EESTI INFOTEHNOLOOGIA KOLLEDŽ

Mauno Pihelgas

EESTI INFOTEHNOLOOGIA KOLLEDŽI ROBOTI JUHTIMISPLATVORMI FUNKTSIONAALSUSE LAIENDAMINE

Diplomitöö

INFOTEHNOLOOGIA SÜSTEEMIDE ARENDAMISE ÕPPEKAVA

Juhendaja: M. Ernits

Tallinn 2010

AUTORIDEKLARATSIOON

Deklareerin, et käesolev diplomitöö, mis on minu iseseisva töö tulemus, on esitatud Eesti Infotehnoloogia Kolledžile lõpudiplomi taotlemiseks Infosüsteemide arendamise erialal. Diplomitöö alusel ei ole varem eriala lõpudiplomit taotletud.

Autor M. Pihelgas…….

(allkiri ja kuupäev)

Töö vastab kehtivatele nõuetele

Juhendaja M. Ernits……

(allkiri ja kuupäev)

Sisukord

Lühendite ja mõistete loetelu

ADC	Analoog-digitaalmuundur (*Analog-to-digital Converter*) muudab analoog sisendsignaali digitaalseks väljundsignaaliks.
API	*Application Programming Interface* – programmiliides, mille abil programm kasutab operatsioonisüsteemi või teiste rakenduste teenuseid.
APT	Debiani paketihaldur.
ASUS Eee PC	Väike sülearvuti, mis on ITK Robootikaklubis kasutusel kui roboti juhtarvuti.
Atmel	Mikrokontrollereid ja teisi tooteid disainiv ja tootev ettevõtte. [1]
bitikiirus	Sekundis edastatavate või töödeldavate bittide arv.
C++	Programmeerimiskeel, mida ITK Robootikaklubi kasutab Robovision juhtimisplatvormi arenduses.
CSV	*Comma-separated Values* – komaga eraldatud väärtused.
Doxygen	Programm, mis võimaldab mitmete erinevate programmeerimiskeelte lähtekoodile dokumentatsiooni genereerida (GNU GPL litsents). [33]
draiver	Välisseadet operatsioonisüsteemiga liidestav juhtimisprogramm.
FIFO	*First In, First Out* printsiip - esimesena salvestatud infoüksus töödeldakse või teisaldatakse esimesena.
GNU GPL	*GNU General Public License* ehk GNU Üldine Avalik Litsents. [9]
GNU LGPL	GNU Lesser General Public License ehk GNU Vähem Üldine

	Avalik Litsents. [9]
GNU projekt	1984. aastal algatatud projekt, mille eesmärgiks oli luua *Unix*-laadne operatsioonisüsteem, mis sisaldab ainult vaba tarkvara. [9]
GTK+	Graafilise kasutajaliidese loomise tööriist.
H-sild	Juhitav võimsusvõimenduselement näiteks mootori juhtimiseks.
IEEE1394 e. FireWire	Sidestandard välisseadmete ühendamiseks arvutiga.
ITK	Eesti Infotehnoloogia Kolledž.
ITK Robootikaklubi	2002. aastal asutatud ITK Robootikaklubi on ITK üliõpilaste organisatsioon, mille üks peamisi eesmärke on osaleda Robotex võistlustel ja abistada võistluste korraldamisel. Käesolevas töös lühemalt kutsutud ka robotiklubiks. [6]
ITK Robootikaklubi Wiki	ITK Robootikaklubi privaatne wiki, mis on klubisisene dokumentatsiooni ja muu informatsiooni allikas.
JPEG, PNG, TIFF	Graafikavormingu formaadid piltide edastamiseks.
libbgrab	*Framegrabber* pildivõtu teegid Linuxile.
makefile	Fail, mis lihtsustab keerulisemate projektide kompileerimist.
Matrox pilditöötlusteegid	Matroxi poolt arendatud pilditöötlusteegid, mis võimaldasid *framegrabberi* ehk „kaadripüüdja“ hangitud pilti töödelda. [7]
megaAVR e. ATmega	Atmeli poolt toodetav mikrokontrollerite perekond. [1]
omni-rattad	Rattad, mis on kaetud pöörlevate rullidega ning tänu sellele võimelised samaaegselt liikuma kahes suunas.
OpenCV	OpenCV ehk Open Source Computer Vision on vabavaraline (BSD litsents) pilditöötlusfunktsioonide teek. [19]
paarsus	Andmesides edastatud andmete veakontrollimeetod.

PlayStation Eye	Sony Computer Entertainment veebikaamera PlayStation 3 mängukonsoolile.
PWM	*Pulse-Width Modulation* – Signaali modulaator, mis on sisuliselt lüliti. Kiireid lülitusi tehes on võimalik saavutada etteantud efektiivvõimsusi seadmetel, mis muidu võimaldavad vaid täisvõimsusel töötamist.
QT	Trolltech-i loodud ning alates 2008. aastast Nokia omandis olev tarkvaraarendus raamistik (GNU LGPL litsents). [17]
Robotex	Robotex on Tallinna Tehnikaülikooli, Tartu Ülikooli ja IT Kolledži korraldatud avalik rahvusvaheline robotivõistlus, mis toimub alates aastast 2001. [24]
Robovision	ITK robotite juhtimis- ning pilditöötlusplatvorm.
RS232	*Recommended Standard* 232 – telekommunikatsiooni standard, mis on kasutusel ka arvutite jadaliidese pesades.
SDL	*Simple DirectMedia Layer* multimeediateegid arvuti riistvara (heli, klaviatuur, hiir, video) kasutamiseks.
ServoBasic	ITK Robootikaklubis kasutatav mikrokontrolleri juhtprogramm.
SVN	Apache SubVersioN on avatud lähtekoodiga versioonihaldustarkvara.
symbolic link e. symlink	Nimeviit.
TCP/IP	Edastusohje protokollistik internetiprotokolli peal.
udev	Linuxi seadmehaldur (alates kerneli versioonist 2.6)
väljaviik, viik	Viikudeks nimetatakse kiibikorpusest välja ulatuvaid metallkontakte, mis on ette nähtud kiibi elektriliseks ühendamiseks trükkskeemiga või kiibipesaga [32]
versioonihaldus	Tarkvara (või muude dokumentide) erinevate versioonide

	haldamine, kus igale versioonile omistatakse kokkulepitud süsteemi alusel number.
VGA	*Video Graphics Array* – analoogsignaalidel põhinev graafikastandard.
Video4Linux (V4L)	Video salvestamise liides Linuxile.

Sissejuhatus

Käesoleva diplomitöö teema on Eesti Infotehnoloogia Kolledži (ITK) roboti juhtimisplatvormi Robovision funktsionaalsuse laiendamine. **Lisatud funktsionaalsus võimaldab lugeda andmeid rohkematelt anduritelt ja suurendada täiturite arvu.** Töö raames on plaanis Robovision raamistikule lisada mitme mikrokontrolleri ning kaamera tugi.

Teema valikul otsustas autor eelnimetatud eesmärgi kasuks, sest on ise ITK Robootikaklubi aktiivne liige ning osales Robovisionit kasutava robotiga 2008. aasta Robotex võistlusel. Tänaseks päevaks kasutavad kõik ITK keerulised võistlusrobotid Robovision pilditöötlus- ning juhtimisplatvormi. Robootikaklubis tegeletakse aktiivselt antud raamistiku arendamisega.

IT Kolledžis on Robovision platvormi arendatud juba 2002. aastast. Arendajaid on olnud käesolevaks hetkeks üle 10. Robovisioni eelkäijaid on Robotex võistlustel edukalt kasutatud mitmete erinevate ülesannete täitmisel. Näiteks jalgpall (2009), toa koristamine (2008), võrkpall (2006), purkide ümberpööramine (2004), pallide otsimine (2002).

Jalgpallirobotite ehitamisel 2009. aastal ilmnes, et kätte on jõudnud olukord, mil Robovision hakkab ajale jalgu jääma ning pärssima roboti funktsionaalsust. Praegune platvorm toetab ainult ühte mikrokontrollerit ühe jadaliidese küljes. **Roboti külge ei saanud enam andureid ega täitureid lisada, kuna kasutatava kontrolleri väljaviigud olid seadmetega ühendatud.** Antud puudus on platvormi peamine probleem. Samuti oleks tarvis lisada tugi veel vähemalt ühele täiendavale kaamerale, sest ühe kaameraga hangitud info ei ole ülesannete lahendamiseks piisav.

Robovision kasutab *OpenCV* ja *QT4* teeke ning on realiseeritud objektorienteeritult *C++* programmeerimiskeeles. Tarkvara on avaldatud *GNU GPL* litsentsi all. Iga aastaga on koodibaas läinud järjest keerulisemaks ning funktsionaalsus on lisandunud kiiremini, kui seda on dokumenteeritud. Enamik platvormi arendusest on leidnud aset Robotexi eelses pinges ja ajapuuduses. Tihtipeale tuleb veel viimasel minutil enne võistlust teha kiireid parandusi lähtekoodis. Kahjuks on sellises olukorras tehtud muudatused jäänud dokumenteerimata. Ühtlasi pole olnud inimest, kes selle eest vastutaks. Sellest tulenevalt on käesolevaks hetkeks muutunud dokumentatsiooni järgmine ohtlikuks, sest joonised on aegunud ning seal kirjutatus sealt võib esineda palju vigu. Ühel jätkusuutlikul programmil peab olema kaasajastatud ning täpne dokumentatsioon.

Ülalnimetatud probleemide ühe võimaliku lahendina näeb autor seda, kui Robovision oskaks suhelda mitme mikrokontrolleriga ja töödelda mitmest kaamerast loetud pildiinfot. Ühtlasi peaks Robovisioni jätkusuutlikkuse ning töökindluse säilimiseks käima koodiga kaasas asjakohane ning korrektne dokumentatsioon.

Lähtudes eelkirjeldatud probleemidest saab sõnastada diplomitöö teema: Eesti Infotehnoloogia Kolledži roboti juhtimisplatvormi funktsionaalsuse laiendamine.

Funktsionaalsuse laiendamiseks oleks tarvis lahendada järgnevad küsimused:

- mitme mikrokontrolleri toe lisamine;
- mitme kaamera toe lisamine;
- koodi parandamine / dokumenteerimine.

Antud diplomitöö tulemusena saab ITK roboti juhtimisplatvormil Robovision olema mitme mikrokontrolleri ja kaamera tugi, korrastatud struktuur, töötav ning dokumenteeritud lähtekood. ITK Robootikaklubis ei ole keegi varem selliste probleemide lahendamist ette võtnud.

Diplomitöö sisu

Analüüsi osas annab autor ülevaate, milline on ITK roboti ja selle juhtimisplatvormi

Robovision ülesehitus. Järgnevalt analüüsitakse, millest eelnevalt püstitatud probleemid tulenevad ning arutleb võimalike lahenduste üle. Ühtlasi uuritakse, et mis mujal maailmas on sarnaste probleemide ja analoogsete platvormide puhul tehtud. Viimaks valib autor välja töö edasise käigu ning üritab leida parima lahendusviisi.

Teostuse ehk realisatsiooni osas kirjeldab autor diplomitöö praktilist poolt – seadistamist, programmeerimist ja dokumenteerimist. Kõigepealt tuuakse välja arenduskeskkonna omadused ning eelseadistamine. Seejärel kirjeldatakse olemasoleva platvormi moodulite dokumenteerimist, mis käis kaasas iga täiendava funktsionaalsuse lisamisega. Täpsemalt on välja toodud konkreetsed muudatused Robovisioni klasside juures. Lõpuks on lühidalt kokku võetud antud diplomitöö tulemused ning kirja pandud edasised plaanid.

Kokkuvõttes sõnastab autor lühidalt püstitatud probleemid ning annab ülevaatliku analüüsi. Põgusalt kirjeldatakse tehtud tööd ja leitud lahendusi. Viimaks tuuakse välja diplomitöö kasu ITK Robootikaklubile.

Diplomitöö lisas leidub kaks küllaltki suuremõõdulist pilti, mille teemaarendusse paigutamisel ei olnud otsest vajadust ning nende puhul kehtis reegel – mida suurem, seda mõistetavam. Lisa 1 on ITK roboti Troller-Roller pilt, millel on välja toodud erinevate komponentide nimed. Eesmärk on näidata, milline on ITK robot ning kui palju on sellel erinevaid sisendeid ja väljundeid. Lisa 2 kujutab Robovision platvormi klassidevahelisi seoseid. Eesmärk on anda detailne, kuid konkreetne ülevaade Robovisionist. Nimetatud lisa sisaldab täiendavaid mooduleid, mida käesolevas diplomitöös ei mainitud.

1. Analüüs

1.1. Esmane analüüs

Mõistmaks, millest lähtuvalt antud diplomitöös otsuseid tehakse, tuleks eelnevalt tutvustada taustalugu, ITK robotite ülesehitust ning juhtimisplatvormi Robovision.

Aastal 2002 arendati pilditöötlusprogrammi Matrox pilditöötlusteekidega Microsoft Windows XP operatsioonisüsteemis. Katsetamise käigus jooksis süsteem teadmata põhjustel pidevalt kokku. Kuna probleemi ei õnnestunud tuvastada, siis programmeerimise lihtsustamiseks mindi üle *SDL* teekidele ning Linux platvormile. Edasine arendus toimus *Video4Linux (V4L)* ning hiljem *Video4Linux2 (V4L2)* vahendeid kasutades. Lühidalt öeldes, igal aastal muutus arenduskeskkond niivõrd palju, et enamik koodi tuli ümber kirjutada. [7]

2008. aastal otsustati üle minna *OpenCV* ja *QT4* teekidele, mis on suures osas püsinud muutumatult kuni tänaseni (va vastavalt ülesandele muutuv pilditöötlusalgoritm). Ühine stabiilne platvorm tuleb kasuks kogu klubile. Sarnase struktuuriga, kuid sisult siiski erinevatel robotitel saab rakendada sama pilditöötlusalgoritmi. Iga meeskond ei pea enda pilditöötlust looma hakkama, vaid saab keskenduda roboti liikumisalgoritmi välja töötamisele.

Autor on tegutsenud ITK Robootikaklubis alates 2008. aastast ning osalenud robotiga Tolmurull meeskonna „Nixie tiim“ kooseisus Robotex 2008 võistlusel. Tegevusvaldkonnad olid meeskonnaliikmete vahel jaotatud. Autori peamine ülesanne oli Tolmurulli programmikoodi kirjutamine ning -testimine. Kasutatud sai Robovision raamistiku olemasolevat funktsionaalsust ning ei olnud tarvis uurida kuidas signaalid

mikrokontrollerisse või kaadrid kaamerast pilditöötlusmoodulisse jõudsid. Sellest tulenevat peab autor siinkohal tunnistama, et Robovisioni rohkem tehnilise küljega ta palju kokku ei puutunud.

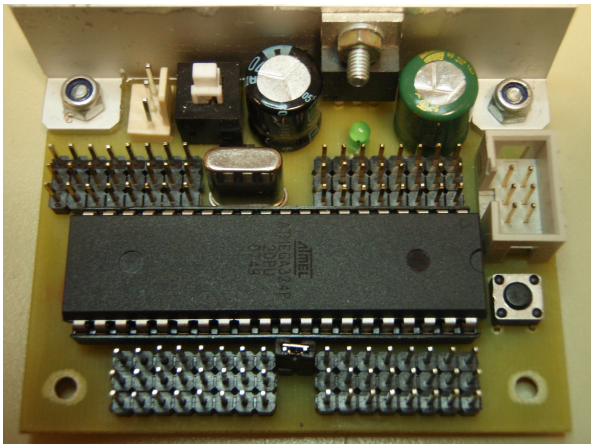
Seega ei saa ka käesoleva töö autor asuda kohe probleemi lahendama, vaid peab eelnevalt selgeks tegema olemasoleva süsteemi.

1.2. ITK roboti struktuur

Iga (käsitsi) ehitatud robot on individuaalne. Neil on oma välimus, otstarve, ehitaja „käekiri“ ja „iseloom“. Samas on võimalik enamikel robotitel leida sama otstarbega komponente. Erandiks ei ole ka ITK robootikute poolt valmistatud robotid. Järgnevalt tuleb juttu ITK robotite komponentidest, Robovisioni moodulitest ning kuidas need kaks omavahel seotud on. Antud peatüki 1.2. ITK roboti struktuur paremaks mõistmiseks käib käesoleva diplomitöö juurde lisa 1, mis toob ITK meeskonna Madistajad roboti Troller-Roller näitel välja roboti ehituse tähtsamad osad.

1.2.1. Mikrokontroller

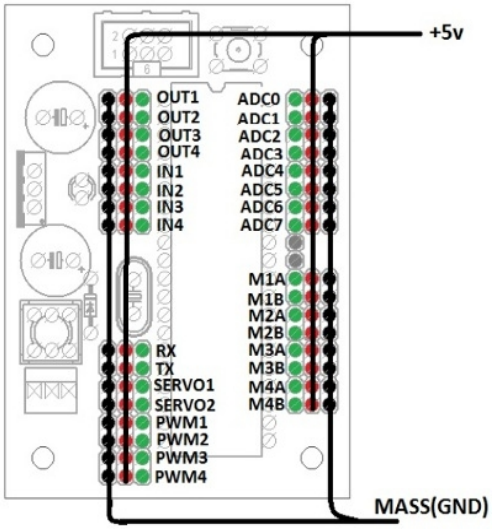
Kasutatakse Atmeli ATmega88, ATmega324P ja ATmega328P mikrokontrollereid. Nimetatud mudelid omavad palju eriotstarbelisi väljaviike, kuid oluline on märkida, et analoogsisendite (*ADC*) arv on kõigest 8 iga kiibi kohta. Mootorite ühendamiseks on 4 väljundit (*PWM*). Kasutamiseks valmistatakse mikrokontrolleriga varustatud protsessorplaat (vt joonis 1 ja 2), mille külge saab mugavalt ühendada kõik vajaliku – andurid, täiturid, toite ning jadaliidese. Suhtlus juhtarvutiga toimub *RS232* ja *USB*-liidese ülemineku abil.



Joonis 1. Roboti Digipallur protsessorplaat [13]

Enne uue mikrokontrolleri kasutamist tuleb tema välkmälu üle kirjutada ITK Robootikaklubis kasutatava ServoBasic programmiga. Käsud võivad erinevatel kontrollerite mudelitel vähesel määral erineda, kuid üldjoontes on ITK Robootikaklubis kasutusel ühine käsustik.

Kontrolleri juhtimiseks saadetakse talle täisarvulisi väärtusi, mis tähistavad erinevaid käske. Mõned kirjutamise käsud vajavad enda järel ka sobivat täisarvulist sisendit. Sellisel kujul toimib suhtlus mikrokontrolleriga ka Robovision programmis. Käskudest annab ülevaate Tabel 1. [11]



Joonis 2. Roboti Digipallur protsessoriplaadi skeem [13]

Tabel 1:
ITK Robootikaklubis kasutatava ServoBasic kontrolleri juhtkäsud [11]

Saadetav väärtus	**Selgitus**	**Tagastatav väärtus**
1-10	Loeb vastava servo väärtuse	Servo väärtus
20-27	Loeb vastava ADC väärtuse	ADC väärtus (anduri näit)
42	Loeb kõikide ADC ja digitaalsete portide väärtuste keskmised	Komaga eraldatud (*CSV*) analoog- ja digitaalportide keskmised väärtused
44	Loeb kõikide ADC ja digitaalsete portide väärtused	Komaga eraldatud (*CSV*) analoog- ja digitaalportide väärtused
129-138 <väärtus>	Kirjutab vastava servo <väärtuse>	-
162-165 <väärtus>	<Väärtustab> PWM-i vastavate mootorite juhtimiseks	-
228	Salvestab kõikide servode väärtused kontrolleri püsimällu	-
230	Alglaadimine ehk Reset	-
231 <väärtus>	Kõik servod sisse(1) / välja (2)	-
233 <väärtus>	Liigutab <väärtustatud> digitaal-pordi biti üles/alla	-

1.2.2. Andurid

Andurite abil saab robot informatsiooni välismaailmast. Need ühendatakse mikrokontrolleri analoogsisendite külge, mis muudavad andurist saadud pinge diskreetseks väärtuseks. ITK Robootikaklubis kasutatavate ATmega kiipide *ADC* sisenditel on 10bitine resolutsioon – see tähendab, et pinge analoogväärtused (0 kuni võrdluspinge, näiteks 5V) teisendatakse diskreetsesse väärtusvahemikku 0 - 1023. Andureid on väga erinevaid, kuid ITK robotitel esinevad eelkõige infrapuna kaugusandurid, infrapuna vastuvõtjad, valgusandurid, puuteandurid ning akude täituvusandurid.

1.2.3. Täiturid

Täituriteks nimetatakse roboti väljundeid, mille kaudu robot keskkonda mõjutab. Sõltuvalt roboti kasutusalast võib nende otstarve muutuda. Kõige tüüpilisemad täiturid roboti küljes on mootorid, mille külge on ühendatud rattad. Aeglasemate ja ühtlasi täpsemate manipulaatorite juhtimiseks kasutatakse servomootoreid – vajalik näiteks esemete haaramiseks. Kasutatakse ka väikesemõõdulisi *LCD* ekraane ja heliväljundeid.

1.2.4. Kaamera

Kaamera abil saab robot välismaailmast hankida pildiinfot. Roboteid saab ehitada ka ilma kaamerata, kuid siis peab usaldama teistest anduritest kogutud informatsiooni. ITK robotite puhul on sobivad paljud veebikaamerad, kuid robootikas on oluline võimalikult suur kaadrisagedus, madal kosteaeg, kaamera väikesed mõõtmed ning maksumus. Robovision töötab Linux operatsioonisüsteemis, seetõttu on vajalik, et veebikaamera oleks toetatud Linux keskkonnas. Eelnevaid omadusi arvestades on ITK Robootikaklubis alates 2009. aastast kasutusel PlayStation Eye veebikaamera. Varasematel aastatel on kasutatud erinevaid Logitech veebikaameraid. Viimasel kahel aastal on olnud ITK robotite juhtarvutitel (ASUS Eee PC) olemas ka integreeritud veebikaamera, kuid seda ei ole kehva asukoha tõttu kunagi võistlustel kasutatud.

1.2.5. Juhtarvuti

Töötlusplatvorm võib sisuliselt olla ükskõik milline Linuxit toetav arvuti. Minimaalsed

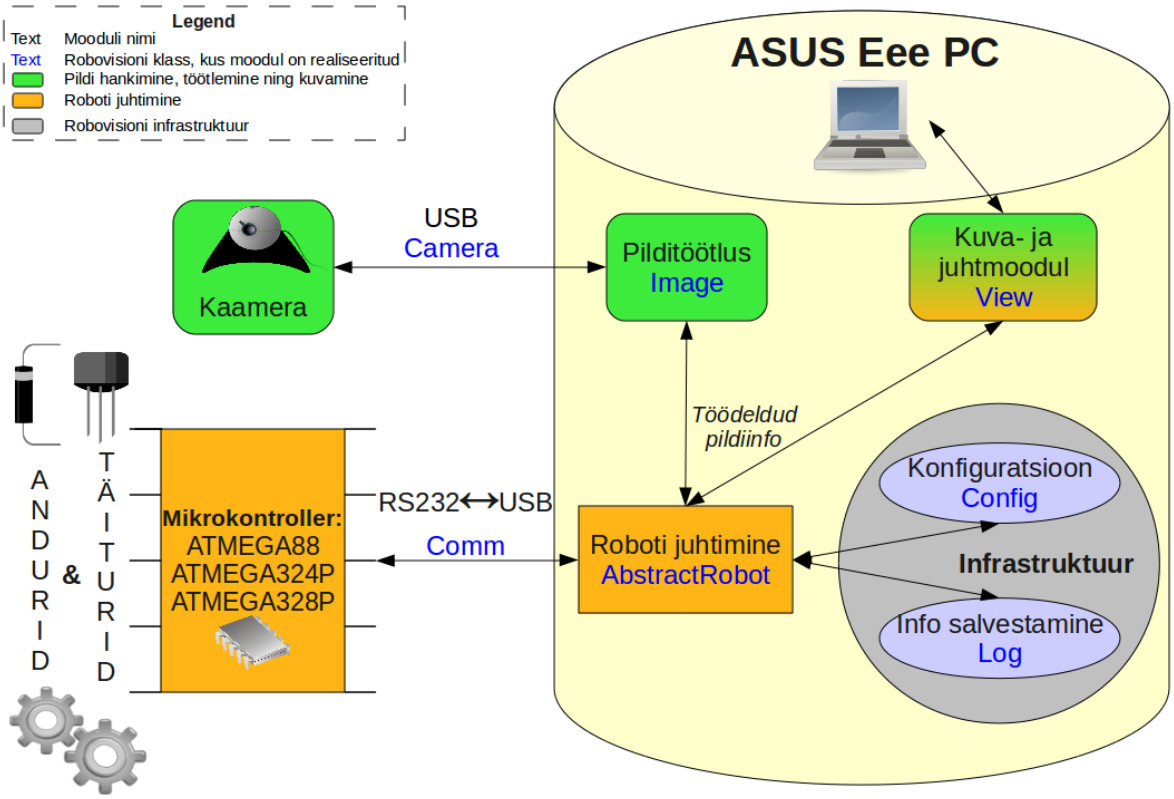
nõuded riistvarale on 600MHz protsessor, 512MB muutmälu, *USB* port, *VGA* väljund. Lisaks peab arvestama roboti mõõtmete piiranguga (allika [25] alusel: 2010. aastal 35cm põhjadiameetri ja kõrgusega silinder). Eelnevast lähtudes on IT Kolledži robootikaklubi kasutusse valitud ASUS Eee PC 700 series, millele on paigaldatud Ubuntu Eee distributsioon. Vajalik on lisateekide paigaldamine ning Robovisioni allalaadimine ITK Robootikaklubi SVN-i hoidlast.

1.2.6. Robovision

Tegemist on C++ programmeerimiskeeles arendatava programmiga. Käesoleva diplomitöö alguseks hõlmab Robovision 2010 endas järgnevat funktsionaalsust:

- kommunikatsioon: suhtlus maksimaalselt ühe mikrokontrolleriga;
- kaamera: pildi küsimine maksimaalselt ühelt kaameralt;
- pilditöötlus: pildilt valgete pallide otsimine;
- roboti juhtimine: igal robotil individuaalne klass, mis omab kõikide robotitega ühiseid funktsioone;
- kuva: töödeldud pildi kuvamine juhtarvuti ekraanile ja programmi teadete trükkimine standardväljundisse;
- juhtmoodul: juhtarvuti klaviatuurilt nupuvajutuste lugemine;
- roboti konfiguratsioon: käivitamisel iga roboti individuaalsest konfiguratsioonifailist robotile omaste parameetrite lugemine;
- logimine: programmi teadete ja roboti tööinfo kirjutamine logifaili.

Joonis 3 Illustreerib, kuidas ITK robotite erinevad osad ning Robovisioni moodulid omavahel kokku sobituvad.



Joonis 3. ITK robotite struktuur

Kommunikatsiooni moodul

Kommunikatsiooni protokoll on realiseeritud Comm klassis, mis avab ühenduskanali mikrokontrolleriga aadressil */dev/usbserial*.

Tegelikult luuakse *USB-RS232* ülemineku ühendamisel arvutiga seade */dev/ttyUSB0*, kuid muutumatu aadressi ning väiksema viite saamiseks tekitatakse *udev* reegli abil *usbserial* nimeline nimeviit (*symlink*). Kuna töö hilisemas faasis tuleb kasutatavatest *udev* reeglitest veel täpsemalt juttu, siis hetkel sellel pikemalt ei peatu.

ITK robotitel kasutatava asünkroonse jadaliidese omadused: 115200-8N1

- bitikiirus: 115200 bitti sekundis;

- andmebitte: 8 bitti;
- paarsus: ei kontrollita;
- sõnumit lõpetavaid bitte: 1 bitt.

Comm klassi funktsioonid võimaldavad:

- luua ühenduse mikrokontrolleriga (openSerial);
- lugeda üksiku anduri, digitaalsisendi või servo väärtust (readSerial);
- lugeda korraga kõikide andurite ja digitaalsete portide väärtused ning genereerida nendest väärtuste massiiv (readSerialMulti);
- kirjutada mikrokontrollerisse valitud servo või mootori väärtus (sendCommand);
- mõne ülalmainitud funktsiooni ebaõnnestumisel sulgetakse ning üritatakse taasavada ühendus mikrokontrolleriga (reopen).

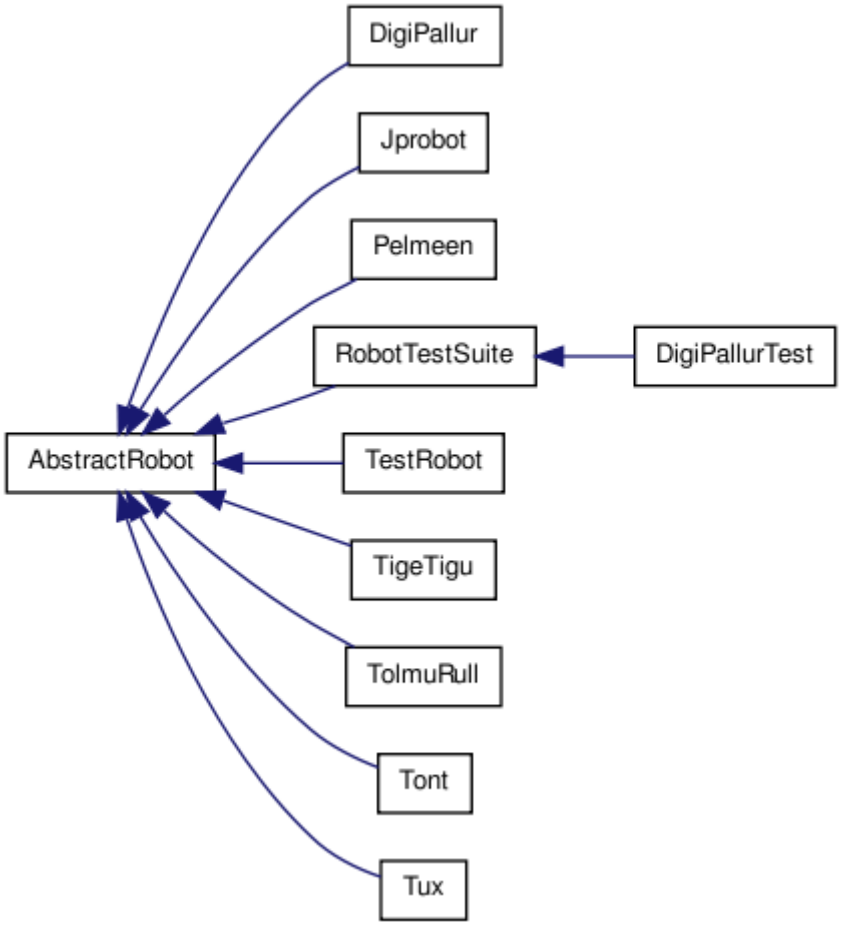
Roboti juhtimine

Igal robotil on omanimeline klass, mis pärineb abstraktsest AbstractRobot klassist (vt Joonis 4). Abstraktne klass tähendab, et seda klassi ennast ei saa kasutada robotina, vaid loodud robotid saavad pärida selle klassi omadused. Seega jagatud funktsionaalsus on kõikidel ITK robotitel ühine. Klass sisaldab funktsioone, mis mugandavad roboti programmeerimist ning robotiehitaja ei pea süvenema kommunikatsiooni protokolli detailidesse.

AbstractRobot klassi funktsioonid võimaldavad:

- küsida servo väärtust (getServo)
- küsida analooganduri väärtust (getAnalog)
- küsida kõikide andurite ja digitaalportide väärtuste massiive (getAnalogs ja getSensors)
- küsida kõikide andurite ja digitaalportide (mitme päringu) aritmeetiliste keskmiste väärtuste massiivid (getAnalogsAvg) – vältimaks juhuslikke valeandmeid
- Lülitada kõik servod välja (setServoOff)
- Lülitada kõik servod sisse (setServoOn)

- Juhtida mootoreid (setDcMotor)
- Väärtustada digitaalporte (setDigital)
- Nullida digitaalporte (clearDigital)
- Programmeerida igale robotile individuaalne juhtimisalgoritm (virtuaalne funktsioon go(), mis kirjutatakse iga roboti klassis üle)



Joonis 4. AbstractRobot klassi hierarhia graaf

Pilditöötlus

Pilditöötlus on realiseeritud Camera ja Image klassides. Camera klass käivitab kaamera ja

küsib sellelt kaadreid. Image klassis toimub pildi töötlemine vastavalt konkreetse ülesande vajadusele ja see muutub igal aastal. Sellepärast ei ole siinkohal mõtet detailset ülevaadet pilditöötlusmoodulist anda.

Kuva- ja juhtmoodul

Robovisioni View klass sisaldab endas kuva- ja juhtmoodulit. Kuvamoodul väljastab loodud aknasse töödeldud pildiinfo. Juhtmoodul võimaldab arvuti sisendseadmete abil robotit juhtida – näiteks klahvivajutuse peale roboti käivitamine. Praeguse kuva- ja juhtmooduli peamine puudus on, et seda on võimalik efektiivselt kasutada ainult juhtarvuti peal.

Infrastruktuur

Config klassis toimub iga roboti individuaalse konfiguratsioonifaili laadimine, millest loetakse kaamera ja jadaliidese aadressid ning pildiparameetrid. Neid on võimalik ette anda ka käsurea parameetritena.

Log klass on mõeldud roboti tööinfo salvestamiseks logifaili, sest programmi standardväljundisse kirjutamine on võrdlemisi aeglane ning vähendab töökiirust märgatavalt. Kahjuks ei ole see klass siiani eriti suurt kasutust leidnud.

1.3. Probleemi analüüs

Järgnevalt uurib autor, miks eelkirjeldatud roboti platvormil laiendatavuse probleemid esinevad. Ühtlasi kaalutakse lahendusi, mis ITK Robootikaklubile ja robotitele sobida võiksid. Analüüsitakse ka analoogseid platvorme mujalt maailmast, et leida, mis neil sarnaste probleemide korral on ette võetud.

1.3.1. Täiendav kaamera

Olukord

ITK robotid kasutavad pilditöötluseks ühte veebikaamerat, mis on paigutatud küllaltki madalale roboti esiossa. Kaamera asetseb maapinnaga paralleelselt. Selline paigutus tuleneb pilditöötlusalgoritmist. Kui kaamera asetseb pallide ja väravaga samal kõrgusel, siis on lihtne palli poole sõita ning kontrollida, kas lüües satub pall väravasse või mitte.

Kahjuks on sellise kaamera vaatenurk liiga piiratud ning ümberringi asetsevate pallide leidmine võtab palju aega. Selleks oleks tarvis täiendavat parema positsiooniga kaamerat, mille abil oleks võimalik kiiresti leida vastane ja pallid. Idee on kasutada kõrgemal asuvat lainurk või isegi 360° kaamerat.

Probleem

Teise kaamera ühendamine tundub lihtne, kuid Robovision hetkel ei toeta kahe kaamera kasutamist. Tarvis oleks teistsugust roboti juhtplatvormi või siis täiendada olemasolevat selliselt, et oleks võimalik eraldi töödelda erinevate kaamerate pilte. Sama pilditöötlust erinevatele kaameratele rakendada ei oleks mõtet, sest informatsioon, mida pildilt otsitakse, on erinev.

1.3.2. Täiendavad andurid ja täiturid

Olukord

Probleem, et roboti külge ei saa ühendada rohkem andureid ega täitureid tuleneb ITK robotitel kasutatavatest mikrokontrolleritest - nimelt nende väikesest sisend ja väljund väljaviikude arvust. ITK robotitel esindatud ATmega88, ATmega324P ja ATmega328P mudelid võimaldavad enda külge ühendada maksimaalselt 8 analoogsisendit (*ADC*) ja 4 H-silda (*PWM*), mida on robotite jaoks liiga vähe (vt joonist 2).

Näiteks, 2009. aasta meeskonna Madistajad robot Troller-Roller kasutas 2 alumist ja 2 ülemist infrapuna kaugusandurit (kaugus objektidest), 2 infrapuna vastuvõtjat (värava andurid), 2 valgusandurit (palli andur). Käisid arutelud, et tarvis oleks lisada kaugusandurid roboti tagaossa, vastase värava andurid ja akude täituvuse andurid, kuid kuna selleks hetkeks olid 8 analoogsisendit juba hõivatud, siis ei olnud see enam võimalik. Sama kehtib ka täiturite kohta. Troller-Rolleri neli *PWM* väljundit kulusid 3 omni-ratta mootori ja palli hoidva rulli kontrollimiseks (vt lisa 1). [14]

Vahetada mikrokontroller

Siinkohal tundub, et lihtne ja loogiline lahendus oleks antud ATmega mikrokontrollerid välja

vahetada. Võimalus oleks kasutada rohkemate väljaviikudega mikrokontrollereid või hoopis mõnda vastava otstarbega disainitud valmislahendust, mis võimaldaks enda külge rohkem andureid ühendada. Need variandid aga tõstataks kaks uut probleemi: mikrokontrolleri käsiteldavus ning komponentide hind.

ITK Robootikaklubis joodetakse elektroonikakomponendid liikmete poolt ise kokku. Klubi üks peamisi eesmärke ongi, et liikmed õpiksid praktiliste ülesannete käigus selgeks õiged elektromontaaži võtted. Valmislahenduste puhul ei ole nendega enamasti ise midagi peale hakata, sest võimalused modifitseerimiseks üldjuhul puuduvad. Rohkete väljaviikudega mikrokontrolleri kasutamine muudaks selle jootmise trükiplaadile keeruliseks. Seda isegi kogenud robootikutele ning rääkimata uutest liikmetest, kes tihtipeale pole enne klubisse tulekut ühtegi elektroonikakomponenti kokku jootnud.

Valmislahenduste ja rohkemate väljaviikudega mikrokontrollerite puhul kaasneb nendega märksa kõrgem hind. Näiteks 16 *ADC* sisendi ja 8 *PWM* väljundiga kiipide hinnad algavad 500kr/tk. Seevastu 8 *ADC* sisendiga mudelite hinnad jäävad üldiselt suurusjärku 50-150kr/tk. See on probleem, sest ITK Robootikaklubi eelarve on igal aastal küllaltki piiratud olnud. Kui eelmises lõigus mainitud elektroonikakomponentide ise kokkumonteerimine ei oleks enam võimalik, siis klubi kasvaksid klubi väljaminekud märgatavalt. [18]

Lisada teine mikrokontroller

Sarnaselt kaamera probleemile oleks ka siin üks lahendus lisada juhtarvuti külge teine mikrokontroller ning täiendada Robovisionit selliselt, et programmeerijal oleks võimalik pöörduda valitud seadme poole. Selline lahendus võimaldaks jätkuvalt kasutada soodsaid kiipe. Samuti oleks ka edaspidi võimalik kasutada ainult ühte mikrokontrollerit, seega olemasolevate asjade jaoks oleks muudatus väike. Võimalik oleks lisada rohkemgi kontrollereid, kuid siis on probleemiks ASUS Eee PC väike *USB* portide arv (3 tk). Kui lisada veel täiendav kaamera, siis tuleks mikrokontrollerite ühendamiseks kasutada *USB* jaoturit. Suurt kaadrisagedust vajavat veebikaamerat ei tasu *USB* jaoturisse ühendada, sest ühe *USB 2.0* kontrolleri maksimaalne ribalaius on kõigest 480 Mbit/s. Kui kaks kaamerat seda jagama peaksid, siis kaadrisagedus langeks.

1.3.3. Puudlik dokumentatsioon

Probleemiks on ka ITK juhtimisplatvormi puudulik dokumentatsioon. Diplomitöö alguseks olid Robovisioni lähtekoodis vaid mõned üksikud kommentaarid ning enamik neist olid vaid klasside autorluse kirjeldamiseks.

Põhjuseks on aastast aastasse kasvanud koodibaas. Palju muudatusi on jäänud dokumenteerimata, sest arendus toimub enamasti sügisel enne Robotexi, mil peamine eesmärk on uus lahendus tööle saada ning roboti peal ära testida. Katsetusjärgus tihtipeale aega dokumenteerimiseks ei ole. Peale Robotexi on teema unustatud ning dokumentatsioon ei ole enne järgmiseks aastaks valmistuma hakkamist enam päevakorras. Probleemi vastu aitaks võidelda, kui arenduse juurde oleks määratud inimene, kes vastutaks dokumentatsiooni loomise ning värskendamise eest. Kahjuks on sellise inimese leidmine keerulisem kui võiks arvata, sest ITK Robootikaklubi on vabatahtliku algatusega ning kooliga seotuse tõttu vahetuvad liikmed võrdlemisi kiiresti.

Lahendusena pakub autor välja, et Robovisioni lähtekood tuleb tagantjärgi kommenteerida ning selle alusel luua üldkasutatav dokumentatsioon.

1.3.4. Lahendusi mujalt maailmast

Kaamera

Mitut kaamerat kasutavad erinevad robootikaprojektid mitmes valdkonnas. Kasutusalade hulka kuuluvad näiteks:

- erineva otstarbega kaamerate kasutamine ühes süsteemis;
- humanoidrobotite ruumiline nägemine - inimnägemine (*Stereo Vision, Stereo Correspondence*); [29]
- robotitel erinevates suundades liikumiseks eraldi kaamerate kasutamine.

ITK robootikaklubis kasutatav *OpenCV* teegid võimaldavad mitme kaamera ühendamist. Olemas on isegi teegid esemete ruumilisuse analüüsimiseks kahe lähestikku asetatud kaamera (*Stereo Vision*) abil. Töötavaid näiteid leiab isegi populaarsest videoportaalist Youtube. [28]

Mikrokontroller

Analüüsi käigus leidis autor uurimustöö – mitte küll päris robootika valdkonnast, aga hoopis tsemendi tootmisprotsessi juhtivast reaalajasüsteemist. Võrreldakse erinevaid mikrokontrollerite kommunikatsiooni arhitektuure. Tänu soodsale hinnale ning töökindlusele otsustakse juhtimisplatvorm valmistada ühest kesksest juhtarvutist ning mitmest selle külge ühenduvast mikrokontrollerist. [3]

Lisaks artiklitele leidus ka foorumeid, kus arvatakse samuti, et tihti on andurite ühendamiseks vähe väljaviike ning suuremad mikrokontrollerid on kallid. Pakutud on mitme mikrokontrolleri ühendamist omavahel sedasi, et üks on vahelüliks järgnevale. Soovitati ka mitme eraldiseisva mikrokontrolleri ühendamist, et mitmelõimelises (*multithreaded*) programmis saaks samaaegselt lugeda andureid ja juhtida mootoreid. [34], [16], [10]

1.3.5. Analoogsed robootika platvormid

Analüüs

Järgnevates lõikudes tuleb juttu, et milliseid alternatiivseid robootika platvorme võiks kaaluda kirjeldatud probleemide lahendamiseks. Nõuded alternatiivsetele platvormidele on:

- tasuta;
- vaba tarkvara litsents;
- on jätkuvalt arenduses;
- omab põhjalikku dokumentatsiooni;
- võimalus platvormi ise edasi arendada;
- arendusvahendid on tasuta ning lihtsasti kättesaadavad;
- võimeline juhtima füüsilisi roboteid (pole ainult simuleerimiseks);
- võimalus kasutada mitut mikrokontrollerit ja kaamerat;
- Vähene ressursinõudlikkus.

Põgusalt sai uuritud mitmeid platvorme, kuid ideaalselt sobivat lahendust ei leitud. Enamike puhul esines siiski mõni puudujääk esitatud nõuetest. Tabel 2 teeb kokkuvõtte tulemustest.

Tabel 2:
Mittesobinud robootika platvormide kokkuvõte

Platvorm	Miks ei sobinud
Evolution Robotics ERSP [8]	Pole tasuta. Allika [5] väitel pole kuigi paindliku funktsionaalsusega
Microsoft Robotics Studio [15]	Pole tasuta (va õppe/hobi eesmärgil), suurem ressursinõudlikkus, kasutab robootikas vähemlevinud *C#* ja *VB.NET* keelt
CLARAty [4]	Arendus näib olevat peatunud – viimane väljalase aastast 2007. Lisaks avalikele moodulitele omab tasulisi/piiratud mooduleid.
Gostai Urbi [31]	Allika [2] kohaselt on selle platvormi sihtgrupp humanoid ja loomarobotid – näiteks Sony Aibo. Ratasrobotite jaoks soovitatakse Player/Stage/Gazebo projekti.

Uuritud platvormidest sobisid edasiseks analüüsiks kolm järgnevat:

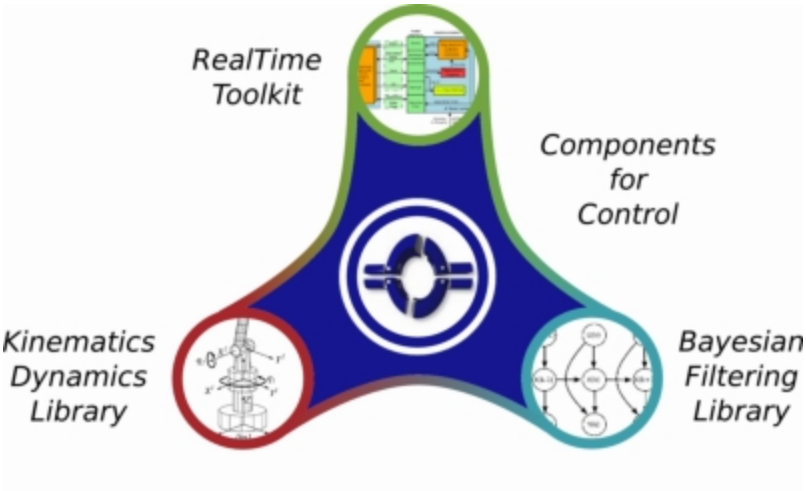
- Player/Stage/Gazebo
- Orocos
- Orca

Player/Stage/Gazebo projekt – Populaarne avatud lähtekoodiga (*GNU GPL* litsentsiga) robootika platvorm. Hõlmab endas kolme erinevat keskkonda. Stage (*2D*) ja Gazebo (*3D*) on võimsad keskkonnad robotite simuleerimiseks erinevates olukordades. Player on serveri rakendus, mis võib joosta nii simuleeritud kui ka füüsilise roboti peal. Neljas erinevas keeles programmeeritav klientprogramm ühendub serveriga ning kogu suhtlus toimub üle *TCP/IP* protokolli. Võimalik on kasutada mitut instantsi samast *draiverist*, ehk mitme kaamera tugi on olemas. Toetudes eelnevale väitele võib öelda, et võimalik on ka kommunikatsioon mitme mikrokontrolleriga. Väidetavalt ei ole tegu eriti ressursinõudliku platvormiga, kui parema jõudluse nimel on siiski soovitatud pildi- ja andmetöötlus teha üle võrgu klientarvutis, kuid see satub vastuollu Robotexi autonoomse roboti nõudega. [26], [25]

Antud projekti kasutatakse ka Eestis. Näiteks simuleerimise keskkonda Gazebo on kasutatud

Tallinna Tehnikaülikooli Biorobootika Keskuse professori Maarja Kruusmaa ja doktorant Juri Gavšini poolt antud Robootika kursuse (IBX0020; LOTI.05.010; I366) kodutööde lahendamisel. [30]

Orocos projekt – Koosneb kolmest *C++* keeles arendatud komponendist. Kinematics Dynamics ja Bayesian Filtering teegid soodustavad robootikaga seotud keeruliste arvutuste tegemist – näiteks keeruliste mitmelüliliste robotkäppade liigutamine. Kolmas, RealTime Toolkit võimaldab reaalajas robotite ning muude masinate juhtimist. Allikas [27] on välja toodud Orocos projekti hea lõimtöötluse (*multithreading*) tugi. Mitme mikrokontrolleri võimalust ei ole mainitud, küll aga on olemas mitme kaamera tugi. Samas ei ole selle raamistiku puhul soovitatud kasutada *USB* kaameraid. Pigem on toetatud *IEEE1394* ehk *FireWire* kaamerad ning nendele arendatud funktsionaalsuski on palju suurem. Seda tõestab ka Orocose *API*. Pilditöötlust antud projektis arendatud ei ole, kuid väidetavalt on võimalik isegi *OpenCV* liidestada. [21]



Joonis 5. Orocos teekide ülesehitus [22]

Orca – Orocos projekti kõrvalharu, mida on iseseisvalt edasi arendanud. Avatud lähetekoodiga raamistik sisaldab *C++* keelseid teeke. Mitme mikrokontrolleri tuge ei ole, kuid toetab mitut kaamerat ning pilditöötluseks kasutatakse *OpenCV* teeke. Dokumentatsioon on põhjalik. Dokumentatsioonist ilmneb, et kaamera funktsionaalsus on lahendatud sedasi, et kõikidelt kaameratelt küsitakse korraga kaadreid, mis nõuab kindlasti Robovisioniga võrreldes suuremat jõudlust. *QT* teeke ei ole projektis kasutatud. [20]

Robovision teiste platvormide kõrval

Robovision ennast eelnevatel aastatel edukalt tõestanud. Kuna järgmisel ehk 2010. aasta Robotexil on taaskord tarvis lahendada robotite jalgpalli ülesanne, siis ei ole tänavu mõistlik tervet robotiplatvormi vahetust ette võtta, sest võib endaga kaasa tuua tagasilöögi. Hetkel on funktsionaalsus saavutatud ning põhjalikult testitud, järgneb vaid täiendamine ja täppisseadistamine. Arvestades, et Robovision sisaldab endas kümnekonna roboti klasse, siis oleks protsessi tarvis kaasata ka eelnevatel aastatel osalenud robootikud. Muutmata kujul robotite lähtekoodi ümbertõstmine uuele platvormile oleks ilmselt raske. Otsust toetab, et 2010. aasta Robotexist kavatsevad paljud eelmisel aastal võistelnud ITK robootikud jälle osa võtta. Koodibaasi asendamine raskendaks varasema koodi korduvkasutust. Sama kehtib pilditöötluse osas. Alustada tuleks praktiliselt nullist.

Kui kaaluda tulevikus üleminekut mõnele teisele platvormile, siis käesoleva töö autor soovitab ühe variandina kaaluda Orca robootika platvormi. Teiste ees annab eelise *GPL* litsents, *OpenCV* pilditöötlusteekide kasutamine ja lai valik väljatöötatud funktsionaalsust.

1.3.6. Probleemi analüüsi tulemus

Lõputöös käsitletav probleem on üldine – teistel robootikutel ja raamistikel on esinenud sarnaseid probleeme. Enamik on suutnud probleemi kõrvaldamiseks oma platvormi täiendada.

Mitme kaamera toe puhul on üheks töötavaks näiteks Orca, mis sarnaselt Robovisionile kasutab videotöötluseks *OpenCV* teeke. Teisedki uuritud projektid on kasutanud *OpenCV* pilditöötlust kahe või enama kaameraga.

Toetudes analüüsile oleks autori arvates mõistlik lahendada täiendavate andurite ning täiturite probleem Robovisionile mitme mikrokontrolleri toe lisamisega.

Dokumentatsiooni loomine on tarkvaraarenduse loomulik osa, seega selle loomine tulevasele muudetud Robovisionile on iseenesest mõistetav.

Autori arvates ideaalset robootika-platvormi ilmselt ei eksisteeri. Nii Robovisionil kui ka kõikidel uuritud platvormidel esinevad mingid omapärad ja puudujäägid, mis nende

efektiivsust pärsivad. Võttes arvesse, et 2010. aastal tuleb Robotexil lahendada täpselt sama ülesanne, siis ei tasu 2009. aastal I ja III koha toonud Robovisionist veel loobuda.

Analüüsi otsus on lisada ITK praegusele roboti juhtimisplatvormile Robovision mitme mikrokontrolleri ning kaamera tugi. Sealjuures oma tegevust dokumenteerides ning olemasolevale koodibaasile dokumentatsioon luua.

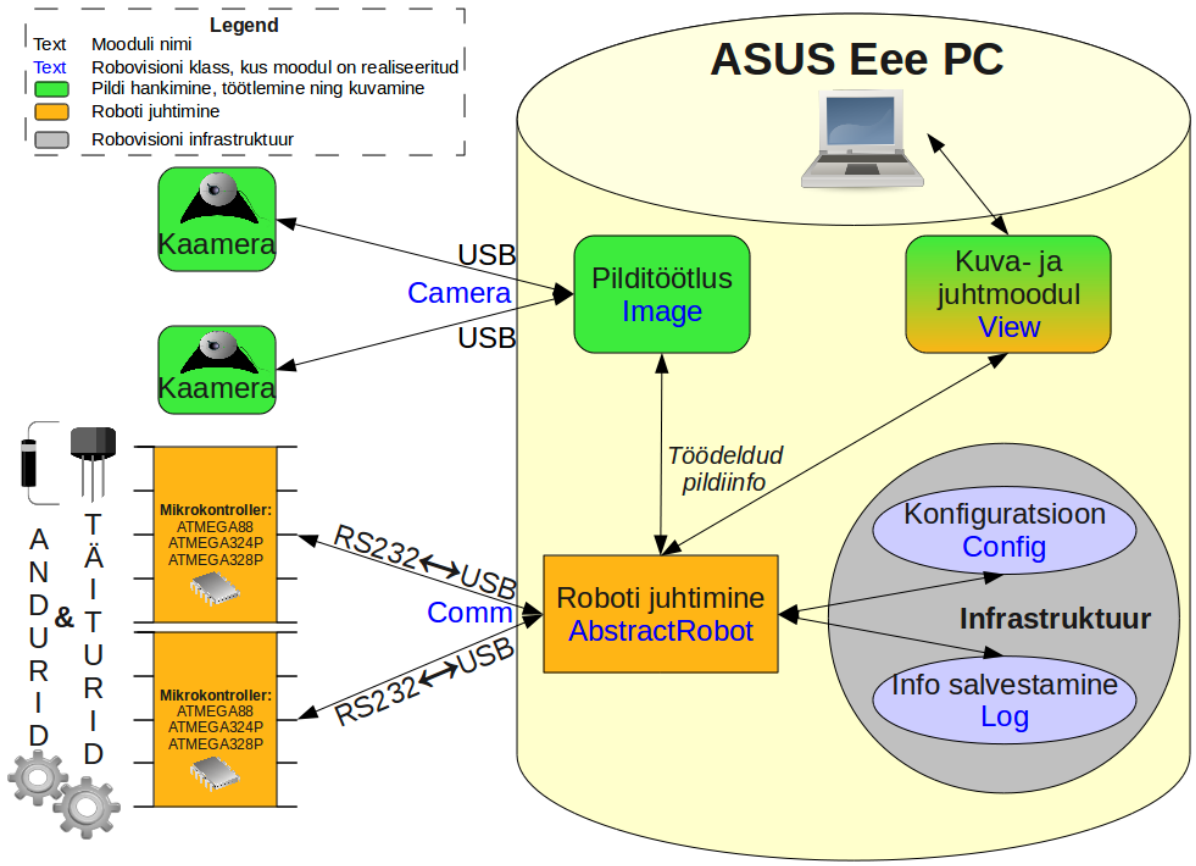
1.4. Muudatuste disaini printsiibid

Lähtuvalt analüüsi otsusest, täiustada olemasolevat Robovision platvormi, tuleb vastu võtta mõned disainiotsused, et milline peaks olema tulevane Robovision. Täiendava funktsionaalsuse lisamine peaks tehtama selliselt, et säiliks tagasiulatuv tugi vanadele ITK robotitele.

Kuna Robovisionit on arendanud mitmed inimesed ning dokumentatsioon on puudulik, siis on keeruline leida neid kohti, mida tuleks muuta. Samuti on raske ennustada ühe või teise muudatuse tagajärgi. Sellepärast on peale igat muudatust tarvis testida, et olemasolev funktsionaalsus säilis.

Enne kui autor sai hakata Robovision platvormi muutma pidi ta endale selgeks tegema kõik klasside vahelised seosed ning suhted. Selleks oli tarvis joonistada ITK roboti struktuur praegu (vt joonis 3). Selgub, et täiendava mikrokontolleri toe lisamiseks on tarvis muuta Comm ja AbstractRobot klasse ning teise kaamera jaoks Camera ja Image klasse.

Autori arvates pole igale seadmele oma klassi lisamine otstarbekas. Sellisel juhul tuleks ennustada lisatavate seadmete hulka ning teha neid toetavad klassid valmis. See võib saada lähitulevikus piiravaks teguriks ning paari aasta pärast on jälle tarvis Robovisioni struktuuri muutma. Dünaamilisem lahendus oleks täiendada olemasolevaid Comm ja Camera klasse selliselt, et nad võimaldaks vajadusel (kui rohkem kui üks seade on ühendatud) juhtida mitut seadet, kuid vanemate robotite jaoks jääks liidese tasemel kõik muutumatuks. Autori arvates peaks tulevased ITK roboti ja Robovisioni vahelised seosed kujunema nagu on näidatud joonisel 6.



Joonis 6. Robovisioni ennustatav struktuur pärast muudatusi

1.4.1. Dokumentatsioon

Vajadused

Tarvis oleks luua kaks erinevat dokumentatsiooni. Esimene neist peaks olema tehniline ning sisaldama detailset klasside kirjeldust, funktsioonide sisend- ja väljundparameetreid ning muutujate otstarvet. Mida põhjalikum, seda parem.

Teiseks tuleks ITK Robootikaklubi wikisse luua dokumentatsiooniga analoogne ülevaade Robovisioni kasutamisest, võimalustest ning eesmärkidest. Seda peamiselt eesmärgiga ülevaatlikult tutvustada ITK robotite juhtimisplatvormi. See peaks sisaldama tehnilisest dokumentatsioonist erinevalt rohkem vabas vormis juttu ning pilte. Erinevatel meeskondadel on sealjuures võimalik artiklit täiendada ning oma roboti artiklile viidata.

Dokumentatsioonivahendi valik

Lähtekoodile dokumentatsiooni loomiseks on tarvis välja valida selleks sobilik vahend. Püstitatud nõuded valitavale dokumentatsiooni generaatorile oleks järgnevad:

- C++ programmeerimiskeele tugi;
- tasuta;
- lähtekoodist automaatselt dokumentatsiooni loomine;
- toetab Linux operatsioonisüsteemi (soovitavalt ka teisi);
- arendatakse aktiivselt edasi (ei ole lõpetatud arendusega toode);
- võimaldab lähtekoodi analüüsides graafe ja diagramme joonistada;
- võimalus kujundada / seadistada väljastatavat dokumentatsiooni;
- võimalus valida väljastatava dokumentatsiooni formaati.

Kuna töö eesmärk ei ole testida erinevaid dokumentatsiooni loomise vahendeid, siis võtab autor arvesse ka Wikipedias kajastatud dokumentatsioonisüsteemide võrdlust. [35]

Lisaks mainitud nõuetele kasutatakse kogemustepõhist eelistust. Toetudes käesoleva töö analüüsi osa teiste analoogsete robootika platvormide võrdlemise uuringule teeb autor otsuse kasutada Robovisioni dokumentatsiooni loomiseks Doxygen tarkvara. Tegu on populaarse vahendiga – seda kasutatakse näiteks Player, Orocos ning Orca projektide dokumenteerimiseks.

Doxygen dokumentatsiooni generaator vastab esitatud nõuetele. Täpsemalt: [33]

- *GPL* litsents;
- toetab *C++*, *C*, *Java*, *Python*, *IDL*, *Fortran*, *VHDL*, *PHP*, *C#* jt programmeerimiskeeli
- toetab Linuxit ja Mac OS X operatsioonisüsteeme;
- töötab enamikel UNIX süsteemidel ning ka Microsoft Windowsis;
- aktiivne arendus – viimane versioon 1.6.3 (avaldatud 21. veebruar 2010);
- ulatuslike seadistusvahendite abil on võimalik automaatselt joonistada ka graafe ja

diagramme;

- toetab *HTML, LATEX, RTF, PostScript, PDF ja Unix-man* formaadis väljundit.

Dokumentatsiooni loomeprotsess

Dokumentatsiooni loomine on kavas korraldada vahelduvate etappidena klasside kaupa. Enne uue klassi muutma hakkamist tuleks dokumenteerida olemasolev funktsionaalsus. Seejärel uut funktsionaalsust lisades ei tohiks kindlasti unustada seda dokumenteerimast.

Oodatav tulemus

Kui eelmainitud protsessi järgida, siis peaksid Robovisioni olulisemad klassid selle töö käigus korralikult dokumenteeritud saama. Hea oleks, kui jõuaks dokumenteerida käesolevas lõputöös mittemuudetavad klassid, kuid see selgub töö käigus. Individuaalsete robotite klasside dokumenteerimist ette ei ole nähtud, sest see poleks antud aja raames mõeldav - iga roboti klass sisaldab 800-1500 programmirida (Robotex 2009 puhul).

2. Tehniline teostus

Käesolev peatükk sisaldab diplomitöö käigus tehtud toimingute kirjeldusi. Alustuseks paigaldatakse arenduskeskkond ning tehakse vajalik eelseadistamine. Sellele järgneb Robovisioni dokumenteerimine ning platvormile täiendava funktsionaalsuse lisamine. Viimaks antakse ülevaade töö tulemustest ning planeeritakse edasine tegevus.

2.1. Arenduskeskkonna paigaldamine

2.1.1. Operatsioonisüsteem

Enne, kui saab hakata Robovision platvormi täiendama, tuleb see paigaldada ning täita mõned eelnõuded. Käesoleva diplomitöö kogu arendus toimub Linuxi, täpsemalt Ubuntu 9.10 – *the Karmic Koala* keskkonnas. Autori valik on võrdlemisi sarnane ITK robotite juhtarvutil kasutatava Ubuntu Eee (uue nimega EasyPeasy) distributsiooniga – mis peamine, kasutatakse sama paketihaldurit *APT*.

Operatsioonisüsteemi paigaldamisel mingeid erilisi nõudeid ei ole. Vajalik on vaid hilisem ligipääs juurkasutaja õigustele, selleks et paigaldada täiendavad tarkvarapaketid ning modifitseerida mõningaid seadistusfaile. Kõik näidatud korraldused, mis algavad käsuga *sudo*, nõuavad juurkasutaja õigusi.

2.1.2. Vajalik tarkvara

Kõik vajalikud installatsioonipaketid on võimalik alla laadida Ubuntu vaikimisi hoidlatest.

Enne tarkvara paigaldamist on soovitav värskendada hoidla pakkide nimekirja.

Tuleb avada terminal ning sisestada käsk:

```
sudo apt-get update
```

Robovisioni kasutamiseks tuleb paigaldada järgnevad tarkvarapaketid:

automake	kompileerimise lihtsustamiseks loodavate nn *Makefile*-ide generaator;
build-essential	Debian pakettide kompileerimiseks vajalik informatsioon;
cmake	avatud lähtekoodiga kompileerimistööriist;
doxygen	dokumentatsiooni generaator;
g++	*GNU C++* kompilaator;
graphviz	Doxygeni poolt kasutatav graafikateek;
libcv-dev	*OpenCV* arendusteegid;
libgtk2.0-dev	*GTK+* teegi arendusfailid;
libjasper1	*JPEG-2000* formaadis piltide lugemiseks ja kirjutamiseks;
libjasper-dev	*JPEG-2000* formaadis piltide lugemiseks ja kirjutamiseks (arendusfailid);
libjpeg-dev	*JPEG* formaadis piltide lugemiseks ja kirjutamiseks (arendusfailid);
libpng12-0	*PNG* formaadis piltide lugemiseks ja kirjutamiseks;
libqt4-dev	*QT4* koos arendusfailidega;
libtiff4	*TIFF* formaadis piltide lugemiseks ja kirjutamiseks;
libtiff4-dev	*TIFF* formaadis piltide lugemiseks ja kirjutamiseks (arendusfailid);
make	avatud lähtekoodiga kompileerimistööriist;
pkg-config	kompilaatori ja linkuri võtmete haldussüsteem;
subversion	versioonihaldustarkvara;
swig	liides, mis võimaldab erinevates keeles koode integreerida;
zlib1g	pakkimisfunktsionaalsusega teek;
zlib1g-dev	pakkimisfunktsionaalsusega arendusteek.

Paigaldamine:

```
sudo apt-get install automake build-essential cmake doxygen g++ graphviz
libcv-dev libgtk2.0-dev libjasper1 libjasper-dev libjpeg-dev libpng12-0
libqt4-dev libtiff4 libtiff4-dev make pkg-config subversion swig zlib1g
zlib1g-dev
```

Oluline on jälgida teateid ekraanil ning veenduda, et kõikide nimetatud pakettide paigaldamine õnnestus.

2.1.3. Udev reegli lisamine mikrokontrollerile

Udev reeglite eesmärk on seadmete tuvastamine, seadistamine ning nimetamine vastavalt otstarbele. Analüüsi peatükis Kommunikatsiooni moodul täpsustati, et töö hilisemas faasis tuleb *udev* reeglite koostamisest lähemalt juttu.

Siinkohal ongi sobilik lisada paigaldatud operatsioonisüsteemile üks *udev* reegel, mis *USB-RS232* ülemineku ühendamisel tekitab seadme */dev/usbserial*, mis on tegelikult nimeviit (*symlink*) seadmele */dev/ttyUSB0*. Selle toimingu taga on tegelikult *USB-RS232* ülemineku kiirendamine – vähendatakse jadaliidese *FIFO* puhvri andmeedastuse ooteaega 4ms peale. Reegli koostamisel on kasutatud juhendit allikast [12].

Uurimisel selgus, et Ubuntu 9.10 versioonis asuvad reeglid kaustas */lib/udev/rules.d* (varasemate juhendite alusel */etc/udev/rules.d*). Reeglid loetakse sisse tähestiku järjekorras ning ühe seadme kohta võib kehtida mitu erinevat reeglit. Kui mingid seaded kattuvad, siis varem laetud reegel jääb kehtima.

Seadmetuvastuse *udev* reegli, mis käivitab järgnevalt esitatud skripti ja loob */dev/usbserial* seadme saab luua järgnevalt:

Luua ning avada fail tekstiredaktoriga (näiteks Nano):

```
sudo nano /lib/udev/rules.d/60-usb-serial.rules
```

Lisada järgnev sisu (ühel real), salvestada ning sulgeda fail:

```
KERNEL=="ttyUSB*", SUBSYSTEMS=="usb-serial",
RUN+="/usr/local/bin/usb_serial_udevRule.sh %n", SYMLINK+="usbserial"
```

Kasutatud reegli selgituseks:

KERNEL	võrdleb seadme nime süsteemi kernelis
SYBSYSTEMS	võrdleb seadme alamsüsteemi nime
RUN	käivitab skripti, kui eelnevad tingimused on tõesed

SYMLINK loob seadmele alternatiivse nime (nimeviit)

Seejärel luua ning avada Nanoga skript, mille antud reegel käima paneb:

```
sudo nano /usr/local/bin/usb_serial_udevRule.sh
```

Lisada järgnev sisu, salvestada ning sulgeda fail:

```
#!/bin/bash
echo "$(date) /sys/bus/usb-serial/devices/ttyUSB$1/latency_timer :)" >>
/tmp/usb_serial_udevRule.txt

echo "4" > /sys/bus/usb-serial/devices/ttyUSB$1/latency_timer
```

Kasutatud skripti selgituseks:

- Esimese *echo* käsklusega tekitatakse */tmp/usb_serial_udevRule.txt* faili uus rida – kontrollimaks, kas skript töötas
- Teise *echo* käsklusega kirjutatakse */sys/bus/usb-serial/devices/ttyUSB$1/latency_timer* faili string „4“, mis tähendab ooteaega 4ms.

Skriptile tuleb anda käivitamise õigused:

```
sudo chmod +x /usr/local/bin/usb_serial_udevRule.sh
```

Taaskäivitada udev teenus:

```
sudo service udev restart
```

Nüüd võib mikrokontrolleri ühendada *USB-RS232* ülemineku abil arvutiga. Kontrollime, kas tekkisid 2 uut seadet.

Sisestada käsk:

```
ls -lrt /dev/
```

Viimati ühendatud seadmete hulgas peaks olema nii */dev/ttyUSB0* kui ka viit sellele nimega */dev/usbserial*

2.1.4. Kaamera

Enne Robovisioni käivitamist on soovitav, et arvutiga oleks ühendatud veebikaamera. Kui kaamera on ühendatud, siis kontrollime, kas operatsioonisüsteem selle tuvastas.

Sisestada käsk:

```
ls -l /dev/video*
```

Kuvatakse kõik arvutiga ühendatud kaamerad. Hetkel näiteks */dev/video0.*

Kui kaamerat käepärast ei ole, siis testimiseks on võimalik Robovisionit käivitada ka etteantud pildifailiga. Sellest täpsemalt peatükis 2.1.5. Robovision lõigus Selgituseks.

2.1.5. Robovision

Allalaadimine

Robovision tuleb endale hankida ITK Robootikaklubi SVN-i hoidlast. Allalaadimiseks on tarvis kasutajanime ning parooli, mille saab ITK Robootikaklubi juhendajalt. Lisaks tavapärasele redigeerimise (*revision*) versioonile hoitakse Robovision projektist ka aastatepõhiseid versioone. See võimaldab kergesti kätte saada näiteks 2008. aasta robotite võistluskoode koos sobiva pilditöötlusalgoritmiga. Käesoleva diplomitöö kirjutamise ajal on kõige värskem versioon nimega „robotvision2010“.

Versioonihaldusest väljavõte (*checkout*) toimub eelmises lõigus paigaldatud versioonihaldusvahendiga Subversion (SVN). Valime kausta, kuhu soovime Robovisioni tööversiooni paigaldada. Hetke näites kasutame sisseloginud kasutaja kodukausta (~/).

Käivitada käsk:

```
svn checkout http://robot.itcollege.ee/svn/vision/robotvision2010
```

Käivitamine

Tuleb siseneda robotvision2010 kausta ning kompileerida lähtekood:

```
cd ~/robotvision2010
make all
```

Kui probleeme ei esinenud, siis nüüd on võimalik **Robovision käivitada käsuga:**

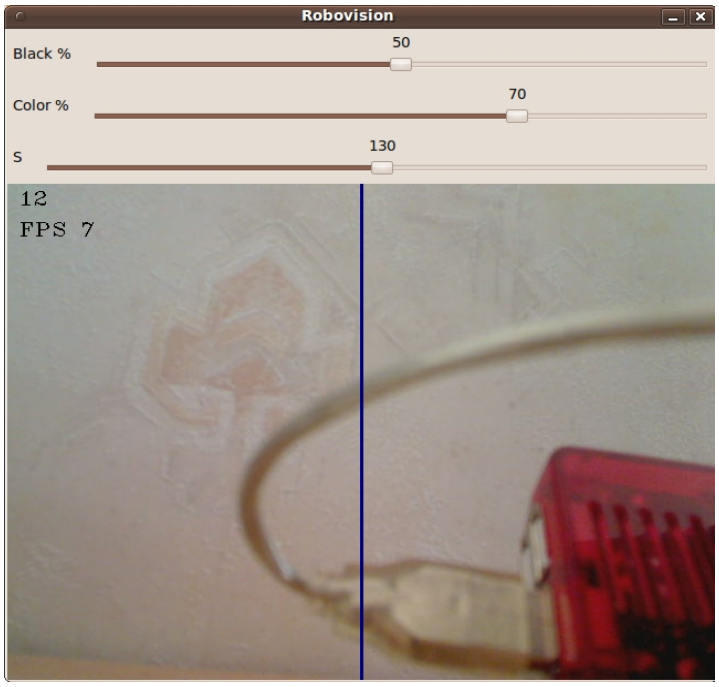
```
./main.bin -r testrobot
```

Õnnestunud käivitamisel peaks avanema joonisel 7 esitatud kuva.

Ülaosas asuvate liuguritega saab jooksvalt muuta pilditöötluse andmeid.

Veebikaamera pildil kuvatakse antud kaadri töötlemiseks kulunud aeg (ms) ning kaadrisagedus.

Robotitel kasutatavad PlayStation Eye kaamerad suudavad kuvada üle 20 kaadri sekundis. (Asus Eee PC puhul)



Joonis 7. Robovision peakuva (kuvatakse veebikaamerast võetud pilt)

Selgituseks

Kasutame üldist testimiseks mõeldud roboti klassi TestRobot, mille sisu hetkel ongi vaid käivitada pilditöötlus ning kuvada pilti. Selgituseks, et Robovision võimaldab käivitamisel käsurealt ette anda järgnevaid (*GNU-style options*) argumente:

-r, --robot	Roboti nimi. Tõstutundlik. Kohustuslik argument;
-s, --sdev	Jadaliidese nimi. Tõstutundlik;
-v, --videodev	Video seadme nimi. Tõstutundlik;
-f, --file	Pildifaili nimi. Tõstutundlik;
-c, --contrast	Kontrastsuse väärtus;
-b, --brightness	Heleduse väärtus.

Näiteks:

```
./main.bin -r testrobot --file pilt.bmp -b 100
```

Argumente, mida käsurealt ei määrata, loetakse iga roboti individuaalsest konfiguratsioonifailist, mis asuvad kaustas configs. Erandina tuleb mainida jadaliidese (mikrokontrolleri) aadressi, mis on */dev/usbserial* kujul otse lähtekoodi sisse kirjutatud. Käsurealt või konfiguratsioonifailist jadaliidese muutmine ei mõjuta midagi – tulevikus peaks see siiski muudetav olema.

Siinkohal väike teemast kõrvalekalle. Juhuslikult leidis autor vea (*bug*) – käsurealt ning konfiguratsioonifailist argumentide lugemisel oli rakendatud vale loogikat. Lähemalt uurides selgus, et kõigepealt loeti sisse käsurealt antud argumendid ning seejärel väärtustatakse nad konfiguratsioonifailist loetud andmetega üle. Tegelikult peaks olema vastupidi. Lisaks oli failist lugemisel tehtud viga. Praegu siiski selle vea parandamise üksikasjadel ei peatuks. Täpsemalt võib selle kohta lugeda hilisemas peatükis 2.4.1. Konfiguratsiooniparameetrite lugemine.

2.2. Mitme mikrokontrolleri toe lisamine

Peale tarkvara paigaldamise ja Robovision kompileerimist, saab alustada püstitatud ülesannete lahendamist. Esmalt lisatakse olemasolevale juhtimisplatvormile mitme mikrokontrolleri tugi, et roboti külge oleks võimalik ühendada rohkem andureid ning täitureid.

Analüüsis selgus, et antud funktsionaalsuse lisamist tuleks alustada Comm klassist, mis realiseerib ühenduse mikrokontrolleriga. Muudatusi tehes tuleb üle vaadata kõik muudatustest sõltuvad funktsioonid – nii Comm kui ka teistes klassides. Comm klassi kasutavad AbstractRobot klassi funktsioonid, seega teisena modifitseeritakse seda. Oluline on, et töö tulemusena säiliks koodis ka vana funktsionaalsus.

2.2.1. Comm klassi dokumentatsioon

Dokumenteerimata programmikoodis muudatuste tegemine on keeruline ning aeganõudev. Sellepärast seadis autor eesmärgiks eelnevalt kogu Comm klassi funktsionaalsus enda jaoks selgeks teha ning dokumenteerida. Esialgsed märkmed said kirja pandud läbivaatuse käigus koos ITK Robootikaklubi juhendajaga. Toetudes märkmetele ning lähtekoodile

dokumenteeriti Comm klassis kokku viie funktsiooni käitumine ning sisend- ja väljundparameetrid.

2.2.2. Comm klassi muudatused

Comm klassi funktsioon *int Comm::openSerial(const char * modemDevice)* avab faili */dev/usbserial* ning salvestab failipideme (*file descriptor*) *int* tüüpi avalikku muutujasse fd. Funktsioon tagastab sama eelmainitud muutuja fd. Sisendparameetrit modemDevice ei kasutata.

Kuna avada on tarvis mitu erinevat faili, siis esimese asjana on tarvis muutujat, kuhu saaks salvestada mitu failipidet. Poleks otstarbekas teha koodi iga potentsiaalse seadme jaoks muutujat – tarvis oleks veidi dünaamilisemat lahendust. Kuna tegu on lihtsa *int* tüüpi muutujaga, siis otsustas autor luua eelprotsessoris defineeritud suurusega (MAX_DEVICES) int tüüpi massiivi fds. Suuruseks sai esialgu määratud 16 ning tegemist on samuti avaliku muutujaga.

Massiiv fds on esialgu täidetud väärtustega -1, sest siis on lihtne kontrollida, millised elemendid on tegelikult failipidedena kasutuses. Sama kontroll oli kasutuses ka eelnevalt. Kui jadaliidese avamine ebaõnnestus, siis fd saab väärtuse -1 ning programmi veavoogu (*stderr*) kirjutatakse veateade.

Järgnevalt on tarvis kohandada openSerial funktsiooni selliselt, et oleks võimalik ette anda ühendatud seadme number (näiteks */dev/ttyUSB**0***). Muudetud funktsioon on järgmine *int Comm::openSerial(const char * modemDevice, **const int nr**)*. Nagu võib arvata, siis tuleb ära muuta koodi sisse kirjutatud aadress */dev/usbserial*. Plaanis on hakata kasutama süsteemi poolt loodud seadme aadressi *„/dev/ttyUSB“+**nr***.

Kahjuks sai tehtud muudatus hetkel sama kehv, kui seda oli eelmine. Nimelt on jadaliidese aadress jätkuvalt otse koodi kirjutatud ning sellele lisatakse vaid funktsioonile etteantud seadme number. Põhjuseks on see, et millegi pärast ei toimi roboti konfiguratsioonist etteantud nime kasutamine. Tagastatakse vaid tühi string. See probleem tuleks tulevikus platvormi paindlikkuse nimel kindlasti ära lahendada.

Peale tehtud muudatusi võib öelda, et fd == fds[nr]. Seega saab kogu Comm klassis esinevad fd muutujad asendada uue fds[nr] variandiga. Kuna fd näol on tegemist avaliku muutujaga, siis tagasiulatuva toe säilimise nimel seda päris lõplikult koodist ei eemaldatud, vaid hoopis väärtustatakse openSerial funktsioonis parasjagu avatud faili pidemega.

Kuna ka teised funktsioonid peale openSeriali on avalikult välja kutsutavad, siis tuleb nendelegi tekitada võimalus seadme numbrit sisendparameetrina anda.

Selleks, et vana koodiga robotite klassid jätkuvalt edasi töötaks, peavad jääma alles kõik varem eksisteerinud funktsioonid. Kuna ühe ja sama koodi kordamine erinevates (vanades ja uutes) funktsioonides ei ole eriti mõistlik, siis sai kasutatud *C++* keele oskust eristada samanimelisi, kuid erinevate parameetritega funktsioone. Seega on nii uued kui ka vanad funktsioonid samanimelised, kuid erinevate sisendparameetritega. Koodikorduste vältimine on lahendatud sedasi, et alati kui pöördutakse vana funktsiooni poole, siis tegelikult kutsutakse välja hoopis uus, millele antakse vaikimisi seadme number 0 (*/dev/ttyUSB0*).

Kõik ülalpool lisandunud funktsioonid ning tehtud muudatused funktsiooni sisendparameetritega tuleb kajastada ka Comm klassi päises (failis *comm.h*).

2.2.3. AbstractRobot klassi dokumentatsioon

Sarnaselt Comm klassile algab ka AbstractRobot klassi muutmine dokumenteerimisest. Dokumenteeriti 12 funktsiooni, 2 muutujat ning 4 klassiobjekti.

Võrreldes eelnevaga oli siin märksa rohkem funktsioone, kuid muudatused olid lihtsamat laadi. Funktsioonid kutsuvad välja Comm klassi funktsioone, mida kirjeldati eelnevas peatükis.

2.2.4. AbstractRobot klassi muudatused

Selleks, et vana koodiga robotite klassid jätkuvalt edasi töötaks, ei tohtinud muudatuste käigus vanu funktsioone eemaldada. Need kutsuvad jätkuvalt välja Comm klassi vanu funktsioone.

Nagu Comm klassis, nii ka siin üritas autor vältida koodikordusi, mis hilisemate muudatuste

puhul tähendaks seda, et sama muudatust tuleks sisse viia kahes kohas. Sellised kordused on ohtlikud, sest keegi teine ei oska esmapilgul arvata, et sama koodi esineb kahes kohas.

Olemasolevate funktsioonide kõrvale lõi autor 10 uut funktsiooni, mis on võimelised sisendparameetrina võtma mikrokontrolleri numbrit. Ainult virtuaalne go() funktsioon jäi puutumata, sest see kirjutatakse nagunii individuaalsete robotite poolt üle.

2.2.5. Udev reeglite kohandamine

Kuna nüüd on võimalik kasutada rohkem, kui ühte mikrokontrollerit, siis on tarvis kohendada varasemas peatükis loodud *udev* reeglit.

Võtta lahti varem loodud reeglifail:

```
sudo nano /lib/udev/rules.d/60-usb-serial.rules
```

Muuta faili sisu järgnevaks, salvestada ning sulgeda fail:

```
KERNEL=="ttyUSB*", SUBSYSTEMS=="usb-serial",
RUN+="/usr/local/bin/usb_serial_udevRule.sh %n"
```

Taaskäivitada udev teenus:

```
sudo service udev restart
```

Selgituseks:

Enam ei ole tarvis, et mikrokontolleri ühendamisel loodaks nimeviit */dev/usbserial*, sest Robovisioni koodis kasutatakse nüüd süsteemi poolt automaatselt tekitatud */dev/ttyUSB* nimealgusega seadmeid. Samas oleks jätkuvalt hea, kui seadme ühendamisel puhvri ooteaeg ikkagi 4ms peale seadistataks. Sellest tulenevalt on ka edaspidi tarvis *udev* reegleid kasutada.

2.2.6. Funktsionaalsuse testimine

Programmeerimise käigus testis autor peale iga muudatust, et kas kood kompileerub (staatiline analüüs) ning iga moodul käitub ootuspäraselt (moodultestimine). Kirjutati mõned testkoodid, millele muudetud programmikood pidi tagastama oodatud väärtuse.

Peale Comm ja AbstractRobot klasside muutmist oli võimalik alustada nende kahe integratsioonitestimisega. AbstractRoboti uued ja vanad funktsioonid olid võimelised suhtlema nii kõikide Comm klassi funktsioonidega ning seega oli tulemus ootuspärane.

Kahe mikrokontrolleri samaaegse kasutamise testimiseks pidi autor ise testmeetodid kirjutama, sest programmeerimise ajal veel ühtegi kahe või enama mikrokontrolleriga robotit ITK Robootikaklubis ei olnud. Loodi funktsioon, mis avab ühenduse kahe juhtarvuti külge ühendatud mikrokontrolleriga. Seejärel saadab neile käske ning oodati korrektseid tagastusväärtusi (vt tabel 1). Tulemus oli ootuspärane (vastavalt tabelile) ning võis peaaegu väita, et mitme mikrokontrolleri tugi oli sellega lisatud.

Testida jäi vaid tagasiulatav funktsionaalsus. Selleks küsis autor luba roboti Digipallur kasutamiseks. Lähtekood tuli roboti juhtarvutis teekide versiooni erinevuste tõttu uuesti kompileerida. Testi käigus ei ilmnenud Digipalluri funktsionaalsuses ühtegi muutust, ning kõik toimis ootuspäraselt.

Robovisionile sai edukalt lisatud mitme mikrokontrolleri tugi, mis lahendas andurite ning täiturite laiendatavuse probleemi.

2.3. Mitme kaamera toe lisamine

Teise probleemina võttis autor käsile roboti piiratud vaatenurga probleemi lahendamise. Nagu varasemalt selgitatud, siis roboti üks ainuke kaamera (vt lisa 1) on paigutatud sedasi, et pilditöötluse abil oleks võimalikult lihtne pall väravasse lüüa - hinnata, kas robot, pall ja värav on ühel sirgjoonel. Täiendav kaamera laiendaks roboti vaatenurka ning aitaks kiiremini palle leida.

2.3.1. Camera klassi dokumentatsioon

Sarnaselt teiste klasside muutmisega alustab autor ka siin dokumentatsioonist. Camera klass on võrdlemisi lihtne. Konstruktoris laetakse roboti konfiguratsioon, käivitatakse logimine ning luuakse CvCapture tüüpi capture objekt, mis *OpenCV* abil võimaldab kaameraga ühenduda. Kasutatakse ainult ühte kaamerat indeksiga 0 ehk */dev/video0*. Siin klassis on lahendatud pildifaili avamine img objekti, juhul kui kasutaja otsustab kaamera asemel pilti

kasutada.

Kogu klass sisaldab kõigest kolme muutujat ja kolme funktsiooni. Funktsioonide otstarve on tagastada capture ja img objektid. Selline lahendus tuleneb sellest, et nende objektide loomisel kasutatakse singleton mustrit.

2.3.2. Image klassi dokumentatsioon

Võrreldes Camera-ga on Image klass märksa mahukam, sisaldades endas 7 funktsiooni, 26 muutujat. Kuna pilditöötlusalgoritm on keeruline ning suures osas ühe kolmanda Robovisioni arendaja poolt loodud, siis dokumenteeritud said kõik elemendid, mille otstarvet ITK Robootikaklubi juhendaja ja autor teadsid.

2.3.3. Camera klassi muudatused

Camera klassis tuleb luua kaks kaamera objekti ning võimaldada nende mõlema kasutamine pilditöötlusklassi Image poolt. Oluline on, et oleks võimalik mõlema kaamera pilti eraldi küsida, sest tõenäoliselt ei ole tarvis samaaegselt töödelda mitme kaamera pilti. See kasutaks asjatult niigi piiratud arvutusjõudlust.

Kaamerate arvutiga ühendamise järjekorrast sõltub, et milline kaamera millise indeksiga süsteemi luuakse. Samas programmikoodi koha pealt on oluline, et erinevad kaamerad oleks identifitseeritavad ning neile saaks õiget pilditöötlust rakendada. Muidu võib juhtuda, et näiteks tahavaate kaamerale rakendatakse eesmise kaamera pilditöötlust.

Probleemi lahendamine on jõudis töö käigus nii kaugele, et erinevad kaamerad on tuvastatavad *udev* reeglite abil, millest räägib järgmine peatükk. Robovisioni poole pealt on autoril olemas lahenduse idee, kuid see on alles teostusjärgus. Testimisel esines probleeme mitme pildi töötlemisega, mille lahendamisega ei ole autor veel tegeleda jõudnud.

2.3.4. Kaameratele udev reeglite loomine

Selleks, et ühendatud kaameraid teineteisest eristada tuleb nende ühendamisel luua nimeviidad, mis viitavad alati õigele kaamerale. Näiteks roboti ees olevale kaamerale luua viit eesmine (*/dev/eesmine*) ning tagaosa kaamerale tagumine (*/dev/tagumine*). Järgnevalt

tulebki juttu, kuidas vastavad reeglid seadistada. Antud näites kasutatakse kahte Logitechi veebikaamerat. Erinevate kaamerate puhul tuleb tootja ja mudeli unikaalsed tunnuskoodid vastavalt kohandada.

Luua ning avada fail tekstiredaktoriga (näiteks Nano):

```
sudo nano /lib/udev/rules.d/70-usb-camera.rules
```

Lisada järgnev sisu (kahel real), salvestada ning sulgeda fail:

```
KERNEL=="video*", ATTRS{idVendor}=="046d", ATTRS{idProduct}=="0990",
SYMLINK+="eesmine"
KERNEL=="video*", ATTRS{idVendor}=="046d", ATTRS{idProduct}=="080f",
SYMLINK+="tagumine"
```

Kasutatud reegli selgituseks:

KERNEL	võrdleb seadme nime süsteemi kernelis
ATTRS{idVendor}	võrdleb seadme tootja unikaalset tunnuskoodi
ATTRS{idProduct}	võrdleb seadme mudeli unikaalset tunnuskoodi
SYMLINK	loob seadmele alternatiivse nime (nimeviit)

Tootja ja mudeli unikaalsed tunnuskoodid leiab näiteks peale seadme arvutiga ühendamist, kui jooksutada terminali aknas käsk *dmesg*. Antud identifitseerimise lahendus töötab vaid juhul, kui kasutatavad kaamerad on teineteisest erinevad mudelid.

Taaskäivitada udev teenus:

```
sudo service udev restart
```

Nüüd võib juhtarvutiga ühendada kaks videokaamerat, mis peaksid loodud nimeviitade abil olema teineteisest eristatavad.

Sisestada käsk:

```
ls -lrt /dev/
```

Viimati ühendatud seadmete hulgas peaks olema */dev/video0, /dev/video1* kui ka viidad nendele nimedega */dev/eesmine* ja */dev/tagumine*.

Täiendused programmis

Kuna *OpenCV* võimaldab kaamera avamiseks kasutada ainult süsteemi poolt loodud */dev/video** indeksit, siis eelnevalt kaamerate indeksi tuvastamiseks lõi autor funktsiooni, mis küsib udev reegli abil loodud nimeviidalt tema sihtfaili nime. Selle abil on võimalik kätte saada soovitud kaamera number.

2.4. Mitmete töö käigus avastatud vigade parandamine

2.4.1. Konfiguratsiooniparameetrite lugemine

Muudatuste käigus avastas autor, et Robovisionis ei tööta korrektselt robotite individuaalsetest *.ini* failidest parameetrite lugemine. Sama probleem esines käsurealt parameetrite etteandmisel.

Lähemalt uurides seisnes probleem selles, et sisseloetavate muutujate väärtustamisel oli tehtud loogikaviga – kõigepealt loeti sisse käsurealt antud väärtused ning seejärel kirjutati need üle failist loetud väärtustega. Tegelikult peaks olema vastupidi, sest faili sisu on püsivam kui käsurealt antud väärtused, mille abil on mugav kiiresti mõnda muudatust testida.

Ühtlasi esines failist lugemisel vigu. Osasid parameetreid ei loetudki failist sisse. Samas mõned kordusid, mistõttu neid väärtustati mitmeid kordi üle.

Autor parandas antud peatükis toodud vead ning täiendas käsurealt antavaid korraldusi.

2.4.2. Funktsioonide tagastusväärtused

Robovisionis esines mitmel pool int tüüpi funktsioone, mille tagastusväärtus oli nii õnnestumise kui ka ebaõnnestumine korral 0. Selline stiil on eksitav, sest tagastusväärtusest ei ilmnenud, kui funktsiooni töös tekkis mingi viga.

Algselt muutis autor tagastusväärtusi täisarvude positiivses suunas, kuid ka see tekitas segadust – polnud aru saada, mis võib olla veakood ning mis tegelik anduri tagastusväärtus.

Seega sai funktsioone muudetud selliselt, et erinevate vigade korral tagastatakse vastavaid negatiivseid väärtusi.

2.5. Dokumentatsiooni tulemus

Käesoleva diplomitöö raames lõi autor Robovisioni tähtsamatele klassidele dokumentatsiooni ning täiendas ITK Robootikaklubi wiki Robovisioni artiklit. Wiki on eestikeelne, kuid lähtekoodi dokumentatsioon on inglise keeles, sest kasutatud muutujate, meetodite ja klasside nimed on kõik inglise keelsed. Kood, mis oleks sisaldanud kahte erinevat keelt, oleks vaid segadust tekitanud. Samuti võib sellest abi olla tulevikus, kui ITK Robootikaklubi plaanib oma robotitega minna välisvõistlustele, kus tihtipeale on tarvis enda robotite lähtekoodi tutvustada.

Töö käigus dokumenteeriti järgnevad klassid:

- AbstractRobot
- Camera
- Comm
- Config
- Image

Antud klassid on Robovisioni ühed tähtsamad komponendid. Dokumentatsioon genereeriti Doxygen vahenditega, mis lisaks kirjeldatud klassidele võimaldab luua ka klassidiagramme ning sõltuvuste graafe. Doxygeni poolt genereeritud diagrammide alusel lõi autor käesoleva töö lisas number 2 oleva joonise, millel on kujutatud robotite pärinemine AbstractRobot klassist ning Robovisioni erinevate klassidevaheliste objektide kasutamine.

2.6. Diplomitöö tulemus

Käesoleva diplomitöö raames tehtud töö tulemusena on kaks probleemi kolmest on leidnud lahenduse - viimasega autor veel tegeleb.

Mitme mikrokontrolleri lahendus on testitud arvutiga, mille külge oli ühendatud kaks *USB-RS232* üleminekut koos mikrokontrolleritega. Robovision oli võimeline suhtlema mõlema mikrokontrolleriga. Tänu sellele on ITK robotitele võimalik lisada täiendav mikrokontroller, mis lahendab analoogsisendite ja -väljundite väikesest arvust tuleneva probleemi. Tagasiulatuv funktsionaalsus on testitud robotil Digipallur.

Mitme kaamera toe lisamine osutus oodatust keerulisemaks ning on veel pooleli. Ettevõetud projekt oli küllaltki suur ja ajanappuse tõttu ei ole autor jõudnud antud probleemi veel lahendada.

Dokumentatsioon on sisuliselt valmis ning kasutatav. Kuna kaamera toe lisamine on alles pooleli, siis tuleb selles osas veel muudatusi. Tänaseks päevaks on loodud dokumentatsiooni kasutatud juba uute robootikaklubi liikmete kaasamiseks tulevasse Robovisioni arendusse.

2.7. Edasine tegevus

2.7.1. Mitme kaamera toe lisamine

Esimese asjana on plaanis lõpule viia kõikide käesolevas töös püstitatud probleemide lahendamine. Nimelt jäi pooleli praeguse roboti liigselt piiratud vaatenurga laiendamine. Lahendusena pakkus autor teise kaamera lisamist, kuid selleks oli vaja Robovisionis teha muudatusi, mis osutusid arvatust raskemaks. Programmi töös tekkisid tõrked, mille kõrvaldamisega autor veel tegeleb. Analoogsete platvormide järgi otsustades on probleem kindlasti lahendatav. Arvestama peab ka muudatustega, mis võivad kaasneda järgnevas lõigus 2.7.2. kirjeldatud tegevustega.

2.7.2. Integreerimine teise projektiga

Robovisioni arendamiseks tehti 2010. aastal samaaegselt kaks projekti. Lisaks käesolevale diplomitööle valmis Mikk Pärast-i lõputöö teemal „Võrgutoe lisamine robotite pilditöötlusraamistikule“, mille käigus lisati Robovisionile võrgutugi. See töö keskendus kuva- ja juhtmooduli (View klass) viimisele roboti juhtarvutist väljapoole, et lisaks juhtarvutile oleks võimalik jälgida roboti tööd ka üle võrgu teisest arvutist. Nende kahe projekti integreerimisel võib olla tarvis teha täiendusi ka käesolevas töös muudetavatele roboti kaamera- ja pilditöötlusmoodulitele. [23]

Nende kahe projektiga seoses ilmnes ITK Robootikaklubis kasutatava versioonihaldustarkvara SVN üks puudus. Nimelt ei võimalda SVN paralleelselt arendada ühe projekti erinevaid funktsionaalsusharusid ja sellepärast on loodud kaks erinevat projekti, mis tuleb hiljem integreerida. See on problemaatiline ning seetõttu on ITK Robootikaklubis

otsustatud üle minna Git nimelisele hajus-versioonihaldustarkvarale, mis taolist arendust paremini toetab.

2.7.3. Tutvustav seminar

Käesoleva töö valmimise ajal ei ole ITK Robootikaklubi liikmetele veel Robovisioni uut funktsionaalsust tutvustatud. Selleks on plaanis korraldada tutvustav seminar, kus autor tutvustab Robotite ehitajatele ja programmeerijatele uusi võimalusi. Kaasnema peaks väike demonstratsioon ning praktiline harjutus. See seminar on mõttekas korraldada alles sügisel, kui Robotex 2010 ITK võistkonnad on selgunud.

2.7.4. Ilmnevate puuduste likvideerimine

Sügisel, kui ITK robootikud hakkavad lahendama käesoleva aasta ülesannet, siis võib loodud funktsionaalsuse juures ilmneda uusi probleeme, mis tuleb jooksvalt kõrvaldada. Praeguses olukorras on raske ennustada probleeme, mis uutel robotitel esineda võivad. Kuna käesoleva töö autor plaanib ka edaspidi ITK Robootikaklubi liikmeks jääda, siis on tal võimalik ise jätkuvalt Robovisioni arendusega tegeleda.

Kokkuvõte

Käesoleva diplomitöö tulemusena lisati Eesti Infotehnoloogia Kolledži robotite juhtimisraamistikule mitme mikrokontrolleri tugi. **Lõputöös lahendatav probleem oli robotite piiratud funktsionaalsuse laiendamine, mis seisnes suurema arvu mikrokontrollerite ning kaamerate kasutusvõimaluse loomises.**

Ühest küljest tulenes probleem ITK Robootikaklubis kasutatavate mikrokontrollerite väikesest väljaviikude arvust, mistõttu ei olnud võimalik protsessoriplaadile rohkem andureid ega täitureid ühendada. Teisest küljest aga roboti juhtimisplatvormi Robovision piirangutest, võimaldades samaaegselt kasutada ainult ühte mikrokontrollerit ja kaamerat.

Töö käigus uuriti võimalikke lahendusi ning analüüsiti raamistiku olemasolevate moodulite omavahelisi seoseid, millest selgus, et need olid halvasti dokumenteeritud. Diplomitöö raames lisas autor vajaliku funktsionaalsuse teise mikrokontrolleri kasutamiseks, jättes sealjuures muutmata olemasolevad liidesed. Teostuse käigus avastas autor lähtekoodis uusi programmivigu, mis töö raames ära parandati.

Programmeerimise osa ei olnud kuigi mahukas, kuid tuli koostada hulgaliselt dokumentatsiooni, leidmaks minimaalne vajalik muudetavate klasside ja meetodite hulk, et pakkuda uut funktsionaalsust, samas säilitades olemasolevate robotite programmikoodiga ühilduvus liidese tasemel.

Lõputöö ülesande püstitusel määratleti kolm eesmärki:

- mitme mikrokontrolleri toe lisamine;
- mitme kaamera toe lisamine;

- koodi parandamine / dokumenteerimine.

Kõik probleemid peale mitme kaamera toe said lõputöö käigus lahenduse.

Uute liikmete kaasamine Robovisioni pilditöötlusplatvormi arendusse on tänu kaasaegsele dokumentatsioonile nüüd tunduvalt lihtsam.

Autor loodab, et 2010. aasta Robotexil võistlevatel ITK robotitel ei ole andurite ja täiturite ühendamisega probleeme, nagu oli möödunud aastal.

Summary

The purpose of this thesis was to solve the problem of expandability regarding the Estonian Information Technology College (ITC) robot control platform called Robovision. The problem was caused by the necessity to use more sensors, actuators and cameras than Robovision enabled. This paper offered a solution to add support for additional micro-controllers and cameras.

On the one hand, the problem originated from the small amount of input/output (I/O) pins on the micro-controllers used by the ITC Robotics Club. These micro-controllers limit the number of supported I/O devices to a smaller amount than required for building an effective robot. On the other hand, the origin of this problem lies within the framework's limit to support only one micro-controller and camera at the same time.

This thesis researched possible solutions and analyzed the module interfaces of the initial framework. The analysis showed that the framework was mostly undocumented. As a part of the thesis, the author provided the necessary functionality required to connect the secondary micro-controller. During the implementation the author discovered several other minor problems, which were corrected as well.

The programming during the implementation was not extensive, however the author had to document the framework in order to find the minimal required amount of changes to offer new functionality. In addition, the initial interfaces had to remain unchanged for backward compatibility with existing robots.

The thesis had three objectives regarding Robovision:

- add support for multiple micro-controllers;

- add support for multiple cameras;
- document the source code and correct any newly found problems.

This thesis offered a solution to all the problems above except for multiple-camera support.

With an updated documentation it is easier to introduce Robovision and its development to new members of the Robotics Club.

The author hopes that ITC robots competing in Robotex 2010 can connect additional sensors and actuators without any problems.

Kasutatud materjalid

1. Atmel Corporation. (10.05.2010). Atmel Corporation - Industry Leader in the Design and Manufacture of Advanced Semiconductors. [http://www.atmel.com/corporate/?source=main_nav]
2. Baillie, J-C. (11.05.2010). URBI: Towards a Universal Robotic Low-Level Programming Language. [http://cogrob.ensta.fr/papers/iros05-baillie.pdf]
3. Camerona R.G., Zhao, J. (05.05.2010). A communication method for an industrial multiple-microcontroller distributed control system. [http://linkinghub.elsevier.com/retrieve/pii/096706619391878Z]
4. CLARAty. (11.05.2010). CLARAty Robotic Software. [http://claraty.jpl.nasa.gov/man/overview/index.php]
5. Department of Computer Science, University of Copenhagen. (11.05.2010). ERSP. [http://image.diku.dk/mediawiki/index.php/ERSP]
6. Eesti Infotehnoloogia Kolledži Robootikaklubi. (09.05.2010). Põhikiri. [http://robot.itcollege.ee/?q=node/3]
7. Ernits, M. (16.05.2010). ROBOTEX OPEN G1 ja G2 Aruanne. [http://enos.itcollege.ee/~mernits/robot/]
8. Evolution Robotics. (11.05.2010). ERSP 3.1 Robotic Development Platform OEM Software. [http://www.evolution.com/products/ersp/]
9. Free Software Foundation, Inc.. (11.05.2010). The GNU operating system. [http://www.gnu.org/]
10. FreshPatents.com. (09.05.2010). Autonomous multi-microcontroller system and the control method thereof. [http://www.freshpatents.com/Autonomous-multi-microcontroller-system-and-the-control-method-thereof-dt20070920ptan20070220234.php]
11. ITK Robootikaklubi. (07.05.2010). ServoBasic kontroller. [http://robot.itcollege.ee/wiki/index.php/ServoBasic_kontroller]
12. ITK Robootikaklubi. (12.05.2010). Udev rule usb serial. [http://robot.itcollege.ee/wiki/index.php/Udev_rule_usb_serial]

13. Meeskond Digipallur. (11.05.2010). Digipallur. [http://robot.itcollege.ee/wiki/index.php/Digipallur]
14. Meeskond Madistajad. (12.05.2010). Madistajad. [http://robot.itcollege.ee/wiki/index.php/Madistajad]
15. Microsoft Developer Network. (11.05.2010). Microsoft® Robotics Developer Studio 2008 Overview. [http://msdn.microsoft.com/library/bb483024]
16. Microsoft Developer Network. (09.05.2010). Need Help with Mobile Robotics Platform. [http://social.msdn.microsoft.com/Forums/en-US/roboticstroubleshooting/thread/34bcbdee-8be8-4822-acfa-6b239631b2dc]
17. Nokia Corporation. (05.05.2010). Qt - A cross-platform application and UI framework. [http://qt.nokia.com/products]
18. Oomipood. (05.05.2010). ATmega. [http://oomipood.ee/?t=k_o&otsi=ATmega]
19. OpenCV Community. (05.05.2010). OpenCV Wiki. [http://opencv.willowgarage.com/wiki/Welcome]
20. Orca Robotics. (11.05.2010). Orca: Components for Robotics. [http://orca-robotics.sourceforge.net/]
21. Orocos. (11.05.2010). Orocos Component Library. [http://www.orocos.org/stable/documentation/ocl/v1.10.x/api/html/index.html]
22. Orocos Project. (18.05.2010). About the Orocos Project. [http://www.orocos.org/orocos/whatis]
23. Pärast, M. (2010). Võrgutoe lisamine robotite pilditöötlusraamistikule. Tallinna Polütehnikum, Eesti.
24. Robotex. (09.05.2010). Robotex. [http://www.robotex.ee/]
25. Robotex. (10.05.2010). Robotex 2010 võistlusülesanne. [http://www.robotex.ee/et/node/341]
26. Sikorski, K. (11.05.2010). Playing with the Player Project. [http://www.linuxjournal.com/magazine/playing-player-project?page=0,0]
27. Soetens, P. (11.05.2010). Lock-Free Data Exchange for Real-Time Applications. [http://people.mech.kuleuven.be/~psoetens/doc/Lock-Free-FOSDEM.pdf]
28. Starlino. (09.05.2010). OpenCv Stereo Vision Software.

[http://www.youtube.com/watch?v=WHBMXzBPZS4]

29. Starlino. (16.05.2010). Stereo Vision with OpenCV and QT. [http://www.starlino.com/opencv_qt_stereovision.html]
30. TTÜ Biorobootika Keskus. (16.05.2010). Robootika erikursus. [http://www.biorobotics.ttu.ee/tikiwiki/tiki-index.php?page=Robootika%20erikursus]
31. Urbi Forge. (11.05.2010). Urbi Forge Main/Home Page. [http://www.urbiforge.com/index.php/Main/HomePage]
32. Vallaste, H. (11.05.2010). e-Teatmik: IT ja sidetehnika seletav sõnaraamat. [http://www.vallaste.ee/index.htm]
33. van Heesch, D. (06.05.2010). Doxygen. [http://www.stack.nl/~dimitri/doxygen/]
34. VEX Forum. (09.05.2010). Cennecting multiple microcontrollers. [http://www.vexforum.com/showthread.php?t=17251]
35. Wikipedia. (02.05.2010). Comparison of documentation generators. [http://en.wikipedia.org/wiki/Comparison_of_documentation_generators]

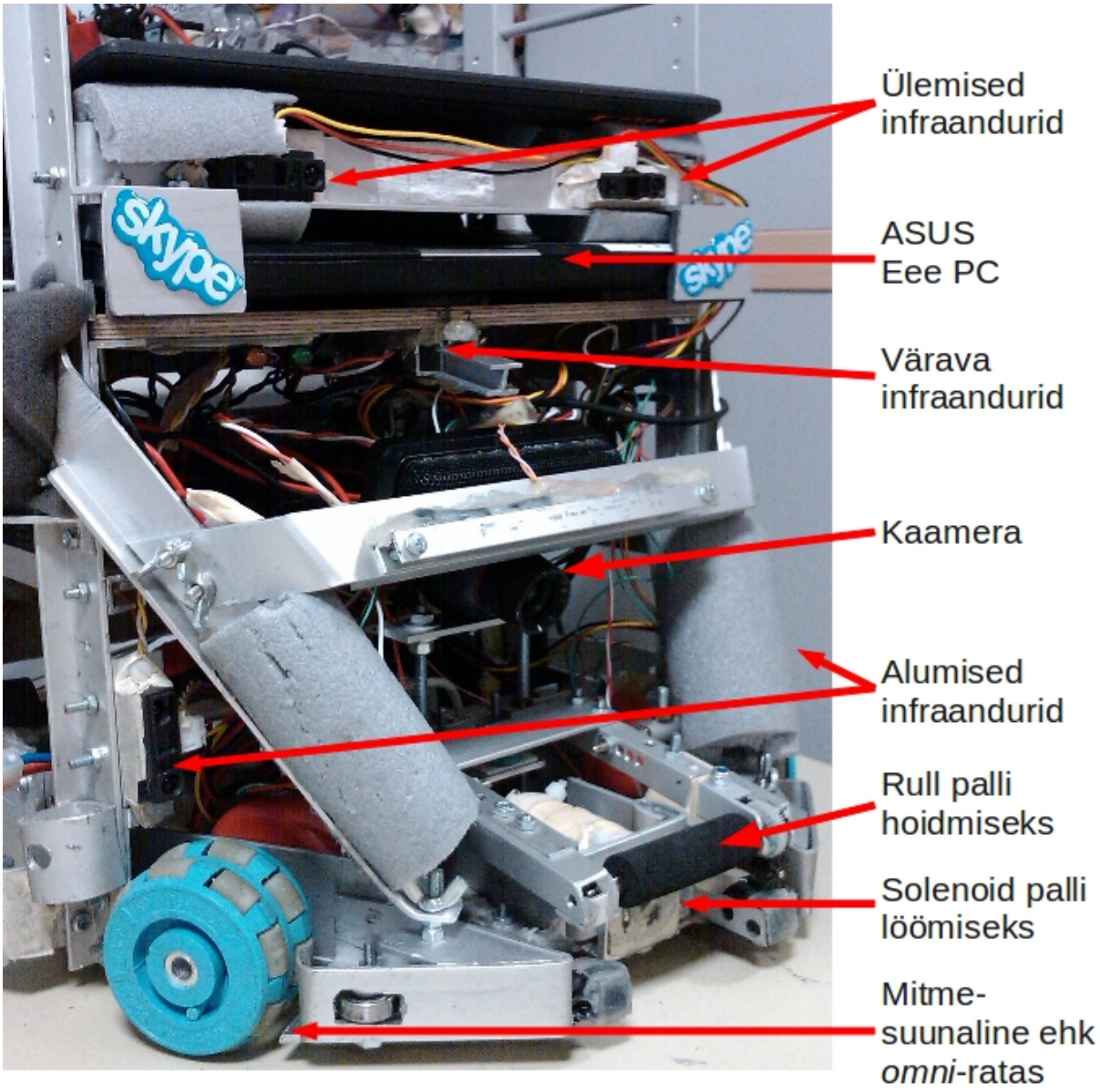
Lisad

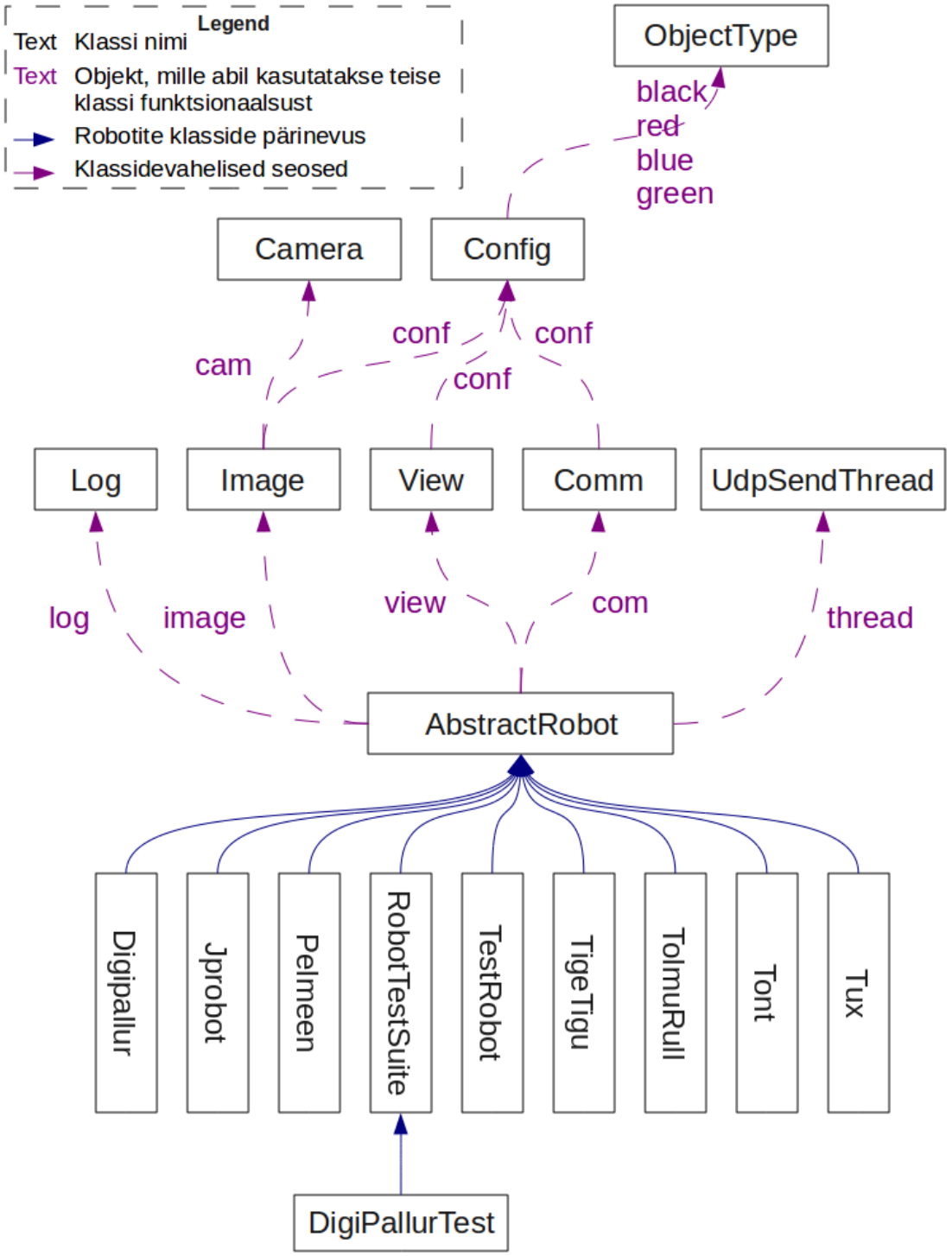
Lisa 1 ITK roboti põhielemendid Troller-Rolleri näitel

Lisa 2 Robovisioni ülesehitus Doxygeni põhjal

Lisa 3 CD Robovisioni lähtekoodi ja dokumentatsiooniga

Lisa 1 ITK roboti põhielemendid Troller-Rolleri näitel




Legend
Text Klassi nimi
Text Objekt, mille abil kasutatakse teise klassi funktsionaalsust
Robotite klasside pärinevus
Klassidevahelised seosed
ObjectType
black
red
blue
green
Camera
Config
cam
conf
conf
conf
Log
Image
View
Comm
UdpSendThread
log
image
view
com
thread
AbstractRobot
Digipallur
Jprobot
Pelmeen
RobotTestSuite
TestRobot
TigeTigu
TolmuRull
Tont
Tux
DigiPallurTest