

*AS-0.3200 Automaatio- ja systeemitekniikan projektityöt*

*Kevät 2009*

## S09-09 Development of AUTT-1 Robot

---

*Ajoradan seuranta-anturi, ultraäänisensori, SSH-palvelin ja uusi akkulaturi*



*Juho Cederström*

## Sisällysluettelo

Sisällysluettelo .....	1
Johdanto .....	2
Rata-anturi merkityn kulkureitin seuraamiseen.....	3
Rata-anturin periaate .....	3
Rata-anturin kehitys ja prototyypit .....	4
Prototyyppi 1: Yksi IR-ledi ja fotodiodi .....	4
Prototyyppi 2: Kolmikanavainen anturi telineessä, jossa kulmaa ja etäisyyttä voi säädellä.....	5
Rata-anturin toteutus .....	8
Ultraäänianturi törmäysten estämiseksi .....	9
Anturi.....	9
Anturin kytkentä.....	9
Anturin käyttäminen PC/104:n IO-kortilla.....	11
Ultraäänianturin mekaaninen asennus .....	12
Rata- ja ultraäänianturien kytkentä PC/104:ään.....	13
Ohjelmat .....	16
Yhteenveto projektin aikana toteutetuista testausohjelmista .....	16
Rata-anturin lukemishojelman pääkohdat.....	17
Rata-anturin nykyinen ajoalgoritmi.....	18
Anturin perusteella rataa seuraavan ohjelman pääkohdat.....	19
Ultraäänianturia käyttävän ohjelman pääkohdat .....	21
Muut parannukset.....	24
SSH-palvelimen asennus EMU-tietokoneen QNX-järjestelmään .....	24
Uuden akkulaturin käyttäminen.....	26
Jatkokehitys .....	27
Yhteenveto ja tiivistelmä .....	29
Liite 1: Työpäiväkirja.....	30
Liite 2: Tärkeitä oheisdokumentteja.....	31

## Johdanto

AUTT-1 on Automaation tietotekniikan harjoitustöissä käytettävä pienikokoinen robotti, jota hyödynnetään esimerkiksi kurssilla AS-116.2120 Automaation tietotekniset järjestelmät. Kyseisen kurssin harjoitustyössä mallinnetaan Rhapsody-työkalulla pienimuotoinen ohjelmisto, jota ajetaan robotin QNX-pohjaisessa PC/104-tietokoneessa.

Koska AUTT-1 on syntynyt useiden projektitöiden ja kesätyöntekijöiden työn tuloksena ja sen tekemiseen on käytetty paljon muusta käytöstä poistettuja laitteita, robotti ei ole kaikilta osa-alueiltaan yhtenäinen kokonaisuus. Lisäksi robotin voi ajatella olevan keskeneräinen, sillä se ei nykyisessä muodossaan kykene lainkaan autonomiseen toimintaan.

Syksyllä 2008 toteutettiin projektityö *Kehityssuunnitelma AUTT-1 –robotille*, jossa eräs kehitysidea oli rakentaa robottiin anturi, jolla se kykenisi seuraamaan lattiaan teipillä merkittyä kulku-uraa. Monet oikean maailman autonomisesti ohjatut ajoneuvot (automated guided vehicle, AGV), esimerkiksi Roclan automaattitrukit, navigoivat seuraamalla lattiaan upotettua johtoa tai värillistä teippiuraa. Näistä värillinen teipillä tehty rata on myös pienessä mittakaavassa yksinkertainen toteuttaa, ja sitä käytetäänkin monissa harrastelija- ja oppilastyöprojekteissa. (Esimerkkejä: <http://www.youtube.com/watch?v=V9CFqJYi6nw> ja <http://www.youtube.com/watch?v=dcWNLBBdrU>)

Kehityssuunnitelmassa esitettiin, että yksinkertaisuudestaan huolimatta tällainen kulkutapa voitaisiin yhdistää haarautuvaa reittiverkkoa kuvaavaan karttaan, jolloin AUTT-1 kykenisi navigoimaan esimerkiksi usean eri huoneen välillä käyttäen tällaista kustannustehokasta laitteistoa. Harjoitustyöalustana laitteisto olisi mielekäs, koska se vastaa esimerkiksi todellisen maailman autonomisesti liikkuvia ja tehtäviä suorittavia robottitrukkeja.

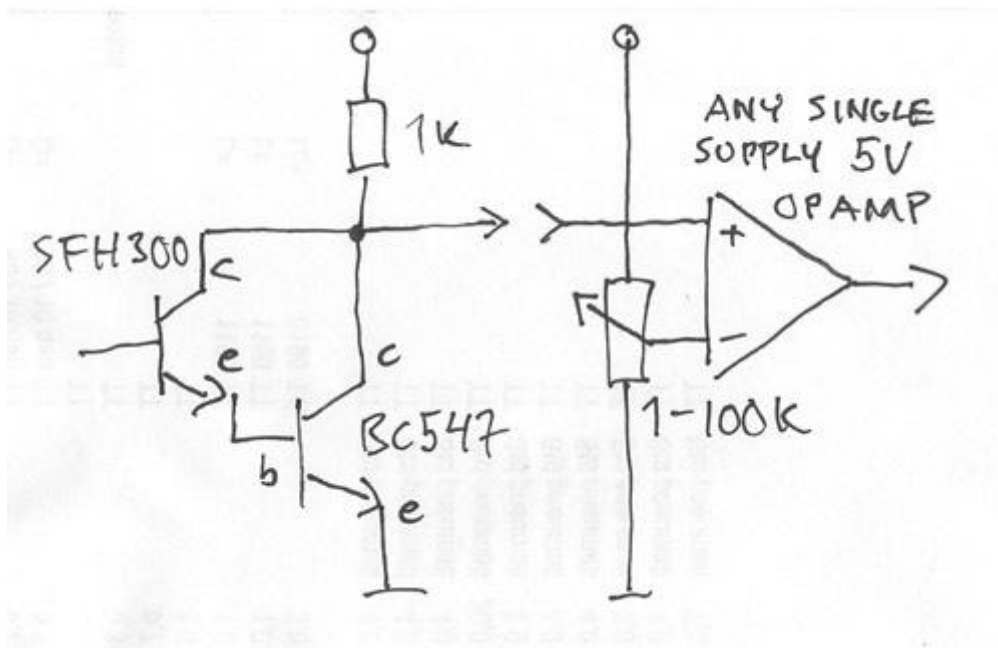
Tässä projektityössä toteutettiin anturi, jolla robotti kykenee tunnistamaan lattiaan merkityn teippiuran, liitettiin robottiin törmäysten estämiseen käytettävä ultraäänianturi, asennettiin robottiin SSH-palvelin robotin etäkäytön mahdollistamiseksi sekä hankittiin robottiin mm. uusi akkulaturi ja uudet hakkuriteholähteet.

## Rata-anturi merkityn kulkureitin seuraamiseen

### Rata-anturin periaate

Rata-anturi ajateltiin alussa toteuttaa siten, että anturin alla olevaa lattiaa valaistaan ledeillä ja lattiasta takaisin heijastuvaa valoa mitataan kulkureitin tunnistamiseksi. Mittauskomponenteiksi suunniteltiin aluksi joko LDR-valovastuksia, fototransistoreita tai fotodiodeja. Alkuperäisenä tavoitteena oli toteuttaa anturi näkyvän valon aallonpituuksilla, jotta vianetsintä ja toiminnan seuraaminen olisi mahdollisimman helppoa.

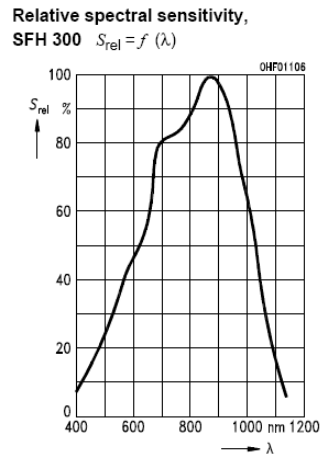
Internet-sivustojen keskusteluita ja kytkentäkaavioita sekä elektroniikkakauppojen sivustoja tutkimalla päädyttiin fotodiodipohjaiseen kytkentään. Slotforum.com –sivun keskustelualueelta löydettiin yksinkertainen fotodiodikytkentä, jossa käytettiin helposti saatavilla olevaa SFH300-fotodiodia. Kytkentäkaavio, jossa SFH300-fotodiodin signaalia vahvistetaan BC547-transistorilla ja signaali kynnystetään digitaaliseksi säätövastuksella ja operaatiovahvistimella, on esitetty alla.



Lähtöideana toiminut kytkentä. Lähde:

<http://www.slotforum.com/forums/index.php?showtopic=28367&hl=lordjw>

SFH300-fotodiodin suurin ongelma on sen toimintaspektri. Kyseinen fotodiode on herkimmillään 850 nanometrin aallonpituudella, eli näkymättömällä infrapunavalolla. Muidenkin elektroniikkakaupoista saatavien fotokomponenttien datalehtiä selaamalla havaittiin kuitenkin, että käytännössä kaikki helposti saatavilla olevat ja kohtuuhintaiset fotodiodit toimivat infrapuna-aallonpituudella, joten vaatimuksesta toimia näkyvällä valolla luovuttiin ja anturi päätettiin toteuttaa n. 850 nanometrin aallonpituuden infrapunavalolla.



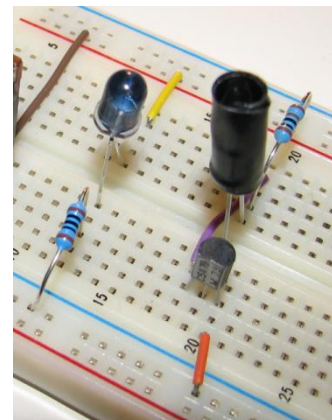
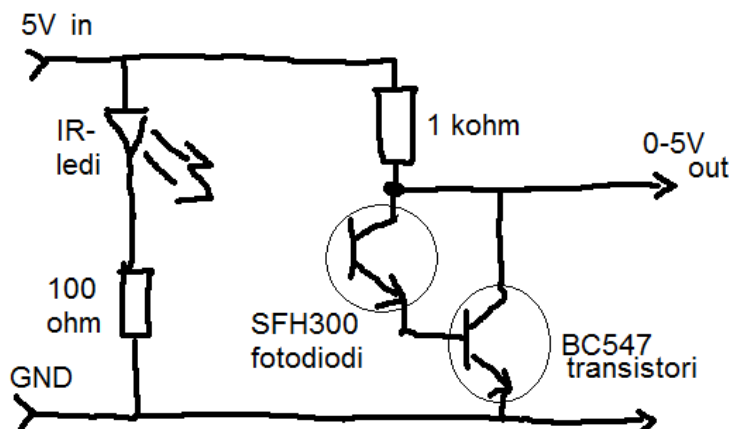
SFH300-fotodiodin toimintaspektri. Lähde: datalehti.

## Rata-anturin kehitys ja prototyypit

Koska fotodiodien käyttämisestä ei ollut entuudestaan kokemusta, projekti aloitettiin yksinkertaisten prototyyppien rakentamisella.

### Prototyyppi 1: Yksi IR-ledi ja fotodiodi

Ensimmäinen prototyyppi rakennettiin suoraan keskustelupalstalta löydetyn, edellä esitellyn kytkennän pohjalta siten, että koekytkentälevylle rakennettiin yksinkertainen IR-ledistä ja SFH300-fotodiodista rakentuva kytkentä. Näiden lisäksi tarvittiin 7805-jänniteregulaattori tasaisen 5V jännitteen muodostamiseksi, etuvastus IR-ledille sekä transistorikytkentä fotodiodin signaalia vahvistamaan. Ensimmäinen prototyyppi on esitetty alla olevissa kuvissa.



Ensimmäisen prototyypin kytkentäkaavio sekä kuva rakennetusta prototyypistä.

Jo alussa havaittiin, että SFH300 on varsin herkkä havaitsemaan valoa lähes joka suunnasta, myös alapuolelta. Tämän vaikutuksen pienentämiseksi fotodiodeja kääritin kuvassa esitetyllä tavalla valoa rajaavan teipin sisään. Lisäksi havaittiin, että SFH300 on herkkä myös esimerkiksi kattovaloista tulevalle hajavalolle, koska sen aallonpituus alkaa jo 420 nanometristä, mikä on näkyvän valon alueella.

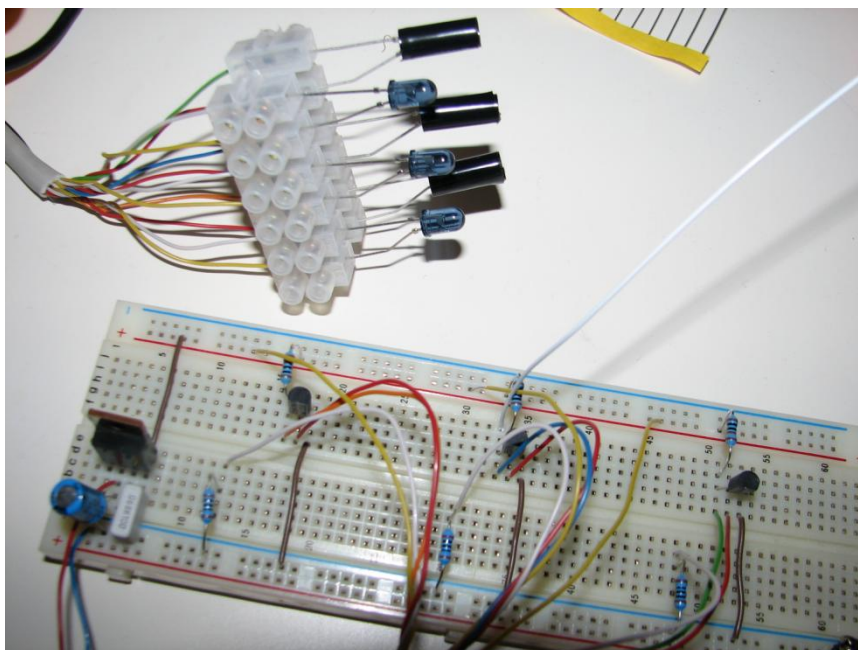
Ensimmäisellä prototyypillä suoritetuissa mittauksissa saatiin seuraavat tulokset:

Täysin valaistu fotodiodeja (saturaatio)	0,8 V
Valkoinen IR-LEDillä valaistu pahvi 5-10 cm etäisyydellä	0,8 V
Musta sähköteippi valkoisella pahvilla	3,0 V
Täysin pimennetty fotodiodeja	4,5 V

Mittaustuloksista havaittiin, että konsepti on toimiva ja kyseisellä kytkennällä on mahdollista erottaa ihmissilmälle mustalta näyttävällä teipillä merkitty kulku-ura. Prototyyppi osoitti kuitenkin, että kulku-uran ja täysin valkoisen pohjan välinen kontrasti ei ole kytkennällä kuin suuruusluokkaa 2,0 V, ja että ympäristöstä tulevan valon määrää ja sen heijastuksia lattiasta täytyy mahdollisesti kontrolloida esimerkiksi varjostimella tai anturin asennolla.

## Prototyyppi 2: Kolmikanavainen anturi telineessä, jossa kulmaa ja etäisyyttä voi säädellä

Anturin geometrian ja antureiden asettelun suunnittelemiseksi rakennettiin projektin seuraavassa vaiheessa sama kytkentä kolminkertaisena. Prototyyppiin tuli siten IR-LED:jä ja SFH300-fotodiodeja oheiskomponentteineen kolme kappaletta.



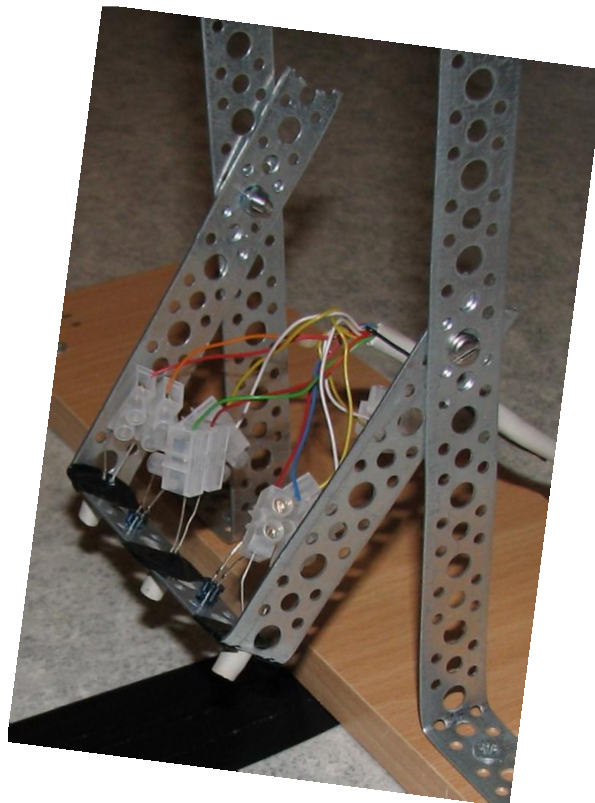
*Prototyyppi 2, kytkentä kolmella IR-LEDillä ja fotodiodilla oheiskomponentteineen.*

Tätä konstruktiota kokeiltiin robotin sijaintipaikan, TUAS-talon laboratoriohuoneen siniharmaata lattiaa vasten. Heti aluksi huomattiin, että TUAS-talolla on erittäin kirkkaat kattoloisteputket, jotka aiheuttavat erittäin voimakkaita heijastuksia lattialla olevista teipeistä. Tämän seurauksena mustat teippiviivatkin heijastavat takaisin niin paljon valoa, ettei niitä voi erottaa lattiasta. Ratkaisuksi keksittiin anturin asentaminen viistosti maahan nähden, jolloin kattovalojen heijastuksen vaikutus pienenee. Lisäksi koe osoitti, että anturin fotodiodien ja ledien symmetria on erittäin tärkeää. Prototyypillä havaittiin, että pienikin ero symmetriassa voi aiheuttaa jopa volttien jännite-eron mittaustulokseen.

TUAS-talon olosuhteissa, 2,5 cm etäisyydelle maasta asennetulla anturilla saatiin mittaustulokseksi 0,9 V paljaalla lattialla ja 4,5 V normaalin sähköteipin kohdalla. Tämä on erittäin hyvä kontrasti, mutta toisaalta havaittiin että anturi olisi hyvä saada nostettua vähintään 5 cm korkeudelle maasta jotta kynnyksen ylittäminen olisi edes teoriassa mahdollista.

Eräs tärkeimpiä prototyypillä todettuja asioita olikin se, että leveämmän teipin käyttäminen mahdollistaisi anturin nostamisen korkeammalle lattian pinnasta. Samalla tulisi kannattavammaksi sijoittaa fotodiodit kauemmaksi toisistaan, jotta näköalue kasvaisi ja tulisi mahdolliseksi ajaa robotilla kovempaa.

Lopuksi prototyyppiä jatkokehitettiin niin, että fotodiodit ja IR-ledit asennettiin säädettäväkulmaiseen telineeseen. Tämä mahdollisti erilaisten kulmien ja etäisyyksien helpon testaamisen anturin geometrian päättämiseksi. Anturit asennettiin telineeseen 4 cm välein, jotta 5 cm leveän sähköteipin tunnistaminen olisi vielä mahdollista. Samalla fotodiodien varjostinkaulukset vaihdettiin lyhyempiin, jotta niiden näkemä alue laajenisi. IR-ledit asennettiin fotodiodien väleihin puoleen väliin.



*Teline, johon fotodiodit ja IR-ledit asennettiin eri kulmien testaamista varten.*

Mittaustuloksiksi saatiin:

Maavara 4,5 cm, kulma 45 astetta: lattia 0,82V, 4 cm leveä teippi 3,4V

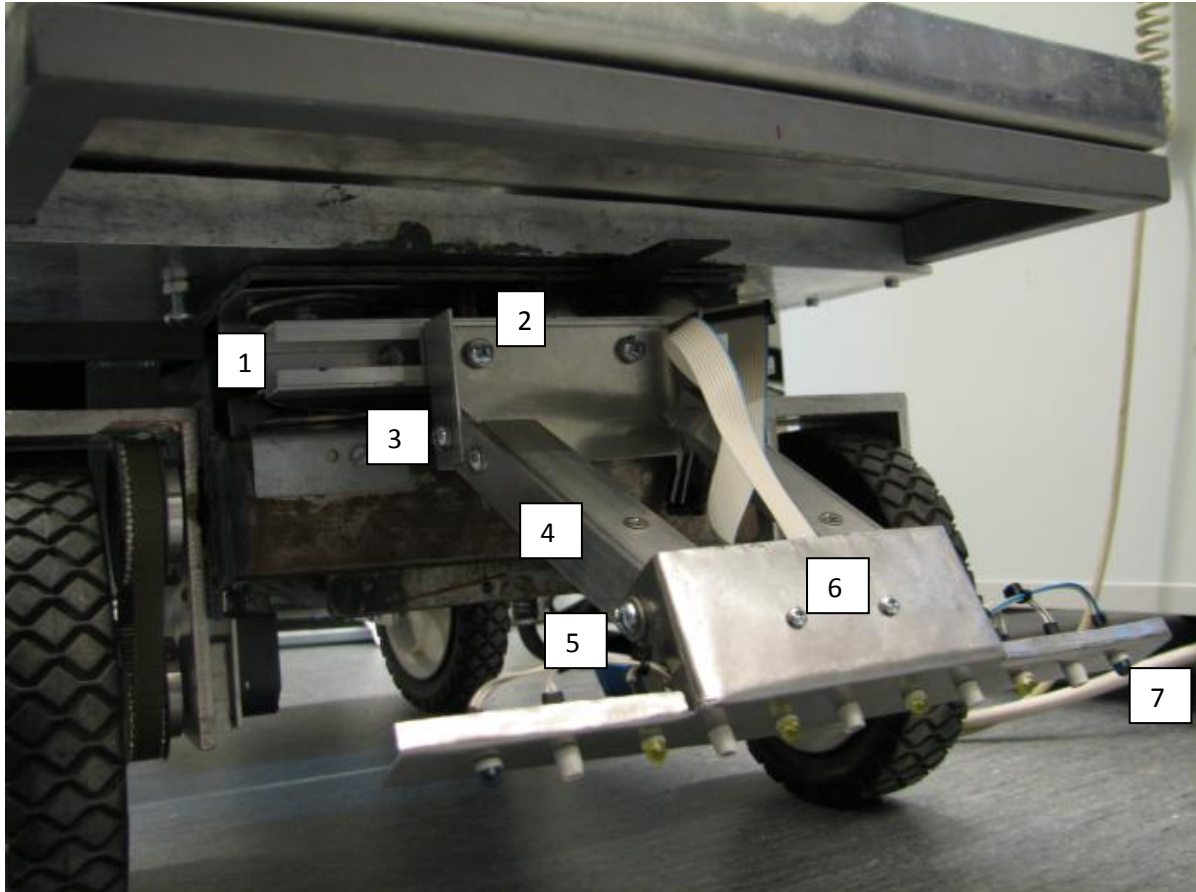
Maavara 4,0 cm, kulma 60 astetta: lattia 0,81V, 4 cm leveä teippi 3,8V

Maavara 6,0 cm, kulma 35 astetta: lattia 0,84V, 4 cm leveä teippi 3,0V

Mittaustuloksista havaittiin, että n. 45 asteen kulmalla on mahdollista nostaa anturi jopa n. 5 cm korkeuteen. Lisäksi kokeilut prototyypillä osoittivat, että 4 cm etäisyydellä toisistaan olevat fotodiodit soveltuvat hyvin 5 cm leveän teippiuran havaitsemiseen.

## Rata-anturin toteutus

Rata-anturin mekaniikka rakennettiin TUAS-talon prosessihallissa metallista. Mekaniikka toteutettiin siten, että anturi tulee n. 5 cm korkeudelle maasta, ja että sen kulma on säädettävissä parhaan toiminnan varmistamiseksi. Anturi asennettiin alla olevan kuvan mukaisesti etupyörien kiinnitykseen, jotta anturi kääntyy pyörien mukana ja on siten aina robotin edessä kulkusuuntaan päin.



*Rata-anturin kiinnitysmekaniikka ja anturiosa*

1. Projektityössä robottiin kiinnitetty kiinteä asennuskisko anturin sivuttaissäätöä varten.
2. Anturin kiinnitysruuvit (2 kpl), joita löysäämällä anturi vapautetaan. Tämän jälkeen anturin voi irrottaa liu'uttamalla sitä sivulle kiskoa pitkin.
3. Nivel, joka säättää anturin korkeuden lattiasta mitattuna.
4. Kiinnitysvarsi.
5. Nivel, joka säättää anturin kulmaa maahan nähden.
6. IR-ledien ja fotodiodien johtojen vedonpoiston kiinnitysruuvit.
7. IR-ledit ja fotodiodit. Ledit ja diodit on kiinnitetty pujottamalla niihin johdonkuorista tehdyt kaulukset. Fotodiodien kaulukset ovat pidemmät, jotta ne toimivat hajavalon varjostimina. Uloimmat IR-ledit ovat eriväriset kuin muut, koska ne ovat peräisin eri komponenttitilauksesta ja toimivat fotodiodien kannalta hieman huonommalla aallonpituudella (n. 950 nm).

# Ultraäänianturi törmäysten estämiseksi

## Anturi

Rata-anturin lisäksi robottiin haluttiin asentaa kehityssuunnitelman mukaisesti ultraäänianturi törmäysten estämiseksi. Teollisuustasoiset ultraäänianturit ovat varsin kalliita, mutta Elfalta löydettiin harrastelijarobottikäyttöön tarkoitettu Parallaxin valmistama PING-ultraäänianturi, jonka hinta oli n. 40 euroa (ilman ALV:a). Näitä antureita ostettiin yksi robotin etupuolelle asennettavaksi.



*Parallax PING-ultraäänianturi. Lähde: Parallaxin kotisivu.*

Valmistaja lupaa PING-anturin toimivan 5V käyttöjännitteellä, kuluttavan 20 mA virtaa ja tunnistavan kohteen 2 cm – 3 metrin etäisyydellä.

## Anturin kytkentä

Parallax PING -ultraäänianturissa on kolme kytkentäjohtinta: maa, 5V käyttöjännite sekä yksi signaalijohdin. Mittaus käynnistetään vetämällä signaalijohdin ylös n. 2-5 mikrosekunnin ajaksi. Tämän jälkeen seuraa n. 750 mikrosekunnin viive, jonka jälkeen ultraäänimoduuli kytkee signaalijohtimen ylös ajaksi, joka vastaa kohteen etäisyyttä. Anturin lähettämä paluusignaali on pituudeltaan 115 us - 18.5 ms.

Tällainen kytkentä on erittäin helppo käytettäessä mikrokontrolleria, jonka IO-kanavia on mahdollista käyttää sekä lähtöinä että korkeaimpedanssisina tuloina. Kyseinen nk. three-state -toiminta mahdollistaa sen, että kumpikin laite voi määrätä saman signaalin tilan vuorotellen toisen häiritsemättä. AUTT-1:n IO-kortilla ei kuitenkaan ole three-state -lähtöjä, joten anturin kanssa on käytettävä kahta eri signaalia, yhtä lähtöä ja yhtä tuloa, sekä melko yksinkertaista sovituselektronikkaa.

AUTT-1:n IO-kortilla on kahdeksan sisääntulon ja kahdeksan ulostulon lisäksi myös kaksi ajastinta/laskuria: yksi yleiskäyttöinen 16-bittinen laskuri (counter 0) ja yksi 32-bittinen laskuri (counterit 1+2 yhdistettynä), jota käytetään AD-muunnoksiin. Koska ultraäänianturin lähettämä signaali on kestoltaan melko lyhyt, sen mittaaminen tarkasti sisäänmenokanavan tilaa kyselemällä ohjelmasilmukasta toistuvasti on työlästä ja

mahdollisesti epätarkkaa. Tämän vuoksi pulssin pituuden mittaamiseen kannattaa käyttää kortin yleiskäyttöistä 16-bittistä laskuria.

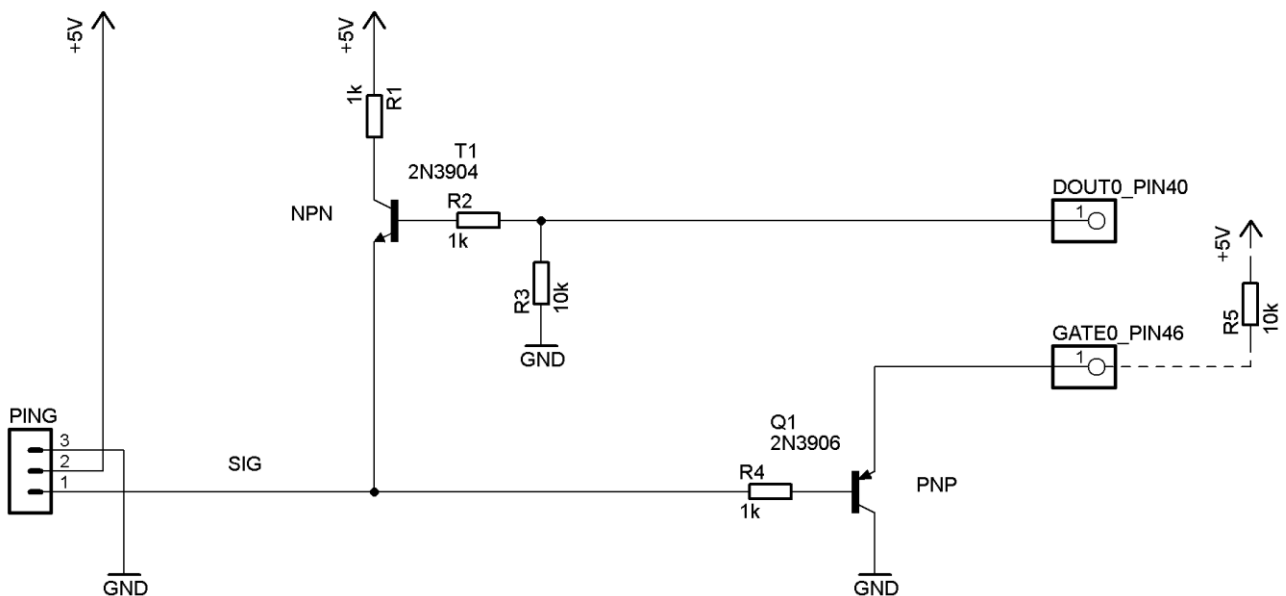
Counter 0:n laskuri toimii kahden signaalin ohjaamana: laskettavat pulssit pienentävät laskurin arvoa yhdellä, ja ns. gate-signaali estää pulssien laskemisen ollessaan arvossa 0. Yleensä laskureita käytetään joko siten, että ulkoisen signaalin reunat lasketaan, tai siten että laskuri laskee kellopulsseja ja gate-signaali määrää laskentajakson pituuden. Tässä sovelluksessa halutaan määrittää ultraäänianturin antaman pulssin pituus, joten AD-muunninkortin laskuri asetaan laskemaan kortin sisäisen 100 kHz kellon pulsseja niin kauan kuin ultraäänianturin ohjaama gate-signaali on arvossa 1.

Periaatteessa anturi kytketään siis siten, että AUTT-1:n AD-kortin digitaalilähtö Dout0:lla tehdään 5 us pituinen signaali joka käynnistää mittauksen, ja AD-kortin counter 0 laskee kuinka pitkään gate-signaali on tämän jälkeen päällä ultraäänianturin ohjaamana.

Kytkenässä on kuitenkin joitakin ongelmia. Ensinnäkin, mikäli Dout0 kytketään suoraan signaalijohtimeen, ultraäänianturi ei pysty ohjaamaan signaalijohdinta lainkaan koska digitaalilähtö Dout0 ohjaa sen joko arvoon 0 tai 1. Tämä ongelma voidaan korjata ohjaamalla Dout0:lla NPN-transistoria, joka vetää signaalin ylös tarvittaessa, muttei muulloin vaikuta signaaliin merkittävästi.

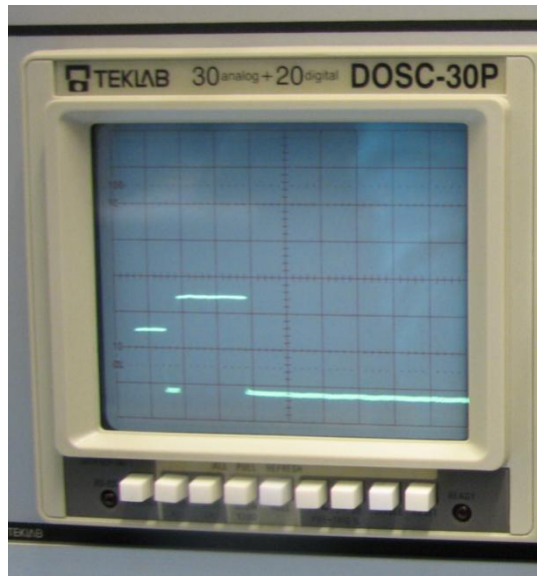
Toinen ongelma on AD-kortin gate-signaalin tulossa oleva 10 kilo-ohmin ylösvetovastus. Mikäli ultraäänianturin signaali kytketään suoraan gate-signaaliin, ylösvetovastus pitää signaalin jatkuvasti arvossa 1. Tämä ongelma voidaan korjata ohjaamalla ultraäänianturin signaalilla PNP-transistoria, joka pakottaa gate-signaalin maahan kun ultraäänianturin signaali on 0, ja antaa ylösvetovastuksen nostaa signaalin ylös kun ultraäänianturin signaali on 1.

Lopulliseksi kytkennäksi saatiin näistä syistä johtuen seuraava:



*PING-ultraäänianturin kytkentä PC/104:n IO-korttiin*

Alla olevasta kuvasta nähdään oskilloskoopin ruudulta, kuinka anturin kytkentä toimii. Alussa näkyvä lyhyt pulssi on transistorin T1 avulla tehty mittaauksen käynnistyspulssi, sitä seuraava pidempi pulssi on transistori Q1:n ja laskuripiirin avulla luettava mittauspulssi jonka pituus vastaa mitattua etäisyyttä. Pulssien erilaiset tasot johtuvat siitä, että kytkentä on mitoitettu kokemuksen perusteella ilman tarkkoja laskutoimituksia. Lopputuloksena saadut jännitetasot eivät ole täydelliset, mutta ne ovat selvästi sekä PING-anturin että IO-kortin tunnistamilla toiminta-alueilla.



*PING-ultraäänianturin signaalit luennan aikana*

## Anturin käyttäminen PC/104:n IO-kortilla

### IO-kortin rekisteri base address + 10, Counter/timer control register, ja sen bittiasetukset

bit 0 = counters 1 and 2 gate control, haluttu asetus: 0=vapaasti, ei gatea

bit 1 = counter 0 input source, haluttu asetus: 1=käytä sisäistä 100 kHz kelloa laskurin referenssitaajuutena ja IN0-signaalia (pin 29) gatena

bit 2 = external gate enabled, haluttu asetus: 0=ei gatea AD-muunnokselle

### Ultraäänianturin tarvitsemat IO-kortin signaalit

pin 29: IN0- counter 0 input, ei käytetä koska laskurin tulosignaalina käytetään 100 kHz sisäistä kelloa

pin 46: Din 2 / Gate 0, käytetään laskurin gatena. 10 kohm sisäinen ylösvetovastus.

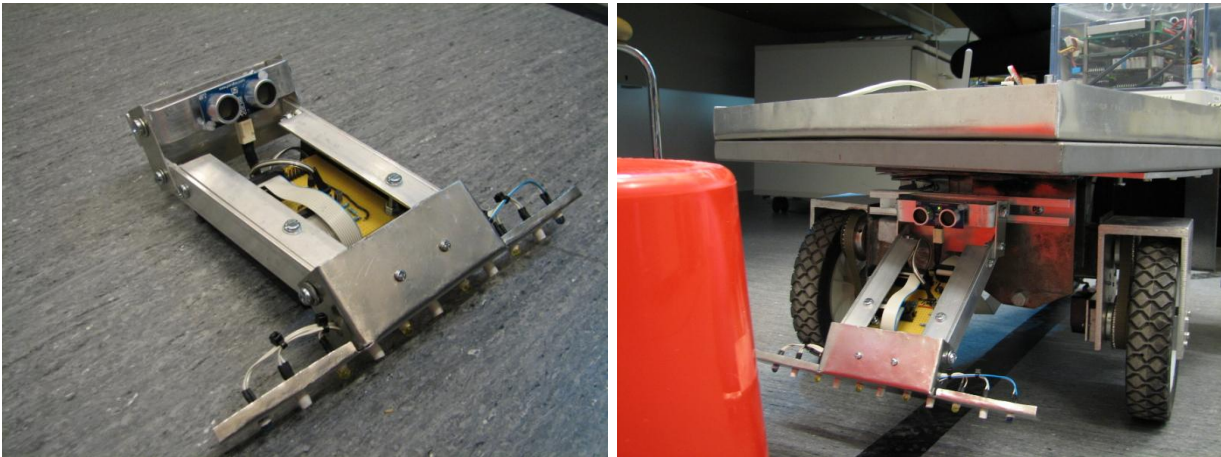
pin 40: Dout1, käytetään käynnistämään ultraäänimittaus

### Counter 0:n kirjoittaminen ja lukeminen ohjelmassa

AUTT-1:n IO-kortti perustuu kolmilaskuriseen 82C54-laskuripiiriin, jonka yksittäistä laskuria ohjataan kahdella 8-bittisellä rekisterillä: ohjaus- ja arvorekisterillä. Koska laskuri on 16-bittinen, ohjausrekisteriin täytyy kertoa että haluamme lukea molemmat arvot peräkkäin. Näin toimiva ohjelma on esitelty myöhemmin luvussa Ultraäänianturia käyttävän ohjelman pääkohdat.

## Ultraäänianturin mekaaninen asennus

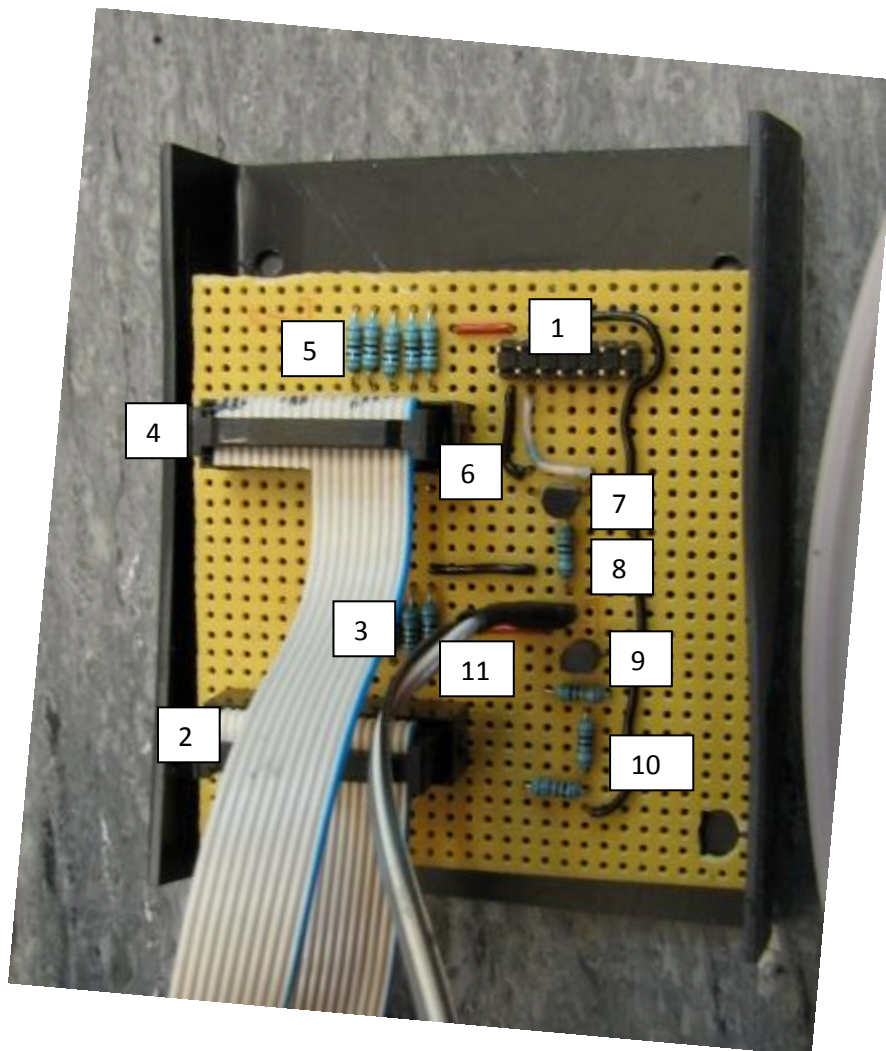
Koska ultraäänianturi haluttiin saada kääntymään renkaiden mukana, se asennettiin samaan kiinnitysmekaniikkaan rata-anturin kanssa. Samaan telineeseen rakennettiin toinen, pystysuunnassa kierrettävä tuki ultraäänianturia varten oheisen kuvan mukaisesti.



*Ultraäänianturin kiinnitys samaan kiinnitysmekaniikkaan rata-anturin kanssa*

## Rata- ja ultraäänianturien kytkentä PC/104:ään

Rata- ja ultraäänianturien luvuissa edellä esitetyt kytkennät rakennettiin juovakuparoidulle prototyyppilevyille. Komponenttivalinnat tehtiin suureksi osaksi sen mukaan, millaisia osia hyllystä sattui löytymään tilaamatta, minkä vuoksi esimerkiksi nauhakaapeliliittimet ovat pituudeltaan huonosti tarkoitukseen sopivia. Piirilevy asennettiin kuvassa näkyvän mustan muovisuojuksen kanssa kiinnitysmekaniikan varren alapuolelle.



*Rata- ja ultraäänianturin sovituslektroniikka*

1. Nauhakaapeliliitin kaapelille, jolla kytkentälevy kiinnitetään PC/104:ään. Ensimmäinen johdin on vasemmassa alakulmassa oleva GND.
2. Nauhakaapeliliitin IR-LED:ille (6 kpl). Joka toinen johdin on ledin anodi, joka toinen katodi.
3. IR-LED:ien etuvastukset (6 kpl), 100 ohm.
4. Nauhakaapeliliitin fotodiodeille (5 kpl).
5. Fotodiodikytkennän ylösvetovastukset (5 kpl), 1 kohm.
6. (piirilevyn alapuolella) Fotodiodikytkennän transistorit (5 kpl), BC547.
7. Ultraäänianturin luontakytkenän transistori Q1, 2N3906.

8. Transistorin Q1 etuvastus R4, 1 kohm.
9. Ultraäänianturin liipaisukytkennän transistori T1, 2N3904.
10. Transistorin T1 kytkennän vastukset, 2x 1 kohm ja 1x 10 kohm.
11. PING-ultraäänisensorin liitäntä. Johdossa musta on GND, valkoinen +5V, harmaa signaalijohdin. Piirilevyllä GND on kuvassa vasemmalla.

Numerolla 1 merkityn, PC/104:lle menevän nauhakaapelin liliittimen kytkentä on seuraavanlainen:

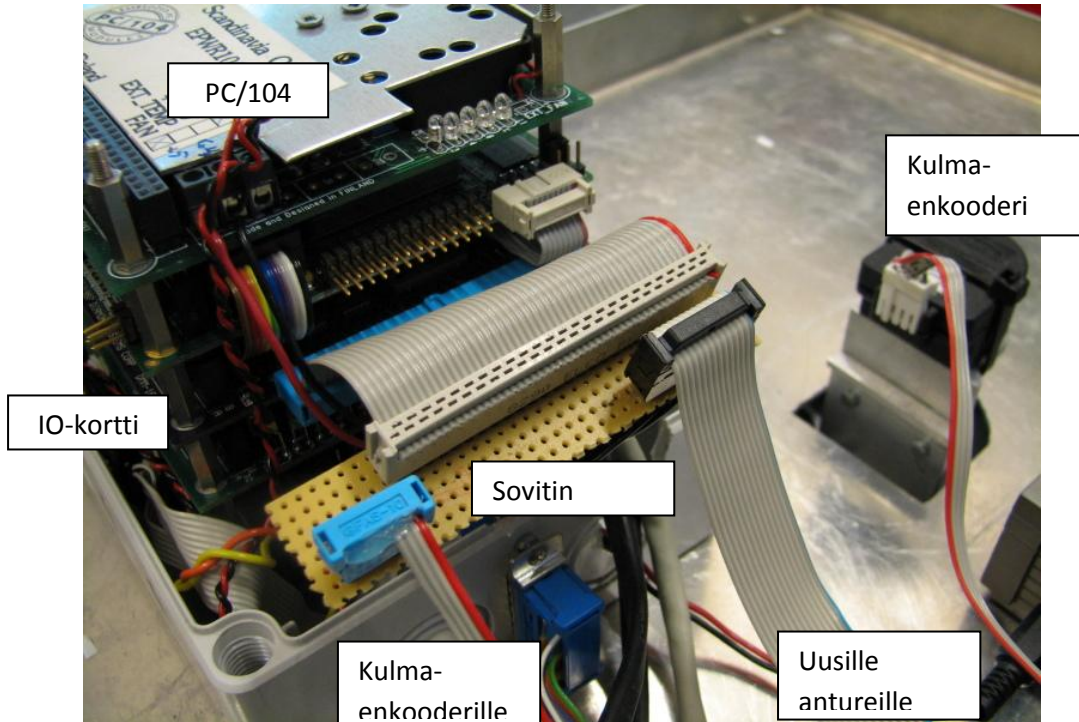
Kosketin	Signaali	Kohteneea oleva IO-kortin kosketin	IO-kortin signaali
1	GND	50	Dgnd
2	+5V	49	+5V
3	Ultraäänen mittaussignaali	46	Gate 0
4	Ultraäänen liipaisusignaali	40	Dout 0
5	Rata-anturi, kanava 1 (vasen)	14	Vin 1
6	Rata-anturi, kanava 2	12	Vin 2
7	Rata-anturi, kanava 3	10	Vin 3
8	Rata-anturi, kanava 4	8	Vin 4
9	Rata-anturi, kanava 5 (oikea)	6	Vin 5
10	(Ei kytketty)		
11	(Ei kytketty)		
12	(Ei kytketty)		
13	(Ei kytketty)		
14	(Ei kytketty)		

PC/104:n IO-kortissa on 50-napainen nauhakaapeliliitin, jossa on kaikki IO-kortin signaalit.

Vin 15 / 7-	1	2	Vin 7 / 7+
Vin 14 / 6-	3	4	Vin 6 / 6+
Vin 13 / 5-	5	6	Vin 5 / 5+
Vin 12 / 4-	7	8	Vin 4 / 4+
Vin 11 / 3-	9	10	Vin 3 / 3+
Vin 10 / 2-	11	12	Vin 2 / 2+
Vin 9 / 1-	13	14	Vin 1 / 1+
Vin 8 / 0-	15	16	Vin 0 / 0+
Agnd	17	18	Vref Out
Agnd	19	20	Vout 0
Agnd	21	22	Vout 1
Agnd	23	24	+15V
-15V	25	26	Vout 2
Agnd	27	28	Vout 3
In 0-	29	30	Dgnd
Out 0	31	32	Out 2
Dout 7	33	34	Dout 6
Dout 5	35	36	Dout 4
Dout 3	37	38	Dout 2
Dout 1	39	40	Dout 0
Din 7	41	42	Din 6
Din 5	43	44	Din 4
Din 3	45	46	Din 2 / Gate 0
Din 1	47	48	Din 0 / Gate 1/2
+5V	49	50	Dgnd

PC/104:n IO-kortin liitäntä. Lähde: IO-kortin manuaali.

Koska IO-kortille täytyy pystyä kytkemään yllä esitellyn antureiden piirilevyn signaalit, jo ennestään robottiin asennettu etupyörien kulmaenkooderi sekä mahdollisesti tulevaisuudessa muita laitteita, PC/104:n kotelon sisään rakennettiin alla olevan kuvan mukainen sovitinpiirilevy, joka kytketään toisaalta IO-korttiin 50-napaisella nauhakaapelilla ja johon toisaalta voidaan kytkeä edellä mainitut anturit kapeammilla nauhakaapeleilla.



*PC/104 ja kotelon sisään tehty sovitin*

Kulmaenkooderin liitännän signaalit on esitetty alla:

Kosketin	Signaali	Kohteneva oleva IO-kortin kosketin	IO-kortin signaali
1	+5V	49	+5V
2	GND	50	Dgnd
3	Kulman mittaussignaali	16	Vin 0
4	(Ei kytketty)		

## Ohjelmat

### Yhteenveto projektin aikana toteutetuista testausohjelmista

Projektin aikana toteutettiin lukuisia pieniä C-kielisiä ohjelmia, jotka testaavat kehitetyn laitteiston toimintaa. Yksinkertaisimmillaan ohjelmat vain lukevat IO-kortin kanavia ja tulostavat arvot näytölle, monimutkaisimmillaan ohjelmat lukevat rata-anturia ja ohjaavat sen ohjaamana pitkin rataa.

Ohjelmat on tallennettu AUTT-1:n ohjelmistokehitystietokoneelle (antero) käyttäjän autt-1 kotihakemistossa olevaan jhceders-hakemistoon. Tähän hakemistoon pääsee myös robotin PC/104-tietokoneella (emu), joko /net -hakemiston tai pääkäyttäjän kotihakemistossa olevan jhcederslinkki -nimisen symbolisen linkin kautta.

Kaikenkaikkiaan jhceders-hakemistossa on seuraavat ohjelmat:

analog_io_test.c	Yksinkertainen itsenäinen ohjelma, joka lukee IO-kortin analogiakanavia ja näyttää lukemat näytöllä.
angle.c	Ennen tätä projektityötä toteutettu kulma-anturia lukeva kirjasto. Sisältää kulma-anturin skaalauksen, mutta toisaalta luennasta puuttuu joitakin tärkeitä viiveitä joten tämä kirjasto ei välttämättä toimi oikein.
kbhit.c	Kirjasto, jonka kbhit-funktiolla on mahdollista saada selville onko jotakin näppäimistön näppäintä painettu. Lähde: <a href="http://gcc.gnu.org/ml/gcc-help/2006-03/msg00101.html">http://gcc.gnu.org/ml/gcc-help/2006-03/msg00101.html</a>
motor.cpp	Ennen tätä projektityötä toteutettu moottorinohjauskirjasto.
ping.cpp	Kirjasto, jota voi käyttää ultraäänianturin raakadatan lukemiseen.
ping_test.c	Yksinkertainen ohjelma, joka näyttää näytöllä ultraäänianturin lukemat.
track.cpp	Kirjasto, jota voi käyttää rata-anturin raakadatan lukemiseen.
track_ajo.cpp	Ohjelma, joka ajaa lattiaan merkittyä rataa pitkin.
track_calib.cpp	Ohjelma, joka näyttää näytöllä pienimmän ja suurimman suorituksen aikana mitatun anturiarvon kullekin rata-anturin kanavalle, eli jota voidaan käyttää kalibrointiin.
track_sensor_test.c	Ohjelma, joka näyttää näytöllä lasketun radan sijainnin, muttei ohjaa moottoreita.

## Rata-anturin lukemiohjelman pääkohdat

Alla on esitetty rata-anturin lukemiseen tarkoitettun track.cpp –ohjelman koodi. Ohjelmassa on kaksi funktiota: `initTrackSensor` joka alustaa anturin ohjelman suorituksen alussa ja `readTrackSensor` joka lukee antureiden tiedot skaalaamattomana `float[]`-muotoiseen taulukkuun.

```
#include "track.h"

void initTrackSensor(void)
{
    // Request permission to use in8 & out8
    ThreadCtl(_NTO_TCTL_IO, 0);

    // Set channels to read (MSB=last, LSB=first)
    out8(BASE_ADDRESS + CHANNEL_REGISTER,
          (TRACK_SENSOR_LAST_CHANNEL << 4) |
          TRACK_SENSOR_FIRST_CHANNEL);

    // Set AD converter range 0..5V
    // 0x08->RANGE = 10V, 0x04->ADBU=unipolar, 0x01->GAIN=2 ==> 0..5V
    out8(BASE_ADDRESS + ANALOG_RANGE, (0x08 | 0x04 | 0x01));
    usleep(10); // >10 microseconds
}

void readTrackSensor(float *values)
{
    unsigned char data_lsb;
    unsigned char data_msb;
    short int data;
    int i;

    // Set the channel numbers. The high byte sets the last channel
    // number, the low byte sets the first.
    // The channel register is incremented automatically after each
    // conversion.
    out8(BASE_ADDRESS + CHANNEL_REGISTER,
          (TRACK_SENSOR_LAST_CHANNEL << 4) |
          TRACK_SENSOR_FIRST_CHANNEL);
    usleep(10); // >10 microseconds

    for(i=0; i<TRACK_SENSOR_CHANNEL_COUNT; i++)
    {
        // Start conversion
        out8(BASE_ADDRESS + AD_LSB, 0x00);
        // Writing (any value) starts the ad-conversion

        // Wait for conversion to finish
        while(in8(BASE_ADDRESS + STATUS_REGISTER) & 0x80);

        // Read data
        data_lsb = in8(BASE_ADDRESS + AD_LSB);
        data_msb = in8(BASE_ADDRESS + AD_MSB);
        data = (data_msb << 8) | data_lsb;

        // Set the value to array
        values[i] = (float) (data+32768) / 65535.0;
    }
}
```

## Rata-anturin nykyinen ajoalgoritmi

Robotille kehitettiin projektityön aikana yksinkertainen radanseurantaohjelma, jonka algoritmi toimii verrattain vaatimattomalla ad-hoc –tyyppisellä säätöalgoritilla. Algoritmi on toteutettu track\_ajo.cpp – tiedostossa.

1. Analogiakanavien arvot mitataan ja tallennetaan taulukkoon siten, että 0V vastaa arvoa 0.0 ja 5V arvoa 1.0. Tämä suoritetaan readTrackSensor()-funktioilla.
2. Koska anturin elektroniikka antaa mittaustuloksia n. väliltä 0.8V-4.5V, mittaustulokset skaalataan isommiksi. Koska mekaniikan ja elektroniikan toleransseista johtuen eri kanavat antavat aavistuksen verran eri lukemia samanlaisissa olosuhteissa, kanavat skaalataan yksitellen.

Skaalausta varten kullekin kanavalle määritellään minimi- ja maksimiarvot väliltä 0.0 ja 1.0. Tämän voi suorittaa esimerkiksi ajamalla track\_calib –ohjelman ja kääntämällä anturin kokonaan ajoradan toiselta puolelta kokonaan toiselle puolelle. Track\_calib näyttää tällöin näytöllä suurimman ja pienimmän käännön aikana mitatun lukeman. Minimi- ja maksimiarvot tallennetaan track\_ajo.cpp-tiedostossa oleviin low\_calibration- ja high\_calibration–taulukoihin.

Kanavien arvot skaalataan lineaarisesti väliltä 0.0-1.0 välille min-max kaavalla

$$\text{skaalattu arvo} = \frac{\text{mitattu arvo} - \text{minimiarvo}}{\text{maksimiarvo} - \text{minimiarvo}}$$

3. Tässä vaiheessa kunkin kanavan skaalattu arvo välillä 0.0-1.0 ilmaisee kuinka tummaa kanavan fotodiodin alla on, eli kuinka todennäköisesti fotodiodin alla on kulku-ura.

Kun näillä todennäköisyyksillä kerrotaan kanaville taulukkoon sensor\_positions määriteltyjä koordinaatteja -10000, -5000, 0, 5000, 10000 ja tulokset summataan, saadaan koordinaatti välillä -10000 – 10000 jossa kulku-ura todennäköisesti sijaitsee.

4. Kaikista kanavista valitaan suurin skaalattu arvo, ja tämän arvon perusteella päätellään onko rata anturin alla vai onko rata kadotettu. Ohjelma käyttää kokeellisesti määritettyä kynnyсарvoa 0.3: mikäli kaikkien kanavien skaalattu arvo on tätä pienempi, robotti pysäytetään varmuuden vuoksi.
5. Mikäli kynnyсарvo ylittyy, lasketaan moottoreille ajonopeudet.

Jos kulku-uran sijainti on välillä -800 – 800, annetaan moottoreille yhtä suuret ajonopeudet eli ajetaan suoraan.

Jos kulku-uran sijainti on negatiivinen, annetaan oikeanpuoleiselle moottorille täysi ajonopeus ja vasemmanpuoleiselle sitä pienempi mitä kauempana origosta kulku-ura on, ei kuitenkaan koskaan negatiivista.

Ajonopeus annetaan samalla logiikalla mutta käänteisesti mikäli kulku-uran sijainti on positiivinen.

Algoritmi on kaukana täydellisestä, mutta se toimii vakio-olosuhteissa suljetulla radalla erittäin hyvin.

Algoritmin suurimmat ongelmat ovat:

- Ohjelmaan kovakoodatut kalibroitirajat, jotka riippuvat valaistusolosuhteista, lattiamateriaaleista ja robotin jousituksen asennosta. Kiinteät kalibroitirajat voisi korvata siten, että ennen ajon alkua robotti mittaisi alkukalibraatioarvot kääntämällä anturia ensin 90 astetta vasemmalle ja sitten 90 astetta oikealle. Lisäksi kalibraatioarvot voisivat mukautua ympäristöön ja muuttuviin valaistus- ja jousitusolosuhteisiin ajon aikana. Tämän toteuttamisessa voisi ehkä hyödyntää myös esimerkiksi Kalman-suodinta.
- Tapa, jolla ohjelma tunnistaa suurimman skaalatun arvon pienuudesta radan kadonneen on epävarma, eikä robotti aina pysähdy radan kadotessa.
- Tässä esitetty algoritmi ei ole yleensäkään toiminnaltaan erityisen tieteellinen, vaan sen voisi korvata luultavasti esimerkiksi sumealla logiikalla tai neuroverkolla.
- Nykyinen algoritmi ei tarjoa suoraan mahdollisuutta kehittyneemmille navigointitavoille, kuten risteysajolle.

Seuraavassa on esitetty pääkohdittain ohjelma, joka toteuttaa tässä kuvatun yksinkertaisen algoritmin.

## Anturin perusteella rataa seuraavan ohjelman pääkohdat

track\_ajo.cpp

```
// ...
#include "motor.h"
#include "track.h"

// ...

int main()
{
    // ...

    // Hard-coded calibration data determined by the track_calib software
    float low_calibration[5] = {0.223, 0.181, 0.178, 0.190, 0.220};
    float high_calibration[5] = {0.714, 0.548, 0.623, 0.661, 0.707};

    // Weights for sensor channel positions
    float sensor_positions[5] = { -10000, -5000, 0, 5000, 10000 };

    float values[TRACK_SENSOR_CHANNEL_COUNT];
    float actual[TRACK_SENSOR_CHANNEL_COUNT];
```

```

float position = 0.0;
float maxvalue = 0.0;

// ...
initTrackSensor();
// ...

while(!kbhit()) {
    // ...

    // Read raw values for each channel
    readTrackSensor(values);
    position=0.0;
    maxvalue = 0.0;

    // Scale all channels and determine the maximum value and
    // the position
    for(i=0; i<TRACK_SENSOR_CHANNEL_COUNT; i++)
    {
        actual[i] = (values[i]-low_calibration[i]) /
                    (high_calibration[i]-low_calibration[i]);
        if(actual[i] > maxvalue) maxvalue = actual[i];
        position += actual[i] * sensor_positions[i];
    }

    // ...

    if(maxvalue < 0.3)
    {
        // Track lost, stop motors.
        // ...
    }
    else if (abs((int)position)<800)
    {
        // Go straight on
        velocity0 = 3000;
        velocity1 = 3000;
    }
    else if (position<0)
    {
        // Turn left
        velocity0 = 1200-abs((int)position/5);
        if(velocity0<0) velocity0=0;
        velocity1 = 3000;
    }
    else
    {
        // Turn right
        velocity0 = 3000;
        velocity1 = 1200-abs((int)position/5);
        if(velocity1<0) velocity1=0;
    }

    if (velocity0>0) motor0.moveVelocity(velocity0*2, FORWARD);
    if (velocity1>0) motor1.moveVelocity(velocity1*2, FORWARD);

    usleep(50000);

}
// ...
}

```

Ohjelma on tallennettu AUTT-1:n ohjelmistokehitystietokoneen (antero) käyttäjän autt-1 kotihakemiston alle hakemistoon jhceders, ja se voidaan kääntää komennolla

```
g++ -o track_ajo track_ajo.cpp track.cpp kbhit.c motor.cpp ping.cpp
```

Käännöksen tuloksena syntyy ohjelma track\_ajo, joka voidaan käynnistää robotissa ottamalla siihen ssh-yhteys ja siirtymällä ohjelmistokehitystietokoneen jhceders-hakemistoon joko pääkäyttäjän kotihakemistossa olevan jhcederslinkki -nimisen symbolisen linkin tai /net -hakemiston kautta sekä antamalla komento `./track_ajo`.

## Ultraäänianturia käyttävän ohjelman pääkohdat

Pääkohdat ohjelmasta ping\_test.c.

```
#include <hw/inout.h>
#include <sys/neutrino.h>
#include <stdio.h>

#define BASE_ADDRESS          0x300
#define AD_LSB                0x00
#define AD_MSB                0x01
#define CHANNEL_REGISTER     0x02
#define DIGITAL_IO_REGISTER  0x03
#define STATUS_REGISTER      0x08
#define COUNTER_SELECT       0x0A
#define ANALOG_RANGE         0x0B
#define COUNTER0_DATA        0x0C
#define COUNTER_CONTROL      0x0F

int main(void)
{
    unsigned short int d1, d2, d3;
    unsigned short int value;
    int i;

    // Request permission to use in8 & out8
    ThreadCtl(_NTO_TCTL_IO, 0);

    // Counter 0 input source = 1 (use on-board 100kHz reference
    // frequency as counter input)
    out8(BASE_ADDRESS + COUNTER_SELECT, 0x02);

    while(1)
    {
        // 1.
        // Generate a 5 us pulse to digital out 1 to trigger the
        // ultrasonic measurement
        out8(BASE_ADDRESS + DIGITAL_IO_REGISTER, 1);
        usleep(5);
        out8(BASE_ADDRESS + DIGITAL_IO_REGISTER, 0);
    }
}
```

```

// 2.
// Configure the counter for writing two counter bytes.
//
// For info on bit meanings, see the 82C54 data sheet.
// SC1,SC0=0 -> Select counter 0.
// RW1,RW0=1 -> Write LSB first, then MSB.
// M2,M1,M0 = 0 -> Mode 0.
// BCD=0 -> Binary counter.
out8(BASE_ADDRESS + COUNTER_CONTROL, 0x30);

// 3.
// Initialize the counter with the maximum value, 0xFFFF
out8(BASE_ADDRESS + COUNTER0_DATA, 0xFF);
out8(BASE_ADDRESS + COUNTER0_DATA, 0xFF);

// 4.
// Wait for the measurement to finish
//
// (300 ms is more than the longest possible measurement.)
usleep(300000);

// 5.
// Configure the counter for reading two bytes and read
// the value (LSB first, then MSB).
out8(BASE_ADDRESS + COUNTER_CONTROL, 0x30);
d1 = in8(BASE_ADDRESS + COUNTER0_DATA);
d2 = in8(BASE_ADDRESS + COUNTER0_DATA);

// 6.
// Combine LSB and MSB. Subtract from 65535 because the
// counter counts downwards.
value = 65535 - ((d2<<8) | d1);
printf("%d ",value);

// 7.
// Print a graphical bar of # characters.
for(i=0; i<value/25; i++) printf("#");
printf("\n");

}

return 0;

}

```

Ohjelma suorittaa jatkuvasti mittauksia ja tulostaa näytölle mittauslukeman (anturin vastauspulssin pituus 100 kHz kellopulsseissa mitattuna) sekä mittauslukemaa vastaavan #-merkeillä piirretyn palkin. Ohjelma ei toistaiseksi skaalaa mittauslukemaa esimerkiksi senttimetreiksi, vaikka skaalauksen toteuttamisen ei pitäisi olla kovinkaan vaikea toimenpide.

Ohjelma on tallennettu AUTT-1:n ohjelmistokehitystietokoneelle (antero) autt-1 –käyttäjän kotihakemistossa olevaan jhceders-hakemistoon. Ohjelma käännetään komennolla

```
gcc -o ping_test ping_test.c
```

Ohjelman voi suorittaa ottamalla robottiin ssh-yhteyden, menemällä pääkäyttäjän kotihakemistossa olevan jhceders-linkin tai hakemiston /net kautta antero-tietokoneen jhceders-hakemistoon ja komentamalla `./ping_test`.

Kun ohjelma suoritetaan, se tulostaa näytölle ultraäänianturin mittapulssien pituudet. Yksikkönä on 100 kHz pulssien lukumäärä, eli suurempi luku tarkoittaa pidempää pulssia ja siten pidempää etäisyyttä. Ohjelma havainnollistaa lukuja myös näyttämällä luvun vieressä sen pituutta kuvaavan #-merkeistä koostuvan vaakapylvään.

```
913 #####  
981 #####  
920 #####  
112 ##  
121 ##  
138 ##  
124 ##  
890 #####  
926 #####  
957 #####
```

← tässä robotin eteen on tullut este

## Muut parannukset

### SSH-palvelimen asennus EMU-tietokoneen QNX-järjestelmään

Projektityön puitteissa haluttiin asentaa AUTT-1 –robottiin SSH-palvelin, jotta robotti ei olisi riippuvainen näppäimistön ja kuvaputkinäytön läheisyydestä, vaan voisi toimia kauempana lattialla etähallinnan alaisena.

Asennus suoritettiin seuraavasti:

1. Haettiin openssh:n qnx-version binääripaketti osoitteesta

[http://sourceforge.net/project/showfiles.php?group\\_id=21249&package\\_id=49071](http://sourceforge.net/project/showfiles.php?group_id=21249&package_id=49071)

2. Purettiin paketti pöytätyökoneelle tiedostojen kopioimista varten. PC/104:n levyyn pääsee käsiksi sen ollessa päällä suoraan /net/emu.automationit.hut.fi -hakemiston kautta.

3. Kopioitiin pöytätyökoneelta seuraavat tiedostot PC/104:n levyille:

```
cp /sbin/devc-pty /net/emu.automationit.hut.fi/sbin
    (taustaohjelma, joka mahdollistaa /dev/pty* -virtuaaliterminaalit)
cp /usr/bin/random /net/emu.automationit.hut.fi/usr/sbin
    (taustaohjelma, joka tarjoaa satunnaislukuvirran /dev/urandom)
```

```
cp /usr/lib/libz.so.2 /net/emu.automationit.hut.fi/usr/lib
```

```
cp .../usr/sbin/sshd /net/emu.automationit.hut.fi/usr/sbin
cp .../usr/etc/moduli /net/emu.automationit.hut.fi/usr/etc
cp .../usr/etc/sshd_config /net/emu.automationit.hut.fi/usr/etc
cp .../usr/libexec/sftp-server /net/emu.automationit.hut.fi/usr/libexec
cp .../usr/bin/scp /net/emu.automationit.hut.fi/usr/bin
    (... tarkoittaa hakemistoa johon asennuspaketti purettiin)
```

4. Ajettiin /usr/bin/random -t -p jotta avainten luontiin saatiin satunnaislukuja.

5. Mukailleen ssh:n asennusohjelman skriptiä, suoritettiin asennuspaketin usr/bin-hakemistossa oleva ssh-keygen PC/104:ssä seuraavasti host-avainten luomiseksi:

```
ssh-keygen -t rsa1 -f /usr/etc/ssh_host_key -N ""
ssh-keygen -t dsa -f /usr/etc/ssh_host_dsa_key -N ""
ssh-keygen -t rsa -f /usr/etc/ssh_host_rsa_key -N ""
```

6. Luotiin uusi käyttäjä ssh-yhteyksiä varten komennolla `passwd autt-1`.

7. Muokattiin `/etc/passwd`-tiedostoa siten, että uusi käyttäjä on pääkäyttäjän alias ja voi täten ajaa laitteistoriippuvaista koodia. Käytännössä tämä tapahtui muuttamalla käyttäjän UID ja GID nolliksi.

```
root:x:0:0:Superuser:/:bin/sh
autt-1:x:0:0:Remote user:/:bin/sh
```

8. Muokattiin Jouko Virran luomaa käynnistyskriptiä `/etc/system/sysinit` siten että `sshd` käynnistyy tietokoneen käynnistyksen yhteydessä. Lisättiin tiedoston loppuun:

```
echo "Starting sshd"
/usr/sbin/random -t -p
/sbin/devc-pty
sleep 2
/usr/sbin/sshd
```

9. Uudelleenkäynnistyksen jälkeen PC/104:ään saa yhteyden komennolla `"ssh 10.0.0.3"` tai `"ssh autt-1@10.0.0.3"` ja syöttämällä salasanan (projektin aikana salasanaksi asetettiin mielikuvituksettomasti 'salakala'). Mikäli yhteyden luominen ei onnistu, `ssh`-palvelimen voi lopettaa katsomalla sen PID:n komennolla `"ps -A"` ja lopettamalla sen `kill`-komennolla. Tämän jälkeen `sshd`:n voi käynnistää komennolla `"/usr/sbin/sshd -D -e"`, jolloin `sshd` ei siirry tausta-ajoon (-D) ja tulostaa virheilmoitukset suoraan näytölle (-e).

10. Koska oletuksena pöytä tietokoneella ja PC/104:llä on samat komentorivikehotteet (prompt), on helppo erehtyä `ssh`:ta käytettäessä siitä kummalla tietokoneella käskyjä suorittaa.

Tämän ongelman kiertämiseksi luotiin hakemistoon `/etc` tiedosto `profile`, joka asettaa kehotteen koulun UNIX-tietokoneita mukaillen seuraavasti:

```
export PS1="\h \w \! \\\$ "
```

Näin kehotteeksi tulee merkkijono, joka on muotoa

```
emu /etc 125 #
```

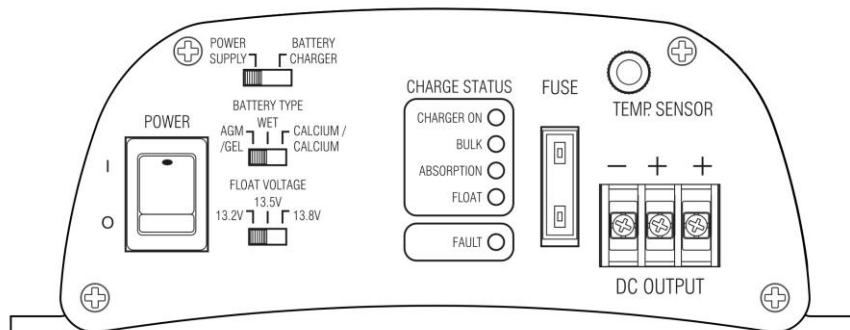
jossa `emu` on tietokoneen nimi, `/etc` työhakemisto, `125` nykyisen komennon historianumero ja `#` tarkoittaa että käytössä on pääkäyttäjän oikeudet.

## Uuden akkulaturin käyttäminen

Projektityön hankintojen yhteydessä AUTT-1 –robottiin ostettiin uusi akkulaturi aiemmin laaditun kehityssuunnitelman mukaisesti. Uusi akkulaturi kykenee samanaikaisesti ylläpitämään PC/104:n ja muun laitteiston toimintaa ja lataamaan akkuja, joten laitteistolla voidaan tehdä kehitystyötä myös akkujen ollessa tyhjiä.

Projektin aikana ei tutkittu, onko robotin vanha, pääasiassa audiokäyttöön tarkoitettu virtaliitin soveltuva uuden laturin suuremmille virroille. Tästä syystä uusi laturi kytketään toistaiseksi robottiin kuten apulaturi autoon: ensin hauenleuka positiiviseen (+) –napaan, sitten hauenleuka negatiiviseen (-) –napaan, virtajohto pistorasiaan ja lopuksi laturin virta päälle.

Akkulaturin käsikirjasta selvitettiin akkulaturille seuraavat asetukset:



*Uusi akkulaturi. Lähde: akkulaturin manuaali.*

### Power supply

Ohjeen mukaan Power supply / battery charger -kytkimessä tulee käyttää Power supply -asentoa, jolloin akkulaturi käyttää kolmesta mahdollisesta latausmoodista vain kahta, jotka ovat turvallisia lisäkuorman kanssa.

Tällöin laturin BULK-ledi ilmaisee akun varauksen olevan < 80% ja FLOAT-ledi tätä enemmän.

Mikäli BULK-ledi palaa jatkuvasti, kytketty laitteisto kuormittaa laturia liikaa.

### Battery type AGM/GEL

AUTT-1:n akku on tyypiltään "maintenance free sealed type (VRLA)", jolloin oikea valinta on manuaalin mukaan AGM/GEL.

### Float voltage 27.6 V

Laturin ohjekirjan mukaan 27.6 V float voltage on paras valinta, kun laturia käytetään power supply -toimintatilassa.

## Jatkokehitys

Projektityössä toteutetut parannukset olivat hyvä lisä AUTT-1:een, koska ne mahdollistavat jo nyt alkeellisen autonomisen toiminnan. Syksyllä 2008 laaditun kehityssuunnitelman mukaan AUTT-1 –robottia jatkokehitetään tulevaisuudessa yhä monipuolisemmalla tavalla autonomiseksi ja sellaiseksi, että se mallintaa todellisia autonomisesti liikkuvia työkoneita paremmin.

Alkuperäisen kehittämissuunnitelman tehtävien lisäksi tämä projektityö osoitti seuraavat tarpeet:

### **Antureiden liittäminen CAN-väylään (Matala prioriteetti)**

AUTT-1:n gyro-anturi on itsenäinen moduuli, joka on kytketty tietokoneeseen CAN-väylällä. Vastaavalla tavalla myös rata- ja ultraäänianturilaitteistolle voitaisiin laatia väyläohjain, jolloin robotin kaikki laitteet olisivat väylässä eikä suoraan PC:n IO:ssa ja robotti havainnollistaisi kenttäväyläparadigmaa nykyistä paremmin. Todennäköisesti gyro-anturin ohjelmistoa ja elektroniikkaa voitaisiin uudelleenkäyttää hyvin suuressa määrin, koska gyro-anturi ja rata-anturi ovat molemmat pohjimmiltaan vain väylään liitettävä AD-muuntimia.

### **Akkulaturin liitännän parantaminen (Keskitärkeä prioriteetti)**

AUTT-1:n vanha akkulaturi kytkettiin robottiin helposti Neutrik-liittimellä. Koska uuden akkulaturin latausvirrat ovat moninkertaiset, uutta laturia ei kiinnitetty vanhan laturin liittimeen. Siksi olisi tarpeen tutkia vanhan liittimen spesifikaatioista sen virrankesto, ja liittää uusi akkulaturi robottiin joko vanhalla liittimellä tai sen ollessa alimitoitettu vaihtaa latausliitin järeämpään.

### **Nyt laadittujen ohjelmien ja algoritmien jatkokehitys (Erittäin korkea prioriteetti)**

Projektityön aikana laaditut ohjelmat olivat hyvin alkeellisia ja lähinnä osoittivat laitteiston toimivuuden.

Aivan aluksi olisi syytä integroida ultraäänianturin lukeminen rata-ajoon niin, että AUTT-1 osaa pysähtyä radalla olevan esteen kohdatessaan. Tämä on verrattain yksinkertaista, joskin ohjelma täytyy toteuttaa mahdollisesti niin että radan seuranta ja ultraäänianturin luenta tehdään rinnakkain koska ultraäänianturin luenta kestää nykyisellään liian kauan ollakseen mahdollinen radan seurannan lomassa.

Lisäksi ohjelmaa tulisi kehittää niin, että rata-anturin kalibroitiedot eivät ole ohjelmaan kovakoodatut. Kiinteät kalibroitirajat voisi korvata siten, että ennen ajon alkua robotti mittaisi alkukalibraatioarvot kääntämällä anturia ensin 90 astetta vasemmalle ja sitten 90 astetta oikealle. Lisäksi kalibraatioarvot voisivat mukautua ympäristöön ja muuttuviin valaistus- ja jousitusolosuhteisiin ajon aikana. Tämän toteuttamisessa voisi ehkä hyödyntää myös esimerkiksi Kalman-suodinta. Lisäksi samalla tulisi kehittää tapaa, jolla robotti tunnistaa radan kadonneen anturin alta.

Mahdollisuuksien mukaan ohjelmien käyttämän ad-hoc –tyyppisen laskenta-algoritmin voisi korvata myös tieteellisemmällä ja kehittyneemmällä, esimerkiksi sumeaa logiikkaan tai neuroverkkoon perustuvalla algoritmilla.

### Verkko-ohjelmisto tietojen siirtämiseksi toiseen tietokoneeseen ohjausta varten (Korkea prioriteetti)

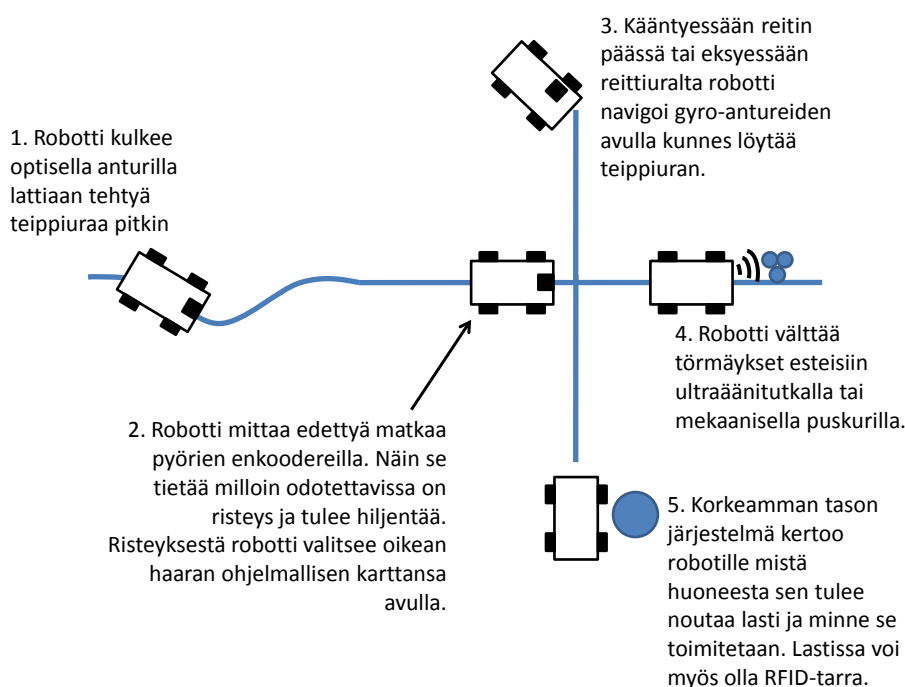
Projektityön aikana robottiin hankittiin nykyisen PC/104:n rinnalle uusi PC, johon on tarkoitus asentaa reaaliaika-Linux. Uuden PC:n olisi tarkoitus toimia AUTT-1:n päätietokoneena vanhan PC/104:n jääsessä IO- ja moottorinohjaustietokoneeksi.

Tällaisen toiminnan mahdollistamiseksi olisi toteutettava ohjelma, joka lukee rata- ja ultraääniantureita, lähettää mittaustulokset esimerkiksi Ethernet-verkon yli ja ottaa vastauksena moottorinohjaustiedot.

### Risteysajoon kykenevän karttaohjelmiston ja korkeamman tason tehtävähallintaohjelmiston suunnittelu ja toteuttaminen (Keskitärkeä prioriteetti)

Syksyllä 2008 laaditussa kehittämissuunnitelmassa esitettiin AUTT-1:n navigoinnista visio, jossa robotti seuraa matalimmalla tasolla teippiuraa nyt kehitetyllä anturilla, keskimmaisella tasolla toimiva karttaohjelmisto ohjaa robottia risteyksistä oikeisiin suuntiin ja korkealla tasolla toimii robotin ulkopuolella ajettava tehtävienhallintaohjelmisto joka voisi jakaa ajotehtäviä usealle identtiselle robotille. Näiden lisäksi navigoinnissa voisi hyödyntää luovasti etupyörien kulmaenkooderia, pyörien matkaenkoodereita, ultraääntä sekä vaikkapa konenäköä ja RFID:tä.

Alla on havainnollistava kaavio kehityssuunnitelman navigointivisiosta.

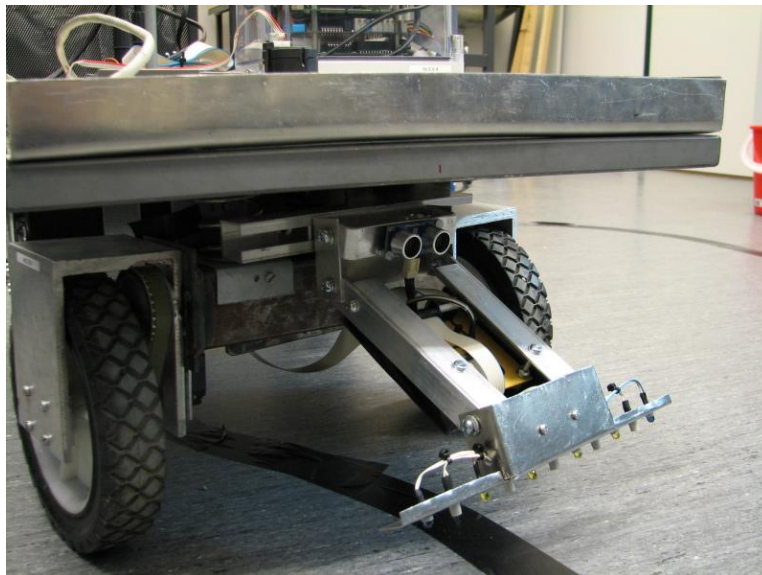


Visio AUTT-1:n monipuolisesta liikkumisesta teippiuraa pitkin. Lähde: kehityssuunnitelma 2008.

## Yhteenveto ja tiivistelmä

AUTT-1 on Automaation tietotekniikan harjoitustöissä käytettävä robotti, jota hyödynnetään esimerkiksi kurssilla AS-116.2120 Automaation tietotekniset järjestelmät. Työssä toteutettiin AUTT-1 –robottiin erittäin halpa anturi, jonka avulla robotti kykenee seuraamaan itsenäisesti lattiaan mustalla teipillä piirrettyä viivaa, asennettiin robottiin ultraäänianturi törmäysten estämiseksi, asennettiin SSH-palvelin robotin helpompaa käyttämistä varten ja otettiin käyttöön uusi akkulaturi.

Rata-anturin rakentamiseksi projektityössä tutkittiin infrapunaledien ja fotodiodien käyttämistä lattian ja mustalla teipillä piirretyn kulku-uran erottamiseksi toisistaan. Työssä rakennettiin kaksi yksinkertaista prototyyppiä, joiden perusteella toteutettiin lopulta alla olevassa kuvassa näkyvä anturi kiinnitysmekaniikkoinen ja sovitinelektroniikkoinen. Anturi kytkettiin robotin PC/104-tietokoneen analogia-IO-korttiin ja sille toteutettiin algoritmeineen yksinkertainen ohjelma, jonka avulla robotti kykenee seuraamaan mainiosti umpinaista lattiaan teipillä piirrettyä lenkkiä.



Rata-anturin kanssa samaan kiinnitysmekaniikkaan kytkettiin halpa harrastelजारobottikäyttöön tarkoitettu Parallax PING-ultraäänimoduuli, joka kykenee mittauksiin alueella 2 cm – 3 m. Ultraäänimoduulille suunniteltiin ja toteutettiin sovitinelektroniikka, jonka avulla se saatiin kytkettyä rata-anturin kanssa saman IO-kortin digitaalilähtöön ja laskuripiiriin. Ultraäänimoduulille toteutettiin yksinkertainen ohjelma, joka suorittaa mittauksen ja näyttää tulokset näytöllä.

Projektityössä asennettiin robottiin QNX:lle valmiiksi käännetty SSH-palvelinohjelmisto, jotta robotti ei ole riippuvainen ulkoisesta näppäimistöstä ja lyhytjohtoisesta kuvaputkinäytöstä. Tuloksena robotin käytettävyyden parani huomattavasti.

Robottiin ostettu uusi akkulaturi osoittautui erittäin hyväksi, sillä se mahdollistaa robotin kehittämisen myös akkujen ollessa tyhjiä ja latauksessa, toisin kuin vanha liian pieneksi mitoitettu akkulaturi.

Projektityön raportissa esiteltiin lisäksi joitakin parannusehdotuksia robotin jatkokehittämiseksi.

## Liite 1: Työpäiväkirja

21.1.	Aloitusluento	2h
28.1.	Alustava suunnitelma	2h
4.2.	Palaveri henkilökunnan kanssa	1h
5.2.	Projektisuunnitelman teko	3h
11.2.	Projektitöiden esittelytilaisuus	3h
12.2.	Hankintojen suunnittelu	3h
19.2.	Hankintojen suunnittelu	3h
20-21.2.	Anturin ensimmäisen prototyypin rakentaminen	7h
24.2.	Anturin prototyypin kokeilu AUTT-1:ssä	3h
5.3.	Säädettävä anturin prototyyppi, mittauksia	8h
17.3.	Väliraportti	1h
18.3.	Väliraportointitilaisuus	2h
19.3.	Anturin kiinnitysmekaniikan rakentaminen	7h
23.3.	Anturin mekaniikan rakentaminen	7h
24.3.	Anturin elektroniikan rakentaminen	4h
25.3.	Anturin kytkeminen PC/104:ään, ensimmäinen testiohjelma	6h
31.3.	Anturin mukaan ajava testausohjelma, sshd:n asennus	6h
1.4.	Anturin lopullisen johdotuksen tekeminen	5h
2.4.	Ultraäänianturin kytkennän suunnittelu	2h
3.4.	Ultraäänianturin kytkennän suunnittelua ja kokeilua, testiohjelma	7h
16.4.	Ultraäänianturin kytkentäelektroniikan rakentaminen	6h
17.4.	Anturin viimeisten osien rakentaminen, kokoonpano	10h
18.4.	Ultraäänianturin kytkentäkaavion piirtäminen	3h
21.4.	Loppuraportin kalvojen tekeminen	2h
22.4.	Loppuraportointitilaisuus	3h
11.5.	Loppudokumentin kirjoitus	5h
12.5.	Loppudokumentin kirjoitus	8h
13.5.	Loppudokumentin kirjoitus	5h

## Liite 2: Tärkeitä oheisdokumentteja

<http://www.alldatasheet.com/datasheet-pdf/pdf/45635/SIEMENS/SFH300.html>

SFH300-fotodiodin datalehti

<http://www.manson.com.hk/downloads/4/7673-2210-0001.pdf>

Akkulaturin (Manson SBC-2205, Insmat 513-8045) käsikirja

<http://www.parallax.com/Portals/0/Downloads/docs/prod/acc/28015-PING-v1.5.pdf>

Parallax PING –ultraäänisen sensorin dokumentaatio

<http://www.diamondsystems.com/files/binaries/dmm16v11.pdf>

PC/104:n IO-kortin manuaali

<http://www.datasheetcatalog.org/datasheet/intersil/fn2970.pdf>

PC/104:n IO-kortin laskuripiirin 82C54 datalehti