

Emmi Katila

KONENÄKÖOHJATTU YHTEISTYÖROBOTTI -
HAVAINNOINTIESITYS

Kone- ja tuotantotekniikan koulutusohjelma
2018

KONENÄKÖOHJATTU YHTEISTYÖROBOTTI -HAVAINNOINTIESITYS

Katila, Emmi
Satakunnan ammattikorkeakoulu
Kone- ja tuotantotekniikan koulutusohjelma
Toukokuu 2018
Sivumäärä: 24
Liitteitä: 2

Asiasanat: konenäkö, robotiikka, automaatio

Opinnäytetyön aiheena oli suunnitella ja toteuttaa havainnointiesitys siitä, miten konenäköjärjestelmä ohjaa yhteistyörobottia toimimaan juomatarjoilijana. Esitys kehitettiin osoittamaan, miten konenäöllä voidaan tunnistaa kappaleita ja miten konenäöllä ohjataan robotin liikkeitä.

Tässä työssä käydään läpi perinteistä konenäköjärjestelmää, robotiikkaa ja havainnointiesityksen tekoa. Konenäkö toteutettiin perinteisellä konenäköjärjestelmällä, joka koostuu värikamerasta ja linssistä. Kuvan analysointi toteutettiin MvTec Halcon -konenäköohjelmistolla. Analysointi koostui segmentoinnista ja hahmontunnistuksesta. Robottina oli Universal Robotsin UR5 -yhteistyörobotti, jota ohjelmoitiin graafisella Polyscope käyttöliittymällä. Robotin ohjelmointiin kuului paikkapisteiden opettaminen, liikekäskyjen valinta ja yhteyden muodostus konenäköjärjestelmään.

Valmis havainnointiesitys pystyy poimimaan 0,33 litraisia kierrätysmuovipulloja, joissa on valkoiset korkit. Tulevaisuudessa, säätämällä konenäköohjelman valintoja ja robotin paikkapisteitä, voidaan korista nostaa myös muunlaisia pulloja.

MACHINE VISION GUIDED ROBOT -DEMONSTRATION

Katila, Emmi

Satakunnan ammattikorkeakoulu, Satakunta University of Applied Sciences

Degree Programme in mechanical and production engineering

May 2018

Number of pages: 24

Appendices: 2

Keywords: machine vision, robotics, automation

The purpose of this thesis was to make a demonstration with machine vision and collaborative robot. Demonstration was made to show how machine vision can identify objects and demonstrate how machine vision can guide robot.

Thesis presents the principles of machine vision and robotics. The planning of machine vision robot demonstration is presented and explained. Machine vision system consisted of color camera and optics. Image analysis was made with MvTec Halcon machine vision software. Analysis consisted of segmentation and matching. Universal Robots UR5 collaboration robot was part of demonstration and it was programmed with Polyscope graphic interface. Programming consisted of teaching waypoints choosing the motion commands and creating the data transfer.

Finished demonstration can pick 0,33liter plastic bottles that have white bottle caps. In the future it's possible to pick different kind of bottles by adjusting the machine vision system and reprogramming the robot's waypoints.

SISÄLLYS

| | | |
|-------|--|----|
| 1 | JOHDANTO..... | 5 |
| 2 | KONENÄKÖ | 6 |
| 2.1 | Konenäköjärjestelmän osat | 6 |
| 2.2 | Analysointi konenäköjärjestelmässä | 8 |
| 3 | ROBOTIT..... | 10 |
| 3.1 | Yhteistyörobotit | 11 |
| 3.2 | Konenäköohjatut robotit | 12 |
| 4 | JUOMATARJOILIJAHAVAINNOINTIESITYKSEN LAITTEET..... | 14 |
| 4.1 | Universal Robots -yhteistyörobotti | 14 |
| 4.1.1 | Työkalu | 14 |
| 4.1.2 | Käyttöjärjestelmä..... | 15 |
| 4.1.3 | Ohjelmointi | 15 |
| 4.2 | Konenäköjärjestelmä..... | 16 |
| 4.2.1 | Konenäköjärjestelmän laitteet | 16 |
| 4.2.2 | Ohjelmointi | 17 |
| 5 | KONENÄKÖJÄRJESTELMÄN JA ROBOTIN YHDISTÄMINEN | 18 |
| 5.1 | Kalibrointi | 18 |
| 5.2 | Yhteyden luominen ja kommunikointi | 18 |
| 5.3 | Konenäköjärjestelmän toteuttaminen..... | 19 |
| 5.4 | Robotin ohjelman toteuttaminen..... | 20 |
| 6 | YHTEENVETO | 22 |
| | LÄHTEET..... | 23 |
| | LIITTEET | |

1 JOHDANTO

Opinnäytetyön aiheena oli suunnitella ja toteuttaa havainnointiesitys siitä, miten konenäköjärjestelmä ohjaa yhteistyörobottia toimimaan juomatarjoilijana. Opinnäytetyön konenäkö- ja robotiikkaosiossa hyödynnettiin jo valmiina olevia ohjelmistoja ja laitteita.

Havainnointiesitykseen valmistettiin automaattisesti toimivat pullonavaaja ja mukien annostelija. Satakunnan ammattikorkeakoulun Robotiikka Akatemia valmisti nämä laitteet.

Konenäköjärjestelmän osina käytettiin IDS uEye UI-5260CP-C-HQ -värikameraa, optiikkana Kowa, LM12JC10M-linssiä, ohjelmistona MVTec Halcon -konenäköohjelmaa ja robottina käytettiin Universal Robots UR5 -yhteistyörobottia. Robotin ja konenäköjärjestelmän välille muodostettiin kommunikaatioyhteys TCP/IP-protokollaa käyttäen.

Havainnointiesitys toimii niin, että robotti hakee mukeit annostelijasta ja asettaa ne valmiiksi paikoilleen. Mukien haun jälkeen robotti hakee korista pullon. Robottikäsivarten asennetulla kameralla katsotaan, missä kohtaa koria pullo on. Pulloa käytetään pullonavaajassa ja tämän jälkeen juoma kaadetaan mukeihin. Pullon kuljetukseen, käyttämiseen avaajassa ja juoman kaatamiseen mukeista ei tarvita konenäköä, koska mukien ja avaajan paikat ovat aina samat.

Tässä opinnäytetyössä suunniteltiin konenäköohjelma, joka tunnistaa pulloja korista. Tehtävänä oli kuvien analysointi Mvtech Halcon -ohjelmalla. Opinnäytetyön toisena tehtävänä oli saada ohjattua robottia konenäön opastuksella. Konenäköjärjestelmä ohjaa robottia tunnistamalla pullojen korkkeja ja lähettämällä niiden sijaintitiedot robotille. Annetulla sijaintitiedolla robotti osaa poimia pullon.

2 KONENÄKÖ

Konenäöllä voidaan tarkoittaa kameralla tapahtuvaa esineiden tunnistusta, paikan havaitsemista ja värien tunnistusta. Konenäkö käsittää kaikki teolliset ja ei-teolliset sovellutukset, joissa yhdistetään laitteisto ja ohjelmisto kuvanottoon ja sen analysoimiseen. Se, mitä konenäkö analysoi, on ohjelmoitu etukäteen. (Cognex 2018).

Konenäkö voidaan hoitaa perinteisellä konenäköjärjestelmällä tai älykameralla. Perinteisessä konenäköjärjestelmässä osina voivat olla kamera, optiikka, tietokone & ohjelmisto, valaistus ja liitännät. Perinteisessä järjestelmässä kamera ottaa kuvan kohteesta ja lähettää sen tietokoneelle, jossa kuva analysoidaan konenäköohjelmalla, josta tulokset lähetetään niitä hyödyntäville järjestelmän osille. Älykamerassa nämä kaikki osat voivat olla integroituna itse kameraan. (SAMK, Automaation tutkimusryhmän Internetsivut 2018).

Tarkan kuvan ja tulosten saamiseksi, on oikea valaistus edellytys. Valaistus voi muuttaa kuvauksen olosuhteita hyvinkin paljon. Valaistuksen tavoitteena on valaista kappale yksinkertaisesti, mutta tarpeeksi hyvin, jotta siitä saadaan haluttu tieto, joten valaistuksen valinnassa täytyy huomioida kuvaamisen vaatimukset ja olosuhteet. (Voutilainen 2014; Lempiäinen 2011).

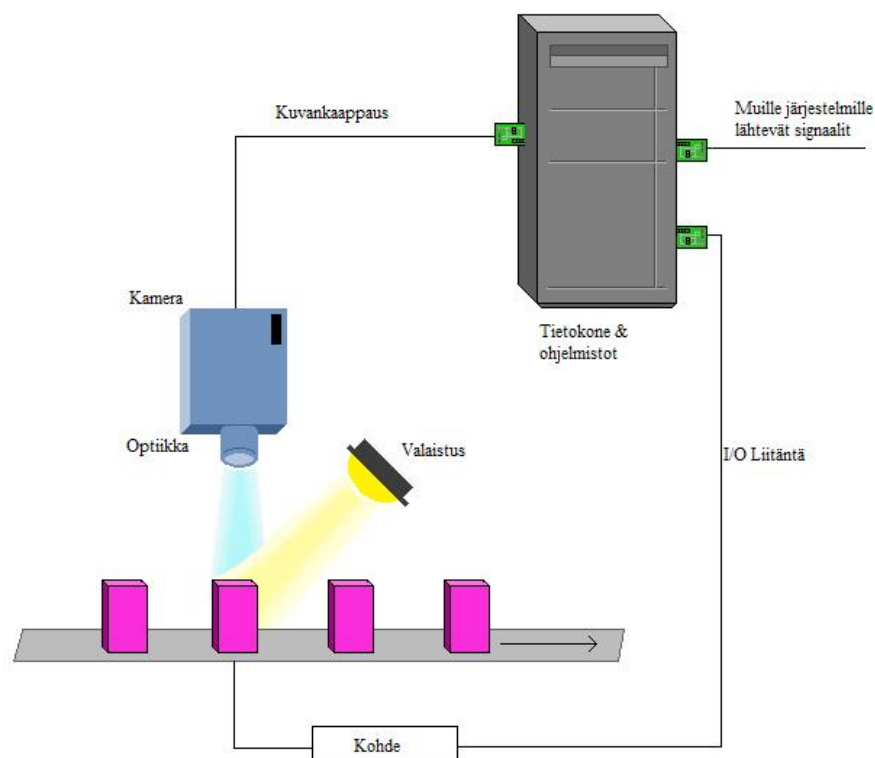
2.1 Konenäköjärjestelmän osat

Perinteiseen konenäköjärjestelmään kuuluu kamera, jolla kuva otetaan. Perinteisesti kamera voi olla matriisi- tai viivakamera. Matriisikameralla tarkoitetaan kameraa, jolla saadaan suorakulmionmuotoinen kuva, jossa pikseleitä on matriisimuodossa eli joku määrä vaakasuunnassa ja joku määrä pystysuunnassa. Viivakamera muodostaa toiseen suuntaan ohuen, yhdestä kolmeen pikselin levyisen kaistaleen ja toiseen suuntaan pitkän pikselijonon eli kuva on viivan muotoinen. Peräkkäin kuvattuja viivoja yhteen liittämällä saadaan kuva suorakaiteen muotoinen kuva kohteesta. (SAMK, Automaation tutkimusryhmän Internetsivut 2018).

Kamerassa on kiinni optiikka eli linssi. Optiikan tehtävänä on kerätä kohteesta heijastuneet valonsäteet kameran kennolle. Optiikassa on yleensä yksi tai kaksi säätörenkasta, joista voidaan säätää aukon kokoa sekä tarkennusetaisyys. Myös optiikan laatu vaikuttaa lopullisen kuvan laatuun yhdessä kameran kanssa.

Valaistuksella on suuri vaikutus konenäköjärjestelmän toimivuuteen. Valaistuksella tuotetaan kohteeseen sellainen valaistus, jolla kohteen kiinnostavat osat saadaan näkyväksi kuvassa hyvin. Vastaavasti valaistuksella voidaan poistaa häiritseviä heijastuksia. Valaistuksen tulee olla aina samanlainen kuvaustilanteessa, sillä pienetkin muutokset valaistuksessa voivat aiheuttaa suuria muutoksia kuvaan. Valaistuksena voi olla taustavalo, kappale voidaan tuoda valonlähteen päälle, jolloin sen reunat näkyvät hyvin. Salamavalo on käytännöllinen, kun kappale liikkuu kovaa vauhtia. Valaistus voi olla myös epäsuoraa tai suoraa. Suorassa valaistuksessa syntyy teräviä varjoja ja heijastuksia. Epäsuora valaistus heijastetaan jonkin pinnan kautta, jolloin kappaleeseen syntyy tasaisempi valo ja varjot pehmenevät. (Lempiäinen 2011).

Kuva lähetetään kamerasta tietokoneelle analysoitavaksi. Kuvan analysointi tapahtuu konenäköohjelmistolla, johon on laadittu jollekin tietylle sovellutukselle analysointiohjelma. Kun kuva on analysoitu, ohjelma lähettää tulokset eteenpäin. Perinteisen konenäköjärjestelmän periaate on esitetty kuvassa 1 (SAMK, Automaation tutkimusryhmän Internetsivut 2018; Metropolia n.d.).



Kuva 1. Perinteinen konenäköjärjestelmä (SAMK, Automaation tutkimusryhmän Internetsivut 2018).

2.2 Analysointi konenäköjärjestelmässä

Kuva-analyysissä kuvasta etsitään oleellista tietoa. Kuva-analyysi on kuvan havainnointia, kokonaisuuden pilkkomista osiin ja tulkintaa. Kuva-analyysin helpottamiseksi konenäköjärjestelmään tulee luoda hyvä valaistus ja suunnitella muutkin kuvausjärjestelyt huolella. Kuva voidaan muokata sellaiseen muotoon, josta sitä on helpompi analysoida. Kuvan esikäsittelyssä suodatetaan kohinaa tai muita sävyvaihteluja. Kuva-analyysin vaiheina voivat olla esimerkiksi segmentointi ja hahmontunnistus. (Pietikäinen & Silven 2011, 5).

Kuva-analyysi alkaa usein segmentoinnilla. Segmentointi tarkoittaa kuvan jakamista osiin, esimerkiksi taustan ja esineen erottamista toisistaan. Harmaasävykuvassa voidaan havaita reunoja tunnistamalla rajapintoja, eli etsimällä alueita, joissa on kahta eri harmaasävyä vierekkäin. Halutut harmaasävyarvot löytyvät kynnystämällä. Kynnys-

tämisessä muodostetaan histogrammijakauma, josta nähdään intensiteettiipiikit. Intensiteettiipiikki syntyy histogrammiin, kun sävyarvo nousee suureksi verrattuna taustansa nähden, esimerkiksi vaalea kappale tummaa taustaa vasten muodostaa intensiteettiipiikin. Histogrammista valitaan arvo, jonka toisella puolella arvot ovat vaalean kappaleen ja toisella puolla tumman taustan, näin on saatu valittua kynnysarvo. (Jauhiainen 2006, 43-44, 47).

Alueen segmentoinnilla tunnistetaan samankaltaiset pikselit yhtenäisiksi alueiksi. Tällä tavoin voidaan esimerkiksi erottaa vaaleat kappaleet tummasta taustasta. Segmentoinnin tulos riippuu määritellyistä kynnysarvoista. Saatu segmentoitu alue voidaan edelleen jakaa osiin, jolloin esimerkiksi erisävyisiä vaaleita alueita määritellään omiksi alueikseen. Päinvastoin myös alueiden yhdistäminen on mahdollista. (Jauhiainen 2006, 48).

Kuvan segmentointi ei ole pakollista hahmontunnistuksessa, mutta se helpottaa työtä. Analysoinnilla yleensä halutaan tunnistaa tai tulkita asioita. Kohteiden tunnistaminen on kuitenkin vaikeaa, jos ei ole ennalta määritelty, miltä jokin näyttää. Hahmontunnistukseen on monenlaisia menetelmiä ja oikean menetelmän valinta perustuu siihen, minkälaista muotoa haetaan. Joissain konenäköohjelmissa on määriteltynä erilaisia muotoja, joihin kuvasta löydettyjä alueita voidaan verrata. Muoto voidaan myös hakea referenssikuvasta. Joissain menetelmissä muotoa haetaan väriarvojen eroavaisuuksien perusteella, joissain reunaviivan muodon perusteella. (Jauhiainen 2006, 48; MvTec Halcon 2017, 7).

3 ROBOTIT

Robotti on mekaaninen yleiskäyttöinen laite, joka pystyy suorittamaan erilaisia tehtäviä tietokoneohjauksella. Robotit voidaan ohjelmoida erilaisiin tehtäviin ja samaa robottia voidaan käyttää moniin käyttötarkoituksiin. Robotteja on niin teollisia kuin ei-teollisiakin. Robotit voivat olla ihmisen kokoisia ja kantaa suuriakin taakkoja ilman ongelmia. Robotit myös toistavat hyvinkin varmasti samaa liikettä yhä uudelleen. Robotteja voidaan sijoittaa sellaisiin paikkoihin, jotka ovat ihmiselle vaarallisia. Ei-teolliset robotit ovat palvelurobotteja, jotka tekevät tehtäviä ihmisen hyväksi. Palvelurobotteja on kehitetty ammattilaisten ja kotitalouksien käyttöön. Kotitalouksien palvelurobotit voivat olla muun muassa pölynimureita ja ikkunanpesijöitä. (Hooper 2014; Salmi 2014).

Teollisuusrobotilla tarkoitetaan robottia, jolla on vähintään kolme liikeakselia ja yksi työkalu. Teollisuusrobotteja ovat muun muassa scara-, cartesian- ja käsivarsirobotti. Käsivarsirobotissa on yleensä 6 liikkuvaa niveltä, joten se voi liikkua vapaasti ulottuvuusalueellaan. Robotti pystyy yleensä liikkumaan x-, y- ja z-suunnassa. Jos robotilla on enemmän kuin viisi akselia, voi se omalla ulottuvuusalueellaan liikkua mihin tahansa pisteeseen missä tahansa asennossa. Kuvassa 2 on kuvattuna käsivarsirobotteja. (Hooper 2014).



Kuva 2. Käsivarsirobotteja (Panasonic Industry Europe GmbH 2018).

Ensimmäiset teollisuusrobotit kehitettiin korvaamaan ihmisiä vaarallisissa, yksitoikkisissa ja raskaissa työtehtävissä. Roboteissa ei ollut turva-antureita, joten ne olivat yleensä turva-aidan sisällä, johon ihminen ei päässyt robotin työskennellessä. Robotit suorittivat yksinkertaisia nosta ja aseta -tehtäviä. Robotteja kuitenkin kehitettiin ja niihin asetettiin antureita ja ne ohjelmoitiin tekemään monimutkaisempia tehtäviä. Ensimmäisten robottien ohjelmointi oli hyvin monimutkaista verrattuna tämän päivän kehittyneisiin ja yksinkertaisempiin ohjelmointiympäristöihin. Aluksi robotteja käytettiin pääsääntöisesti maalauksessa, hitsauksessa ja kokoonpanossa. (Wallén 2008, 6-8).

Nykyään teollisuusrobottien työtehtävät ovat lähes rajoittamattomat. Teollisuusrobotit voivat työskennellä itsenäisesti tai ihmisen kanssa. Toimivassa järjestelmässä robotti hoitaa raskaat työt ja ihminen ohjelmoi robotin ja huolehtii häiriönpoistoista ja huollosta. Työn ruumiillinen kuormittavuus on näin vähentynyt. Jos ihminen toimii robotin kanssa samassa tilassa, ihmisen turvallisuus täytyy taata. Yhtenä keinona voi käyttää mekaanisia esteitä, turva-aitoja ja muita kiinteitä rakenteita. Turvallisuus voidaan saavuttaa myös henkilön paikanseuranta-antureilla, robotin asentoantureilla ja keskuslogiikalla, joka seuraa antureita. Paikanseuranta-anturina voi olla muun muassa turvamatto tai valoverho. Paikanseuranta voidaan hoitaa myös turvakameroilla. Nämä havaitsevat robotin alueella liikkuvan ihmisen ja lähettävät tiedon keskuslogiikalle. Logiikasta riippuen robotti voi hidastaa liikettään, pysähtyä tai lopettaa toimintansa kokonaan suorittamalla hätäseis-komennon. (Robotiikka yleinen 2016, 79, 85, 90-92).

3.1 Yhteistyörobotit

Yhteistyörobotti voi työskennellä yhdessä ihmisen kanssa. Yhteistyörobotin ympärille ei tarvitse rakentaa erillistä turva-aitaa, vaan se on ohjelmoitu pysähtymään, jos se törmää esteeseen tai havaitsee olevansa törmäämässä esteeseen. Yhteistyörobotti voi tunnistaa esteeseen törmäämisen esimerkiksi moottorin virrankulutuksen lisääntymisestä. Yhteistyörobotti voidaan myös varustaa muilla turvatekniikoilla. Nämä tekniikat ovat älykkäitä ja havainnoivia. Robotti voi havaita ihmisen muun muassa erilaisten anturien avulla, RFID-paikannuksella, laserskannerilla, ultraäänitutkalla tai kokenäöllä. Osa yhteistyöroboteista työskentelee suurimmaksi osaksi itsenäisesti, mutta

toisinaan ihminen saattaa työskennellä sen toiminta-alueella. Yhteistyörobottien nostovoimat ovat rajalliset verrattuna isoihin teollisuusrobotteihin, tosin tavallinen teollisuusrobotti voidaan muuntaa yhteistyörobotiksi lisäämällä siihen tarvittavat turvajärjestelmät. (Bélanger-Barette 2015; Shikany 2014: Robotiikka yleinen 2016, 90-97).

Voimarajoitetulla robotilla on voima-anturit nivelissään, joten se tuntee siihen kohdistuvat voimat. Kun robotti havaitsee siihen kohdistuvia odottamattomia voimia, se voi joko pysäyttää itsensä tai palata johonkin ennalta määrättyyn asentoon. Voima-anturien ansiosta robottia voi ohjelmoida niin sanotusti käsin. Robotista voi ottaa kiinni ja liikuttaa sen haluttuun asentoon tai näyttää sille polun, jota pitkin kulkea. Tällaisten robottien ohjelmointi on suhteellisen helppoa ja nopeaa. (Bélanger-Barette 2015).

3.2 Konenäköohjatut robotit

Robotista saadaan älykkäämpi, kun sen ohjaukseen yhdistetään esimerkiksi konenäköä. Konenäköohjattu robotti on dynaaminen ja adaptiivinen. Myös näiden robottien konenäköjärjestelmässä niin kuin perinteisessä konenäköjärjestelmässä pitää ottaa huomioon kameran ja optiikan valinta, valaistus, optiikka, tiedonsiirto ja kuvan analysointi. (Dechow 2018). Kuvassa 3 on esitettynä robotti, jonka ohjaukseen käytetään konenäköjärjestelmää.



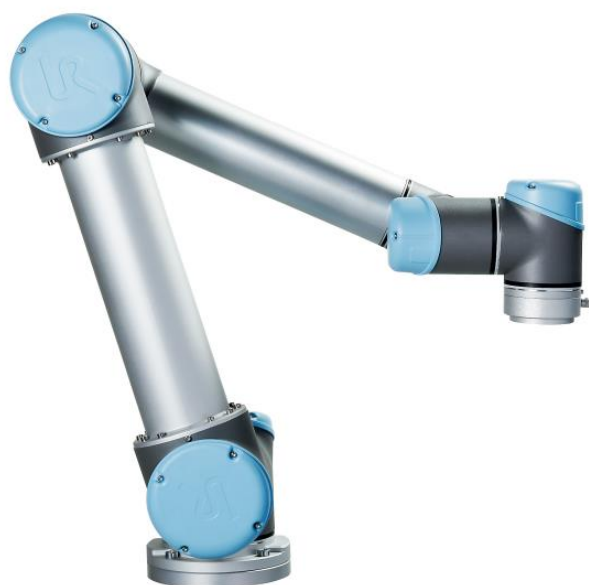
Kuva 3. Konenäköohjattu robotti (Keyence 2018).

Kamera voidaan kalibroida robotin kanssa samaan koordinaatistoon ja tämän myötä kamera voi antaa robotille tietoja sen ympäristöstä. Tiedoilla robotti voi liikkua haluttuun paikkaan tai toteuttaa komentoja. Konenäköohjattu robotti tekee automaatiosta joustavampaa ja havaitsevampaa. Kommunikatio robotin ja konenäköohjelman välillä on välttämätön. (Dechow 2018).

4 JUOMATARJOILIIJA-HAVAINNOINTIESITYKSEN LAITTEET

4.1 Universal Robots -yhteistyörobotti

Tässä työssä käytettävä robotti on Universal Robots UR5 -yhteistyörobotti. Robotti on kuuden vapausasteen käsivarsirobotti ja sen runko on valmistettu pääasiassa alumiiniputkista ja nivelistä. UR5:n kantokyky on 5 kg ja sen ulottuvuus on 850 mm. Kuvassa 4 robotti on esitettynä. (Universal Robots 2014, 77).

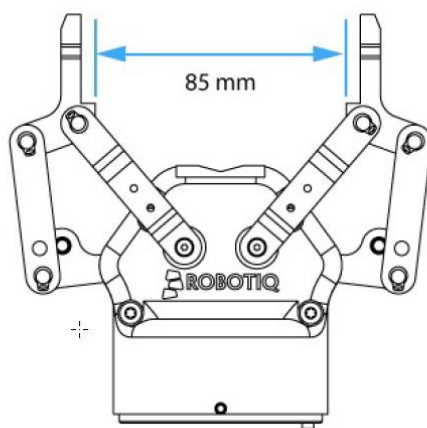


Kuva 4. UR5-käsivarsirobotti (Universal Robots 2014, 9).

4.1.1 Työkalu

Robotissa on kiinni Robotiq 2-finger 85 -sormitarttuja, joka on esitetty kuvassa 5. Tarttujassa on kaksi sormeaa, joissa kummassakin on kaksi niveltä. Tarttuja pystyy pitämään kappaleesta kiinni viidellä tartuntapinnalla. Tarttuja painaa 850 g ja sillä pystytään poimimaan 5 kilogramman painoisia kappaleita. (Robotiq 2018. 7-8,122).

2-FINGER 85



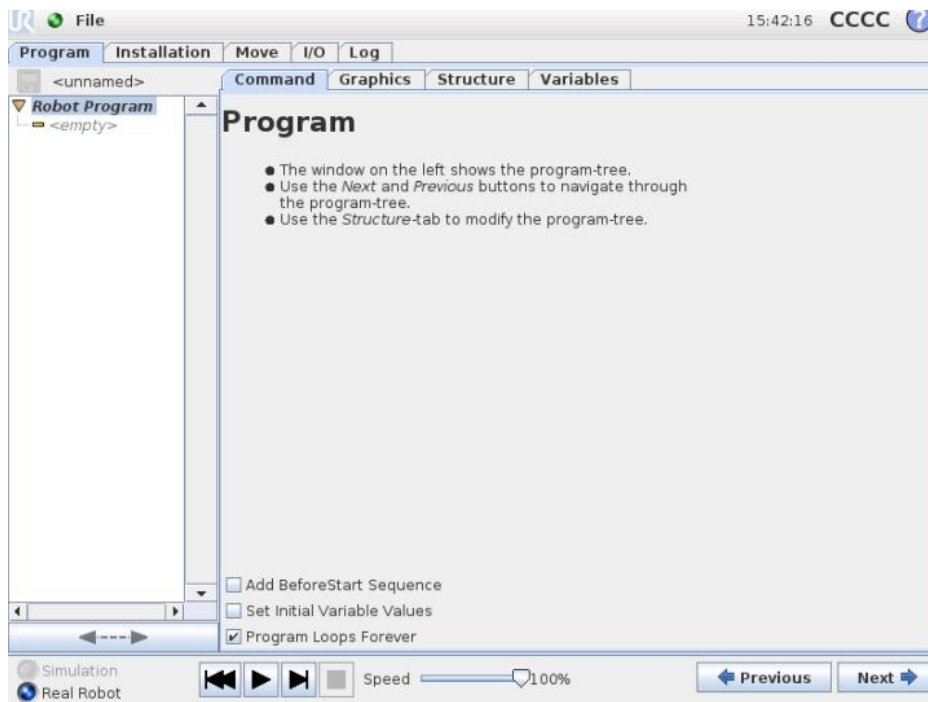
Kuva 5. Sormitarttuja (Robotiq 2018, 8).

4.1.2 Käyttöjärjestelmä

Polyscope on Universal Robotsin graafinen käyttöliittymä, jolla operoidaan robottia ja ohjelmia sekä luodaan ohjelmia. Polyscope toimii kosketusnäytöllä, joka on kytketty ohjauslaatikkoon. Käyttöliittymän ikkunarakenne on porrastettu ja se koostuu valikoista, valintanapeista ja kuvakkeista. Polyscopeen voidaan lisäksi liittää näppäimistö ja hiiri, mutta se ei ole tarpeellista, koska osoitinta ja näppäimistöä voidaan käyttää myös kosketusnäytöltä. (Universal 2014, 75).

4.1.3 Ohjelmointi

Ohjelmointi toimii niin sanotusti ikkunoiden kautta. Ohjelmointiympäristössä ikkunat on järjestelty välilehtien tapaan. Program-välilehdellä on näkyvissä sen hetkinen ohjelma. Vasemmassa reunassa näkyy ohjelmakomennot ja oikealla on ohjelmointiympäristö. Kuvassa 6 näkyy UR5-robotin ohjelmoinnin käyttöliittymä.



Kuva 6. Universal Robotsin ohjelmointi-ikkuna (Universal Robots 2014, 112).

Robotin liikkeitä pystyy ohjaamaan joko kosketusnäytöltä painikkeilla ja tietoja syöttämällä tai ns. freedrive –tilassa eli käsin robottia liikuttamalla. Näytössä näkyy robotin sen hetkinen sijainti ja suunniteltu sijainti haamukuvana, kun robottia ei ohjata käsin.

4.2 Konenäköjärjestelmä

4.2.1 Konenäköjärjestelmän laitteet

Käytössä on IDS uEye UI-5260CP-C-HQ -värikamera. Kamera on matriisikamera CMOS-kennolla. Kameran resoluutio on 2.35 MPix. Pikselit ovat jakautuneet muotoon 1936 x 1216 pikseliä ja yhden pikselin koko kennolla on 5.9 μm . Kameran CMOS-kennon mitat ovat 11,345mm x 7,126mm. Kamera painaa 50 grammaa ja se on kytketty Ethernet-johdolla tietokoneeseen. (IDS-imaging 2018).

Optiikkana käytetään Kowan LM12JC10M-linssiä. Käytettävän linssin polttoväli on 12 mm ja sen suurin aukkokoko on F/1.8. Kameran kennon ja polttovälin perusteella voidaan laskea, että kuva-alaksi muodostuu 472,7mm x 296,9mm, kun kuvataan 50cm

etäisyydellä kohteesta. Järjestelmään ei luotu erillistä valaistusta, vaan valaistus on luokkahuoneen kattovalojen varassa.

4.2.2 Ohjelmointi

Halcon on MVTechin ohjelmisto konenäköjärjestelmän analysointitehtäviin. Halconilla tehdään kuva-analyysejä ja luodaan konenäköohjelmia. Halconissa itsessään on laaja kirjasto, josta voi hakea eri komentoja ja valmiita ohjelmia. Halconilla voi muun muassa tunnistaa muotoja, mitata etäisyyksiä, segmentoida ja käsitellä 3D-malleja. (MvTec Halcon 2017, 9, 52).

5 KONENÄKÖJÄRJESTELMÄN JA ROBOTIN YHDISTÄMINEN

Konenäköjärjestelmän ja robotin välille on luotava yhteys, jotta niiden kommunikaatio toimii. Yhteyden luomisen jälkeen konenäköjärjestelmällä saadut arvot lähetetään robotille ja robotti liikkuu niiden perusteella määritettyyn paikkaan.

5.1 Kalibrointi

Koska robotti toimii omassa koordinaatistossaan, joka on esitetty millimetreissä ja kamera laskee etäisyyksiä pikseleiden mukaan, on kamera kalibroitava siten, että sen koordinaatit ovat maailmakoordinaatistossa. Kalibrointi tapahtui ottamalla ensin kuva tietyistä kohdasta ja merkitsemällä kameran sijainti, eli robotin asema, muistiin. Kuvasta valittiin kaksi pistettä, joiden pikselikoordinaatit kirjattiin myös muistiin. Tämän jälkeen robotti ajettiin molempien valittujen pikselipisteiden kohdalle ja katsottiin robotin maailmakoordinaatit. Saatujen tietojen avulla pystyttiin laskemaan, että 1 mm vastaa noin 6,2 pikseliä. Kalibroinnin tiedoilla konenäköohjelman käyttämä pikselikoordinaatisto voitiin muuttaa robotin käyttämään maailmakoordinaatistoon.

5.2 Yhteyden luominen ja kommunikointi

Jotta kommunikaatio toimisi, on robotin ja kameran välille luotava yhteys. Tässä sovelluksessa käytettiin TCP- (transmission control protocol) ja IP- (internet protocol) tietoverkkoprotokollien yhdistelmää.

Kun kaksi konetta käyttävät samaa protokollaa, voidaan niiden välille luoda kommunikointiyhteys. TCP-protokolla luo yhteydet sovellusten välille. IP-protokolla hoitaa viestin saapumisesta oikeaan osoitteeseen.

Konenäköohjelmassa käytettäviä muuttujia ovat Data, BaseProtocol, Timeout, AcceptingSocket, OpenStatus, Error ja Answer. Data on se muuttuja, jonka tiedot lähetetään

robotille, kun halutaan robotin liikkuvan. Timeout-muuttujan arvoksi asetetaan 3 sekuntia, joka on se aika, kuinka kauan yhteydenottoa robotille yritetään. Answer-muuttujasta luetaan robotilta tuleva tieto.

Kun yhteyttä avataan, pitää ensin avata mahdollisuus yhteydenottoon. Yhteydenavauksen mahdollistamisessa määriteltiin käytettävä tiedonsiirtoprotokolla ja yhteyden avaamisen odotusaika sekä portti. Odotusajaksi asetettiin kolme sekuntia eli yhteydenottopyyntöä tarkastetaan kolme sekuntia kerrallaan, jonka jälkeen yritetään muodostaa yhteys uudelleen. Tästä vaiheesta päästään eteenpäin, kun yhteydenotto onnistuu. Ohjelmankierto pysäytetään, jos yhteyttä yritetään muodostaa jollakin muulla kuin TCP- tai HALCON-protokollalla.

Yhteydenmuodostamisen jälkeen voidaan vastaanottaa viestejä. Robotilta otetaan vastaan viesti, jonka arvo tallennetaan muuttujaan Answer. Jos muuttujan Answer arvo on 'Uusikuva', kamera ottaa kuvan. Jos Answer on jotain muuta kuin 'Uusikuva', vastaanotetaan robotilta uudestaan arvo muuttujalle Answer. Tätä tapahtuu niin kauan, kunnes Answer:n arvo on 'end'.

Jos kuvasta löytyy korkkeja, ohjelma valitsee yhden korkin. Valitun korkin keskipiste pikselikoordinaatistossa muutetaan robotin käyttämään maailmakoordinaatistoon. Pullonkorkin keskipisteen koordinaatti on siis (roboX,roboY) -muodossa. Saatu koordinaatti asetetaan Datan arvoksi. Jos kuvasta ei löydy pullonkorkkia, Datan arvoksi määritellään (1000,1000). Lopuksi, kun robotin IP-osoite ja portti on saatu määriteltyä, voidaan muuttujan Data tiedot lähettää robotille. Datan lähettämisen jälkeen sen arvot tyhjennetään.

5.3 Konenäköjärjestelmän toteuttaminen

Konenäköohjelma toteutettiin ottamalla kameralla kohteesta kuva ja analysoimalla sitä halutun tiedon löytämiseksi. Yhteys kameraan avataan erillisellä komennolla. Komennon sisään on kirjattu myös osoite, mistä haetaan tiedosto, johon on säädetty muun muassa valkotasapainoarvot.

Ensiksi kuva segmentoidaan korkkien erottamiseksi taustastaan. Otetaan kuva ja muutetaan se harmaasävykuvaksi. Harmaasävykuvasta on helpompi erotella alueita, koska on vain harmaan arvoja, joita rajataan. Koska pullonkorkit näkyvät kuvassa valkoisena, valitaan threshold-komennolla vaaleat alueet kuvasta, eli valitaan korkeat lukuarvot kuvasta. Metallinen pöytä aiheutti sen verran heijastumia, että pullojen alle laitettiin mustaksi maalattu vanerilevy. Näin heijastumista päästiin eroon.

Korkit eivät kumminkaan vielä ole omia alueitaan vaan kaikki vaaleat kohdat kuvasta mielletään ohjelmassa yhdeksi alueeksi. Alueet erotettiin toisistaan connection-komennolla. Koska korkkeja on monta, alueet rajataan erillisiksi alueiksi. Tämä ei kuitenkaan riitä, koska taustasta tulee mukaan muitakin alueita kuin pullonkorkkeja. Select_shape -komennolla jo valituista alueista valitaan tietyn kokoisia ja muotoisia alueita. Pinta-alaksi valitaan 16000-27000 pikseliä ja muodoksi valitaan tietty pyöreys. Näin saadaan erotettua kuvasta vain halutut kappaleet eli pullonkorkit. Saaduista erillisistä alueista etsitään keskikohdat. Keskikohdat halutaan tietää, koska niistä saa hyvin selville pullonkorkin paikan, joka voidaan muuntaa robotin koordinaatistoon. Nämä koordinaatit lähetetään robotille.

5.4 Robotin ohjelman toteuttaminen

Samoin kuin konenäköohjelmalla luotiin yhteys robotille, myös robotin ohjelmalla luodaan yhteys konenäköohjelmalle. Robotille kerrotaan tietokoneen IP-osoite, johon se ottaa yhteyttä kommunikoidakseen konenäköjärjestelmän kanssa.

Ohjelman alussa robotti ohjataan ensimmäiseen kuvauspisteeseen ja avataan tarttuja. Robotilta lähetetään viestin 'Uusikuva' konenäköohjelmalle ja kamera ottaa kuvan.

Robotin tarvitsee vastaanottaa konenäköjärjestelmältä paikkakoordinaatti, jossa juomapullo on XY -tasokoordinaatistossa, Z-taso on valmiiksi määriteltä, koska juomapullojen korkeus on aina sama. Pullonkorkin sijainti lähetetään koordinaatteina robotin käyttämässä koordinaatistossa.

Jos konenäköjärjestelmä ei havaitse ensimmäisessä kuvauspaikassa pulloa, ohjelma lähettää robotille datan '(1000,1000)'. Juuri tämän kyseisen viestin saatuaan robotti menee toiseen kuvauspaikkaan ja lähettää taas viestin 'Uusikuva'. Robotti liikkuu kuvauspaikkojen välillä niin kauan, että konenäköohjelma havaitsee pullon.

Pullonkorkin koordinaattien saamisen jälkeen robotti menee konenäköjärjestelmältä saamaansa pisteeseen. Tämän jälkeen robotti toteuttaa komennon `move variable`, eli robotti suorittaa liiku-komennon juuri siitä kohdasta halutun matkan alemmas. Robotti menee kohtisuoraan alaspäin ja poimii pullon laittamalla tarttujansa kiinni-asentoon.

Eli konenäköjärjestelmällä on ohjattu robotti poimimaan pullo antamalla robotille pullon koordinaattipisteet. Tämän jälkeen robotti on mennyt suoraan pullon yläpuolelle ja poiminut pullon. Robotti saa myös tiedon, jos pulloa ei ole ensimmäisessä kuvauspisteessä ja robotti siirtyy toiseen kuvauspisteeseen, josta konenäköjärjestelmä ottaa uuden kuvan. Robotti poimii pulloja konenäön ohjauksessa.

6 YHTEENVETO

Opinnäytetyö tehtiin Satakunnan ammattikorkeakoululle. Opinnäytetyössä suunniteltiin ja toteutettiin konenäköohjelma, joka ohjaa yhteistyörobotin toimintaa. Ohjelma yhdistettiin havainnointiesitetykseen. Esityksessä robotti hakee mikit annostelijasta ja asettaa ne paikalleen. Mukien paikoilleen laittamisen jälkeen robotti menee ennalta määrättyyn paikkaan ja kamera ottaa kuvan. Konenäön avulla robotti tunnistaa pullon ja hakee sen korista, käy avaamassa sen pullonavaajassa ja kaataa juomaa mukeihin. Tämän jälkeen pullo käydään viemässä pois.

Satakunnan ammattikorkeakoulun Robotiikka Akatemia valmisti mukin annostelijan ja pullon avaajan, jotka asennettiin kiinni samaan pöytään, jossa robotti on.

Konenäköjärjestelmän toteuttamisessa käytettävien menetelmien valinnassa annettiin varsin vapaat kädet tietyin reunaehdoin. Ohjelmien opettelu tapahtui labratyöharjoituksen ja harjoitusvideoiden avulla.

Konenäköjärjestelmä koostui IDS uEye UI-5260CP-C-HQ –värikamerasta ja Kowa, LM12JC10M-linssistä. Konenäköohjelmointi tapahtui MvTec Halcon -ohjelmalla, jolla analysoitiin otettu kuva ja luotiin yhteys Universal Robots UR5 -yhteistyörobotille.

Konenäköohjelmassa ongelmaksi muodostui metallisesta pöydästä johtuvat heijastumat, jotka näkyivät pullokorin läpi ja vieressä. Heijastumat aiheuttivat virheellisiä tuloksia ohjelmassa. Valkeat heijastumat näyttivät ohjelmassa samalta kuin valkoiset korkit. Ongelma korjattiin maalaamalla vanerilevy mustaksi ja laittamalla se kiinteästi kiinni pullokorin alle.

Tässä järjestelmässä robotti pystyy poimimaan vain tietynlaisia pulloja, koska konenäköohjelma tunnistaa pullon korkin sen perusteella, että korkki on valkoinen. Myös robottiin asetettiin pullon korkeus vakioksi. Jatkossa on kuitenkin suhteellisen helppoa ohjelmoida järjestelmä tunnistamaan erivärisiä pullonkorkkeja vaihtamalla konenäkö tunnistamaan korkki esimerkiksi jonkin tietyn värin perusteella.

LÄHTEET

Bélanger-Barette, M. 'What does collaborative robot mean?'. Robotiq. 19.8.2015. Viitattu 3.5.2018. <https://blog.robotiq.com/what-does-collaborative-robot-mean>

Cognex. 2018. What is machine vision. Viitattu 2.5.2018. <https://www.cognex.com/what-is/machine-vision/what-is-machine-vision>

Dechow, D. 2018. Latest innovations in vision guided robotics. Viitattu 18.5.2018. <https://www.visiononline.org/webinar-detail.cfm/webinars/latest-innovations-in-vision-guided-robotics/id/56>

Hooper, R. 2014. Learn About Robots. Industrial Robots. Viitattu 2.5.2018. <http://www.learnaboutrobots.com/industrial.htm>

IDS-imaging internet sivut. 2018. Viitattu 18.5.2018. www.ids-imaging.com

Jauhainen, J. 2006. Digitaalinen kuvankäsittely. Oulun seudun ammattikorkeakoulu. Viitattu 16.5.2018. <http://www.oamk.fi/~jjauhiai/opetus/DIP/kuvankasittely2.pdf>

Keyence internet sivut. 2018. Vision-guided robotics User Support Site. Viitattu 18.5.2018. <https://www.keyence.com/landing/globalrobotvision.jsp>

Konenäkö-Machine vision. n.d. Metropolia. Viitattu 16.5.2018. <https://wiki.metropolia.fi/download/attachments/11637671/Konenako.pdf>

Lempiäinen. 'Konenäkö tutuksi viikossa: 4/5 Valaistus'. Konenäkö. 17.12.2011. Viitattu 21.5.2018. <http://konenako.blogspot.fi/2011/12/konenako-tutuksi-viikossa-45-valaistus.html>

Mitchell, B. 2018. Understanding transmission control protocol and internet protocol. Lifewire. Viitattu 7.5.2018. <https://www.lifewire.com/transmission-control-protocol-and-internet-protocol-816255>

MvTec Halcon. 2017. Solution Guide II-b matching, Version 17.12

Panasonic Industry Europe GmbH internet sivut. 2018. viitattu 18.5.2018. <https://eu.industrial.panasonic.com/>

Pietikäinen, M & Silven, O. 2011. Konenäkö. Viitattu 21.5.2018. <http://www.cse.oulu.fi/wsgi/CMV/AboutCMV?action=AttachFile&do=get&target=konenako.pdf>

Robotiq. 2016. 2-Finger adaptive robot gripper instruction manual Revision 2017/06/06.

Robotiikka Yleinen (2016). 2016. Lahti: Lahden ammattikorkeakoulu. Automaatiotekniikan luentoja. Viitattu 2.5.2018. http://miniweb.lpt.fi/automaatio/opetus/luennot/pdf_tiedostot/Robotiikka_yleinen.pdf

Salmi, T. 2014. Robotiikka – monien mahdollisuuksien tekniikkaa. Vtt Impulssi. 31.12.2014. Viitattu 22.5.2018. <https://www.vtt.fi/Impulssi/Pages/Robotiikka-%E2%80%93-monien-mahdollisuuksien-tekniikkaa.aspx>

SAMK, Automaation tutkimusryhmän Internetsivut. 2018. Konenäkö. Viitattu 2.5.2018. http://automaatio.samk.fi/?page_id=16

Shikany, A. 2014. Collaborative Robots End User Industry Insights. Robotic Industries Association. Viitattu 3.5.2018. https://www.robotics.org/userassets/riauploads/file/RIA_Collaborative_Robots_White_Paper_October_2014.pdf

Universal Robots. 2014. User manual UR5/CB3 Version 3.0.

Voutilainen, P. 2004. Konenäkö. Opetushallitus. Viitattu 3.5.2018. <http://www03.edu.fi/oppimateriaalit/puutuoteteollisuus/automaatio/konenako/index.html>

Wallén, J. 2008. The history of industrial robots. Tekninen raportti. Linköpingsin Yliopisto.

HALCON-KONENÄKÖOHJELMA

```

*näyttää muuttujien arvot
dev_update_on()
*Esittää data nimisen muuttujan
Data := "
*avaa yhteyden kameralle ja hakee tiedoston, jossa määritellään kameran asetukset
open_framegrabber ('uEye', 1, 1, 0, 0, 0, 0, 'default', 8, 'default', -1, 'false', 'D:/Emmi-
nIniTiedostoMAXRES.ini', '9', 0, -1, AcqHandle)
*muuttujalle protocol asetaan arvo TCP4, jolla määritellään käytetty tiedonsiirtopro-
tokolla
    Protocol := 'TCP4'
*muuttujalle timeout asetetaan arvo 3 sekuntia, joka on aika, jonka ajan yhteydenot-
toa odotetaan
    Timeout := 3.0
    *Avaa mahdollisuuden muodostaa yhteyden porttiin 3000, muuttujaan accepting-
socket tallennetaan muuttujien Protocol ja Timeout tiedot
    open_socket_accept(3000, ['protocol','timeout'], [Protocol,Timeout], Accepting-
Socket)
    *Hakee muuttujasta protocol arvon, joka sisältää joko TCPn tai HALCONn, ja tal-
lentaa sen muuttujaan baseprotocol
    tuple_regexp_match(Protocol, 'TCP|HALCON', BaseProtocol)
    if (BaseProtocol == 'TCP' or BaseProtocol == 'HALCON')
        *Esittää muuttujan error, johon tallennetaan mahdolliset virheet
        dev_error_var(Error, 1)
        *Virheitä ei huomioida, jotta ohjelman suorittaminen ei lopu kesken, kun koite-
taan saada yhteys robottiin
        dev_set_check('~give_error')
        *Esittää muuttujan ja annetaan arvoksi 5
        OpenStatus := 5
        *niin kauan kunnes openstatus muuttuja on jotain muuta kuin 2
        while (OpenStatus != 2)
            *Tarkistaa, onko yhteydenottopyyntöä 3 sekuntia
            socket_accept_connect(AcceptingSocket, 'auto', Socket)
            *Kun errorin arvoksi tulee 2, yhteydenotto onnistui ja päästään ulos while-ko-
mennosta
            OpenStatus := Error
        endwhile
        *aktivoi virheilmoitukset
        dev_set_check('give_error')
        *varmistaa, että timeout arvo on 3. Eli jos yhteys katoaa, odotetaan 3 sekuntia
        set_socket_param(Socket, 'timeout', Timeout)
        *Mikäli ei käytetä tcp tai halcon protokollaa, pysäytetään ohjelmakierto
        else
            stop()
        endif
    *Hakee muuttujaan Address yhteydenottajan IP-osoiteen
    get_socket_param(Socket, 'address_info', Address)

```

```

*Esittää muuttujan Answer
Answer := []
*niin kauan kunnes answer on jotain muuta kuin end
while (Answer != 'End')
    *ottaa robotilta vastaan viestin, joka tallennetaan muuttujaan answer
    receive_data(Socket, ['z', 'A3'], Answer, From)
    *jos answer on 'Uusikuva', suoritetaan if
    if (Answer == 'UusiKuva')
        *suoritetaan while komentoa niin kauan kuin datassa ei ole arvoa
        while (Data = "")
            *ottaa kuvan
            grab_image (Image, AcqHandle)
            *muuttaa kuvan harmaasävykuvaksi
            rgb1_to_gray (Image, HarmaasavyKuva)
            *eroittaa harmaasävyarvot väliltä 100-245 omaksi alueeksi nimeltä korkit
            threshold (HarmaasavyKuva, Korkit, 100, 245)
            *Eroittaa alueet, jotka eivät koske toisiaan
            connection (Korkit, KorkitErikseen)
            *valitsee alueita pikselikoon perusteella
            select_shape (KorkitErikseen, PelkatKorkit, 'area', 'and', 16000, 27000)
            *täyttää kolot alueissa
            fill_up(PelkatKorkit,PelkatKorkitFilled)
            *valitsee vain tietyn pyöreiden omaavat alueet
            select_shape(PelkatKorkitFilled,PelkatKorkitFilled,'circulari-
ty','and',0.3,1.0)
            *etsii yksittäisten alueiden keskikohdat
            area_center (PelkatKorkitFilled, Area, Row, Column)
            *laskee erillisten alueiden lukumäärän
            count_obj(PelkatKorkitFilled,NumberOfKaikkiKorkit)
            *jos alueita, korkkeja, on enemmän kuin 0, suoritetaan if-komento
            if (NumberOfKaikkiKorkit > 0)
                *valitsee yhden pullon
                select_obj(PelkatKorkitFilled, ObjectSelected, 1)
                *määrittää pikselikoordinaatit robotin käyttämään koordinaatistoon
                TestRow := Row[0]-228
                TestColumn := Column[0]-309
                roboX := 192-(TestRow/6.13768)
                roboY := -296-(TestColumn/6.20398)
                *määrittää datan arvoksi koordinaattipisteen
                Data := '(' + roboX + ',' + roboY + ')'
                *jos korkkeja ei löydy suoritetaan else-komento
            else
                *datan arvoksi määritellään 1000,1000
                Data := '(1000,1000)'
                *data-tiedon 1000,1000 saatuaan, robotti siirtyy toiseen kuvauspisteeseen
            endif
        endwhile
    *jos answer on jotain muuta kuin 'Uusikuva', tyhjennetään datan arvo
    else
        Data := ""

```

```
endif
*määritellään muuttujaan 'to' yhteydenottajan ip-osoite ja portti
To := [From[0],From[1]]
*määritellään missä muodossa viesti lähetetään
Format := 'z'
*lähetetään robotille datan arvo
send_data(Socket, Format, Data, To)
*tyhjennetään data
Data := "
endwhile
*suljetaan yhteydet
close_socket(Socket)
close_socket(AcceptingSocket)
close_framegrabber (AcqHandle)
```

Universal Robotsin ohjelma pullojen haun osalta

```

Program
  Init Variables
  BeforeStart
    kuvauspaikka0
  MoveJ
    Waypoint_2
    uuskuva" False
    alkupose"get_actual_tcp_pose()
  Robot Program
    'MoveJ'
    Loop vastaus[0]0
      'socket_send_string("UusiKuva") '
      uuskuva" True
      Wait: 1.0
    If vastaus[1] 1000 and vastaus[2] 1000
      vastaus"[0,0,0]
      If kuvauspaikka0
        kuvauspaikka"1
        MoveJ
          Waypoint_5
      Else
        kuvauspaikka"0
        MoveJ
          Waypoint_4
    Else
      If kuvauspaikka0
        alkupose=p[vastaus[1]/1000,vastaus[2]/1000,0.211,2.22137,-
2.22151,0.0]
      Else
        alkupose=p[(vastaus[1]-200)/1000,vas-
taus[2]/1000,0.211,2.22137,-2.22151,0.0]
      Gripper Move40% (1)
      MoveJ
        alkupose
      Wait: 1.0
      MoveL
        Waypoint_1
      Wait: 1.0
      Gripper Close (1)
      Wait: 1.0
      MoveL
        Waypoint_3
      If kuvauspaikka0
        MoveJ
          Waypoint_2
      Else
        MoveJ
          Waypoint_6
  Thread 1
    var_1"socket_open("10.103.1.86",3000)
    If uuskuva True
      uuskuva" False
      socket_send_string("UusiKuva")
    Else
      socket_send_string("Odota")
    Wait: 0.5

```

```
Thread_2
    vastaus"socket_read_ascii_float(2)
```