sound)

Ääni-toimilohko soittaa nuotin, äänen, tai nimetyn äänitiedoston. Useimmista toimiloh-koista poiketen ohjelma voi, niin haluttaessa, jatkaa tästä heti eteenpäin soiton yhä jatkuessa.

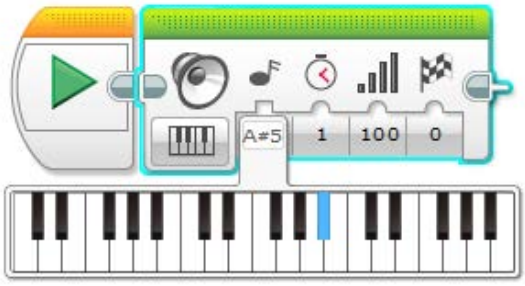
- Pysäytä soitto
- Soita äänitiedosto
- Soita ääni
- Soita nuotti

Ääni-toimilohkon parametrit

Hz — Soitettavan äänen taajuus Herzeinä (300-10000)

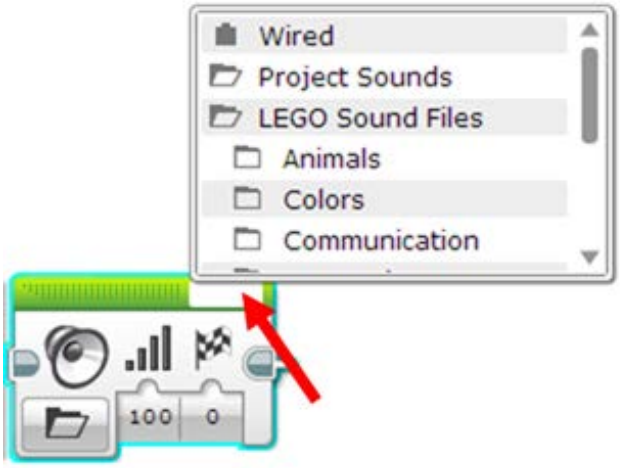
Soitettava nuotti
- Nuotti, katso taulukko (A-G)
- Oktaavi (4-6)
- Puolinuotti = #

A-G	Do,Ré,Mi
C	Do
D	Ré
E	Mi
F	Fa
G	Sol
A	La
B	Si



Näppäimistötoiminto auttaa nuottien valinnassa

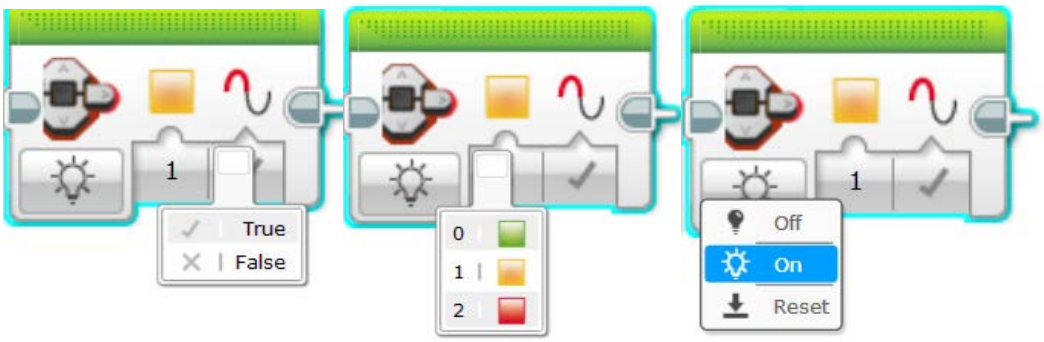
	Kesto sekunteina, Huom! Lyhyet kestot merkitään desimaalilukuina, esim. 0.3 tai vaikka 0.15 sekuntia.
	Äänenvoimakkuuden ohje (0-100)
	Suoritustapa
	Ohjelman suoritus odottaa käsketyn tehtävän suorittamisen loppuun asti ja jatkaa vasta sitten.
	Käsketty tehtävä suoritetaan yhden kerran, ohjelman suoritus jatkaa silti heti eteenpäin
	Käskettyä tehtävää suoritetaan uudestaan ja uudestaan ikuisesti, ellei muuta määrätä. Ohjelman suoritus jatkaa silti heti eteenpäin



Soitettava äänitiedosto valitaan valikosta, joka aukeaa toimilohkon otsikko-kentän ikkunaa klikkaamalla

 Esitettävän äänitiedoston nimi, ulkoinen tulotieto (Wired)

Merkkivalo -toimilohko (Brick Status Light)



Toiminto Väri (Colour) Vilkkuu (Pulse)

V3-ohjaimen painikkeita ympäröi merkkivalo. Sen kolmen erivärisen valon avulla välitetään käyttäjälle tietoa ohjaimen toiminnasta.

Kun käynnistät EV3:n, merkkivalo toimii seuraavalla tavalla:

- Punainen kertoo että ohjainta ollaan käynnistämässä, tai sammuttamassa.
- Vihreä kertoo ohjaimen olevan käynnissä, mutta itse robotin ohjelma, robotti, ei ole käynnissä.
- Vilkkuva vihreä kertoo ohjelman, robotin, olevan käynnissä.
- Oranssi on hälytys (esim. akku vähissä).

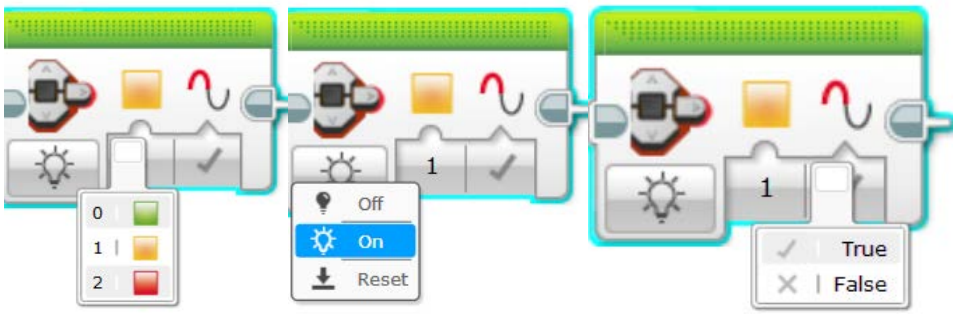
(yleisimmät tapaukset, käyttöohjeessa laajempi kuvaus

Ohjelmoijan kannalta mielenkiintoista on se, että tämän merkkivalon toiminta on täysin ohjelmoitavissa. Äänimerkkien ohella robotti voidaan ohjelmoida kertomaan asioita esimerkiksi ohjelman suorituksen etenemisestä käyttäjälle myös valomerkkien avulla.

Ohjelman suoritus ei pysähdy odottamaan merkkivalo –toimilohkolla annetun käskyn suorittamista loppuun. Se laittaa valon päälle, sammuksiin, tai vilkkumaan, ja jatkaa välittömästi eteenpäin suorittamaan seuraavaa käskyä.

Merkkivalo-toimilohkon toiminnot (Brick Status Light)

- **Off**, sammutetaan merkkivalo
- **On**, vilkkuva tai yhtäjaksoinen valo
- **Reset**, Palautetaan oletusasetukset



Toiminto Väri (Colour) Vilkkuu, (kyllä /ei)

9. Ohjelman kulkua ohjaavat toimilohkot

- **ALOITUS**
- **VIIVE**
- **SILMUKKA, toisto**
- **IF-THEN (-ELSE), ehdollinen suoritus**
- **SILMUKAN KESKEYTYS**

Ohjelman aloitus-toimilohko (Start)

Kaikki ohjelmat alkavat tällä toimilohkolla ja niitä voi olla ohjelmassa vain yksi. Jos robotti on kytkettynä tietokoneeseen, sen ohjelman voi käynnistää klikkaamalla vihreää kolmiota. Mutta varo ettei robottisi tällöin esimerkiksi karkaa pöydältä ja putoa lattialle särkyen.

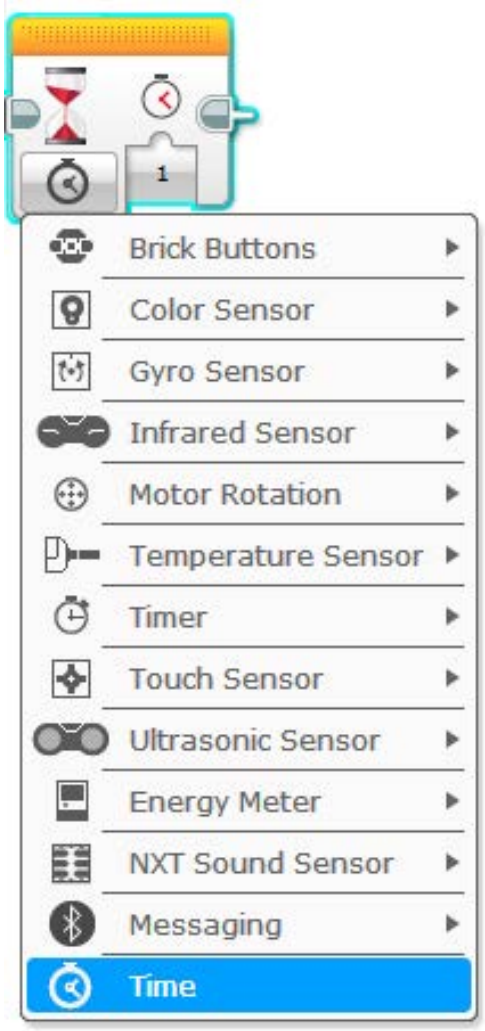
Toiminnot ja parametrit

Tähän toimilohkoon ei liity toiminnon valintaa, tai aseteltavia parametreja.

Viivetoimilohko (Wait)

Viivetoimilohko, ohjelman suoritus pysähtyy tämän toimilohkon kohdalle ja jatkaa vasta annetun **ehdon** täytyttyä.

Toiminnot = ehto



- Ohjaimen painikkeet
- Väri tai valoisuus
- Gyroskooppi
- Infrapuna anturi
- Moottorin pulssianturi
- Lämpötila
- Ajastin (käynnistetty aiemmin)
- Kosketus
- Ultraääni
- Energia, virta, jännite ja teho
- NXT:n äänisensori
- Bluetooth viesti
- Aika

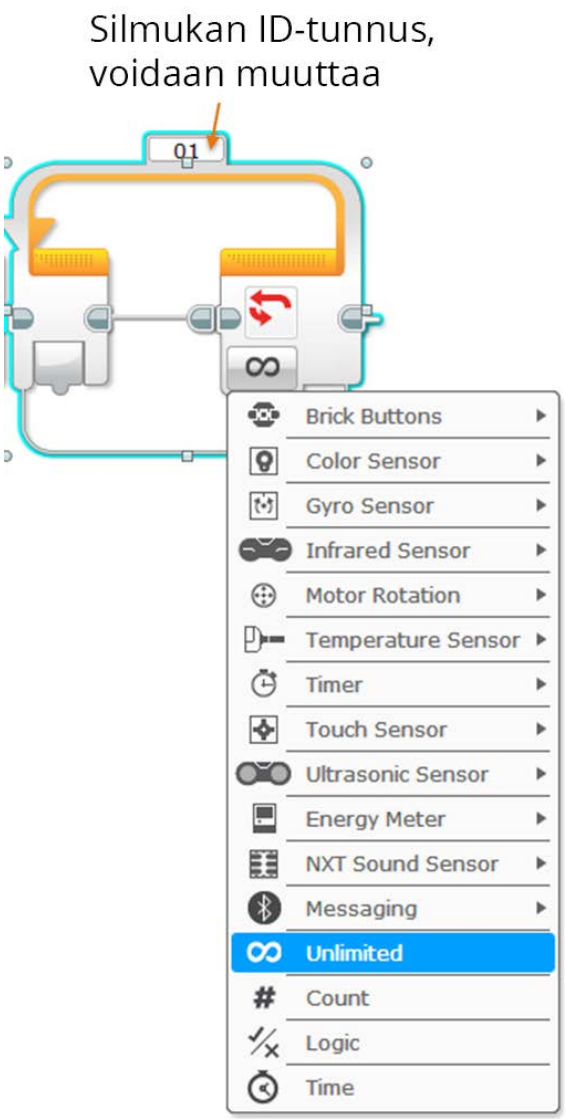
Compare, Vertaile

Update, Päivitä

Change, Muutos

Silmukkatoimilohko (Loop)

Silmukkatoimilohko, silmukan sisään laadittua ohjelman osaa toistetaan, kunnes silmukasta poistumisen **ehto** täyttyy.



Toiminnot = ehto

- Ohjaimen painikkeet
- Väri tai valoisuus
- Gyroskooppi
- Infrapuna anturi
- Moottorin pulssianturi
- Lämpötila
- Ajastin (käynnistetty aiemmin)
- Kosketus
- Ultraääni
- Energia, virta, jännite ja teho
- NXT:n äänisensori
- Bluetooth viesti
- Ikuinen silmukka
- Laskuri
- Looginen ehtolauseke
- Aika

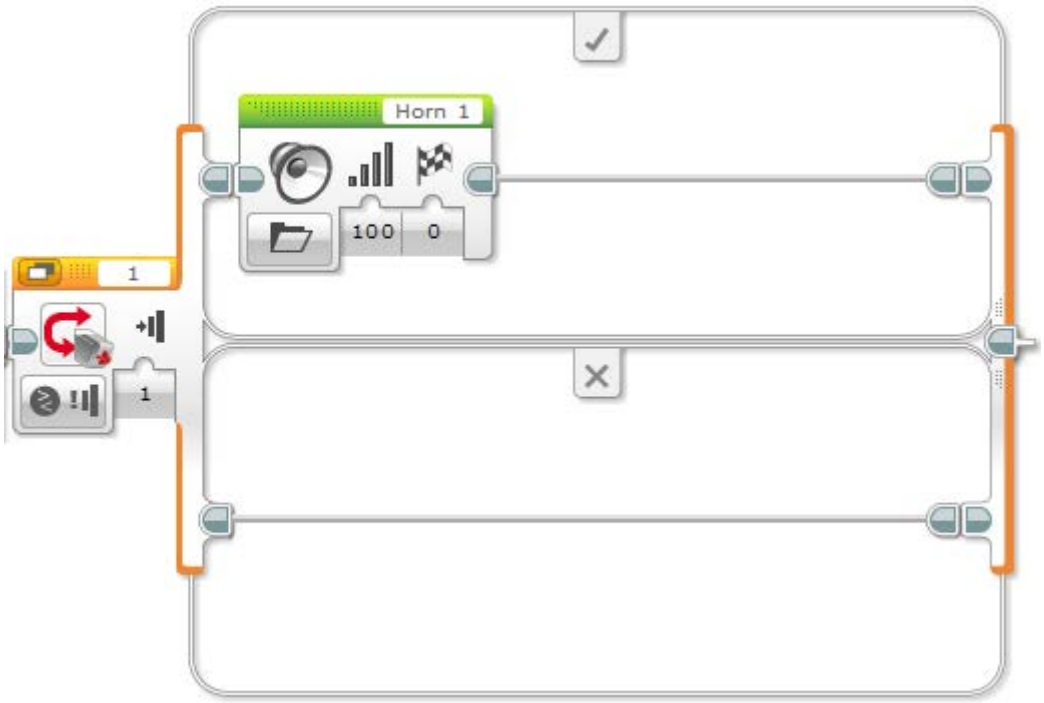
If-Then -toimilohko (Switch)

If-Then -toimilohko, ehdollinen ohjelman osan suorittaminen.

Jos (If) ehto toteutuu

➔ Silloin (Then) ohjelman suoritus kulkee ylempää polkua (✓).

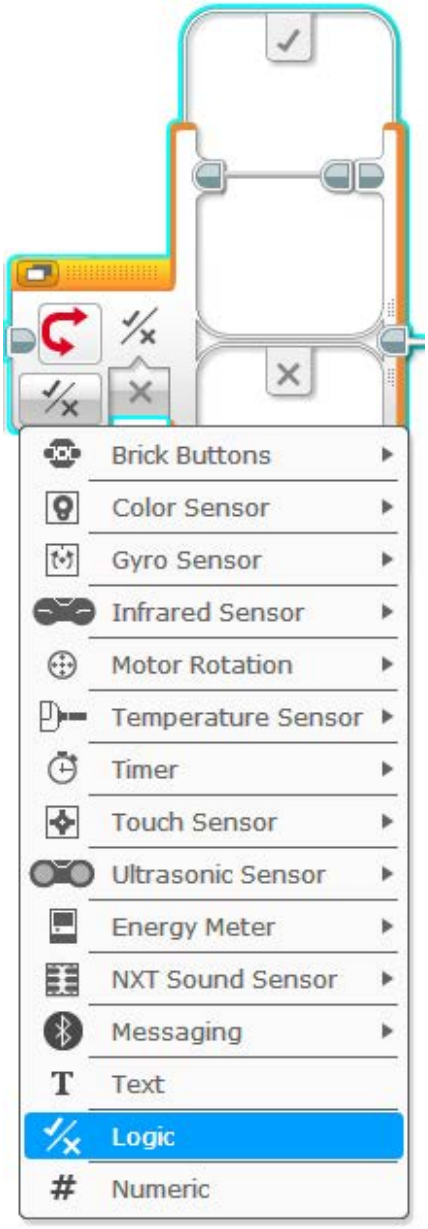
➔ Muutoin (Else) ohjelman suoritus kulkee alempaa polkua (✗).



Ehtona on kosketusanturin painaminen.

- Jos joku painaa kosketusanturin nappia, ohjelma kulkee ylempää polkua ja soittaa torvea.

If-Then toimilohkon toiminnot

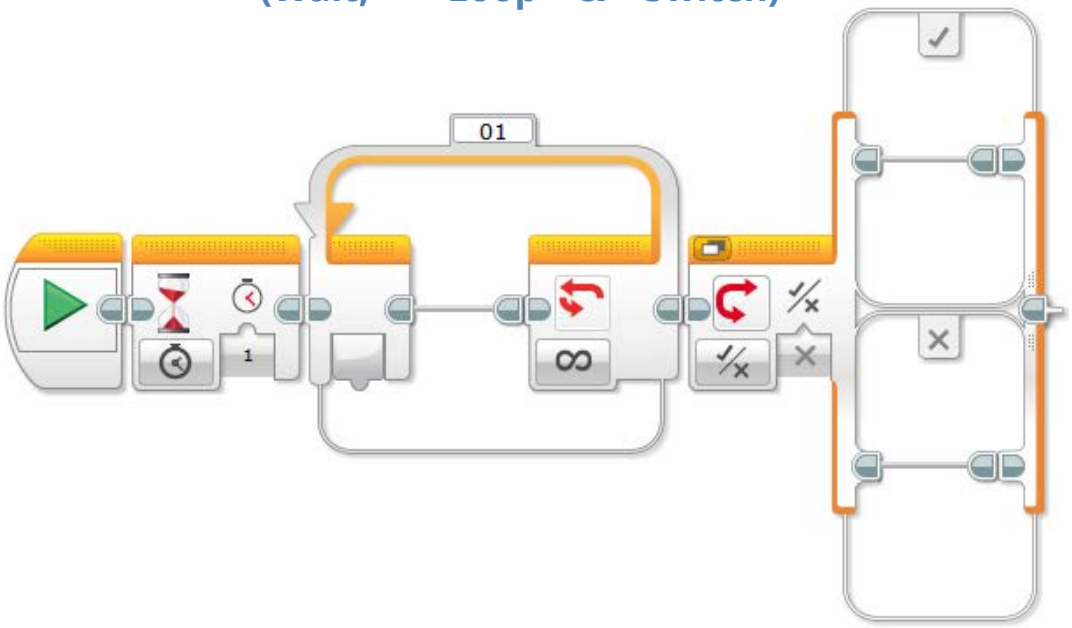


Toimintaa ohjaava ehto

- Ohjaimen painikkeet
- Väri tai valoisuus
- Gyroskooppi
- vvInfrapuna anturi
- Moottorin pulssianturi
- Lämpötila
- Ajastin (käynnistetty aiem-
  min)
- Kosketus
- Ultraääni
- Energia, virta, jännite ja teho
- NXT:n äänisensori
- Bluetooth viesti
- Tekstimuotoinen viesti
- Looginen ehtolauseke
- Numero (kokonaisluku)

Viive, silmukka ja If-Then toimilohkojen parametrit

(Wait, Loop & Switch)



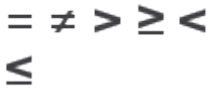
Color, väri jonka valokenno mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Set of Brick Buttons, Käytettävät ohjaimen painikkeet, valitse yksi tai useampi painike valikosta.

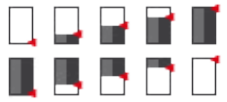
State, tila, edellyttääkö ehdon täyttyminen, että painike on ylhäällä, painettu alas, tai painallusta (alas-ylös).

Button ID, kertoo mikä painike oli painettuna ennen kuin ohjelma jatkoi tästä toimilohkosta eteenpäin (Lähtö).

Set of Colors, värit jotka täyttävät ehdon, yksi tai useampia.



Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.



Treshold Value, vertailuluku (0...100), (-100...100), (tms)

Light, mitattu valon määrä jonka valokenno näki viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Direction, mihin suuntaan mittausarvon muutoksen tulee olla.

Amount, ohjelman jatkamisen ehtona olevan muutoksen vähimmäismäärä.

Number, ohjelman kulkua If-Then toimilohkossa ohjaava kokonaisluku.

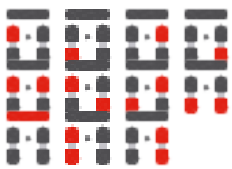
Angle, mitattu asentotieto jonka gyroskooppi mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Rate, kulmakiihtyvyys jonka gyroskooppi mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Proximity, etäisyys (mm) jonka infrapuna-anturi mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Channel, IR-lähettimen kanava

Heading, suunta josta infrapuna-anturin havaitsema signaali tuli ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).



Set of Remote Button IDs, Käytettävät kauko-ohjaimen painikkeet, valitse yksi tai useampi painike valikosta.

Button ID, kertoo mikä kauko-ohjaimen painike oli painettuna ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Degrees, moottorin kulkema matka asteina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Rotations, moottorin kulkema matka kierroksina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Current Power, moottorin nopeus ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Temperature Celcius, lämpötila Celcius asteina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Temperature Fahrenheit, lämpötila Fahrenheit asteina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Timer ID, tarkasteltavan ajastimen tunnus

Elapsed Time, Ajastimen lukema, kun ohjelma jatkoi tästä eteenpäin (Lähtö).

Time, aika jonka ohjelma odottaa ennen jatkamistaan (Tulo)

Measured Value, kertoo oliko painike painettuna kun ohjelma jatkoi tästä eteenpäin (Lähtö).

Distance in Centimeters, etäisyys senttimetreinä

Distance in Inches, etäisyys tuumina

Presence/Listen, Kuuntelu moodi, ohjelman suoritus jatkuu kun ultraäänianturi havaitsee jonkin toisen lähteen lähettämän ultraäänisignaalin.

Measured Value, kertoo kyseisen suureen mittaustiedon hetkeltä kun ohjelma jatkoi tästä eteenpäin (Lähtö).

Sound Level, Äänen voimakkuuden mittaustieto hetkeltä kun ohjelma jatkoi tästä eteenpäin (Lähtö).

Logic, Looginen ehto, tosi (True) / epätosi (False)

Comparison Text, Vertaa tekstiä, käytetään kahden robotin kommunikoidessa keskenään Bluetooth yhteydellä. Lisätietoja EV3 Help, Messaging Block

Message, Teksti johon vastaanotettua saman nimistä, toisen robotin lähettämää Bluetooth viestiä verrataan. Lisätietoja EV3 Help, Messaging Block

Anturin laitetunnus, ulkoinen tulotieto (Wired)

Keskeytä silmukka -toimilohko (Loop Interrupt)

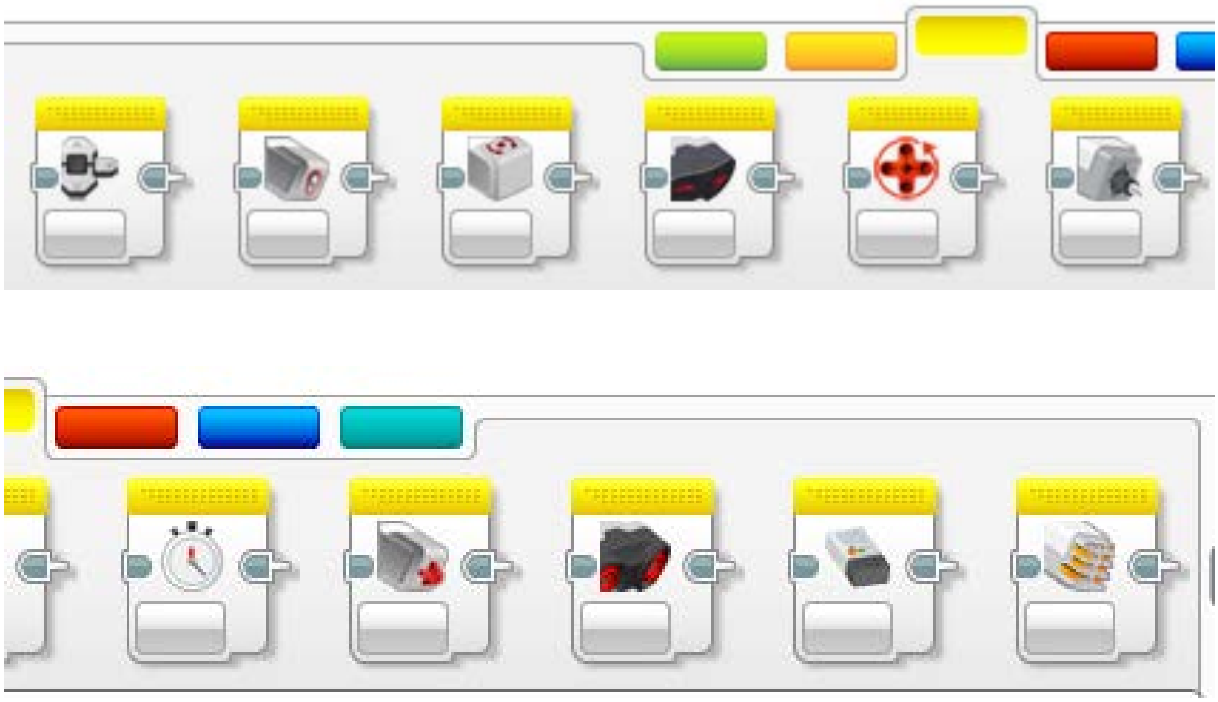
Keskeytettävän silmukan tunnus

Silmukka –toimilohkon keskeytys voidaan määritellä tapahtuvaksi esim. laskurin, ajan, anturin jne. perusteella. Aina ei kuitenkaan voida ennalta määritellä riittävän tarkasti milloin jonkin tehtävän toistaminen pitäisi keskeyttää ja ohjata ohjelman kulku ulos silmukasta.

Tällöin voidaan laatia ohjelman toiseen haaraan silmukan keskeyttämiseen liittyvien tietojen käsittely ja keskeyttää sitten nimetty silmukka Keskeytä silmukka –toimilohkolla.

Tälle toimilohkolle tarvitsee antaa vain yksi parametri, keskeytettävän silmukan tunnus.

10. Anturien toimilohkot



LEGO:n omat anturit, tilanne tätä kirjaa kirjoitettaessa

- OHJAIMEN PAINIKKEET
- VALO ja VÄRI
- GYROSKOOPPI *
- INFRAPUNA
- MOOTTORIN PYÖRIMINEN
- LÄMPÖTILA *
- AJASTIMET, 8kpl
- KOSKETUS
- ULTRAÄÄNI *
- SÄHKÖTEKNIIKAN MITTAUKSET *
- **ÄÄNEN VOIMAKKUUS ***

HARMAALLA MERKITYT ANTURIT EIVÄT SISÄLLY EV3 EDUCATION PERUSPAKETTIIN (LEGO 45544). Antureita on saatavilla erikseen.

*** Tähdellä merkityt toimilohkot (ja anturit) eivät sisälly ohjelman ilmaiseen kuluttajaversioon, mutta ovat ladattavissa osoitteesta**

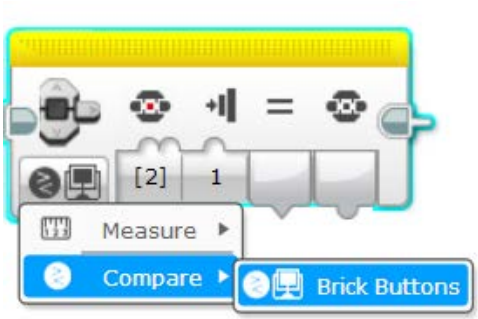
http://www.lego.com/fi-fi/mindstorms/downloads?ignorereferer=true

Ladatun tiedoston asennus tehdään valitsemalla työpöydän Tools -valikosta Block Import, ja seuraamalla ohjeita.

EV3:lle on julkaistu useiden valmistajien toimesta koko joukko muitakin antureita. Näihin palataan tarkemmin tämän kirjasarjan seuraavassa osassa.

Ohjaimen painikkeet -toimilohko (Brick Buttons)

Ohjainyksikön etupuolella, keskellä, on viisi komentopainiketta. Tämä toimilohko tarkastaa tulevat signaalit, eli niistä jotakin painettu, tai painetaan yhä.



Kuudetta, eli näyttöruudun alla vasemmalla sijaitsevaa keskeytyspainiketta ei voi hyödyntää ohjelmissa.

Toiminnot

Mittaus, toimilohkon lähtötietona on **tieto siitä, mitä painiketta painetaan**. Mikäli useita painikkeita on painettuna yhtä aikaa, se voi kertoa niistä vain yhden, katso taulukko.

&	1 vasen	2 keskellä	3 oikea	4 ylös	5 alas
1 vasen		2	3	4	5
2 keskellä	2		2	4	2
3 oikea	3	2		4	5
4 ylös	4	4	4		4
5 alas	5	2	5	4	

Vertailu (Compare), toimilohkoa käytettäessä voidaan nimetä seurattava painike/painikkeet. Tällä toimilohkolla on kaksi lähtötietoa. Tieto siitä, onko nimettyä/nimettyjä painikkeita painettu (Tosi/Epätosi), sekä mitä painiketta painetaan (katso edellinen kohta, Mittaus).

Huom. Jos laitat tämän toimilohkon silmukka-toimilohkon sisään, se kykenee tunnistamaan ja muistamaan myös suorituskertojen välillä tehdyn painalluksen (Bumped).

Ohjaimen painikkeet -toimilohkon parametrit

Set of Brick Buttons, Käytettävät ohjaimen painikkeet, valitse yksi tai useampi painike valikosta.

State, tila, edellyttääkö ehdon täyttyminen, että painike on ylhäällä, painettu alas, tai painallusta (alas-ylös).

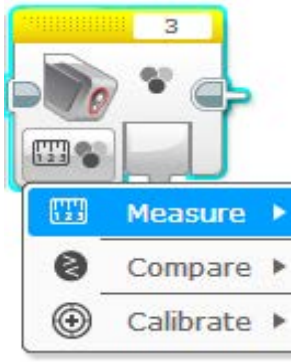
Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö).

Button ID, kertoo mikä painike oli painettuna ennen kuin ohjelma jatkoi tästä toimilohkosta eteenpäin (Lähtö).

Valokennotoimilohko (Color Sensor)

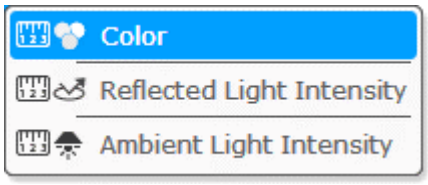
EV3:n valokenno tunnistaa apuvalojensa avulla värejä ja kohteesta heijastunutta valoa. Apuvalot, sinistä lukuunottamatta, sammutettuina se mittaa ympäristöstä tulevan valon määrää.

Toiminnot

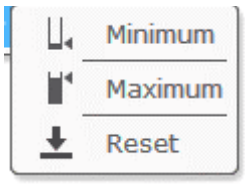


- Mittaus
- Mitattua tietoa verrataan annettuun vertailulukuun
- Kalibrointi, anturille opetetaan miten musta musta on ja miten välkoinen valkoinen on, näissä olosuhteissa.

Mitä halutaan mitata tai vertailla



- **Väri**
- Kohteesta **heijastuneen valon määrä**
- **Ympäröivän valon määrä**



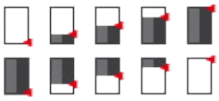
- Kalibroidaan musta (tumma kohde)
- Kalibroidaan valkoinen (vaalea kohde)
- Palautetaan alkuperäiset asetukset

Valokennon parametrit

Set of Colors, värit jotka täyttävät ehdon, yksi tai useampia.

Color, valokennon mittaama väri (Lähtö).

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)



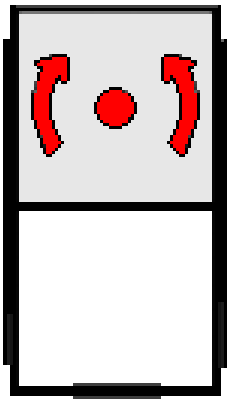
Treshold Value, vertailuluku (0...100)

Light, valokennon mittaama valon määrä (Lähtö).

Anturin laitetunnus, ulkoinen tulotieto (Wired)

Gyro-toimilohko (Gyro Sensor)

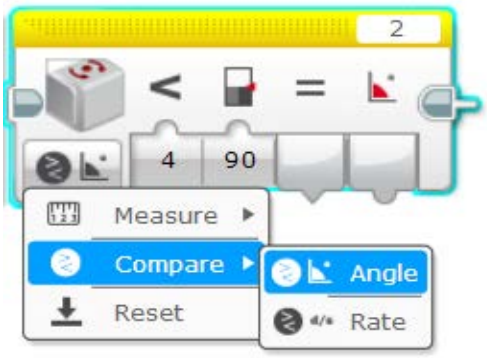
Gyro-toimilohkolla ja gyroskooppi anturilla voidaan mitata asennon muutoksia, eli kulmaa ja kulmakiihtyvyyttä. EV3 gyroskooppi anturi on ns. yksi-akselinen MEMS gyro, eli elektroninen gyro.



Akseli jonka ympäri kääntymistä mitataan on merkitty anturiin punaisella pisteellä, sekä mitattavissa olevia py rimisen suuntia kuvaavilla nuolilla. Kääntyminen esitetä asteina, + merkkinen suunta on myötäpäivään (miinus v tapäivään).

Huomaa myös, että mittaustieto ei nollaudu, kun kierro tulee täyteen, vaan jatkaa kasvamistaan.

Toiminnot



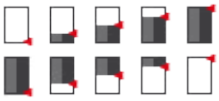
- **Mittaus**, kulma tai kulmakiihtyvyys
- **Vertailu**, kulma tai kulmakiihtyvyys
- **Nollaus**, nollaa mittauksen

Koska toimilohko muodostaa kulma-tiedon laskemalla sen anturin mittaamasta kulma-kiihtyvyys tiedosta, voi siihen kertyä vähitellen virhettä. Siksi on suositeltavaa nollata mittaus, kun siihen on mahdollisuus.

Anturin kiinnittämiseen tulee myös kiinnittää huomiota. Sen tulee olla suorassa asennossa mitattavaan akseliin nähden ja tukevasti kiinnitetty.

Gyro-toimilohkon parametrit

Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.



Treshold Value, vertailuluku

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Angle, mitattu asentotieto jonka gyroskooppi mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Rate, kulmakiihtyvyys jonka gyroskooppi mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Anturin laitetunnus, ulkoinen tulotieto (Wired)

*** Tämä toimilohko ei sisälly ohjelman ilmaiseen kuluttajaversioon, mutta on ladattavissa osoitteesta**

http://www.lego.com/fi-fi/mindstorms/downloads?ignorereferer=true

Ladatun tiedoston asennus tehdään valitsemalla työpöydän Tools -valikosta Block Import, ja seuraamalla ohjeita.

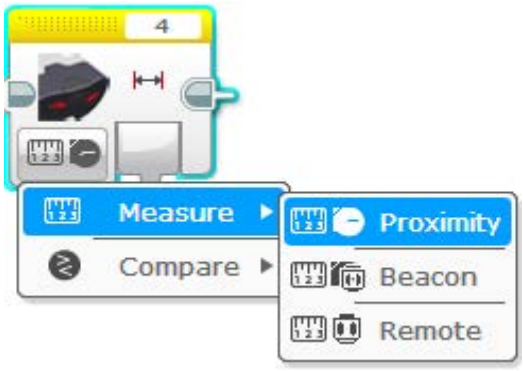
Anturi: EV3 Gyro Sensor, LEGO set 45505

Infrapuna-anturi -toimilohko (Infrared Sensor)

Infrapuna-anturi sisältyy EV3:n kulutta-ja-pakkaukseen, kuten myös tähän liittyvä kaukosäädin.

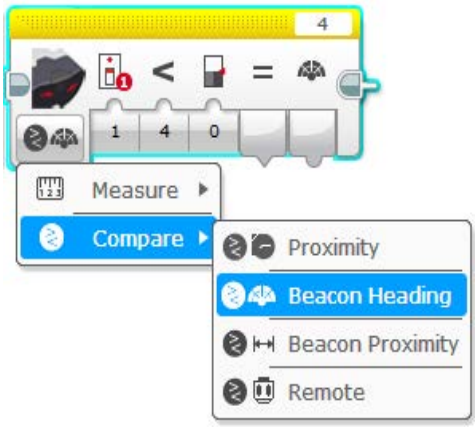
(EV3 Infrared Sensor, LEGO set 45509)

(EV3 Infrared Beacon, LEGO set 45508)

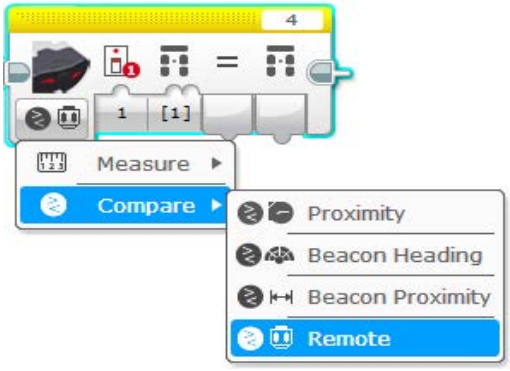


Toiminnot

Sen perustoiminnot ovat **mittaaminen** (Measure) ja **vertailu** (Compare). Siihen liittyy myös kaksi aivan erityistä toimintoa.



Beacon, Majakan etsintä-toiminnon avulla voidaan tunnistaa onko lähettyvillä muita IR-valon lähteitä, miten kaukana ja missä suunnassa ne ovat.
Remote, Etä-ohjaus toiminto puo-lestaan mahdollistaa robotin ohjaamisen kauko-säätimellä.



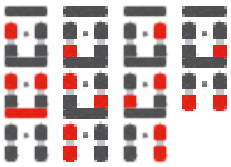
Remote toiminto on myös se syy, jonka takia infrapuna-anturin käyttö on kielletty osassa robottien kilpailulajeja. Toisaalta taas osaan kilpailuista ei voi osallistua ilman Beacon-toimintoa (kuten jalkapallo).

Infrapuna-anturin parametrit

Proximity, etäisyys (mm) jonka infrapuna-anturi mittasi viimeksi ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Channel, IR-lähettimen kanava

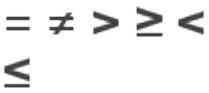
Heading, suunta josta infrapuna-anturin havaitsema signaali tuli ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).



Set of Remote Button IDs, Käytettävät kauko-ohjaimen painikkeet, valitse yksi tai useampi painike valikosta.

Button ID, kertoo mikä kauko-ohjaimen painike oli painettuna ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Detected, Havaittu toinen IR-valon lähde, tosi/epätosi (Lähtö)



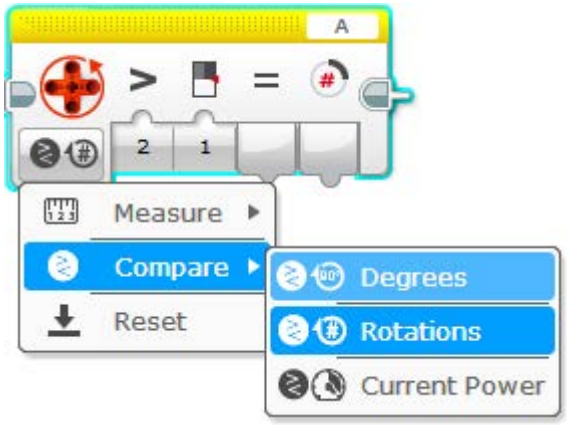
Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.

Treshold Value, vertailuluku (0...1000)

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Anturin laitetunnus, ulkoinen tulotieto (Wired)

Moottorin pyörimistieto -toimilohko (Motor Rotation)



EV3:n servomoottorit toimivat myös pyörimisen mitta-antureina.

Tämän toimilohkon avulla voidaan mitata ja vertailla moottorin pyörimistä kokonaisina kierroksina, tai asteina. Lisäksi tätä toimilohkoa käyttäen voidaan tarkastella moottorin ottaman tehon määrää.

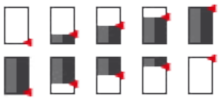
Tarkkuus: 0.5 astetta

Toiminnot

- Mittaus (Measure)
- Vertailu (Compare)
- Nollaus (Reset)

Moottorin pyörimistieto -toimilohkon parametrit

= ≠ > ≥ < ≤ Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.



Treshold Value, vertailuluku (-100..0..100)

=

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Degrees, moottorin kulkema matka asteina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

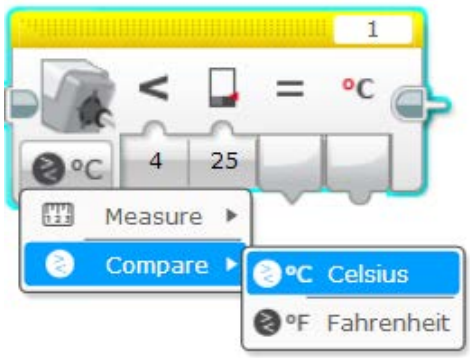
Rotations, moottorin kulkema matka kierroksina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Current Power, moottorin nopeus ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Moottorin laitetunnus, ulkoinen tulotieto (Wired)

Lämpötilamittauksen toimilohko (Temperature Sensor)

Lämpötilan mittauksissa käytetään EV3:n kanssa yhteensopivaa edellisen laite-sukupolven, eli NXT:n lämpötila-anturia.



Teknistä taustaa: NXT:n lämpötila-anturi on A/D-muuntimella varustettu NTC-termistori, jonka resistanssin lämpötilakerroin on negatiivinen ja tyypillisesti voimakkaan epälineaarinen.

Käyttöalue -20°C .. +120°C (-4°F .. +248°F), tarkkuus ±1-2°C

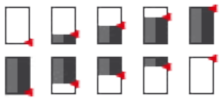
Toiminnot

- Mittaus (Measure), Celsius ja Fahrenheit asteina
- Vertailu (Compare), Celsius ja Fahrenheit asteina

Lämpötilanmittauksen toimilohkon parametrit

= ≠ > ≥ < ≤

Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.



Treshold Value, vertailuluku (-100..0..100)

=

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

°C

Temperature Celcius, lämpötila Celcius asteina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

°F

Temperature Fahrenheit, lämpötila Fahrenheit asteina ennen kuin ohjelma jatkoi tästä eteenpäin (Lähtö).

Anturin laitetunnus, ulkoinen tulotieto (Wired)

*** Tämä toimilohko ei sisälly ohjelman ilmaiseen kuluttajaversioon, mutta on ladattavissa osoitteesta**

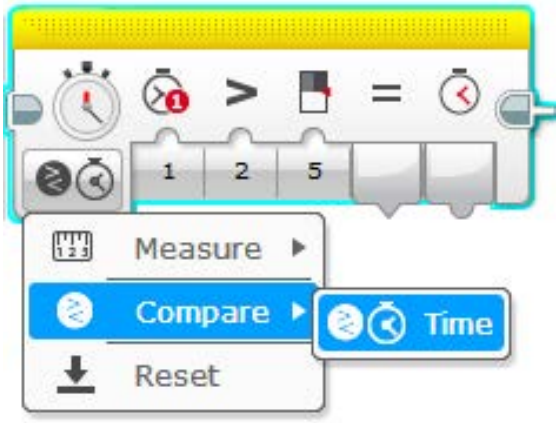
http://www.lego.com/fi-fi/mindstorms/downloads?ignorereferer=true

Ladatun tiedoston asennus tehdään valitsemalla työpöydän Tools -valikosta Block Import, ja seuraamalla ohjeita.

Anturi: NXT Temperatur Sensor, LEGO set 9749

Ajastin-toimilohko (Timer)

Ajastimille on ohjelmoinnissa monenlaista käyttöä. EV3:n ohjelmoijalla on käytettävänään 8 ajastinata.



Toiminnot

MIttaus, vertailu ja nollaus

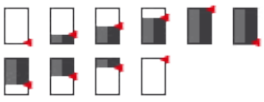
Kaikki ajastimet nollautuvat kun ohjelma käynnistetään. Toimilohkon nollaustoiminto nollaa valitun ajastimen. Nollattu ajastin käynnistyy välittömästi, aloittaen laskemisen taas uudelleen nollasta.

Parametrit

Timer ID, tarkasteltavan ajastimen tunnus

= ≠ > ≥ < ≤

Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.



Treshold Value, vertailuluku (0...100), (-100...100), (tms)

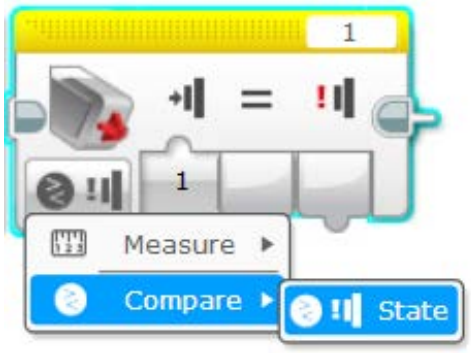
Elapsed Time, Ajastimen lukema, kun ohjelma jatkoi tästä eteenpäin (Lähtö).

=

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Kosketusanturi -toimilohko (Touch Sensor)

Kosketusanturi -toimilohko



Toiminnot

- **Mittaus,** eli **painetaanko painiketta juuri nyt.**
- **Vertailu** (Compare)**:** Ei paineta, Painetaan, Painallus (Bumped)

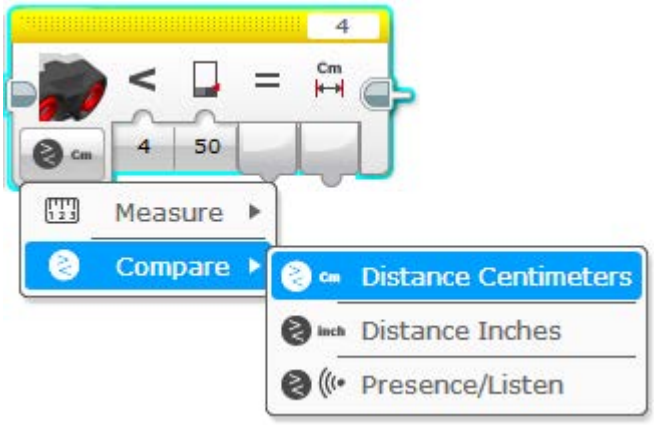
Huom. Jos laitat tämän toimilohkon silmukka-toimilohkon sisään, se kykenee tunnistamaan ja muistamaan myös suorituskertojen välillä tehdyn painalluksen (Bumped).

Parametrit

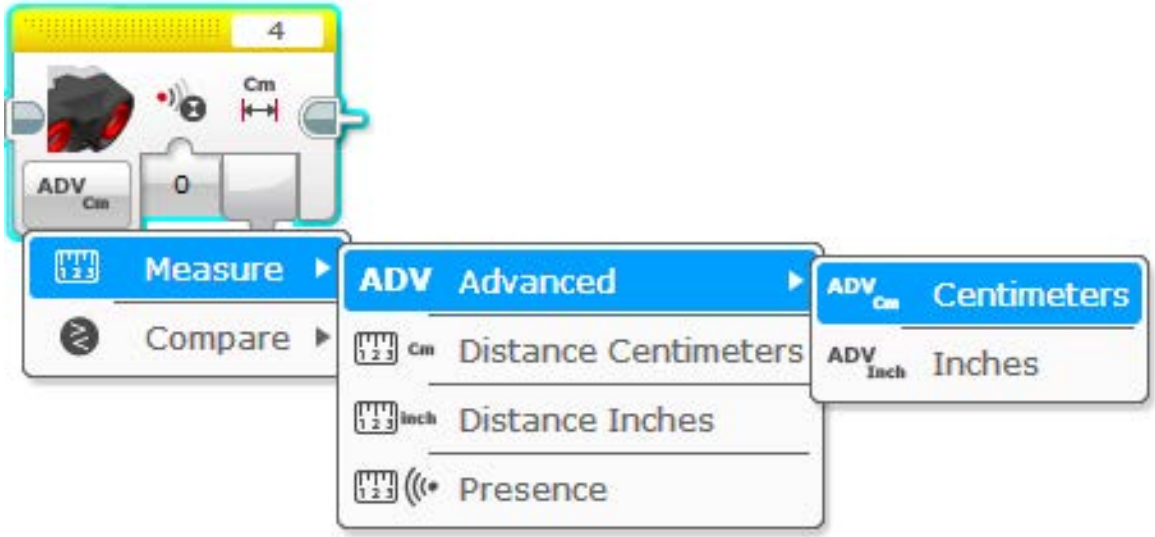
	State, tila, edellyttääkö ehdon täyttyminen, että painike on ylhäällä, painettu alas, tai painallusta (alas-ylös).
=	Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)
	Measured Value, kertoo oliko painike painettuna kun ohjelma jatkoi tästä eteenpäin (Lähtö).
	Anturin laitetunnus, ulkoinen tulotieto (Wired)

Ultraäänianturi -toimilohko (Ultrasonic Sensor)

Ultraäänianturin perustoiminnot ovat mittaus ja vertailu. Advanced toiminto poikkeaa tavallisesta mittaamisesta (Distance) siten, että siinä voidaan valita lähettääkö anturi yksittäisen vai jatkuvaa äänisignaalia.



Se voidaan ohjelmoida myös tarkkailemaan muita mahdollisia ultraäänen lähteitä. Tämä tehdään kuuntelutoiminnon avulla (Presence/Listen).



Ultraäänianturi-toimilohkon parametrit

Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.

Treshold Value, vertailuluku (0...100, tai 0...255)

Measuring Mode, Tapa mitata (Ping=yksi- ja Continuous=jatkuva signaali)

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Distance, Etäisyys sentteinä tai tuumina (1tuuma = 2.5cm)

Ultrasound Detected, Ultraäänen lähde havaittu, tosi/epätosi (Lähtö)

Anturin laitetunnus, ulkoinen tulotieto (Wired)

*** Tämä toimilohko ei sisälly ohjelman ilmaiseen kuluttajaversioon, mutta on ladattavissa osoitteesta**

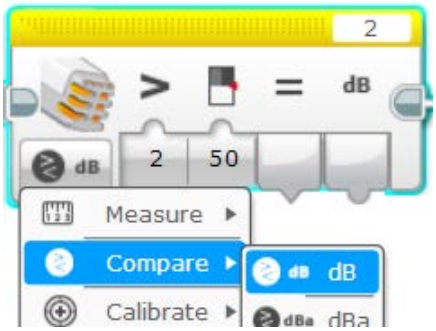
http://www.lego.com/fi-fi/mindstorms/downloads?ignorereferer=true

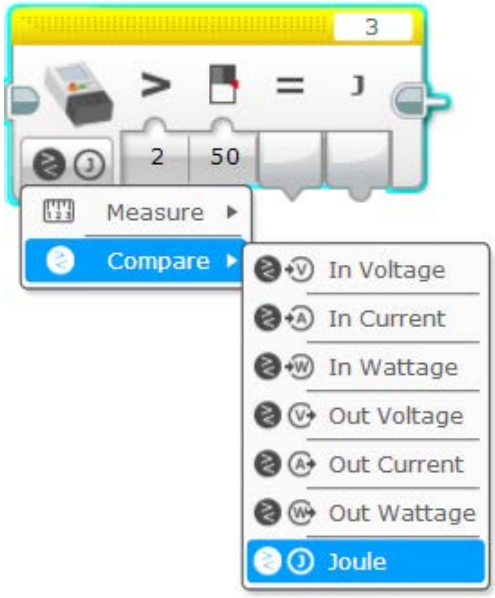
Ladatun tiedoston asennus tehdään valitsemalla työpöydän Tools -valikosta Block Import, ja seuraamalla ohjeita.

Anturi: EV3 Ultrasonic Sensor, LEGO set 45504

Sähkötekniikan mittaukset -toimilohko (Energy meter)

Energiamittari on osa uusiutuvan energian lisävarustesarjaa. Kytkemällä energiamittari EV3:n tuloporttiin, sen avulla voidaan tutustua energian tuotta-miseen, varastointiin ja kuluttamiseen.



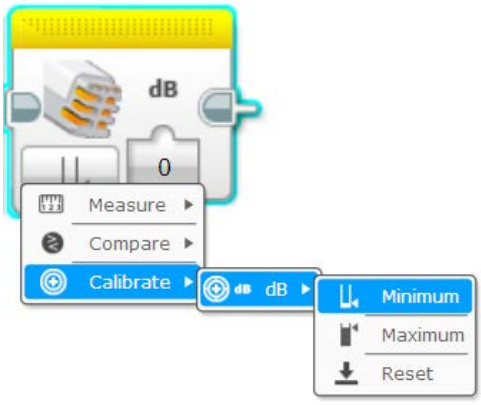


Toiminnot

- Mittaus ja Vertailu

Tutkittavat suureet

- Akun latausjännite [V], -virta [A] ja –teho [W]
- Akun purkujännite [V], -virta [A] ja –teho [W]
- Akkuun varastoidun energian määrä [J]



Sähkötekniikan mittaukset -toimilohkon parametrit

Compare, looginen vertailutoiminto jossa mitattua tietoa verrataan vertailulukuun, valitse yksi annetuista vaihtoehdoista.

Treshold Value, vertailuluku (0...100, tai 0...255)

Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Measured Value, kertoo kyseisen suureen mittaustiedon hetkeltä kun ohjelma jatkoi tästä eteenpäin (Lähtö).

Anturin laitetunnus, ulkoinen tulotieto (Wired)

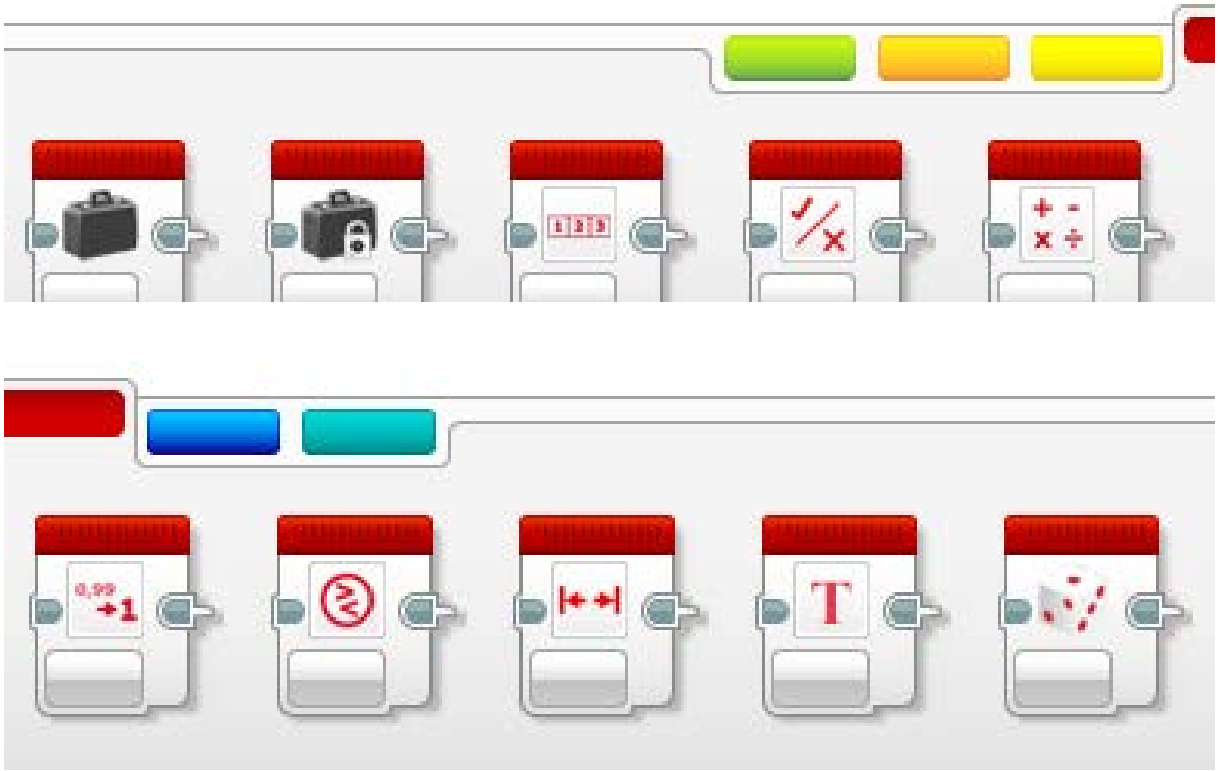
*** Tämä toimilohko ei sisälly ohjelman ilmaiseen kuluttajaversioon, mutta on ladattavissa osoitteesta**

http://www.lego.com/fi-fi/mindstorms/downloads?ignorereferer=true

Ladatun tiedoston asennus tehdään valitsemalla työpöydän Tools -valikosta Block Import, ja seuraamalla ohjeita.

Uusiutuvan energian lisävarustesarja: LEGO Education 9688

11. Tietojen käsittelyyn liittyvät toimilohkot

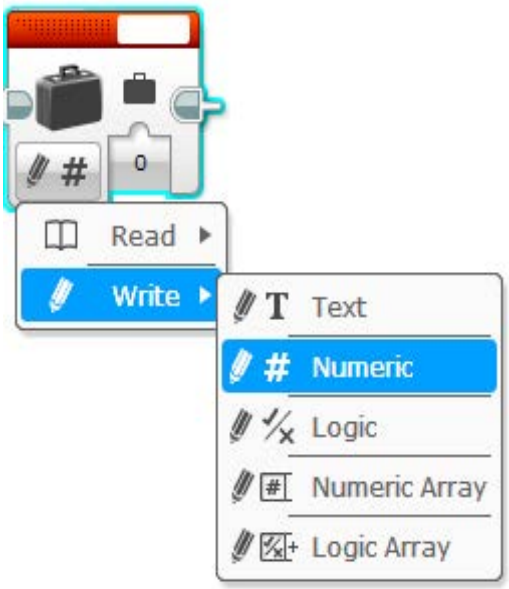


- **MUUTTUJA**
- **KIINTEÄ LÄHTÖTIETO ***
- **VEKTORIMUUTTUJA ***
- **LOOGISET VERTAILUT (BOOLEN ALGEBRA: AND, OR, XOR, NOT)**
- **MATEMAATTISET FUNKTIOT**
- **LUVUN PYÖRISTÄMINEN ***
- **VERTAILU**
- **SALLITTU ALUE ***
- **TEKSTIEN YHDISTÄMINEN ***
- **SATUNNAISLUKU**

TÄHDELLÄ MERKITYT TOIMILOHKOT ESITELLÄÄN KIRJASARJAN SEURAVASSA OSASSA *

Muuttuja, muistitoimilohko (Variable)

Muuttuja voi olla esimerkiksi mittauk-sen arvo, käyttäjän suorittama valinta, teksti, soitettavan äänitiedoston nimi, tai vaikka satunnaisluku.



Huom! Muuttujan arvo, eli muistipaikan sisältö voidaan myös päivittää kesken ohjelman suorituksen.

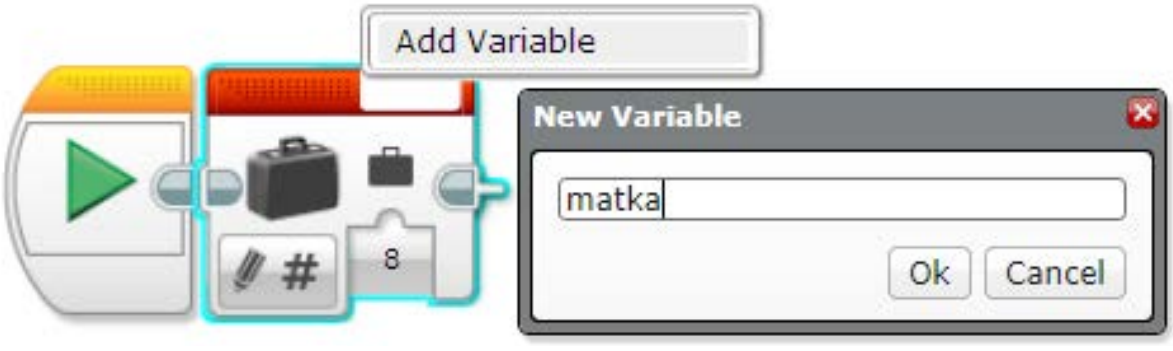
Muistipaikka on kuin matkalaukku, tai laatikko, jossa muuttujan arvoa kuljetetaan myöhempää käyttöä varten.

Toiminnot

Päätoimintoja on kaksi, muuttujan arvon taltioiminen muistiin eli kirjoittaminen (Write), sekä muuttujan arvon

1. **Huom!** Valitse muistipaikan tyyppi ennen sen nimeämistä!

2. Muuttujan nimi tulee ohjelmalohkon yläreunaan, valkoiseen laatikkoon. Selkeiden, tallennettavaa tietoa osuvasti kuvaavien nimien käyttäminen helpottaa ohjelmointia.



Kuvassa muuttujan nimeksi kirjoitetaan ”Matka” ja asetetaan sen arvoksi kahdeksan (8).

Esimerkin muuttujan (Matka) arvoa voidaan päivittää ja lukea useitakin kertoja ja eri tilanteissa ohjelman suorituksen aikana. Muistipaikka on tapa taltioida ja siirtää tietoa ohjelman sisällä.

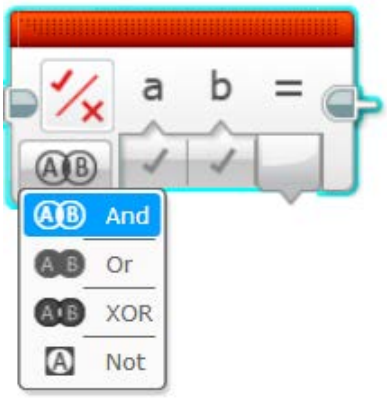
Muuttuja -toimilohkon parametrit

Value, Muistipaikkaan taltioidun muuttujan arvo (Tulo tai Lähtö)

Loogiset vertailut -toimilohko (AND, OR, XOR, NOT) (Logic Operations)

"Robotti on kulkenut 50cm **JA** valokenno on havainnut jotakin vihreää".

Loogisten ehtojen vertaileminen (Boolen algebra), on oleellinen osa ohjelmointia. EV3:n käskykannassa on tähän tarkoitukseen oma toimilohkonsa. Kokenut ohjelmoija voi tehdä loogisia vertailuja myös laskemalla.



Kuvake	Toiminto	Tulot	Lähdön arvo (Laskukaava)
	JA (AND)	A, B	"Tosi", jos A ja B ovat molemmat tosia. Muissa tapauksissa "epätosi". (tosi kun A+B=2)
	TAI (OR)	A, B	"Tosi", jos A tai B on tosi. Muussa tapauksessa "epätosi". (tosi kun A+B>0)
	JOKO TAI (XOR)	A, B	"Tosi", jos vain A tai B on tosi. Muissa tapauksissa "epätosi". (tosi kun A+B=1)
	EI (NOT)	A	"Tosi", jos A on epätosi, "epätosi" jos A on tosi. (tosi kun A=0)

Loogiset vertailut -toimilohkon parametrit

a — Looginen tulotieto A, tosi/epätosi

b — Looginen tulotieto B, tosi/epätosi

= — Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Matemaattiset funktiot -toimilohko (Math)

Laskenta-toimilohkon avulla voidaan suorittaa ueita erilaisia laskutehtäviä. Peruslaskutoimi-tukset löytyvät pikavalikosta.

ADV, Advanced toiminto tarjoaa koko joukon lisää toimintoja, mahdollisuuden kirjoittaa itse suoritettava laskukaava, sekä tuplasti muuttujia tässä kaavassa käytettäväksi (alla).

ADV LISÄFUNKTIOT

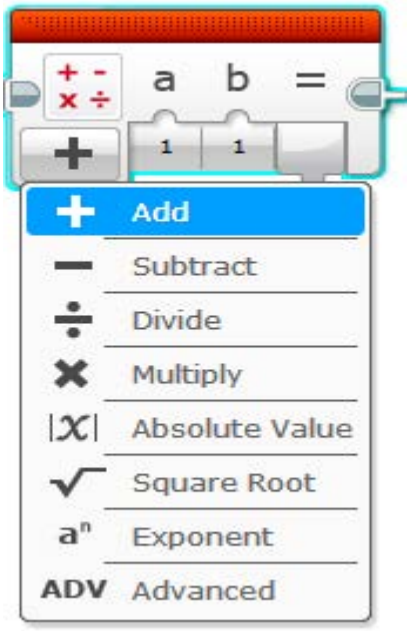
Invertoi, Moduloi, Pyöristä suurempaan, Pyöristä pienempään, Pyöristä lähimpään, Log, Ln, sin, cos, tan, Asin, Acos, Atan.

Kuvake	Toiminto	Tulot	Tulos
+	Yhteenlasku	a,b	a + b
−	Vähennyslasku	a,b	a - b
×	Kertolasku	a,b	a x b
÷	Jakolasku	a,b	a / b
|x|	Itseisarvo	a	a ilman etumerkkiä
√	Neliöjuri	a	Neliöjuuri a
a^n	Potenssilasku	a,n	a potenssiin n
ADV	Lisäfunktiot	a,b,c,d	Annetun kaavan mukaan

Vinkki: Koska looginen ”tosi” = 1 ja ”epätosi” = ”0”, voit ohjelmoida loogisen vertailun myös laskutoimituksen ja vertailun yhdistelmänä.

Vertailu -toimilohko (Compare)

Tätä toimilohkoa käytetään numeeristen tietojen keskinäiseen vertailuun. Tietotyyppien yhteensopivuus mahdollistaa myös loogisen tietotyypin tietojen kytkemisen tähän toimilohkoon (”Tosi=”1”, ”Epätosi”=”0”).



Toiminnot

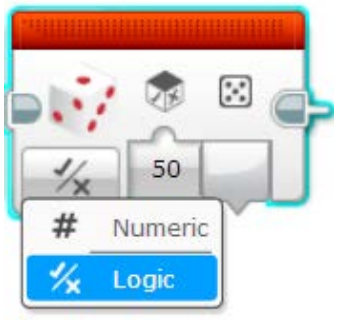
- Yhtäsuuri kuin
- Erisuuri kuin
- Suurempi kuin
- Suurempi tai yhtäsuuri kuin
- Pienempi kuin
- Pienempi tai yhtäsuuri kuin

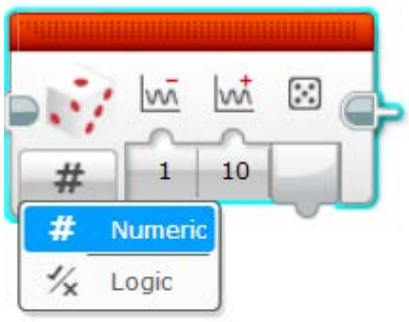
Vertailu -toimilohkon parametrit

a	Numeerinen tulotieto A
b	Numeerinen tulotieto B
=	Compare Result, täyttyykö annettu ehto, tosi/epätosi (Lähtö)

Satunnaisluku -toimilohko (Random)

Laitteiden ja pelien halutaan joskus lisätä satunnaisuutta tai yllätyksellisyyttä. Tällöin tarvitaan satunnaislukuja.

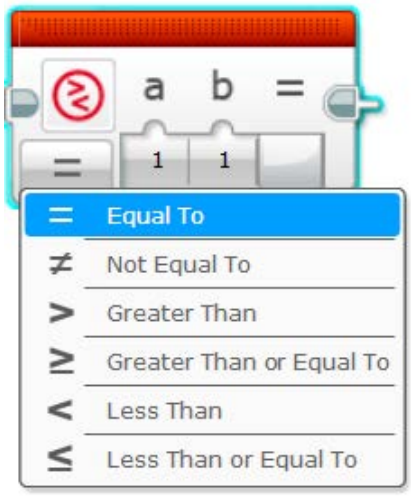




Toiminnot

Numeerinen, antaa lähtöön satunnaisluvun annetulta vaihteluväliltä.

Looginen, antaa lähtöön arvon ”Tosi” tai ”Epätosi”.



Parametrit

Alaraja (Lower Bound) tuotettavalle luvulle.

Yläraja (Upper Bound) tuotettavalle luvulle.

Arvon ”Tosi” todennäköisyys voidaan määritellä (0-100%).

Arvo (Value), tuotettu satunnainen numeerinen / looginen arvo.

Muut toimilohkot

Siniset ”Expertti”-toimilohkot, loput punaisista toimilohkoista, sekä vaaleansinisten omien toimilohkojen tekeminen esitellään tämän kirja-sarjan seuraavassa osassa.