jamk.fi

Mittaus- ja reunanohjausjärjestelmän käyttöliittymä

Anssi Hakala

Opinnäytetyö
Marraskuu 2017
Tekniikan ja liikenteen ala
Insinööri (AMK), automaatiotekniikan tutkinto-ohjelma

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Hakala, Anssi	Opinnäytetyö, AMK	11 2017
	Sivumäärä 55	Julkaisun kieli Suomi
		Verkkojulkaisulupa myönnetty: x

Työn nimi

Mittaus- ja reunanohjausjärjestelmän käyttöliittymä

Tutkinto-ohjelma

Insinööri (AMK), automaatiotekniikan tutkinto-ohjelma

Työn ohjaaja(t)

Ari Kuisma, Matti Siistonen

Toimeksiantaja(t)

Vision Systems Oy

Tiivistelmä

Vision Systems Oy on suunnitellut ja toimittanut useita mittaus- ja reunanohjausjärjestelmiä teollisuudessa käytettyihin metalli- ja paperikelaimiin. Tähän Vision Systemsin tarjoamaan pakettiin ei ole aikaisemmin kuulunut omaa käyttöliittymää ja siihen haluttiin muutos.

Tehtävänä oli siis suunnitella ja toteuttaa käyttöliittymä Vision Systemsin mittaus- ja reunanohjausjärjestelmään. Tavoitteena oli luoda käyttöliittymästä toimiva kokonaisuus, josta löytyy kaikki tarpeellinen sekä järjestelmän ohjauksen kannalta että järjestelmän huollon ja käyttöönoton kannalta.

Tarkoituksena oli aluksi selvittää, mitä eri keinoja tämän käyttöliittymän toteuttamiseen on, ja valita näistä paras. Toteutustavaksi valittiin Beckhoffin Twincat 3 -ohjelma. Tämän jälkeen kerättiin asiakkailta ja Vision Systemsin työntekijöiltä mielipiteitä siitä, mitä käyttöliittymässä tulisi olla ja tehtiin alustava suunnitelma käyttöliittymän rakenteesta ja ominaisuuksista.

Kun suunnitelma oli saatu aikaiseksi, tehtiin sen pohjalta itse käyttöliittymä. Käyttöliittymän toimivuus todettiin useilla testeillä, joissa yritettiin löytää ja korjata kaikki mahdolliset virheet. Joitain käyttöliittymän ominaisuuksia ei päästy testaamaan.

Lopputuloksena saatiin toimiva käyttöliittymä, joka sisältää suurimman osan työntekijöiden ja asiakkaiden toiveiden mukaisista ominaisuuksista. Käyttöliittymän täydellistä toimivuutta ei päästy testaamaan, mutta teoriassa ohjelmallisen osuuden tulisi kattaa kaikki järjestelmän toimintatilat.

Avainsanat (asiasanat)

PID-säätöpiiri, käyttöliittymä, suunnittelu, reunantunnistus, Twincat 3

Muut tiedot

jamk.fi

Description

Author(s)	Type of publication	Date
Hakala, Anssi	Bachelor's thesis	11 2017
		Language of publication: Finnish
	Number of pages: 55	Permission for web publication: x

Title of publication

Interface for measurement and edge guiding system

Degree programme

Electrical and Automation Engineering

Supervisor(s)

Kuisma Ari & Siistonen Matti

Assigned by

Vision Systems Oy

Abstract

Vision Systems Oy designs and delivers several measuring and edge guiding systems for industrial use aimed to be used in metal and paper coilers. This package offered by Vision Systems has not previously had its own interface.

Therefore, the task was to design and implement a user interface to Vision Systems' measurement and edge guiding system. The goal was to make the user interface a functional entity, which would have everything needed for both the system's control and for the system's maintenance and commissioning.

First, the idea was to figure out what different ways there are to design and create the interface and select the best one, which for this purpose was Beckhoff's Twincat 3 program. After this, opinions were gathered from the customers and Vision Systems' employees of what features the user interface should contain, after which an initial plan was completed of the structure and features of the user interface.

When the plan was completed, a user interface was created based on that plan. Its functionality was tested multiple times with various tests where an attempt was made to find and fix all possible mistakes. Some features of the user interface could not be tested.

The project resulted in a functional user interface including most of the features in accordance with the wishes of the customers and the employees. The complete functionality of the user interface has not yet been tested; however, in theory, the programmed part should cover all of the systems modes.

Keywords/tags (subjects)

PID control circuit, user interface, design, edge detection, Twincat 3

Miscellaneous

Sisältö

1 Johdanto 5
1.1 Opinnäytetyön toimeksianto ja tavoitteet 5
1.2 Toimeksiantaja 5
1.3 Tutkimusmenetelmät 6
2 Järjestelmän komponentit 6
2.1 Päälle- tai aukikelain 7
2.2 Proportionaaliventtiili 7
2.3 Lineaarianturi 8
2.4 VisiSmart-integroitu konenäköanturi 9
2.5 Beckhoff CP6606-paneeli-PC 10
2.6 Beckhoff-CX9020 logiikka 11
3 PID-säätöpiiri 12
3.1 Perusteet 12
3.2 Matemaattinen selitys 14
3.3 Säätimen viritys 15
4 Reunantunnistus ja mittausmenetelmät 18
4.1 Reunantunnistuksen perusteita 18
4.2 Järjestelmässä käytetyt mittausmenetelmät 21
5 Käyttöliittymä 23
5.1 Graafinen ja tekstipohjainen käyttöliittymä 23
5.2 Graafisen käyttöliittymän suunnittelu 25
5.3 Värien ja kontrastin käyttö 27
6 Käyttöliittymän toteutusvaihtoehdot 28
6.1 Twincat 3 -ohjelmisto 29
6.2 Indusoft Web Studio -ohjelma 29
6.3 Itse koodattu käyttöliittymä 31
6.4 Yhteenveto toteutusvaihtoehdon valinnasta 31

7 Työn toteutus ja ongelmat 32
7.1 Suunnittelu 32
7.2 Työn tekeminen vaiheittain 33
7.3 Ongelmat 37
8 Työn tulokset 40
9 Pohdinta 41
9.1 Tuotteen jatkokehitys 41
9.2 Luotettavuus 42
9.2.1 Lähteiden luotettavuus 42
9.2.2 Testausten luotettavuus 43
Lähteet 46
Liitteet 49
Liite 1. Ensimmäisiä luonnoksia järjestelmän käyttöliittymästä (1) 49
Liite 2. Ensimmäisiä luonnoksia järjestelmän käyttöliittymästä (2) 50
Liite 3. Columns-välilehti 51
Liite 4. System State -välilehti 52
Liite 5. Other Information- välilehti 53
Liite 6. Profibus IOs -välilehti 54
Liite 7. Guiding Selection -välilehti 55

Kuviot

Kuvio 1. Esimerkki metallirainan kelaimesta 7
Kuvio 2. Proportionaaliventtiili. 8
Kuvio 3. Magnetostriktiivisyyteen perustuvan lineaarianturin toimintaperiaate 9
Kuvio 4. Beckhoffin CP6606 -paneeli-PC 10
Kuvio 5. Beckhoffin CX9020 -logiikka 11
Kuvio 6. PID-säätimen ohjaama prosessi 13
Kuvio 7. Ideaalin PID-säätimen rakenne 14
Kuvio 8. Esimerkki askelvastekokeesta 17

Kuvio 9. Esimerkki Sobelin matriisin vaikutuksesta. Vasemmalla on alkuperäinen kuva ja oikealla Sobelin matriisilla muokattu kuva..... 19
Kuvio 10. Sobelin matriisi (vasemmalla), sekä matriisilla laskettu uusi pikselin väriarvo..... 20
Kuvio 11. Järjestelmässä käytettävä kolmiomittausperiaate. 22
Kuvio 12. Kappaleen korkeuden mittaaminen kolmiomittausperiaatteella. 22
Kuvio 13. Windowsin resurssienhallinta 24
Kuvio 14. Windowsin komentokehote..... 25
Kuvio 15. Esimerkki kontrastin vaikutuksesta..... 28

Taulukot

Taulukko 1. Viritysparametrien vaikutus prosessiin. 16
Taulukko 2. Viritysparametrien määritys prosessiparametrien avulla. 18

Käsitteet

Function block diagram	Graafinen ohjelmointikieli, jossa käytetään blockeja, eli eräänlaisia laatikoita, joille annetaan jokin tehtävä (esim. AND tai OR).
Ladder diagram	Graafinen ohjelmointikieli, joka on tarkoitettu PLC:iden ohjelmointiin.
Metalliraina	Pitkä, yhtenäinen metallilevy, jota kuljetetaan ja varastoidaan kelaksi kerättynä rullana. Näistä jatkojalostetaan erilaisia metallituotteita.
Offset	Poikkeama esimerkiksi keskikohdasta tai alkuperäisestä asemasta.
Referenssiarvo	Vertailuarvo eli arvo, jolla viitataan johonkin toiseen arvoon.
SCADA	Supervisory Control And Data Acquisition. Tunnetaan paremmin nimillä valvomo-ohjelmisto tai PC-valvomo.
Setpoint	Säätöpiirissä oleva asetusarvo, johon mitattava suure halutaan.
Tagi	Toimii samalla tavalla kuin muuttuja tavallisessa ohjelmassa, eli siihen voidaan kirjoittaa ja siitä voidaan lukea arvoja.

1 Johdanto

1.1 Opinnäytetyön toimeksianto ja tavoitteet

Opinnäytetyön tehtävänä on suunnitella, toteuttaa sekä testata mittaus- ja reunanohjausjärjestelmän käyttöliittymä. Työn toteutusvaiheet olivat: sopivan suunnitteluohjelman valinta, käyttöliittymän ulkonäön ja toimintojen suunnittelu, käyttöliittymän toteutus, testaus ja palautteen hankinta ja viimeiseksi hienosäätö ja palautteen mukaiset korjaukset sekä raportointi. Valmis käyttöliittymä soveltuu käytettävien metallirainan tai paperin kelaimien ohjaamiseen ja tarkkailuun.

Opinnäytetyön tavoitteena oli luoda täysin toimiva ja käyttäjäystävällinen käyttöliittymä, joka toimi Windows-pohjaisilla tietokoneilla sekä yleisimmillä nettiselaimilla. Työn aikana tuli myös ilmi, että käyttöliittymän haluttiin toimivan myös Beckhoffin omilla paneeli-PC:illä. Toimintojen osalta käyttöliittymällä tuli pystyä valitsemaan ohjaava reuna, seuraamaan mittaustuloksia sekä syöttämään ohjelmalle offset-arvo. Käyttöliittymästä tuli myös käydä ilmi järjestelmän virheiden tilatiedot. Myöhemmin tavoitteisiin lisättiin myös Profibus-liikenteen seuraaminen sekä kielen vaihto suomen ja englannin välillä.

1.2 Toimeksiantaja

Opinnäytetyön toimeksiantajana toimi Vision Systems Oy, joka sijaitsee Jyväskylän Palokassa. Yritys on optisen mittauksen ja konenäkötekniikan asiantuntijayritys, jonka asiakaskohtaisiin projekteihin kuuluvat teollisuusjärjestelmät, sulautetut järjestelmät, liikkuvien työkoneiden ohjaus- ja mittausjärjestelmät sekä liikenteeseen ja turvallisuuteen liittyvät projektit. Yhtiö on myös kehittänyt anturi- ja järjestelmätasolla itse omia tuotteita, joista osa on myös patentoitu. Vision Systems Oy:n asiakaskunta koostuu pääasiassa paperi-, sellu-, metalli-, teräs-, lasi-, elintarvike- ja rengasteollisuuden yrityksistä, sekä usean alan laite- ja konenäkövalmistajista. (Vision Development n.d.)

Eräs näistä Vision Systemsin suunnittelemista järjestelmistä on mittaus- ja reunanohjausjärjestelmä, joka on suunniteltu paperi- ja metallikelaimille. Tässä järjestelmässä ei ole aikaisemmin ollut omaa käyttöliittymäänsä ja toimeksiantaja halusi korjata tämän puutteen.

1.3 Tutkimusmenetelmät

Työ on kehittämisprojekti, joka on tutkimusotteeltaan kehittämistutkimus, sillä tutkimuksen tuloksena on tarkoitus saada aikaan muutos työn kohteena olevaan projektiin. Teorian ja käytännön suhdetta tässä työssä voidaan sanoa abduktioksi, eli teoria ja käytäntö vuorovaikuttavat toisiinsa. Työ on itsessään empiirinen tutkimus, sillä tutkimuksen kohteena oleva ongelma on oikea reaalimaailman ongelma, eikä ainoastaan teoreettinen ongelma. Aineistonkeruumenetelminä käytettiin kvalitatiivisellekin tutkimusotteelle tyypillisiä aineistoja, kuten dokumentteja ohjeista ja kaavioista, sekä omakohtaisia havaintoja ohjelmiston ja järjestelmän käyttäytymisestä. (Opinnäytetyö-kehittämisprojekti n.d.; Tutkimuksen/opinnäytetyön toteutus n.d.)

Käyttöliittymän toiminta pyrittiin luotettavasti varmistamaan useilla testeillä, joita analysoimalla pyrittiin etsimään virheitä käyttöliittymän ulkonäön, oikeinkirjoituksen ja yleisen toiminnan kannalta. Käyttöliittymän kehittämistyön ollessa loppusuoralla, oli sitä tarkoitus päästä testaamaan aidossa tehdasympäristössä. Tämä ei kuitenkaan ollut mahdollista, joten Vision Systemsin tiloihin rakennetiin tehdasympäristöä mahdollisimman hyvin kuvaava testiympäristö, jossa varmistettiin käyttöliittymän toimivuus.

2 Järjestelmän komponentit

Järjestelmä, johon käyttöliittymä rakennettiin, on käytännössä suuri kelain, jonka tarkoitus on kelata metalli- tai paperirainaa rullalle tai auki rullasta. Järjestelmä seuraa rainan reunaa ja ohjaa kelainta siten, että rullan reuna pysyy mahdollisimman tasaisena. Metallirainarullia kuljetettaessa on tärkeää, että rullat ovat reunoiltaan tasaisia, jotta ne eivät vahingoitu lastauksen, kuljetuksen tai purun yhteydessä.

Järjestelmässä, jonka pohjalta käyttöliittymä suunniteltiin, on seuraavat komponentit: päälle- tai aukikelain, proportionaaliventtiili, lineaarianturi, yksi tai useampi kappale VisiSmartin integroitua konenäköanturia, Beckhoffin CP660-paneeli-PC sekä Beckhoffin CX9020 -logiikka.

2.1 Päälle- tai aukikelain

Kelaimen tarkoitus järjestelmässä on yksinkertaisesti kelata linjastolta tulevaa materiaalia rullalle tai syöttää materiaalia linjastolle kelaamalla rullaa auki. Järjestelmässä käytetään materiaalina ainoastaan terästä, mutta mikään ei estä hyödyntämästä käyttöliittymää myös paperirullien kelaimissa. Kuviossa 1 on esitetty esimerkki siitä, minkälainen kelain voi olla.

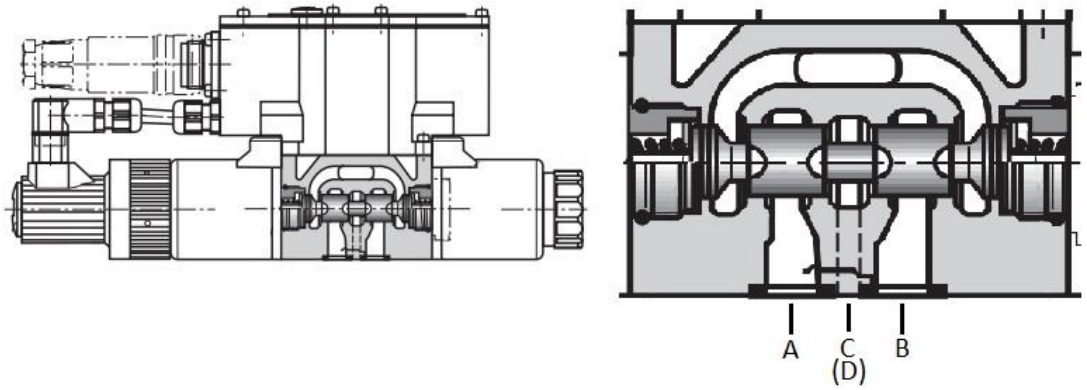
Kuvio 1. Esimerkki metallirainan kelaimesta (Cascadia Metals n.d.)

2.2 Proportionaaliventtiili

Kuten nimestä voikin jo päätellä (proportional = suhteellinen), proportionaaliventtiilillä säädetään ulostulevaa suuretta suhteessa sisääntuloon (Proportional valves 2013). Koska proportionaaliventtiiliä pystytään ohjaamaan tasaisin liikkein, eikä askelmaisesti, vähentää tämä paineiskuja järjestelmissä, joissa vaaditaan useita virtauksen tai paineen muutoksia (Proportional control valves 2009).

Kuviossa 2 on esitetty proportionaaliventtiili, jota mittaus- ja reunanohjausjärjestelmässä käytetään ohjamaan hydraulista painetta, jolla puolestaan ohjataan kelaimen

asentoa sivuttaissunnassa linjastolta päin katsottaessa. Venttiilin puolaa ohjataan joko yhdellä tai kahdella solenoidilla, riippuen siitä, kuinka monta asentoa venttiilillä on. Solenoidit sijaitsevat venttiilin puolan molemmin puolin. Venttiilin kaikki neljä tuloa/lähtöä on merkattu kuvioon 2 kirjaimin A, B, C ja D, ja venttiilin puolan asento määrittää, mitkä näistä tuloista ja lähdöistä on yhdistetty toisiinsa. Puolan molemmissa päissä sijaitsevat palautusjouset, jotka palauttavat puolan keskiasentoon, mikäli kummallakaan solenoidilla ei aktiivisesti ohjata puolan asentoa. Venttiilissä on myös oma elektroninen takaisinkytkentä, jolla venttiili varmistaa, että puola on oikeasti saavuttanut halutun aseman. (4/2 and 4/3 proportional directional valves… 2013)



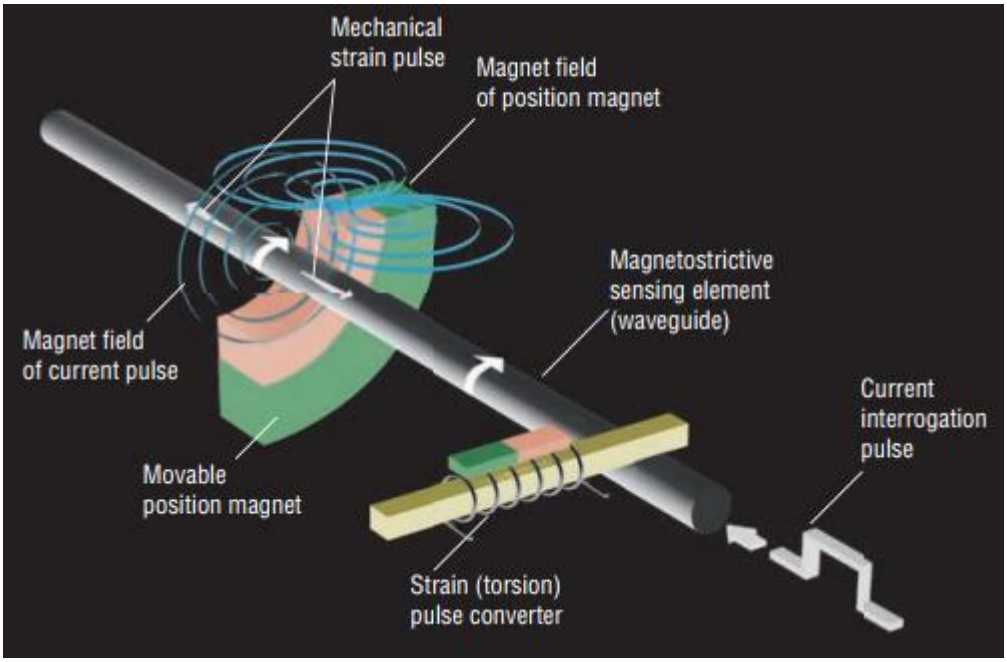
Kuvio 2. Proportionaaliventtiili (4/2 and 4/3 proportional directional valves… 2013, muokattu)

2.3 Lineaarianturi

Lineaarianturin tehtävä on mittaus- ja reunanohjausjärjestelmässä mitata kelaimen asemaa sivuttaissuunnassa. Lineaarianturilta tulevaa tietoa käytetään järjestelmän yhtenä takaisinkytkentänä proportionaaliventtiiliä ohjattaessa.

Järjestelmässä käytettävän lineaarianturin toimintaperiaate perustuu ferromagneettisen mittauselementin magnetostriktiivisyyteen ja se on havainnollistettu kuviossa 3. Anturin toiminnan kannalta tärkeimpänä elementtinä toimii ferromagneettinen puikko, johon mitattavaan kappaleeseen kiinnitetty magneetti luo magneettikentän. Ferromagneettista puikkoa pitkin lähetetään tämän jälkeen virtapulssi, joka luo kappaleeseen oman magneettikentän. Kun pulssin luoma magneettikenttä kohtaa mitattavan kappaleen magneetin luoman magneettikentän, syntyy pulssi, joka liikkuu fer-

romagneettista puikkoa pitkin äänennopeudella. Puikon samassa päässä, josta virtapulssi lähetettiin, sijaitsee kääntäjä, joka kääntää pulssin elektroniikalle ymmärrettävään muotoon. Kun mitataan virtapulssin lähetyksen ja paluupulssin saapumisen välillä kulunut aika sekä tiedetään, millä nopeudella ääni liikkuu kyseisessä puikossa, pystytään laskemaan, millä etäisyydellä mitattava kappale on. (Magnetostriction n.d.; Temposonics Sensor Technology - magnetostrictive! n.d.)



Kuvio 3. Magnetostriktiivisyyteen perustuvan lineaarianturin toimintaperiaate (Temposonics, R-Series Catalog 2014)

2.4 VisiSmart-integroitu konenäköanturi

VisiSmart-konenäköanturin tehtävänä on mittaus- ja reunanohjausjärjestelmässä seurata linjastolla liikkuvan metallirainan reunaa ja mitata sen paikkaa. Konenäköanturilla voidaan myös joissain tapauksissa seurata metallirainan keskipistettä ja mitata sen paikkaa. Näitä paikkatietoja käytetään yhtenä PID-säätimen takaisinkytkentänä proportionaaliventtiiliä ohjattaessa.

VisiSmart käyttää kuvien otossa CMOS-kennoa, jonka tarkkuus on viiden megapikselin luokkaa ja resoluutio on 2592 x 1944 pikseliä. Konenäköanturilla käsitellään tyypillisesti 50 - 1000 kuvaa sekunnissa ja sen tarkkuus on 0,1 - 5 pikselin luokkaa. Perus-

versio on varustettu mustavalkoisia kuvia ottavalla anturilla, mutta tarvittaessa laitteessa voidaan käyttää myös värillisiä kuvia ottavaa anturia, jolloin VisiSmart pystyy myös tunnistamaan ja mittaamaan erilaisia värejä. (Visi 50 Smart n.d.)

Laitteesta löytyy yksi Ethernet-kaapelin liitin, kaksi digitaalista inputia sekä neljä digitaalista outputia. Tarpeiden mukaan VisiSmartiin voidaan myös asentaa yksi analoginen sisääntulo, yksi analoginen ulostulo sekä Profibus-liitin. VisiSmart-konenäköanturia voidaan käyttää joko itsenäiseen mittaukseen ja ohjaukseen tai se voidaan liittää osaksi suurempaa automaatiojärjestelmää I/O -liitäntöjensä ansiosta. Tavallisesti laitetta käytetään reunan mittaukseen, keskipisteen mittaukseen, leveyden mittaukseen, alueen mittaukseen ja kappaleiden lukumäärän laskemiseen. (Visi 50 Smart n.d.)

2.5 Beckhoff CP6606-paneeli-PC

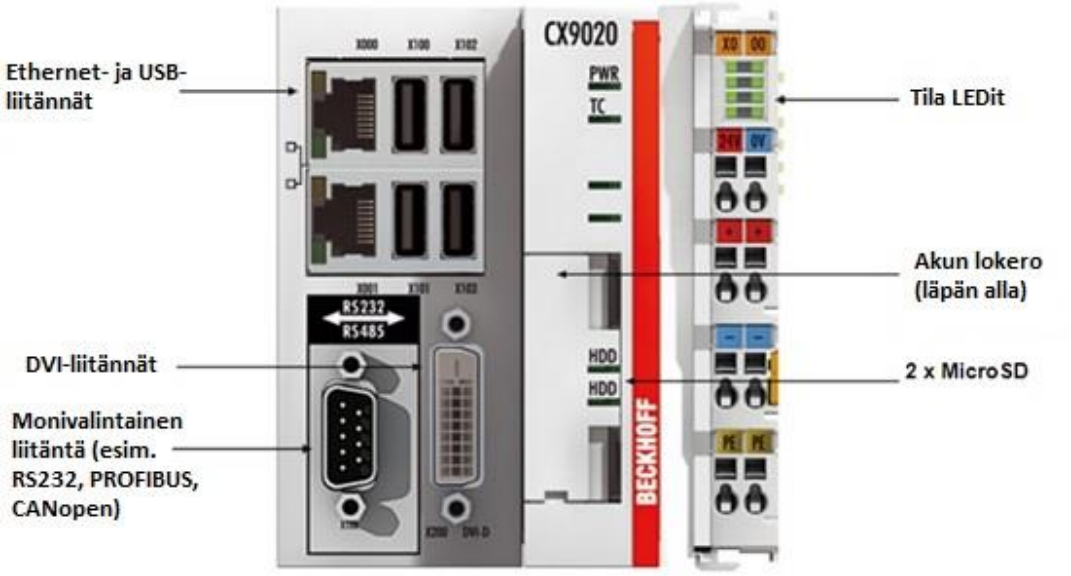
Paneeli-PC:llä on tarkoitus mittaus- ja reunanohjausjärjestelmässä pyörittää järjestelmän käyttöliittymää, jota ohjataan paneelin 7 tuumaiselta kosketusnäytöltä. Paneeli-PC:n käyttöjärjestelmänä toimii Windows Embedded Compact 7:llä ja PC itsessään vaatii toimiakseen 24 voltin virtalähteen. Paneeli-PC ja logiikka kytketään toisiinsa EtherCAT-liitännällä. Kuviossa 4 on esitetty Beckhoffin CP6606 -paneeli-PC. (CP6606, Panel PC with ARM Cortex -A8 n.d.)

Kuvio 4. Beckhoffin CP6606 -paneeli-PC (CP6606, Panel PC with ARM Cortex -A8 n.d.)

2.6 Beckhoff-CX9020 logiikka

Logiikan tehtävä on kerätä tietoa järjestelmässä olevilta antureilta, tehdä näiden pohjalta tarvittavia laskutoimituksia, verrata laskutoimitusten tuloksia käyttäjän syöttämiin arvoihin (kuten offsetiin) ja ohjata näiden pohjalta järjestelmässä olevaa toimilaitetta, eli proportionaaliventtiiliä. Beckhoffin logiikka toimii siis ikään kuin järjestelmän aivoina. Logiikka on esiteltynä kuviossa 5.

Logiikan laskutoimitukset hoitaa sen yksi ytiminen 1 GHz:n ARM Cortex A8 -prosessori. Logiikka vaatii toimiakseen 24 voltin DC-jännitteen, ja mikäli tämä jännitteen syöttö katkeaa yllättäen, logiikan oma 128 kB:n NOVRAM (non-volatile random access memory) kerää tärkeimmän datan muistiinsa ja palauttaa sen taas logiikan käyttöön jännitteen palatessa takaisin. Logiikan käyttöjärjestelmänä toimii Microsoft Windows Embedded Compact 7, mikä takaa sen yhteensopivuuden muiden Windows pohjaisten tietokoneiden kanssa. (CX9020, Basic PLC module n.d.)



Kuvio 5. Beckhoffin CX9020 -logiikka (CX9020, Basic PLC module n.d., muokattu)

3 PID-säätöpiiri

3.1 Perusteet

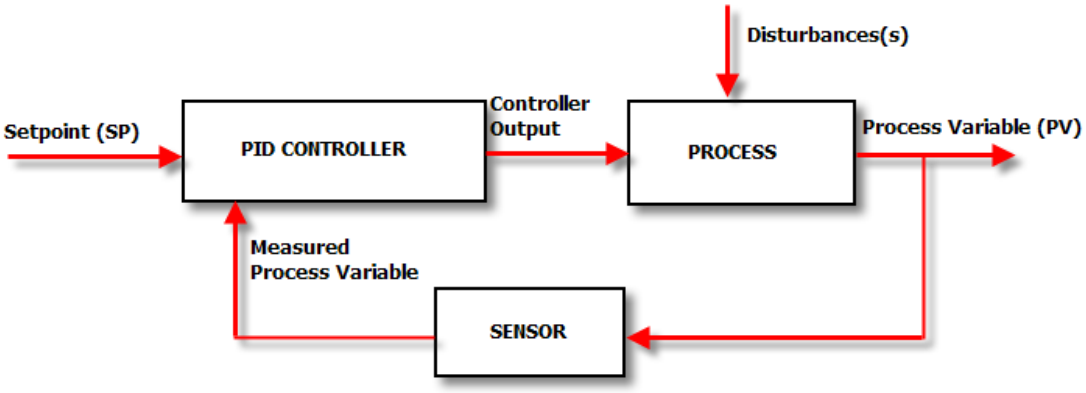
PID-säätöpiirillä on suuri merkitys työssä esiintyvän mittaus- ja reunanohjausjärjestelmän toiminnan kannalta. PID-säätöpiiri on loppujen lopuksi vastuussa siitä, kuinka tasaisesti järjestelmä kelaa metallirainan rullalle ja kuinka hyvin järjestelmä reagoi metallirainan sivuttaiseen liikkeeseen. Tästä syystä on tärkeää ymmärtää, mistä osista PID-säätöpiiri koostuu, sekä kuinka PID-säädin viritetään.

PID-säätimet ovat kaikista teollisuudessa käytetyistä säätöpiireistä yleisimpiä ja jo vuonna 2002 ne käsittivät yli 95 % kaikista säätöpiireistä. Näistä yleisin on PI-säädin. (Åström 2002)

PID-säätöpiirin ideana on säätää haluttua laitetta yhden tai useamman mittauksen perusteella ja se toimii takaisinkytkennän periaatteella. Esimerkiksi jos säädetään nesteen virtausnopeutta haluttuun arvoon venttiilin avulla, tarvitsee aluksi mitata nykyinen virtausnopeus. Tästä säädin päättelee kuinka paljon venttiiliä tarvitsee aukaista tai sulkea, jotta se saavuttaa halutun virtausnopeuden arvon. Tehtyään tarvittavan ohjaustoiminnon, säädin alkaa seuraamaan virtausnopeuden muutosta. Tästä muutoksesta ja sen hetkisestä virtausnopeudesta säädin tekee taas uudet päätelmät siitä, kuinka paljon se säätää venttiiliä ja mihin suuntaan. Säätöpiiri on siis loputon silmukka, joka mittaa ja säätää haluttua suuretta haluttuun arvoon.

Kuviossa 6 on esitetty PID-säätimen toiminta osana prosessia. Kuvion vasemmassa reunassa oleva ”Setpoint”-nuoli on PID-säätimen asetusarvo. Tällä muuttujalla käyttäjä kertoo säätimelle, mihin arvoon mitattava suure halutaan asettaa. Asetusarvon saatuaan säädin tekee tarvittavat laskelmat ja ohjaa tämän jälkeen prosessia haluttuun suuntaan. Kuviossa olevalla ”Disturbances(s)”-nuolella kuvataan prosessiin vaikuttavia häiriöitä, joiden vaikutusta PID-säädin yrittää myös kumota mahdollisuuksien mukaan. Kuvion oikeassa reunassa oleva ”Process Variable” -nuoli on prosessin tuloksena oleva suure, jonka arvoa käyttäjä haluaa alun perin muuttaa. PID-säädin pyrkii siis saamaan tämän arvon mahdollisimman lähelle setpointin arvoa. Jotta säädin tietäisi, miten kaukana nykyinen process variable -arvo on setpointin arvosta, tar-

vitsee säätimelle luoda takaisinkytkentä. Takaisinkytkentä on usein suoritettu anturilla, joka mittaa process variablen arvoa ja lähettää mittaustulokset PID-säätimelle. (PID for Dummies n.d.)

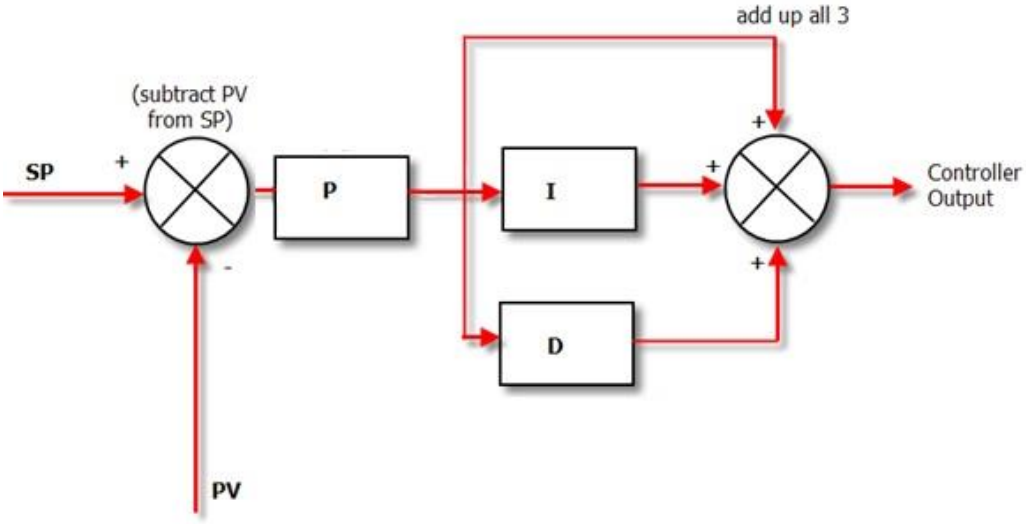


Kuvio 6. PID-säätimen ohjaama prosessi (PID for Dummies n.d.)

PID-säädin on siis eräänlainen funktio, jonka tuloksena on referenssin ja säädettävän suureen erotus ja se koostuu kolmesta osasta: P, I ja D. P-osa tulee sanasta ”proportional” ja se on erosuureeseen suoraan verrannollinen termi, joka pyrkii poistamaan virhettä. P-termin haittana on se, että se luo itsessään pysyvän poikkeaman prosessiin. I-osa tulee sanasta ”integral” ja se on puolestaan erosuureen integraaliin verrannollinen termi. I-termillä pyritään poistamaan pysyvää poikkeamaa, mutta sen käyttö hidastaa vaste-aikaa, sekä aiheuttaa säädettävään suureeseen ylityksen. Ylitys tarkoittaa, että säädettävän suureen arvo menee yli asetusarvon säädön aikana. D-osa tulee sanasta ”derivative”, ja se on erosuureen derivaattaan suoraan verrannollinen termi. D-termi reagoi erosuureen muutoksiin vastustaen näitä, mikä poistaa säädöstä värähtelyjä ja nopeuttaa vastetta. Haittana on herkkyys kohinalle. (PID-säädin 2011; Harju & Marttinen 2000, 44-52)

Kuviossa 7 on esitetty ideaali PID-säädin. Kuviossa vasemmalla on aluksi suoritettu takaisinkytkennällä saadun PV:n (Process Variable) arvon erotus SP:n (SetPoint) arvosta, josta saadaan tulokseksi erosuureen, eli virheen arvo. Tämän jälkeen ideaalisäädin laskee ensin P-termin ohjausarvon. Tällä ohjausarvolla kerrotaan seuraavaksi I- ja D-termien ohjausarvot. Lopuksi nämä kerrotut arvot sekä P-termin ohjausarvo

lasketaan keskenään yhteen. Tästä saadaan säätimen lopullinen ohjausarvo, mikä lähetetään prosessille.



Kuvio 7. Ideaalin PID-säätimen rakenne (PID for Dummies n.d., muokattu)

3.2 Matemaattinen selitys

PID-säätimen ohjaussignaalin arvo voidaan myös esittää matemaattisena funktiona, jonka ulkomuoto riippuu siitä, onko kyseessä P-, PI- vai PID-säädin. P-säätimen ohjaussignaalin funktio on: $u(t) = K_p e(t) + u_0$, jossa u(t) on säätimen ohjaussignaali, $K_p$on säätimen vahvistus, e(t) on erosuureen arvo ja u_0 on ohjaussignaalin vakiotaso kun erosuure on nolla (u_0 on usein myös nolla). (Harju & Marttinen 2000, 45-47, 50-52)

PI-säätimen ohjaussignaali on matemaattisena funktiona puolestaan seuraavanlainen: $u(t) = K_p[e(t) + \frac{1}{T_i}\int e(t)dt]$, missä T_i on integrointiaika sekunneissa. Jos verrataan tätä funktiota pelkän P-säätimen funktioon, huomataan, että P-säätimen u_0-termi on korvattu ajan funktiona muuttuvalla erosuureen integraalilausekkeella. Tämä integraalilauseke on käytännössä verrannollinen kaikkien vanhojen erosuureiden summaan, minkä ansiosta P-säädölle tyypillinen asentovirhe jää pois. Ominaisuuksiensa ansiosta PI-säädin on yleisimmin käytetty säädintyyppi. (Harju & Marttinen 2000, 47-48, 50-52)

PID-säätimen ohjaussignaalin matemaattinen funktio on muotoa: $u(t) = K_p[e(t) + \frac{1}{T_i}\int e(t)dt + T_d\frac{de(t)}{dt}]$, missä T_d on derivointiaika sekunneissa. PI-säätimen funktioon verrattuna PID-säätimen funktioon on lisätty derivaattalauseke, jonka tehtävänä on kuvata suureen muutosnopeutta. Jos muutoksia ei tapahdu, on D-signaali vakio ja $\frac{de(t)}{dt}$:n arvo on näin ollen nolla. Joskus erosuureen muuttumisnopeuden $\frac{de(t)}{dt}$ sijasta käytetään mittaussuureen muuttumisnopeutta $\frac{-dy(t)}{dt}$. Tällöin säätimen D-osa ei reagoi suoraan asetusarvon muutoksiin, eli säädin on käytännössä tarkoitettu tässä tapauksessa kuormitushäiriöiden kompensointiin. (Harju & Marttinen 2000, 48-52).

Joissain tapauksissa P-, I- ja D-osilla on kullakin omat vahvistustekijänsä. Tämä tarkoittaa matemaattisesti sitä, että ohjaussignaalin funktiossa on kolme erillistä termiä. Ohjaussignaalin funktio PID-säätimelle on tässä tapauksessa muodossa $u(t) = k_p e(t) + k_i\int e(t)dt + k_d\frac{de(t)}{dt}$. (Harju & Marttinen 2000, 60).

3.3 Säätimen viritys

PID-säätimen viritys perustuu käytännössä kolmen viritysparametrin arvon määrittämiseen. Nämä parametrit ovat nimeltään vahvistuskerroin, integrointiaika sekä derivointiaika. Tärkeintä virityksessä on selvittää näiden parametrien välinen suhde siten, että säädin painottaa oikeassa suhteessa ohjaustermien P, I ja D vaikutusta säätimen kokonaisohjaukseen. Viritysparametreille ei myöskään ole olemassa yhtä oikeaa suhdetta, vaan eri parametriarvojen suhteilla saatetaan saavuttaa hyvin samanlainen prosessin käyttäytyminen. (Harju & Marttinen 2000, 114-116).

Vahvistuskerroin K_p on säätimen ohjauksen ja erosuureen välinen suhdekerroin, eli vahvistuskertoimen kasvattaminen nopeuttaa järjestelmän käyttäytymistä. P-säätimen virityksessä säädetään siis ainoastaan tätä parametria. (Harju & Marttinen 2000, 50).

Integrointiaika T_i on se aika, jossa säätimen I-osa saa aikaan samansuuruisen muutoksen kuin säätimen P -osa askelmaisessa erosuureen muutoksessa. Aiemmin esitetyn PI-säätimen funktiota tutkimalla huomataan, että T_i on kaavassa muodossa $\frac{1}{T_i}$.

Tämä tarkoittaa sitä, että mitä suurempi integrointiaika on, sitä pienempi I-osan vaikutus on säätimen toimintaan ja jos integrointiaika asetetaan äärettömän suureksi, katoaa I-osan vaikutus lähes kokonaan ja PI-säädin alkaa muistuttamaan toiminnaltaan enemmän P-säädintä. Pieni integrointiaika aiheuttaa puolestaan suuren ohjauksen kasvunopeuden. (Harju & Marttinen 2000, 50).

Derivointiaika T_d vaikuttaa suoraan siihen miten voimakkaasti D-osa vaikuttaa säätöön. Suurempi derivointiaika tarkoittaa siis käytännössä suurempaa D-osan vaikutusta säätöön. (Harju & Marttinen 2000, 50).

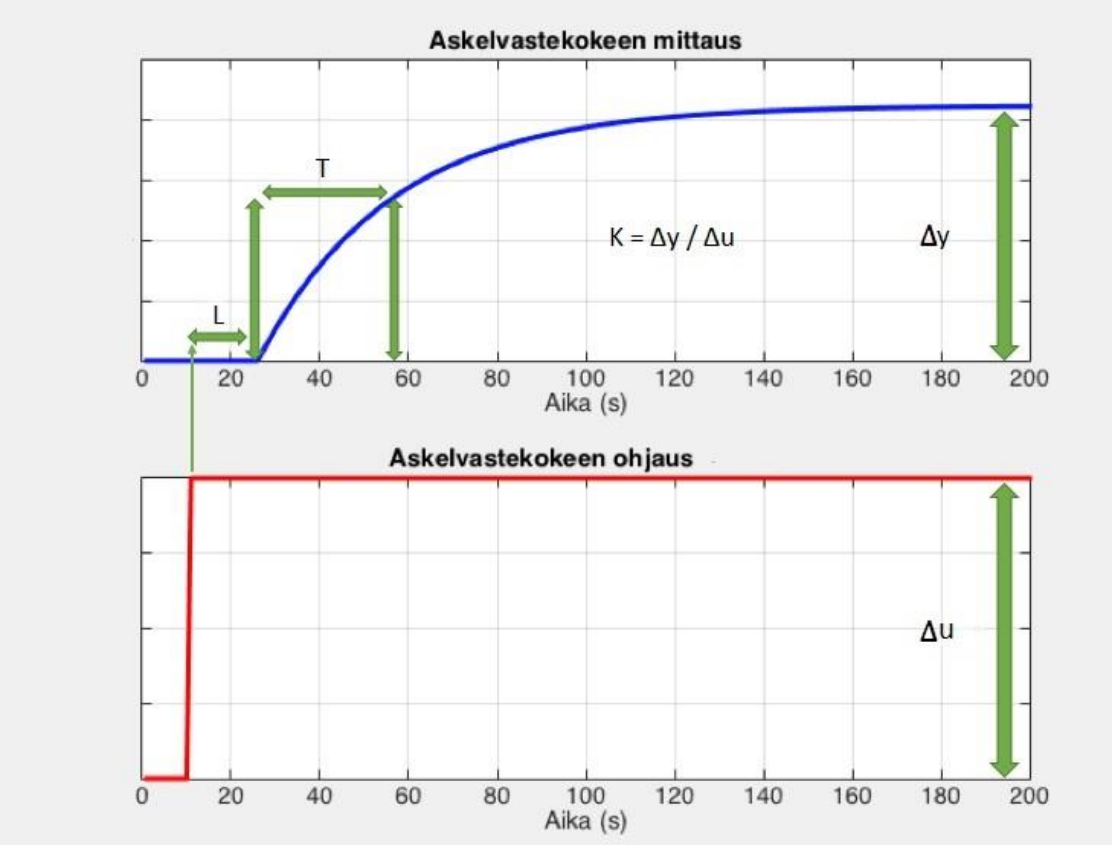
Viritystapoja on kahdenlaisia, kokeellisia ja laskennallisia. Kokeellisessa viritystavassa on ideana selvittää kokeilemalla viritysparametrien arvojen parhaat suhteet. Jos halutaan virittää PID-säädin kokeellisesti, on tärkeää ymmärtää jokaisen parametrin merkitys ja vaikutus järjestelmän käyttäytymiseen. Viritysparametreille löytyy myös suuntaa antava taulukko (ks. taulukko 1), jossa kerrotaan, miten parametrit vaikuttavat prosessiin. Taulukko pitää enimmäkseen paikkansa, vaikka poikkeuksiakin löytyy. (Harju & Marttinen 2000, 114-116).

Taulukko 1. Viritysparametrien vaikutus prosessiin (Harju & Marttinen 2000, 114)

	Muuttujan arvo	**Prosessin nopeus**	**Prosessin Stabiilisuus**
K_p	kasvaa	kasvaa	vähenee
T_i	kasvaa	vähenee	kasvaa
T_d	kasvaa	kasvaa	kasvaa

Laskennallisia viritystapoja on useita, mutta näistä helpoimpia on Ziegler-Nicholsin viritysmenetelmä. Menetelmä on esitetty jo vuonna 1942 ja se perustuu kolmeen prosessiparametriin: vahvistus K, aikavakio T, sekä viive L. Vahvistus K:ta ei pidä sekoittaa vahvistuskerroin K_p:een. Nämä prosessiparametrit saadaan selvitettyä esimerkiksi askelvasteesta. (Harju & Marttinen 2000, 115-116).

Askelvaste on teollisuudessa helpoin ja yleisin tapa kuvata järjestelmän käyttäytymistä. Askelvastekokeen ideana on antaa järjestelmälle porrasmainen ohjauskäsky ja mitata järjestelmän lähdön muutosta siihen asti, kunnes se saavuttaa tasapainotilansa. Kuviossa 8 on esitetty askelvaste sekä se, kuinka prosessiparametrit K, T ja L saadaan selville askelvasteesta. (Harju & Marttinen 2000, 74)



Kuvio 8. Esimerkki askelvastekokeesta (FOPDT integrating model n.d., muokattu)

Kuten kuviosta 3 näkyy, vahvistus K on siis mittaussignaalin muutoksen ja ohjaussignaalin muutoksen suhde tasapainotilassa. K:n laskemisessa on tärkeää muistaa skaalata mittaussignaali ja ohjaussignaali samaan yksikköön. Viive L puolestaan kuvaa sitä aikaa, missä prosessi ei reagoi ohjaussignaalin muutokseen. Aikavakion selvittämiseen on olemassa eri tapoja, mutta yksinkertaisin näistä on selvittää laskemalla koska mittaussignaali on noussut 63 % tasapainoasemaan nähden ja mitata kuinka kauan aikaa tähän kului. Aikavakion määrittämisessä ei oteta huomioon viivettä L, vaan ajanmittaus aloitetaan siitä, mistä prosessisignaali alkaa reagoimaan ohjaussignaalin muutokseen. Kun nämä prosessiparametrit on selvitetty, saadaan niiden avulla selvitettyä laskemalla säätimen viritysparametrit. (Harju & Marttinen 2000, 79-80). Taulukossa 2 on esitetty kunkin viritysparametrin laskeminen kullekin säätimelle.

Taulukko 2. Viritysparametrien määritys prosessiparametrien avulla (Harju & Marttinen 2000, 115)

	K_p	T_i	T_d
P	T/(KL)		
PI	0,9T/(KL)	3L	
PID	1,2T/(KL)	2L	L/2

4 Reunantunnistus ja mittausmenetelmät

4.1 Reunantunnistuksen perusteita

Konenäöstä puhuttaessa liikutaan digitaalisessa maailmassa, jossa kaikki koostuu pohjimmiltaan biteistä. Niinpä myös konenäössä käsiteltävät kuvat koostuvat siis biteistä. Näistä kuvien biteistä rakennetaan aluksi pikseleitä, ja näistä pikseleistä rakennetaan kokonaisia kuvia. Yksi pikseli kuvassa pystyy ottamaan ainoastaan yhden värin, eli mitä enemmän kuvassa on pikseleitä, sen tarkempi kuvasta tulee. Kuvan pikseleiden määrän ilmaisemiseen käytetään termiä resoluutio. Jos kuvan resoluutio on esimerkiksi 1280 x 768, tarkoittaa se sitä, että kuvassa on yhteensä 983040 pikseliä, jotka ovat jakautuneet 1280 sarakkeeseen ja 768 riviin. (Davies 2004, 19-21)

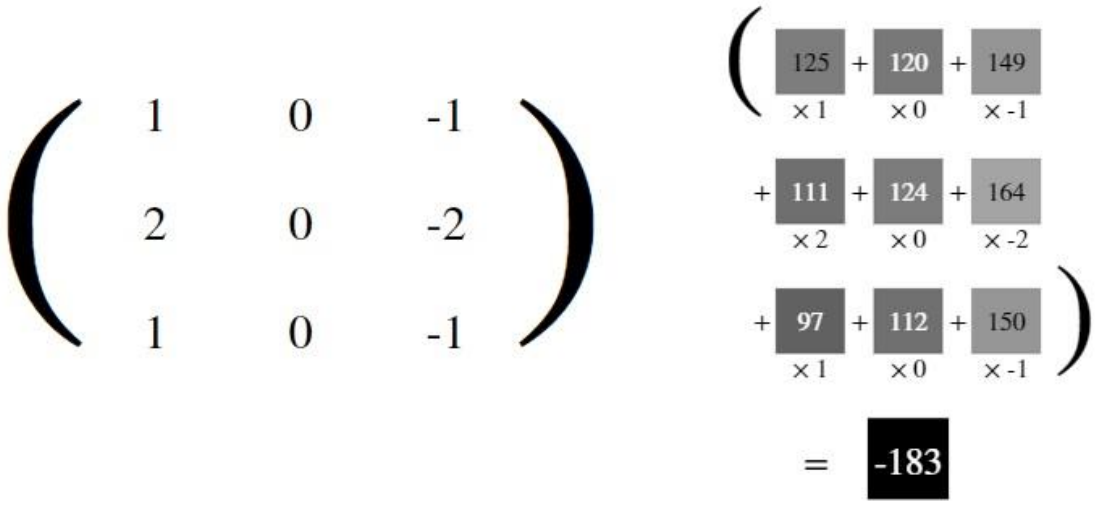
Yksi pikseli koostuu siis biteistä ja näiden bittien tehtävänä on määrittää, minkä värin kyseinen pikseli ottaa. Pikselin bittien määrä siis kertoo kuvan väriskaalasta eli siitä, kuinka monta eri väriä kuvassa pystytään esittämään. Jos yhdessä pikselissä on esimerkiksi kahdeksan bittiä, tarkoittaa se sitä, että yhdellä pikselillä on 256 eri värivaihtoehtoa, jossa 0 on väriltään musta ja 255 valkoinen. (Davies 2004, 19-21)

Konenäössä näillä pikselien bittien arvoilla on suuri merkitys kappaleen reunantunnistuksen kannalta. Yksi yleinen tapa reunantunnistuksessa on käyttää apuna maskia (käytetään myös nimitystä image kernel), joka on eräänlainen matriisi, jolla lasketaan jokaiselle pikselille uusi väriskaalan arvo käyttäen apuna kyseisen pikselin ympärillä olevia pikseleitä. Näitä maskeja on useita ja niitä voidaan käyttää myös muihin kuvankäsittelyyn liittyviin toimintoihin. Hyvänä esimerkkinä näistä reunantunnistuksessa käytetyistä matriiseista voidaan pitää Sobelin matriiseja. Se, mitä Sobelin mat-

riisia käytetään, riippuu siitä, mistä päin reunoja halutaan kuvasta korostaa (esimerkiksi ylhäältä alas tai vasemmalta oikealle). Kuviossa 9 on demonstroitu, mitä Sobelin vasemmalta oikealle -reunantunnistusmatriisi tekee kuvalle. (Davies 2004, 132-135; Powell n.d.)

Kuvio 9. Esimerkki Sobelin matriisin vaikutuksesta. Vasemmalla on alkuperäinen kuva ja oikealla Sobelin matriisilla muokattu kuva. (Powell n.d.)

Kuviossa 10 on puolestaan esitetty Sobelin matriisi, jolla kuvion 9 kuva on muokattu. Sulkeiden sisällä laatikoissa olevat arvot kuvaavat yksittäisten pikseleiden värien arvoja väliltä 0…255, jossa nolla on väriltään musta ja 255 valkoinen. Näiden laatikoiden alapuolella ovat kertoimet, jotka määräytyvät Sobelin matriisin mukaan ja joilla kerrotaan näiden laatikoiden arvot. Sulkeiden ulkopuolella oleva laatikko on lopputulos matriisille ja tätä arvoa käytetään keskimmäisen pikselin uutena väriarvona. Koska Kuvan väriskaala on 0…255, kaikki arvot alle nollan käsitellään mustina, eli nollina ja kaikki arvot yli 255 käsitellään valkoisina, eli pyöristetään 255:een. Tällä menetelmällä käydään läpi kaikki kuvan yksittäiset pikselit ja lopputulokseksi saadaan kuvion 9 mukainen kuva. (Powell n.d.)



Kuvio 10. Sobelin matriisi (vasemmalla), sekä matriisilla laskettu uusi pikselin väriarvo. (Powell n.d.)

Mikäli esimerkiksi Sobelin menetelmällä saadut kuvan reunat eivät ole riittävän selkeitä, voidaan kuvalle tehdä vielä lopuksi kynnystäminen (thresholding). Kuvan kynnystämisessä on kyse siitä, että muutetaan kuvassa olevat pikselit yksibittisiksi, jolloin ne voivat saada vain kaksi väriarvoa; musta tai valkoinen. Tällaista kuvaa kutsutaan myös binäärikuvaksi. Binäärikuva saadaan aikaiseksi kun valitaan sopiva pikselin väriarvo, esimerkiksi 180, jota käytetään kynnysarvona ja kaikki kynnysarvon yläpuolella olevat pikselit muutetaan valkoisiksi ja alapuolella olevat mustiksi, tai päinvastoin. (Davies 2004, 31-32)

Kynnystämistä käyttämällä voidaan myös suoraan tunnistaa kappaleen reunat alkuperäisestä kuvasta, jos mitatun kappaleen ja taustan välillä on selkeä ero kontrastissa ja tausta sekä mitattu kappale ovat suhteellisen tasaisia. Jos kynnystämisen voi tehdä suoraan alkuperäisestä kuvasta, säästää se laskentatehoja kuvankäsittelyssä. (Davies 2004, 131)

Reunantunnistus ei käytännössä ole aina näin virtaviivaista ja yksinkertaista, vaan usein vaaditaan myös jonkinlaisia filttereitä, jotka perustuvat joko samantyyppisiin matriiseihin kuin kuviossa 10, tai muihin kehitettyihin laskukaavoihin. Reunantunnistuksessa valaistus on myös suuressa merkityksessä, sillä oikean valaistuksen avulla saadaan mitattavan kappaleen ja taustan välille suuri kontrasti, jolloin reunan havait-

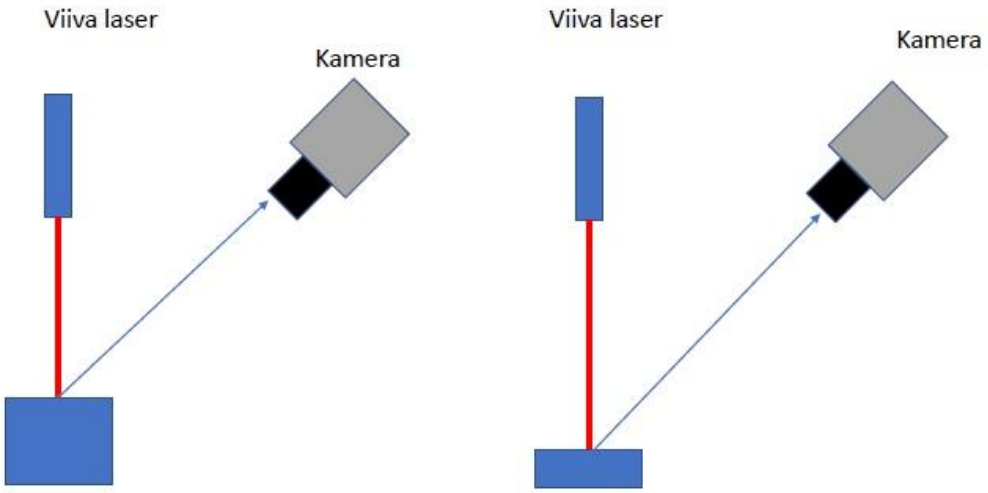
seminen on helppoa. Myös laservaloa käytetään avuksi reunantunnistuksessa. Laservaloa käytettäessä, kameran tarvitsee ainoastaan seurata laservalon katkeamiskohtaa.

4.2 Järjestelmässä käytetyt mittausmenetelmät

Kun konenäköanturi on tunnistanut mitatun kappaleen reunan ja se on valmiiksi kalibroitu, voidaan kappaleelle laskea kaikki tarvittavat parametrit. Järjestelmässä voi olla joko yksi konenäköanturi tai useampi. Yhden kameran järjestelmissä on ainoastaan mahdollista mitata metallirainan toisen reunan paikkaa, sekä tarvittaessa paksuutta. Kahden tai useamman kameran järjestelmissä puolestaan pystytään mittaamaan reunojen paikkojen ja paksuuden lisäksi myös metallirainan leveyttä sekä keskipistettä. (Lauttamus 2017)

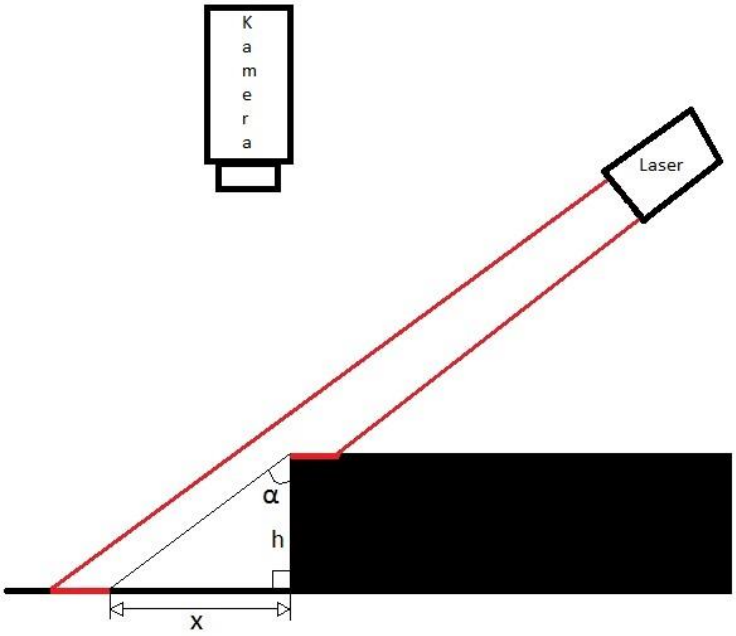
Koko metallirainan leveys pystytään selvittämään laskemalla aluksi molempien kameroiden mitta-alueilla olevat metallirainan leveydet ja lisäämällä tähän lukuun mitta-alueiden väliin jääneen katvealueen leveys. Kun leveys on saatu selville, pystytään siitä laskemaan kappaleen keskikohta lisäämällä kappaleen toisen reunan alkukoordinaattiin puolet koko kappaleen leveydestä. Mikäli on tarvetta, voidaan kappaleen paksuus myös mitata käyttämällä kolmiomittausperiaatetta. (Lauttamus 2017)

Toteutustapoja kolmiomittaukselle on monia, mutta Vision Systemsin usein käyttämä tapa tässä järjestelmässä on esitetty kuviossa 11. Tässä tavassa laservalo on asennettu kohtisuoraan mitattavaan kappaleeseen nähden ja kamera hieman vinoon, jolloin kappaleen korkeus vaikuttaa siihen, mille korkeudelle laservalo piirtyy kameran kennolle. Ideana on tässä tavassa siis mitata mille kohdalle laservalo piirtyy kameran kennolle ja laskea siitä kappaleen korkeus. (Virkajärvi 2017)



Kuvio 11. Järjestelmässä käytettävä kolmiomittausperiaate. (Virkajärvi 2017)

Kolmiomittauksessa kamera voitaisiin myös asettaa kohtisuoraan ja laservalo hieman vinoon, jolloin korkeuden mittaus perustuisi matematiikassa paljon käytettyyn trigonometriseen funktioon nimeltään tangenttiin. Kuviossa 12 on esitetty tämä mahdollinen tapa kameran ja laservalon sijoittelulle. Kappaleen korkeus laskettaisiin tässä tavassa siten, että aluksi kamera mittaisi laservalon luoman kappaleen varjon pituuden x. Koska pikkukolmion kulma α on sama kuin laservalon tulokulma, joka tiedetään, pystyttäisiin kappaleen korkeus h laskemaan suoraan tangentin avulla. Koska $\tan \alpha = \frac{x}{h}$, saadaan siitä korkeudeksi $h = \frac{x}{\tan \alpha}$.



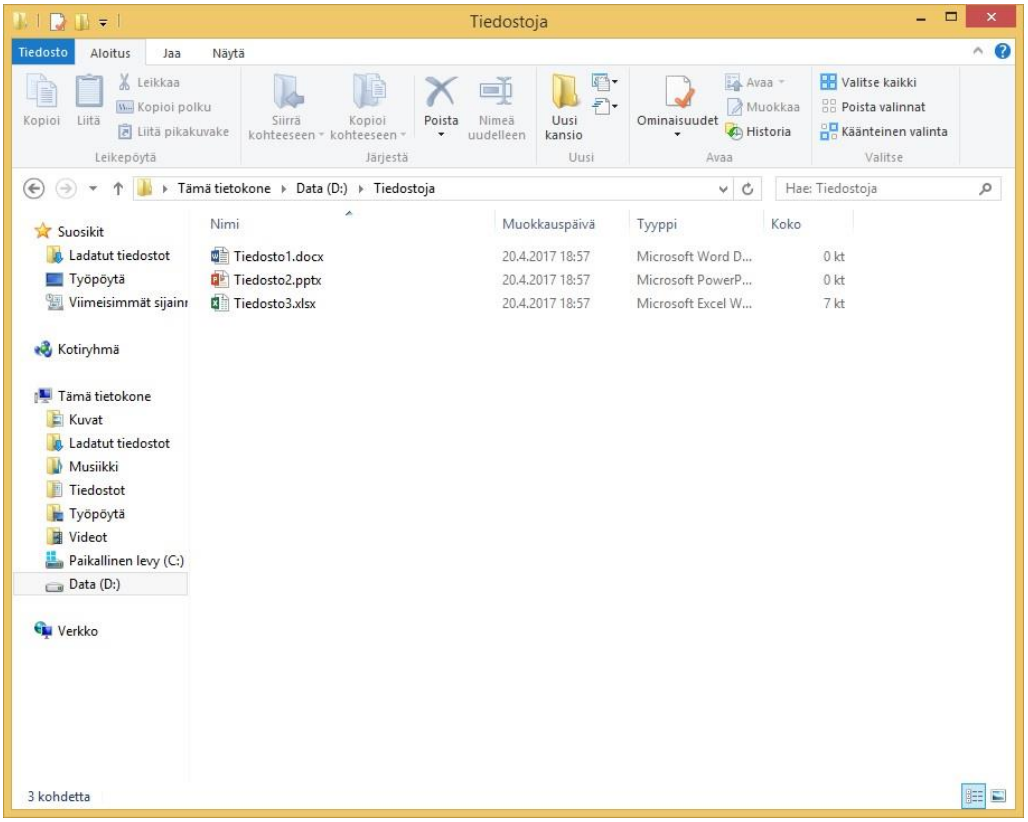
Kuvio 12. Kappaleen korkeuden mittaaminen kolmiomittausperiaatteella.

5 Käyttöliittymä

5.1 Graafinen ja tekstipohjainen käyttöliittymä

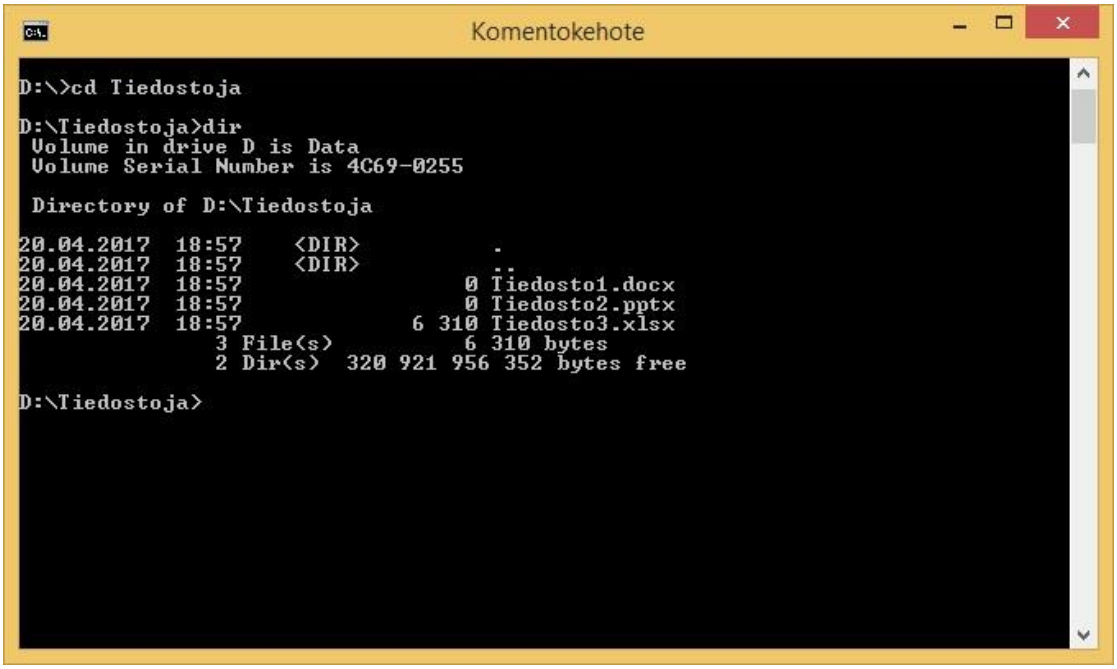
Käyttöliittymä on se rajapinta, jossa ihminen syöttää tai vastaanottaa tietoja koneelta, eli vuoro vaikuttaa koneen kanssa. Käyttöliittymä voi yksinkertaisuudessaan olla auton ratti ja polkimet, sekä mittaristo, joilla auton käyttäjä pystyy antamaan autolle komentoja (kiihdytä, hidasta, käänny…) ja vastaanottamaan tietoja autolta (auton nopeus, kierrokset, polttoaineen määrä…). (Auer 2009)

Tietokonemaailmassa käyttöliittymä voidaan jakaa kahteen eri luokkaan: Graafinen käyttöliittymä ja merkkipohjainen käyttöliittymä. Graafinen käyttöliittymä perustuu graafisten objektien, kuten kuvakkeiden, hallintaan esimerkiksi hiirellä tai näppäimistöllä. Teollisuudessa graafisessa käyttöliittymässä on myös usein erilaisia käyriä tai muita kuvia, joilla kuvataan esimerkiksi laitteen asentoa. Hyvänä esimerkkinä graafisesta käyttöliittymästä voidaan pitää Windowsin resurssienhallintaa (Kuvio 13), jonka avulla käyttäjä etsii tietokoneelle tallennettuja tiedostoja tai kansioita. (TVT-ajokortti n.d.)



Kuvio 13. Windowsin resurssienhallinta

Merkkipohjainen käyttöliittymä puolestaan koostuu pelkästään tekstistä ja kaikki komennot syötetään sille myös tekstimuodossa. Hyvänä esimerkkinä voidaan tässä tapauksessa pitää Windowsin komentokehotetta. Kuviossa 14 on esitetty saman kansion sisältö, kuin kuviossa 13, mutta tekstipohjaisessa käyttöliittymässä. (TVT-ajokortti n.d.)



Kuvio 14. Windowsin komentokehote

5.2 Graafisen käyttöliittymän suunnittelu

Graafisen käyttöliittymän suunnittelussa tulee ottaa huomioon muutakin kuin käyttöliittymän toimivuus tai tehokkuus. Näiden lisäksi käyttöliittymän tulee olla helppokäyttöinen ja helposti ymmärrettävä. Jotta käyttöliittymä täyttäisi nämä ehdot, on hyvä ottaa huomioon ainakin seuraavat asiat suunnittelussa; selkeys, tiiveys, ”tuttuus”, reagoivuus, johdonmukaisuus, tehokkuus sekä anteeksiantavuus (Fadeyev 2009)

Selkeys on graafisen käyttöliittymän suunnittelun kannalta ehkä tärkein asia, sillä graafisen käyttöliittymän tehtävähän on selkeyttää ja havainnollistaa laitteen tai koneen käyttöä. Selkeyteen vaikuttaa käyttöliittymän elementtien organisointi, käytettävä fontti, tooltippien käyttö, sekä käytetyt värit. Tooltipit ovat normaalisti lyhyitä tekstejä, joissa selitetään esimerkiksi mitä jokin käyttöliittymän objekti tekee ja näiden pitäminen lyhyinä ja ytimekkäinä on usein kannatettavaa. (Fadeyev 2009; Martin n.d.)

Vaikka käyttöliittymän tuleekin olla selkeä ja ymmärrettävä, kannattaa myös muistaa pitää se mahdollisimman tiiviinä. Joka kerta kun käyttöliittymään lisätään uusia tooltippejä ja selityksiä käyttöliittymän eri elementeistä, lisääntyy myös käyttöliittymän massa, mikä tarkoittaa lisää lukemista käyttäjälle. Jotta käyttöliittymä pysyisi mahdollisimman tiiviinä, kannattaa jokainen käyttöliittymän elementti nimetä mahdollisimman selkeästi ja pitää jokainen selitys mahdollisimman lyhyenä. (Fadeyev 2009)

Käyttöliittymään kannattaa myös liittää paljon kuvioita ja asioita jotka viittaavat kaikille tuttuihin esineisiin, sääntöihin tai muuhun vastaavaan, esimerkiksi tekemällä keskeytys-napista STOP-merkin näköisen tai zoom-painikkeesta suurennuslasin näköisen. Mitä enemmän käyttöliittymässä on viitteitä jo ennestään tuttuihin asioihin, on käyttäjän helpompi ymmärtää vaistomaisesti miten käyttöliittymä toimii. (Fadeyev 2009)

Käyttöliittymästä tulisi myös tehdä reagoiva, millä tarkoitetaan kahta asiaa: nopeutta ja palautetta antavaa. Käyttöliittymää, joka toimii ja lataa asioita hitaasti ja tökkii, on todella turhauttavaa käyttää verrattuna nopeaan ja sulavaan käyttöliittymään. Käyttöliittymän on hyvä antaa myös palautetta käyttäjälleen tekstien, kuvioiden ja värien muodossa. Esimerkiksi jos käyttäjä painaa jonkin napin pohjaan, muutetaan napin väri vihreäksi tai jos laitteistossa tapahtuu virhe, näytetään punaista varoitustekstiä käyttäjälle mikä kertoo virheen luonteesta. (Fadeyev 2009)

Johdonmukaisuus käyttöliittymän valikoissa on tärkeää varsinkin laajoissa käyttöliittymissä, joissa käyttäjän tarvitsee ohjailla isoja kokonaisuuksia useilla välilehdillä. Tämä saavutetaan siten, että luodaan painikkeet, kuvakkeet ja muut elementit, jotka tekevät tai näyttävät samaa asiaa, samanlaisiksi, jotta käyttäjä tietää mitä painike tekee myös muissa tilanteissa tai mitä kuvake tarkoittaa toisessa tilanteessa. Johdonmukaisuutta saadaan myös suunnittelemalla välilehtien rakenne mahdollisimman paljon samanlaisiksi. Näin käyttäjä löytää helposti tarvitsemansa siirtyessään uudelle välilehdelle. (Fadeyev 2009)

Käyttöliittymän tehokkuuteen liittyy jo aiemmin listatut asiat, kuten johdonmukaisuus ja selkeys, mutta siihen liittyy myös muuta. Nimittäin mikäli käyttöliittymästä halutaan tehokas, tulee suunnittelijan miettiä mitä käyttöliittymän käyttäjä haluaa

saada aikaan tai saavuttaa käyttöliittymällä ja rakentaa sitten kaikki toiminnot sen pohjalta. (Fadeyev 2009)

Lopuksi, käyttöliittymän tulee myös olla anteeksiantava. Käyttöliittymää tulee käyttämään loppupelissä aivan tavalliset ihmiset, jotka yksinkertaisesti tulevat tekemään virheitä. Esimerkiksi jos käyttäjä painaa vahingossa STOP-painiketta, anteeksiantava käyttöliittymä kysyy käyttäjältä varmistuksen, jolloin käyttäjä voi vielä halutessaan peruuttaa tekemänsä. Se, kuinka hyvin käyttäjä pystyy kumoamaan tekemänsä virheet, kertoo paljon siitä, kuinka laadukas käyttöliittymä on. (Fadeyev 2009)

5.3 Värien ja kontrastin käyttö

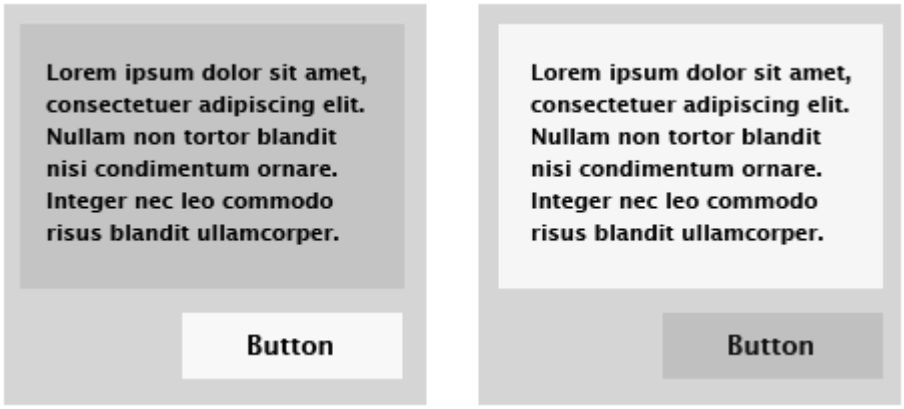
Värien ja kontrastin käyttöä usein laiminlyödään käyttöliittymän suunnittelussa, vaikka sillä on suuri merkitys siihen, miten käyttäjä havainnoi asioita ja mihin käyttäjän huomio kiinnittyy (Fadeyev 2008). Jos käyttöliittymässä käytetään paljon kirkkaita erivärisiä elementtejä, saattaa oleellisin ja tärkein tieto jäädä käyttäjältä kokonaan huomaamatta. Jos taas kaikki on väritykseltään tasaisen harmaata, ei käyttöliittymä ole miellyttävä käyttää ja tärkeimmät asiat saattavat edelleen jäädä huomiotta.

Vaikka käyttöliittymä onkin kaksiulotteisella näytön pinnalla, pystytään silti värin kirkkaudella luomaan mielikuva esimerkiksi painikkeen, syvyydestä. Tämä syvyys saadaan luotua tekemällä kappaleesta joko tummempi, tai vaaleampi, sillä vaaleammat kappaleet vaikuttavat olevan lähempänä kuin tummemmat kappaleet. Tämä on myös yksinkertainen, helppo ja selkeä tapa kertoa käyttäjälle esimerkiksi mitkä painikkeet ovat painettu ja mitkä ei. (Fadeyev 2008)

Värien käyttöön kannattaa kiinnittää myös huomiota, sillä kaikki värit eivät käyttäydy samalla tavalla. Värejä on olemassa kahta eri tyyppiä, kylmiä (esim. vihreä ja sininen) ja lämpimiä (esim. punainen ja keltainen) värejä ja näistä lämpimät värit ovat niitä, jotka kiinnittävät käyttäjän huomion paremmin (Fadeyev 2008). Teollisuudessa on esimerkiksi tärkeää, että käyttöliittymä ei kiinnitä erityistä huomiota mihinkään silloin kun kaikki sujuu hyvin. Kun taas jokin menee vikaan, täytyy käyttäjän huomio kiinnittyä nopeasti siihen käyttöliittymän osaan, joka on oleellisin virheen kannalta. Tämä saadaan aikaiseksi sillä, että käytetään käyttöliittymässä pääsääntöisesti kylmiä värejä ja virheen sattuessa muutetaan oleellisimpien osien värit lämpimiksi.

Käyttäjän huomion kiinnittämiseen on myös värien lisäksi olemassa toinenkin tapa, nimittäin kontrasti. Suuremman kontrastin omaavat elementit erottuvat paremmin joukosta ja pistävät enemmän käyttäjän silmään kuin matalan kontrastin omaavat elementit. (Fadeyev 2008)

Kuviossa 15 on esitetty kaksi esimerkkiä siitä, miten käyttäjän huomion saa kiinnittymään eri kohtiin kuviossa. Vasemmalla ”Button”-painikkeesta on tehty vaaleampi kuin taustasta, joten se erottuu joukosta. Oikealla puolestaan teksti erottuu hyvin, sillä se sijaitsee vaalealla taustalla mikä luo suuren kontrastin tekstin ja taustan välille. Painike sen sijaan lähes katoaa taustaan. Varjostusten, värien ja kontrastien hyvällä yhteiskäytöllä saadaan siis käyttäjän huomiota ohjailtua käyttöliittymän kannalta tärkeisiin kohtiin oikealla hetkellä. (Fadeyev 2008)



Kuvio 15. Esimerkki kontrastin vaikutuksesta (Fadeyev 2008)

6 Käyttöliittymän toteutusvaihtoehdot

Käyttöliittymän tekemiseen on olemassa monia eri tapoja, joita en kaikkia vertaile tai luettele tässä. Näistä mahdollisista tekotavoista esittelen kolme parasta vaihtoehtoa ja valitsen niistä yhden, jolla käyttöliittymä tullaan toteuttamaan. Vaihtoehdot ovat: Beckhoffin Twincat 3, Indusoft Web Studio ja itse koodattu käyttöliittymä.

6.1 Twincat 3 -ohjelmisto

Twincat 3 (The Windows Control and Automation Technology) on Beckhoffin tekemä ohjelmisto, joka on suunniteltu automaatiojärjestelmien ohjelmointiin ja pyörittämiseen. Ohjelmointiin voidaan käyttää Beckhoffin omaa graafista kieltä, C#- tai C++ -ohjelmointikieltä. Twincat on myös integroitu Microsoftin Visual Studioon sekä Matlabiin/Simulinkiin, mikä puolestaan mahdollistaa laajan skaalan erilaisia ohjelmointimahdollisuuksia ja ratkaisuja. Visual Studio mahdollistaa myös näiden ohjelmointikielien rinnakkaisen käytön samassa projektissa. (Twincat 3, Extended Automation 2012)

Hyvät ja huonot puolet

Koska mittaus- ja reunanohjausjärjestelmä on toteutettu Beckhoffin logiikalla ja ohjelma tehty jo valmiiksi Twincat 3 -sovelluksella, on kaikkein helpoin ja selkein tapa tehdä myös käyttöliittymä samalla sovelluksella, se takaa ohjelman ja käyttöliittymän yhteensopivuuden. Twincat 3:lla voi myös ilmaiseksi rakentaa ja testata omaa käyttöliittymää niin usein kuin on tarvetta, mikä on hyvä, sillä silloin suunnitteluvaiheessa ei tarvitse vielä sitoutua ostamaan tiettyä ohjelmaa tai lisenssejä. Tämä mahdollisti myös sen, että käyttöliittymää pystyttiin tekemään omalla koneella silloin kun haluttiin.

Huonona puolena voidaan pitää käyttöliittymäelementtien vähäistä määrää ohjelmassa. Elementtien ulkonäön ja toiminnan muokkaaminen on pääosin riittävän monipuolista, mutta tiettyjen elementtien kohdilla muokkausmahdollisuudet jäävät hieman puutteellisiksi.

6.2 Indusoft Web Studio -ohjelma

Indusoft Web Studio on ohjelma joka on kehitetty teollisuuden automaatioympäristön käyttöliittymien, SCADA-systeemien sekä sulautettujen instrumentointiratkaisujen suunnitteluun ja toteutukseen. Indusoft Web Studiolla voidaan myös luoda käyttöliittymästä Web-käyttöliittymä, jolloin käyttöliittymään päästään käsiksi tavallisella verkkoselaimella, kuten Internet Explorerilla. (Indusoft Web Studio n.d.)

Indusoft Web Studiolla rakennetaan käyttöliittymiä pääsääntöisesti valmiiden objektien ja elementtien avulla, jotka tulevat ohjelman mukana. Objektien lisääminen tapahtuu käytännössä siten, että etsitään aluksi haluttu objekti ohjelman omasta kirjastosta, valitaan se ja asetetaan haluttuun kohtaan käyttöliittymässä. Jokaisella objektilla on myös omia ominaisuuksia, joita pystytään muokkaamaan, jotta saadaan objekti sopimaan mahdollisimman hyvin omiin tarkoitusperiin. Ohjelmalla voidaan myös luoda käyttöliittymään useita välilehtiä, joista jokainen tallennetaan omana tiedostona. Objektien ohjaamiseen käytetään ”tageja”, jotka toimivat käytännössä muuttujina, eli niihin voidaan sijoittaa ja niistä voidaan lukea arvoja. Tagien määrä määräytyy Indusoftilta ostetun lisenssin mukaan. (Indusoft Web Studio 8.0 Quick Start Guide 2016)

Hyvät ja huonot puolet

Indusoft Web Studio on tehty ainoastaan käyttöliittymien ja vastaavien sovellusten suunnitteluun, mikä tarkoittaa sitä, että ohjelman kehittäjät ovat myös keskittyneet vain ja ainoastaan käyttöliittymän suunnitteluun liittyviin ohjelmallisiin tarpeisiin. Tämä johtaa siihen, että Indusoft Web Studiossa on huomattavasti enemmän elementtejä ja objekteja käyttöliittymän suunnittelua varten. Myös näiden objektien ja elementtien ominaisuuksien muokkaaminen on monipuolista. Käyttöliittymien rakentaminen ohjelmalla on myös yksinkertaisen ja helpon näköistä.

Indusoft Web Studion huonoja puolia ovat hinta, sekä tässä tapauksessa integroiminen muuten valmiiseen projektiin. Näistä kolmesta käyttöliittymän toteutusvaihtoehdosta Indusoftin lisenssi on ylivoimaisesti kallein, tosin silti vain pieni osa valmiin projektin hinnasta. Suurempi ongelma Indusoft Web Studiossa on sillä tehdyn käyttöliittymän integroiminen Beckhoffin Twincat 3:lla tehtyyn ohjelmaan sekä oman kokemukseni puuttuminen ohjelman käytöstä. Vaikka Indusoft Web Studio vaikuttaakin helppokäyttöiseltä ja yhteensopivuus Twincat 3:n kanssa on taattua, tuo silti lisää työtä ja mahdollisia ongelmia tehdä käyttöliittymä eri ohjelmalla kuin muu projekti, varsinkin kun ohjelmasta ei ole minkäänlaista aikaisempaa kokemusta.

6.3 Itse koodattu käyttöliittymä

Kolmas vaihtoehto käyttöliittymän tekemiseen oli koodata se itse. Koodaaminen tultaisiin suorittamaan todennäköisimmin C++- tai Visual Basic -ohjelmointikielellä ja ohjelmointiin tultaisiin mahdollisesti käyttämään Visual Studio -ohjelmaa.

Visual Studio on Microsoftin luoma kehitysympäristö, johon on sisällytetty C#-, C++- sekä Visual Basic -ohjelmointikieli. Tämä mahdollistaa ohjelmointikielten välisen kehitystyökalujen jakamisen ja helpottaa näin ollen useamman ohjelmointikielen sekakäyttöä oman sovelluksen rakentamisessa. Visual Studiolla pystytään tekemään ASP.NET Web -sovelluksia, XML Web -palveluja, työpöytä-sovelluksia ja älypuhelinsovelluksia. (Introducing Visual Studio n.d.)

Hyvät ja huonot puolet

Itse koodatussa käyttöliittymässä ei tarvitse tyytyä valmiisiin käyttöliittymä-elementteihin tai -objekteihin, vaan ohjelmoija voi itse tehdä juuri haluamansa objektit joista löytyvät kaikki tarvittavat ominaisuudet ja visuaalisuudet. Koska Twincat 3 on myös integroitu Visual Studioon, pystyy itse koodatun käyttöliittymän siirtämään helposti Twincat 3:lla tehtyyn projektiin ilman että tarvitsee pelätä yhteensopivuusongelmia.

Huono puoli itse koodatussa käyttöliittymässä on se, että se vaatii huomattavasti suuremman määrän työtä. Jokainen käyttöliittymän objekti ja ominaisuus on ohjelmoitava erikseen tyhjästä. Vaikka itselläni onkin hieman kokemusta C-kielen ohjelmoinnista, on se silti vähäistä verrattuna siihen mitä tällaisen käyttöliittymän tekemiseen vaaditaan.

6.4 Yhteenveto toteutusvaihtoehdon valinnasta

Näistä kolmesta vaihtoehdosta keskusteltiin yhdessä Vision Systemsin työntekijöiden kanssa ja vertailtiin jokaisen vaihtoehdon hyviä ja huonoja puolia. Keskustelujen aikana tultiin melko pikaisesti siihen tulokseen, että käyttöliittymää on järkevintä lähteä tekemään Beckhoffin Twincat 3:lla.

Pääpainoksi keskustelussa asettui suunnitteluohjelman toimivuus, yksinkertaisuus sekä ohjelmalla suunnitellun käyttöliittymän yhteensopivuus loppuprojektin kanssa. Vaikka Indusoft Web Studio vaikuttikin hieman selkeämmältä ja monipuolisemmalta

suunnitteluohjelmalta kuin Twincat 3, todettiin silti, että käyttöliittymä on parempi suunnitella samalla ohjelmalla, kuin muukin projekti. Tämä on jatkoakin ajatellen parempi ratkaisu, sillä koko projekti sijaitsee näin ollen yhden kansion alla ja kaikki ohjelman koodista käyttöliittymään on muokattavissa samalla ohjelmalla. Tämä poistaa myös kaikki epäilykset käyttöliittymän yhteensopivuudesta muuhun projektiin. Myös Twincat 3 ilmainen käyttö suunnitteluvaiheessa ja testauksessa vahvisti päätöksen käyttöliittymän suunnitteluohjelman valinnasta.

C-ohjelmointikielellä rakennettu käyttöliittymä oli myös yksi varteenotettavista ratkaisuista, mutta se hylättiin myös melko nopeasti. Keskusteluissa tultiin siihen tulokseen, että oma tietotaito ei ole riittävä kunnollisen käyttöliittymän ohjelmoimiseen opinnäytetyölle suunnitellussa ajassa.

7 Työn toteutus ja ongelmat

7.1 Suunnittelu

Käyttöliittymän suunnitelman pohjana käytettiin pitkälti SSAB:n työntekijöiden kommentteja, sekä heillä nykyisin olevaa kelaimen käyttöliittymää. SSAB on kansainvälinen teräsyhtiö, jolla on tuotantolaitoksia Ruotsissa, Suomessa ja Yhdysvalloissa. Vierailimme suunnitteluvaiheen aikana Hämeenlinnassa sijaitsevalla SSAB:n tuotantolaitoksella. (SSAB lyhyesti n.d.)

Alkuperäisiin suunnitelmiin tuli kuitenkin muutos, sillä käyttöliittymän tuli myös sopia Beckhoffin CP6606 kosketuspaneelille, jolloin alkuperäisesti suunniteltua käyttöliittymää piti jakaa useammalle välilehdelle, jotta kaikki tarvittavat asiat saataisiin mahtumaan. Tekovaiheessa käyttöliittymään tuli myös jonkin verran lisäyksiä Vision Systemsin työntekijöiden toiveista, suurimpana Profibus-viestin inputit ja outputit näyttävä välilehti laitteiden huolto- ja asennustarpeita varten.

Ensimmäiset luonnokset käyttöliittymän ulkonäöstä tehtiin PowerPoint-ohjelmalla. Luonnosten oli tarkoitus olla karkeita hahmotelmia siitä, kuinka kaikki tarvittavat tiedot sijoitetaan käyttöliittymään, joten PowerPoint sopi tähän tarkoitukseen riittävän hyvin. Luonnoksilla oli tarkoitus myös antaa suuntaa sille, miltä käyttöliittymä visuaalisesti voisi näyttää. Pääpaino säilytettiin kuitenkin elementtien sijoittelulle, joten

loppujen lopuksi visuaalinen suunnittelu jäi hieman taka-alalle. Käyttöliittymän ensimmäiset luonnokset on esitetty liitteissä 1 ja 2.

Pidin lähtökohtaisesti käyttöliittymän tärkeimpänä asiana esittää kelaimen sijainti mahdollisimman selkeästi ja kiinnittävän ensimmäisenä huomion käyttöliittymässä, heti järjestelmän virheiden jälkeen. Tästä syystä pyrin pitämään käyttöliittymän värimaailman mahdollisimman neutraalina, lukuun ottamatta kelaimen sijainnin osoitinta. Olin myös epävarma siitä, kannattaako kelaimen sijainti esittää pysty- vai vaakasuunnassa, sillä molemmissa on sekä hyvät, että huonot puolensa. Vaakasuunnassa esitettynä osoittimen liike kuvaa paremmin sitä, missä suunnassa kelain oikeasti liikkuu järjestelmässä, mikä helpottaa operaattoria hahmottamaan paremmin kelaimen aseman. Tämä saattaa kuitenkin tuoda sekaannuksia kelaimen liikkeen suunnasta, jos käyttöliittymä onkin sijoitettu ns. ”väärälle puolelle”, eli vaikka kuvassa näyttäisikin, että kelain on ohjattu hieman keskikohdan vasemmalle puolelle, onkin se tosiasiassa hieman keskikohdan oikealla puolella. Halusin, että tätä mahdollista sekaannusta ei tule, joten päätin esittää kelaimen liikkeen pystysuunnassa, jolloin ei ole väliä minne kosketusnäyttö asennetaan.

Huomioitavaa on myös, että sanoja ”Vasen” ja ”Oikea”, ei ollut tarkoitus käyttää varsinaisessa käyttöliittymässä sekaannusten välttämiseksi. Syynä tähän on se, että oikea ja vasen riippuvat aina katsojasta, minkä vuoksi näiden tilalla oli tarkoitus käyttää kuvaavampia sanoja, kuten esimerkiksi ”Operaattorin puoli” ja ”Ohjainten puoli”.

7.2 Työn tekeminen vaiheittain

Opinnäytetyön tekeminen alkoi valitun ohjelman, eli Beckhoffin Twincat 3:n, asennuksesta ja ohjelmaan tutustumisesta. Ohjelma oli ennestään osittain tuttu JAMKilla aikaisemmin tehdyn harjoitustyön ansiosta, mikä helpotti ohjelman hahmottamista ja varsinkin projektipuun rakenteen ymmärtämistä, sekä uuden projektin luomista.

Itse käyttöliittymän tekeminen aloitettiin aluksi tutustumalla Beckhoffin omiin koulutusmateriaaleihin ja esimerkkeihin aiheesta. Varsinkin Beckhoffin esimerkeistä oli suuri apu oman käyttöliittymän luomisessa. Kun tässä vaiheessa oli päätetty, että

käyttöliittymän tuli myös mahtua toimimaan Beckhoffin kosketuspaneeleissa, oli alkuperäiset suunnitelmat unohdettava ja mietittävä käyttöliittymän rakenne täysin uusiksi.

Kun käyttöliittymän uusi rakenne oli mietitty valmiiksi, aloitettiin käyttöliittymän teko Twincat 3 -ohjelmalla. Käyttöliittymässä päätettiin käyttää alkuperäisen suunnitelman tietoja, mutta jakaa ne useammalle välilehdelle, jotta käyttöliittymä mahtuisi järkevästi paneelille. Käyttöliittymän rakenteeseen otettiin mallia Beckhoffin koulutusmateriaalien esimerkeistä.

Käyttöliittymän rakentaminen aloitettiin tekemällä ensimmäisenä käyttöliittymän pääsivu, jossa esitetään kaikki tärkeimmät tiedot sekä haluttu välilehti. Valittu välilehti esitetään pääsivulla olevassa laatikossa. Kun käyttöliittymän pääsivu oli rakennettu valmiiksi, keskityin haluttujen välilehtien suunnitteluun ja toteuttamiseen. Jokainen välilehti luotiin omaksi ”Visualization Object” -tiedostoksi, joita sitten kutsuttiin näkymään pääsivussa olevaan laatikkoon välilehden valintapainikkeilla. Välilehtien suunnittelussa tuli ottaa huomioon välilehden koko ja tehdä niistä kooltaan sellaisia, että ne mahtuvat pääsivun laatikkoon. Koska välilehdillä on tilaa melko vähän, tuli muistaa pitää elementtien määrä vähäisenä, mutta silti pitää fonttikoot riittävän suurina, että käyttäjä saa jokaisesta välilehdestä selvää.

Kielenvaihdon tekeminen käyttöliittymään Twincatilla oli yllättävän yksinkertaista. Twincat 3 kokoaa automaattisesti kaikki erilaiset tekstit käyttöliittymässä käytettävistä objekteista taulukoksi tiedostoon nimeltä ”GlobalTextList”. Tiedosto itsessään sijaitsee samassa projektipuussa kuin käyttöliittymäkin. Taulukkoon pystyy itse lisäämään kieliä oman mielensä mukaan, jolloin ohjelma yksinkertaisesti lisää uuden sarakkeen ”GlobalTextList”-tiedostoon, mihin voi itse kirjoittaa käännökset kustakin tekstistä. Taulukon pystyy myös helposti kääntämään Excel-tiedostoksi, jolloin taulukon täyttäminen sujuu huomattavasti mukavammin. Excelissä täytetyn taulukon sisällön voi lopuksi viedä takaisin projektiin yhtä helposti.

Kun välilehdet ja pääsivu olivat suunniteltu ja visuaalisesti valmiit, oli aika miettiä tarvittavien muuttujien linkkaukset oikeisiin objekteihin. Onnekseni Vision Systemsillä oli menossa kyseisen järjestelmän testauksia, joten pääsin tutkimaan järjestelmää ja ohjelmaa sen ollessa käynnissä, mikä helpotti hahmottamaan ohjelmassa olevien

muuttujien tehtäviä. Tein myös harjoitusmielessä pienen demo-ohjelman järjestelmälle. Demo-ohjelma sisälsi vain pääsivun, jolla oli setpointin asetus ja kaksi pylvästä, jotka kuvasivat setpointin arvoa ja mitattua reunan arvoa. Tajusin myös demo-ohjelman ansiosta, että pelkkä muuttujien linkkaus ei riitä, vaan jotta käyttöliittymä tulisi toimimaan halutulla tavalla, tuli ohjelmaan lisätä myös muutama rivi koodia ja uusia muuttujia. Suoritin demo-ohjelman ohjelmoinnin ladder diagram -ohjelmointikielellä, mutta päädyin kuitenkin käyttämään päätyössä function block diagram -ohjelmointikieltä.

Oma ohjelmointiosuuteni oli kooltaan 17 riviä yksinkertaisia function block diagrameja. Tein ohjelmassani käytännössä datatyyppien muunnoksia, muuttujien vertailua, AND-fucktioita, jotka kertoivat käyttöliittymälle, mikä tila oli valittu, sekä OR-funktioita, joilla ohjattiin käyttöliittymässä olevia merkkivaloja. Datatyyppien muunnoksia tarvittiin käyttöliittymässä olevia pylväsdiagrammeja varten, jotka hyväksyivät ainoastaan array-muodossa olevaa dataa.

Loin käyttöliittymää varten yhteensä 16 muuttujaa, joista kolme oli array-tyyppisiä ja loput boolean-tyyppisiä. Array-tyypin muuttujiin kopioitiin mitatun reunan arvo, sauva-anturin aseman arvo ja setpointin arvo, joita sitten käytettiin käyttöliittymän pylväsdiagrammeissa. Boolean-tyyppisiä muuttujia puolestaan käytettiin ilmaisemaan profibus-yhteyden tilaa (päällä vai pois), kumpi reuna on valittu ohjauspuoleksi, sekä onko kameroita enemmän kuin yksi. Loppuja boolean muuttujia käytettiin erilaisten virheiden esittämiseen.

Käyttöliittymän oltua mielestäni hyvällä mallilla, päätin siirtää sen pääprojektiin. Käyttöliittymää oltiin siis tähän asti tehty erillisessä projektissa ja tässä vaiheessa siirrettiin ensimmäistä kertaa pääprojektiin, jossa sijaitsi koko projektin ohjelmallinen osuus.

Tähän mennessä käyttöliittymää oli testattu ainoastaan Beckhoffin logiikan kanssa useita kertoja käyttöliittymän luomisen ohella, mutta ei kertaakaan yhdessä muiden toimilaitteiden kanssa. Tässä vaiheessa todettiin siis käyttöliittymän olevan valmis testiin, jossa käytettiin logiikan lisäksi myös kaikkia järjestelmään kuuluvia toimilaitteita ja testiympäristön rakentaminen aloitettiin Vision Systemsin tiloissa. Testiympäristöön kuului paineilmalla ohjattava kaksitoiminen sylinteri, Beckhoffin CX9020-

logiikka, Vision Systemsin konenäkökamera, Beckhoffin CP6606-paneeli-PC sekä kaksi venttiiliä, joista toinen oli käsin säädettävissä ja toista ohjattiin milliampeeriviestillä. Testissä sylinterin männän asema kuvasi metallirainan reunaa ja tätä asemaa mitattiin Vision Systemsin kameralla. Venttiilien tehtävänä oli säätää sylinterin päihin ohjattavan paineilman määrää. Käsikäyttöinen venttiili asetettiin vakiopaineeseen, joka riittää männän työntämiseen. Toista venttiiliä ohjasi logiikka, joka laski tai lisäsi ilmanpainetta, jotta mäntä pääsi liikkumaan joko oikealle tai vasemmalle.

Testeissä ilmeni muutamia ongelmia, isoja ja pieniä, mutta loppujen lopuksi kaikki ongelmat saatiin jollain tasolla korjatuiksi niiden kahden päivän aikana, jolloin tätä testiä suoritettiin. Suurimpina ongelmina olivat offsetin asettaminen ja oikean järjestelmän tilan tunnistaminen.

Järjestelmän tilan tunnistuksella tarkoitetaan sitä, että tunnistetaan, onko järjestelmään kytketty profibus vai ei sekä sitä, että mitataanko kameroilla metallirainan toisen reunan asemaa vai keskipisteen asemaa. Näitä tietoja tarvittiin käyttöliittymässä, jotta osattiin piilottaa oikeita elementtejä käyttäjältä, esimerkiksi kun profibus on kytketty järjestelmään, tuli offsetin asetuspainike piilottaa. Ohjelmassa oli olemassa lähes jokaisesta eri tilasta jokin muuttuja, joita pystyttiin hyödyntämään, joten ainoana ongelmana oli näiden muuttujien löytäminen ohjelmasta.

Käyttöliittymä oli tässä vaiheessa lähes valmis ja viimeisenä asiana oli saada käyttöliittymä toimimaan myös perus nettiselaimilla, kuten Internet Explorer, Mozilla Firefox ja Google Chrome. Web-käyttöliittymä vaati toimiakseen ilmaisen lisäosan, jonka sai ladattua Beckhoffin www-sivuilta. Lisäosan asennuksen jälkeen, Twincat 3 -ohjelmaan ilmestyy automaattisesti uusi tiedosto tai objekti, nimeltään ”WebVisualization, minkä voi lisätä omaan projektiin. Objekti luo automaattisesti tarvittavat tiedostot projektiin, joiden avulla käyttöliittymää pystytään käyttämään nettiselaimella. Kaiken ollessa kunnossa, pääsi käyttöliittymään käsiksi kirjoittamalla selaimen osoiteriville: http://localhost/Tc3PlcHmiWeb/Port_xxx/Visu/webvisu.htm, jossa localhostin tilalle kirjoitettiin laitteen nimi ja Port_xxx tilalle oikea portin numero (oletuksena 851). Käyttöliittymän toimiessa nettiselaimen kautta, todettiin käyttöliittymän olevan omalta osaltani valmis.

7.3 Ongelmat

Työssä vastaan tulleet ongelmat ilmestyivät lähinnä käyttöliittymän toteutuksen aikana, jotka johtuivat joko omasta virheestä, tai joissain tapauksissa käytetyn ohjelman virheistä. Tässä kappaleessa on tarkoitus käydä läpi kaikki työn aikana eteen tulleet ongelmat ja näiden ratkaisut kootusti.

Ensimmäinen ongelma itse käyttöliittymän tekemisessä oli Twincat 3 -ohjelman kaatuminen aina kun uuteen projektiin lisättiin Visualization-objekti, joka toimii projektin visuaalisena käyttöliittymänä. Visualization-objektiin on siis tarkoitus rakentaa käyttöliittymä, joten se oli koko työn kannalta tärkein objekti. Ongelma poistui vasta Twincat 3 -ohjelman uudelleen asennuksella, jolloin ohjelma asennettiin sen ehdottamaan oletuskansioon. Ohjelman asennettiin aluksi omavalintaiseen kansioon, mistä syystä ohjelma ei ilmeisesti toiminut oikein.

Paneeli-PC:tä käytettäessä, nousi ensimmäiseksi ongelmaksi käyttöliittymässä käytettyjen tekstien fonttikoko sekä fontin tyyli. Tämä vaikutti osittain moneen painikkeeseen ja muihin objekteihin, jotka sisälsivät tekstejä, sillä tekstien fonttikokoa kasvatettaessa, tuli myös tekstejä sisältävien objektien kokoa kasvattaa. Suuremmaksi ongelmaksi kuitenkin muodostui fontin tyylin muokkaaminen paneeli-PC:lle. Vaikka fontin tyylin pystyi vaihtamaan ja muutos näkyi myös normaalilla tietokoneella, niin paneeli-PC:llä fontti ei kuitenkaan vaihtunut. Ilmeisesti Beckhoffin paneeli-PC:llä on ainoastaan yksi vakio fontin tyyli, jota ei pysty vaihtamaan, joten jouduimme tyytymään tähän vakiofonttiin.

Käyttöliittymässä on käytetty lukuisia led-objekteja, joita käytetään lähinnä järjestelmän havaitsemien virheiden ilmaisemiseen. Myöskään näiden led-objektien käyttö ei sujunut ongelmitta paneeli-PC:n kanssa. Tavallisella PC:llä pyörivässä käyttöliittymässä nämä led-objektit näkyivät ja toimivat oikein, mutta paneeli-PC:llä led-objektit jostain syystä eivät näkyneet ollenkaan. Ongelma ratkesi vaihtamalla kaikki led-objektit tavallisiksi ympyrä-objekteiksi, joiden väriä muutettiin virheen ilmaisemiseksi.

Yksi alkuperäisten suunnitelmien jälkeen lisätyistä välilehdistä oli nimeltään ”Profibus IOt”, jossa esitettiin eriteltyjä tietoja profibus liikenteestä tekstikenttä-objekteilla.

Ongelmaksi tämän välilehden kanssa nousi se, että vaikka kaikkiin tekstikenttä-objekteihin oli linkattu oikeat muuttujat, joita näiden tekstikenttien tuli näyttää, pysyivät kaikki tekstikentät silti tyhjinä käyttöliittymän ollessa käynnissä. Lyhyen tutkimisen jälkeen selvisi, että tekstikenttiin tuli myös linkkauksen lisäksi kirjoittaa %s, %i tai %b, riippuen siitä mitä tietomuotoa tekstikentässä haluttiin esittää. %s jos haluttiin esittää string-muotoista tietoa, %i jos haluttiin esittää integer-muotoista tietoa ja lopuksi %b mikäli haluttiin esittää boolean-muotoista tietoa.

Ongelmia tuotti myös käyttöliittymän liittäminen pääprojektiin, sillä pääprojektista puuttui kokonaan Visualization manager -objekti. Visualization manager hallinnoi projektissa Visualization -objekteja, minkä vuoksi Visualization managerin on pakko olla projektissa. Syystä, joka ei koskaan selvinnyt, Twincat 3 ei antanut lisätä Visualization manager -objektia pääprojektiin millään tavalla. Ainoa ratkaisu oli siis luoda täysin uusi projekti, jonne kopioitiin kaikki pääprojektin tiedostot sekä erikseen rakentamani käyttöliittymä. Tähän uuteen projektiin Twincat 3 antoi lisätä Visualization manager -objektin, mikä ratkaisi tämän ongelman.

Tässä uudessa projektissa ilmeni heti ongelma, mitä ei aikaisemmin ollut. Käyttöliittymän oikeassa alareunassa on Suomen ja Englannin lippujen kuvakkeet, joista käyttäjä pystyy vaihtamaan käyttöliittymän kielen. Uuden projektin luomisen jälkeen, nämä kuvakkeet eivät näkyneet ollenkaan paneeli-PC:llä, vaikka ne tavallisella PC:llä näkyivätkin edelleen normaalisti. Syyksi paljastui väärä kansio, josta lippujen kuvakkeita yritettiin hakea. Tavallinen PC ei välittänyt, mistä lippujen kuvakkeita haetaan, kunhan ne vain sijaitsivat koneen muistissa, mutta paneeli-PC vaatii, että lippujen kuvakkeet sijaitsevat samassa projektikansiossa, kun muukin projekti. Lippujen kuvakkeiden siirtäminen oikeaan kansioon poisti ongelman ja nämä kuvakkeet toimivat jälleen normaalisti.

Käyttöliittymän rakentaminen sujui tästä eteenpäin ongelmitta viimeisiin testeihin asti, joissa paljastui muutamia isoja toiminnallisia ongelmia. Heti testien alussa havaittiin, että käyttöliittymän tuli peittää automaattisesti joitain painikkeita, jotta käyttäjä ei pääse niitä painamaan järjestelmän ollessa tietyssä toimintatilassa. Esimerkiksi jos käyttöliittymään on kytketty profibuskaapeli, käyttäjän ei tarvitse vaihtaa järjestelmän offset-arvoa, vaan se tulee järjestelmälle profibuksen kautta. Myöskään mi-

tattavaa reunaa ei pidä pystyä vaihtamaan, mikäli järjestelmässä on ainoastaan kytketty yksi VisiSmart konenäköanturi. Ongelma ratkaistiin lisäämällä projektiin uusia boolean-tyyppisiä muuttujia, joita seuraamalla käyttöliittymä osasi peittää tarvittavat painikkeet ja indikaattorit.

Yhden objektin piilottaminen osoittautui kuitenkin ongelmalliseksi, sen puutteellisten ominaisuuksien vuoksi. Normaalisti objektin ominaisuuksista löytyy kohta, nimeltään ”Invisibility”, johon voidaan linkata boolean-tyyppinen muuttuja. Jos tämän muuttujan arvo on tosi, objekti piilotetaan. Tämä ominaisuus kuitenkin puuttui pylväsdiagrammi-objekteista, minkä takia tätä objektia ei voitu piilottaa tavallisin keinoin. Ongelma ratkaistiin asettamalla tämän pylväsdiagrammin päälle täysin taustan värinen tyhjä laatikko, jota puolestaan piilotettiin, kun haluttiin tuoda pylväsdiagrammi esille.

Toinen testeissä havaittu merkittävä virhe käyttöliittymän käytön kannalta oli offset-arvon asettaminen järjestelmälle. Järjestelmä seuraa offset-arvoa samasta muuttujasta, jonka arvoa pyrittiin myös käyttöliittymän kautta vaihtamaan. Muuttujan arvo ei kuitenkaan vaihtunut, vaikka ohjelmallisesti kaikki olikin kunnossa. Ohjelman tarkempi tutkiminen paljasti, että muuttuja, johon offsetin arvoa yritettiin käyttöliittymän kautta kirjoittaa, hakikin arvonsa toisesta ohjelman sisäisestä muuttujasta. Eli käyttöliittymän kautta valittu offsetin arvo kirjoittui kyllä kyseiseen muuttujaan, mutta vain hyvin pieneksi hetkeksi, jonka jälkeen ohjelma korvasi käyttöliittymältä tulleen arvon tällä toisella ohjelman sisälle määritellyllä muuttujan arvolla. Ongelma poistui kun käyttöliittymällä valittu offssetin arvo kirjoitettiin tähän ohjelman sisäiseen muuttujaan.

Viimeisenä pienenä ongelmana käyttöliittymässä havaittiin, että offsetin arvolle ei ollut määritelty ylä- tai alarajaa. Koska Offset-arvolle oli varattu ohjelmassa 16 bittiä, joista yksi ilmaisee arvon etumerkin (plus tai miinus), oli näillä biteillä esitetty maksimi-/minimiarvo +-32768. Mikäli tuo maksimiarvo ylitettiin tai minimiarvo alitettiin, pyörähtivät bitit niin sanotusti ympäri, eli laskeminen aloitettiin uudestaan nollasta. Ongelma pystyttiin kuitenkin ratkaisemaan asettamalla ylä- ja alaraja offsetin asetusarvolle.

8 Työn tulokset

Työn tuloksena saatiin toimiva käyttöliittymä Vision Systemsin mittaus- ja reunantunnistusjärjestelmälle, joka täyttää jollain tasolla kaikki työn alussa määritellyt tavoitteet. Koska käyttöliittymää päästiin testaamaan ainoastaan yhden kameran järjestelmässä, ei voida täysin varmasti sanoa, että käyttöliittymä toimii oikein myös muissa toimintatiloissa. Voidaan ainoastaan todeta, että teoriassa käyttöliittymän tulisi toimia oikein jokaisessa mittauksen toimintatilassa, sillä käyttöliittymä sisältää ohjelmallisesti tarpeelliset asiat näissä tiloissa toimimiseen.

Valmiin käyttöliittymän esittely

Valmiin käyttöliittymän pääsivu kaikkine välilehtineen on esitetty tämän raportin liitteissä 3-7. Käyttöliittymä koostuu käytännössä kahdesta osasta: pysyvästä pääsivusta sekä valitusta välilehdestä. Pääsivu pitää sisällään kaksi tekstikenttää, offsetin asetuspainikkeen, virhe-ledin, välilehtien valintapainikkeet, kielen vaihtopainikkeet sekä laatikon, jossa näytetään valittu välilehti. Tekstikentissä esitetään mitatun reunan positio tai sauva-anturin asema (riippuen järjestelmän tilasta) ja järjestelmän offsetin arvo. Offsetin asetuspainikkeella käyttäjä voi vaihtaa järjestelmän offsetin arvon ja välilehtien valintapainikkeiden avulla käyttäjä pääsee vaihtamaan pääsivulla esitetyn välilehden. Virhe-ledi ilmaisee yleisesti, että järjestelmässä on jokin virhe tai virheitä. Virhe-ledin syttyessä punaiseksi, syttyy myös yksi tai useampi välilehtien valintapainikkeista punaiseksi, ilmoittaen näin käyttäjälle miltä välilehdeltä virhe tai virheet ovat peräisin.

Liitteessä 3 on esitetty käyttöliittymän aloitusnäkymä. Aloitusnäkymän välilehdeksi on valittu Columns-välilehti, joka pitää sisällään kolme tai kaksi pylvästä, riippuen järjestelmän tilasta. Vasemmalta katsottuna kahdessa ensimmäisessä pylväässä esitetään offsetin arvoa ja mitatun reunan position arvoa. Vasemmanpuoleisessa pylväässä esitetään joko lineaarisensorin position arvoa tai todellisen aseman arvoa, riippuen järjestelmän tilasta.

Liitteessä 4 on esitetty System State -välilehti. Tällä välilehdellä on ainoastaan ledmerkkivaloja, joilla ilmaistaan järjestelmän kannalta tärkeitä asioita. Välilehden oike-

alla puolella olevat kolme lediä ilmaisevat, mikä ohjaustila järjestelmässä on kulloinkin valittuna. Vaihtoehdot ovat manuaali, automaatti tai keskitys. Vasemmalla puolella on ylhäältä alkaen esitetty profibus-kommunikaation tila, järjestelmän virhetila, mittausvirheen tila sekä hätäseis-tila.

Liitteessä 5 on esitetty Other Information -välilehti. Tämä välilehti pitää sisällään ainoastaan tekstikenttiä, joissa esitetään käyttäjän kannalta oleellisia lukuja. Nämä luvut ovat ylhäältä alkaen: levyn paksuus, toleranssi plus, toleranssi miinus, muilta offset-kytkimiltä annettu arvo, mitattu kokonaisleveys, sekä venttiilin setpoint. Toleranssi plus ja miinus -arvoilla voidaan kertoa käyttäjälle esimerkiksi metallirainan sallittu maksimi- ja minimipaksuus tai sallittu maksimi- ja minimiheitto reunan aseman ja offsetin arvon välillä.

Profibus IOs -välilehti on esitetty liitteessä 6. Tämän välilehden tarkoitus on näyttää käyttäjälle profibus-liikenteen sisään- ja ulostulon tietoja. Välilehden vasemmalla puolella on näytetty sisään tulevan profibus-liikenteen tietoja ja oikealla puolella puolestaan ulos menevän profibus-liikenteen tietoja. Tätä välilehteä on tarkoitus käyttää pääasiassa järjestelmän käyttöönotossa sekä huollon yhteydessä.

Liitteessä 7 on esitetty Select Guiding Edge -välilehti, josta voidaan valita järjestelmän ohjaava reuna sekä seurata kameroiden virheitä. Välilehden yläreunassa on kolme painiketta, joilla voidaan valita järjestelmän ohjaavan metallirainaa joko jommankumman reunan avulla, tai rainan keskipisteen avulla. Näiden painikkeiden alapuolella ovat virhe-ledit, joilla kuvataan erikseen metallirainan reunoja seuraavien kameroiden virheitä. Jokin kameran virhe-ledeistä syttyy jos kamera ei löydä reunaa, valotaso on liian matala tai lämpötila kohoaa liian korkeaksi.

9 Pohdinta

9.1 Tuotteen jatkokehitys

Vaikka tämän työn tuloksena saatu tuote täyttääkin kaikki työssä määritellyt tavoitteet, on tuotetta silti mahdollista kehittää paremmaksi. Käyttöliittymän suunnittelussa ja toteuttamisessa on keskitytty pääosin teknisiin ongelmiin ja yleiseen toimi-

vuuteen, eikä niinkään ulkonäköön. Koska käyttöliittymä on tuote, jota tarjotaan asiakkaalle, olisi sitä hyvä jatkokehittää ”myyvemmän” näköiseksi. Tämä tapahtuisi pääsääntöisesti käyttöliittymän visuaalisen puolen parannuksilla, kuten värjäämällä käyttöliittymä uudelleen sopivammilla väreillä, sekä korvaamalla käyttöliittymän vakioelementit ja -objektit tyylikkäämmillä, tähän järjestelmään erikseen räätälöidyillä elementeillä ja objekteilla.

Jatkokehityksen kannalta olisi myös tärkeää jatkaa tuotteen toimivuuden testausta varsinkin testaamatta jääneiden toimintatilojen osalta. Vaikka käyttöliittymä toimiikin teoriassa kaikissa toimintatiloissa, on hyvin todennäköistä, että käytännössä käyttöliittymässä on vielä virheitä joidenkin järjestelmän toimintatilojen suhteen.

Jatkossa olisi myös hyvä kerätä palautetta käyttöliittymästä asiakkailta, jotka työkseen seuraavat ja ohjaavat samanlaisia järjestelmiä, mitä tässäkin työssä esiintyy. Palautetta kerättiin työn aikana ainoastaan Vision Systemsin työntekijöiltä.

9.2 Luotettavuus

Tämän työn luotettavuus lopputuloksen kannalta on suoraan verrannollinen työhön käytettyjen lähteiden, sekä työn aikana suoritettujen testausten luotettavuuteen. Lähteiden luotettavuus kertoo lähinnä työn viitekehyksen luotettavuudesta, kun taas testit kertovat enemmän työssä suunnitellun käyttöliittymän luotettavasta toiminnasta, eli laadusta.

9.2.1 Lähteiden luotettavuus

Tekniikan kehittyessä jatkuvasti, on myös tärkeää pitää tämän kaltaisissa töissä lähteet hyvin ajan tasalla välttääkseen vanhentuneeseen tietoon viittaamista. Tässä mielessä osa lähteistä on hieman vanhentuneita, kuten vuonna 2000 julkaistut kirjat PID-säätimistä. Toisaalta taas näistä lähteistä etsityt tiedot ovat perustietoja esimerkiksi PID-säätimien toiminnasta tai virityksen teoriasta ja nämä teoriat ovat pysyneet lähes muuttumattomina tähän päivään asti.

Käyttöliittymän suunnittelun teoriaan käytetyt lähteet ovat suurimmaksi osaksi yksittäisen ihmisen kirjoittamia julkaisuja omille nettisivuilleen. Luotettavuuden kannalta tällaiset tekstit ovat useimmiten huonoja, sillä aina on olemassa riski, että kirjoittaja

ei ole perehtynyt aiheeseensa riittävän laajasti tai jättää tarkoituksella sanomatta asioita jotka eivät hänen mielestään kuulosta hyvältä tai tue hänen omia havaintojaan. Myös vaarana on, että kirjoittaja väittää olevansa alan asiantuntuja, vaikka tosiasiassa ei sitä ole. Tässä tapauksessa pidän kuitenkin näitä lähteitä melko luotettavina, sillä kirjoittaja on itse (ainakin sanoo olevansa) ammattinsa puolesta suunnitellut ja rakentanut useita erilaisia käyttöliittymiä. Tutkin myös ennen tähän lähteeseen viittaamista useita muita samankaltaisia yksittäisten suunnittelijoiden julkaisemia tekstejä, joissa jokaisessa toistui samat kohdat suunnittelun tärkeimmistä pointeista.

Lähteiden luotettavuuden kannalta on myös tärkeää pitää lähteet mahdollisimman monipuolisina. Tähän työhön lähteitä on otettu sekä kotimaisista, että ulkomaisista nettijulkaisuista ja kirjallisuudesta.

Osa lähteistä koostuu myös tuote-esittelylehtisistä, joita käytettiin lähinnä mittaus- ja reunanohjausjärjestelmän eri komponenttien esittelyssä. Näiden esittelylehtisten on tarkoitus esitellä tuote ja tehdä siitä mahdollisimman myyvän kuuloinen. Tässä mielessä tämänkaltaiset esittelylehtiset ovat lähteinä välttäviä, sillä niissä saatetaan liioitella tuotteiden ominaisuuksia ja kykyjä. Tuote-esittelyissä usein myös kerrotaan ainoastaan esiteltävän tuotteen hyvistä puolista, samalla kun jätetään huonot puolet kertomatta. Kuitenkin mielestäni on ymmärrettävää käyttää yrityksen omia tuote-esittelyitä, jos esitellään tämän kyseisen yrityksen tuotteita, koska nämä tuote-esittelyt ovat niin sanotusti virallisia julkaisuja tästä tuotteesta.

9.2.2 Testausten luotettavuus

Käyttöliittymälle tehtiin työn aikana lukuisia testejä, joissa pyrittiin varmistamaan, että käyttöliittymä toimii halutulla tavalla, ja että jokainen käyttöliittymän elementti toimii suunnitellulla tavalla. Testauksen kohteena olivat näissä testeissä käyttöliittymän yleinen toiminta, käyttöliittymässä olevien tekstien koko ja fontti, käyttöliittymän kaikkien elementtien ja välilehtien koko ja toiminta, käyttöliittymän toimiminen paneeli-PC:llä, käyttöliittymän toiminta muiden toimilaitteiden kanssa, sekä käyttöliittymää varten tehdyn ohjelmallisen osuuden toiminta.

Suurin osa testeistä tehtiin pelkästään Beckhoffin logiikan ollessa kytkettynä, sillä kaikkein yleisin testauksen syy oli varmistaa jonkin uuden elementin tai objektin toiminta käyttöliittymässä. Tämän selvittämiseen ei tarvinnut kytkeä muita toimilaitteita logiikkaan, mikä teki näistä testeistä helppoja toteuttaa. Iso osa testauksista koostuikin näistä lyhyistä ja nopeista testeistä jossa testattiin jonkin yksittäisen osan toimintaa käyttöliittymässä. Tämänkaltaiset lyhyet testit eivät välttämättä ole luotettava tapa testata koko käyttöliittymän toimintaa, sillä näissä testeissä keskitytään ainoastaan yhden osan toimintaan, eikä kaikkien osien keskinäiseen toimimiseen. Mutta koska ideana ei ollutkaan vielä näissä testeissä todeta koko käyttöliittymän olevan toimiva kokonaisuus, vaan nimenomaan metsästää näitä pieniä virheitä objektien ja elementtien toiminnasta, sanoisin, että tämä oli luotettava tapa testata käyttöliittymää tässä vaiheessa.

Käyttöliittymän viimeisissä testeissä oli tarkoitus testata käyttöliittymän toimintaa kaikkien järjestelmään kuuluvien toimilaitteiden kanssa. Näissä testeissä pääpaino oli käyttöliittymää varten koodatussa ohjelmallisessa osuudessa, käyttöliittymän kautta annettujen käskyjen oikeassa toiminnassa, sekä käyttöliittymässä olevien indikaattoreiden oikeissa lukemissa. Myös web-käyttöliittymän toimintaa testattiin pikaisesti näissä testeissä. Käytännössä käyttöliittymää testattiin siten, että annettiin käyttöliittymälle jokin offset-arvo ja seurattiin pyrkikö järjestelmä ohjaamaan mäntää oikeaan suuntaan. Kun mäntä oli liikkunut uuteen asemaan, tarkasteltiin käyttöliittymässä olevien pylväsdiagrammien ja muiden osoittimien ja indikaattoreiden lukemat ja tarkasteltiin näiden paikkaansa pitävyyttä.

Koska viimeisissä testeissä päästiin testaamaan ainoastaan yhden kameran järjestelmää, jäi käyttöliittymästä testaamatta kokonaan muut mittauksen toimintatilat. Muut tilat tulevat voimaan, kun järjestelmään on kytketty useampi kuin yksi kamera, sekä kun profibus on kytketty järjestelmään. Viimeiset testit olivat siis käyttöliittymän kokonaisvaltaisen toiminnan testauksen osalta riittämättömiä, mutta yhden kameran järjestelmän osalta riittäviä ja luotettavia.

Kaikkien testien luotettavuuteen vaikuttaa myös paljon itse testaajan kokemus ja asiantuntemus testattavasta asiasta. Myös testien määrä sekä testaajien määrä vaikuttavat suurelta osin siihen, kuinka luotettavasti testeissä havaitaan virheitä testatta-

van asian toiminnan kannalta. Tämän työn testit ja koekäytöt on pääasiassa suorittanut minä, joten on hyvinkin mahdollista, että kaikkia virheitä ei ole testeissä huomattu. Pyrin kuitenkin kompensoimaan vähäistä kokemuksen ja asiantuntemuksen määrää tekemällä runsaasti testejä käyttöliittymälle, jotta huomaisin mahdolliset virheet edes toisella tai kolmannella kerralla, niiden ilmestyessä.

Loppujen lopuksi pidän tekemiäni testejä riittävän luotettavina, mutta mielestäni lisätestejä vaadittaisiin ennen kuin käyttöliittymän voi sanoa toimivan moitteettomasti kaikissa mittauksen toimintatiloissa.

Lähteet

4/2 and 4/3 proportional directional valves, direct operated, with electrical position feedback, without/with integrated electronics (OBE). N.d. Datasheet Rexrothin www-sivuilla. Viitattu 26.10.2017. https://md.boschrexroth.com/modules/BRMV2PDFDownload-internet.dll/re29061_2012-11.pdf?db=brmv2&lvid=1168163&mvid=13009&clid=20&sid=9F17089D25659ED823234B4A3767FB02.borex-tc&sch=M&id=13009,20,1168163

Auer, L. 2009. Käyttöliittymän määrittely. Virtuaalinen ammattikorkeakoulu. Viitattu 20.4.2017. http://www2.amk.fi/digma.fi/www.amk.fi/opintojaksot/030308/1111676348138/1111677021119/1111677160787/1111677410876.html

CP6606, Panel PC with ARM Cortex -A8. N.d. Tuote-esittely Beckhoffin www-sivuilla. Viitattu 11.9.2017. https://www.beckhoff.com/english.asp?industrial_pc/cp6606.htm

CX9020, Basic CPU module. N.d. Tuote-esittely Beckhoffin www -sivuilla. Viitattu 29.8.2017. https://www.beckhoff.com/english.asp?embedded_pc/cx9020.htm

Cascadia Metals. N.d. Viitattu 20.3.2017. http://cascadiametalsmb.com/index.php?pageid=43

Davies, E. R. 2004. Machine Vision: Theory, Algorithms, Practicalities. 3. p. Elsevier Science.

Fadeyev, D. 2009. 8 Characteristics Of Successful User Interfaces. Viitattu 26.4.2017. http://usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces/

Fadeyev, D. 2008. Using Light, Color and Contrast Effectively in UI Design. Viitattu 7.5.2017. http://www.usabilitypost.com/2008/08/14/using-light-color-and-contrast-effectively-in-ui-design/

FOPDT integrating model. 2015. Processcontrolstuff.net. Viitattu 1.9.2017. http://www.processcontrolstuff.net/category/mallinnus_ja_simulointi/

Harju, T. & Marttinen, A. 2000. Säätöpiirin virityksen perusteet. Espoo: Otamedia Oy.

InduSoft Web Studio. N.d. Tuote-esittely Indusoftin www-sivuilla. Viitattu 6.7.2017. http://www.indusoft.com/Products-Downloads/HMI-Software/InduSoft-Web-Studio

Indusoft Web Studio 8.0 Quick Start Guide. 2016. InduSoftVideon julkaisema opasvideo uuden projektin aloittamiseen. Viitattu 7.7.2017. https://www.youtube.com/watch?v=kwbpza9-Ciw

Introducing Visual Studio. N.d. Tuote-esittely Microsoftin www-sivuilla. Viitattu 11.7.2017. https://msdn.microsoft.com/en-us/library/fx6bk1f4(v=vs.90).aspx

Lauttamus, M. 2017. Myyntipäällikkö, Vision Systems Oy. Haastattelu 23.10.2017.

Magnetostriction. N.d. MTS Sensors. Viitattu 7.9.2017. http://www.mtssensors.de/fileadmin/medien/download/mts_measuringprinciple.pdf

Martin, S. N.d. Effective Visual Communication for Graphical User Interfaces. Viitattu 26.4.2017. http://web.cs.wpi.edu/~matt/courses/cs563/talks/smartin/int_design.html

Opinnäytetyö-kehittämisprojekti. N.d. Tutkimus ja kehittäminen -kurssin kurssimateriaalia, JAMK.

PID for Dummies. N.d. Control Solutions, Inc. Viitattu 16.3.2017. https://www.csimn.com/CSI_pages/PIDforDummies.html

PID-säädin. 2011. Oppimateriaali. Aalto-yliopisto, Sähkötekniikan korkeakoulu, Säätötekniikka. Viitattu 8.3.2017. http://autsys.aalto.fi/pub/control.tkk.fi/Kurssit/Verkkokurssit/AS-74.2111/simulointi/oppitunti5/pid.html

Powell, V. N.d. Image Kernels, Explained Visually. Viitattu 27.9.2017. http://setosa.io/ev/image-kernels/

Proportional control valves. 2009. Hydraulics & Pneumatics. Viitattu 23.3.20017. http://hydraulicspneumatics.com/other-technologies/book-2-chapter-14-proportional-control-valves?page=1

Proportional valves. 2013. Tietoisku Feston www-sivuilla. Viitattu 26.10.2017. https://www.festo.com/wiki/en/Proportional_valves

SSAB lyhyesti. N.d. Yritysesittely SSAB:n www-sivuilla. Viitattu 10.10.2017. http://www.ssab.fi/ssab-konserni/tietoja-ssabsta/ssab-lyhyesti

Temposonics, R-Series Catalog. 2014. Tuote-esittely MTS Sensorsin www-sivuilla. Viitattu 7.9.2017. http://www.mtssensor.de/fileadmin/medien/download/Catalog/Catalog_R-Series_551303_EN.pdf

Temposonics Sensor Technology - magnetostrictive!. N.d. MTS Sensorin julkaisema video tuotteen toimintaperiaatteesta. Viitattu 7.9.2017. http://www.mtssensor.de/index.php?id=224

Tutkimuksen/opinnäytetyön toteutus, tutkimusmenetelmät. N.d. Tutkimus ja kehittäminen -kurssin kurssimateriaalia, JAMK.

TVT-ajokortti. N.d. Käyttöjärjestelmä ja käyttöliittymä, Käyttöliittymät. Helsingin yliopiston TVT-ajokortin oppimateriaali. Viitattu 20.4.2017. http://blogs.helsinki.fi/tvt-ajokortti/1-tietokoneen-kayton-perusteet/1-1-tietokoneen-toimintaperiaate/kayttojarjestelma-ja-kayttoliittyma/

TwinCAT 3, eXtended Automation. 2012. Beckhoffin tekemä esittelylehtinen. Viitattu 25.5.2017. https://download.beckhoff.com/download/document/catalog/Beckhoff_TwinCAT3_042012_e.pdf

Virkajärvi, M. 2017. Technology manager, Vision Systems Oy. Sähköpostiviesti 7.11.2017

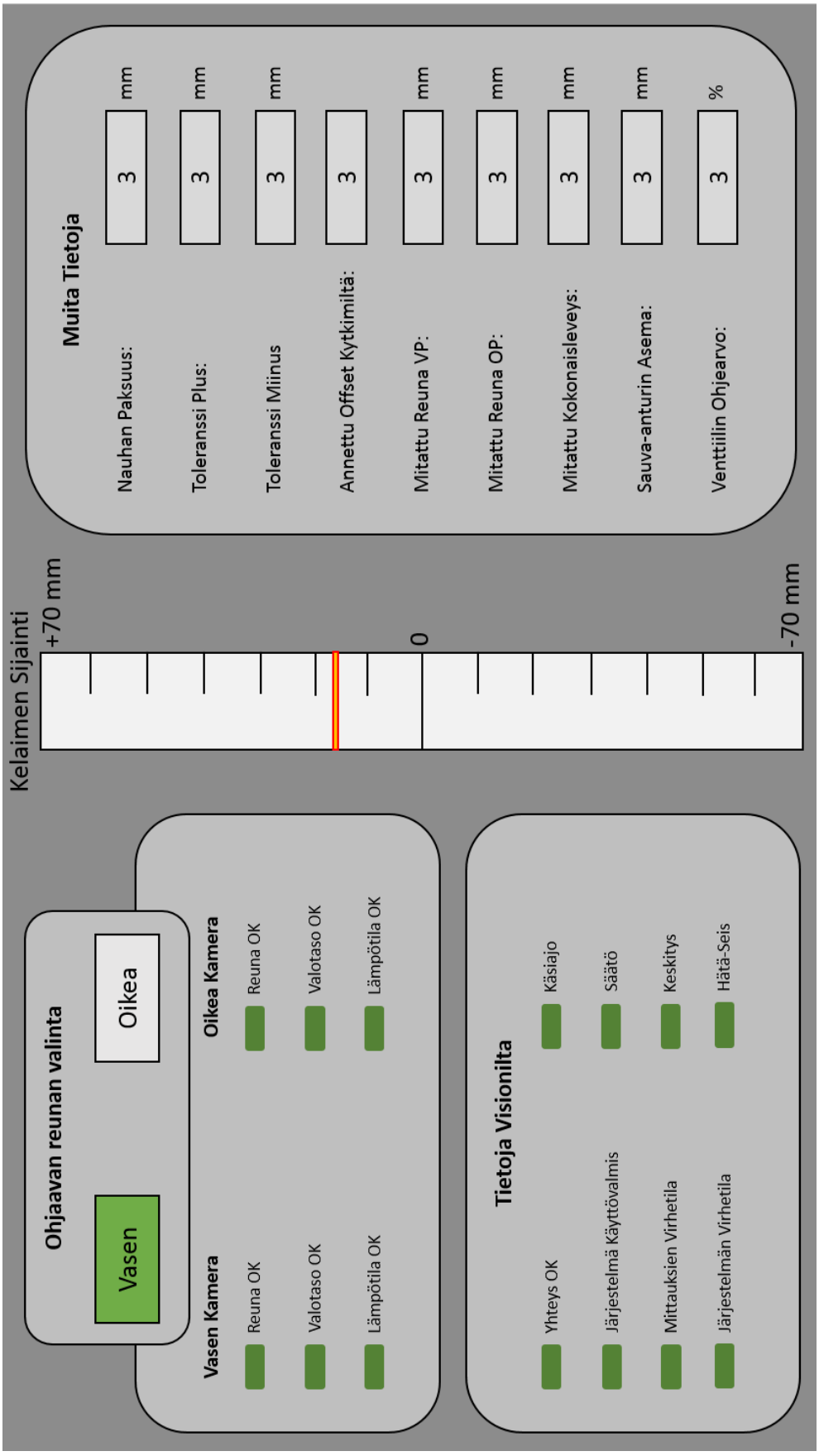
Visi 50 Smart. N.d. Tuote-esittelylehtinen Vision Systemsin www -sivuilla. Viitattu 15.9.2017. http://www.visionsystems.fi/wp-content/uploads/2016/01/VISI_SMART_eng.pdf

Vision Development. N.d. Yritysesittely Vision Systems Oy:sta ja Vision Development Oy:sta. Viitattu 14.7.2017. http://www.visionsystems.fi/wp-content/uploads/2016/01/VISION-DEVELOPMENT-projektit-ja-palvelut-2015.pdf

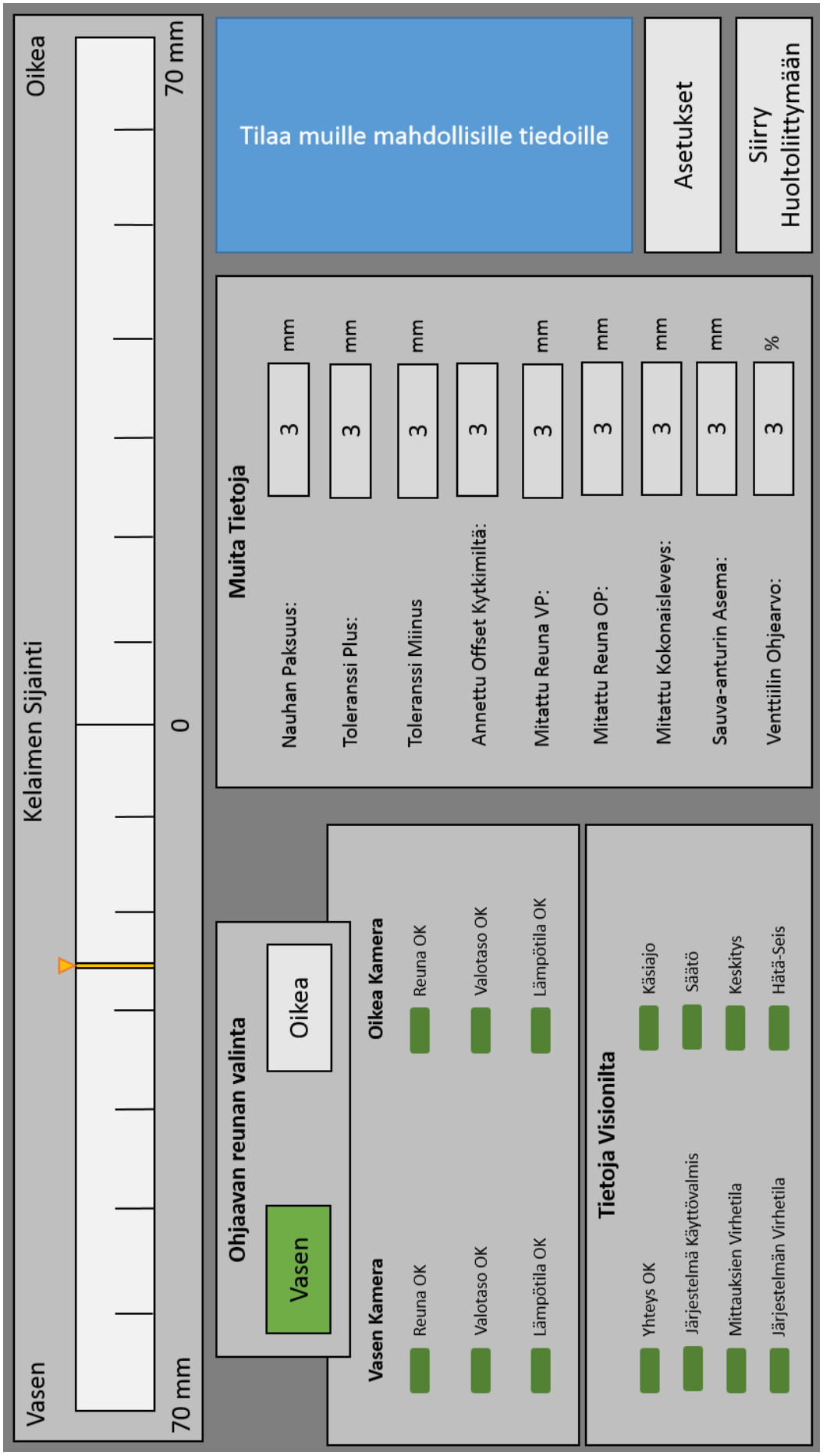
Åström, K. J. 2002. Control System Design, Chapter 6. PID Control. University of California, Department of Mechanical and Environmental Engineering.

Liitteet

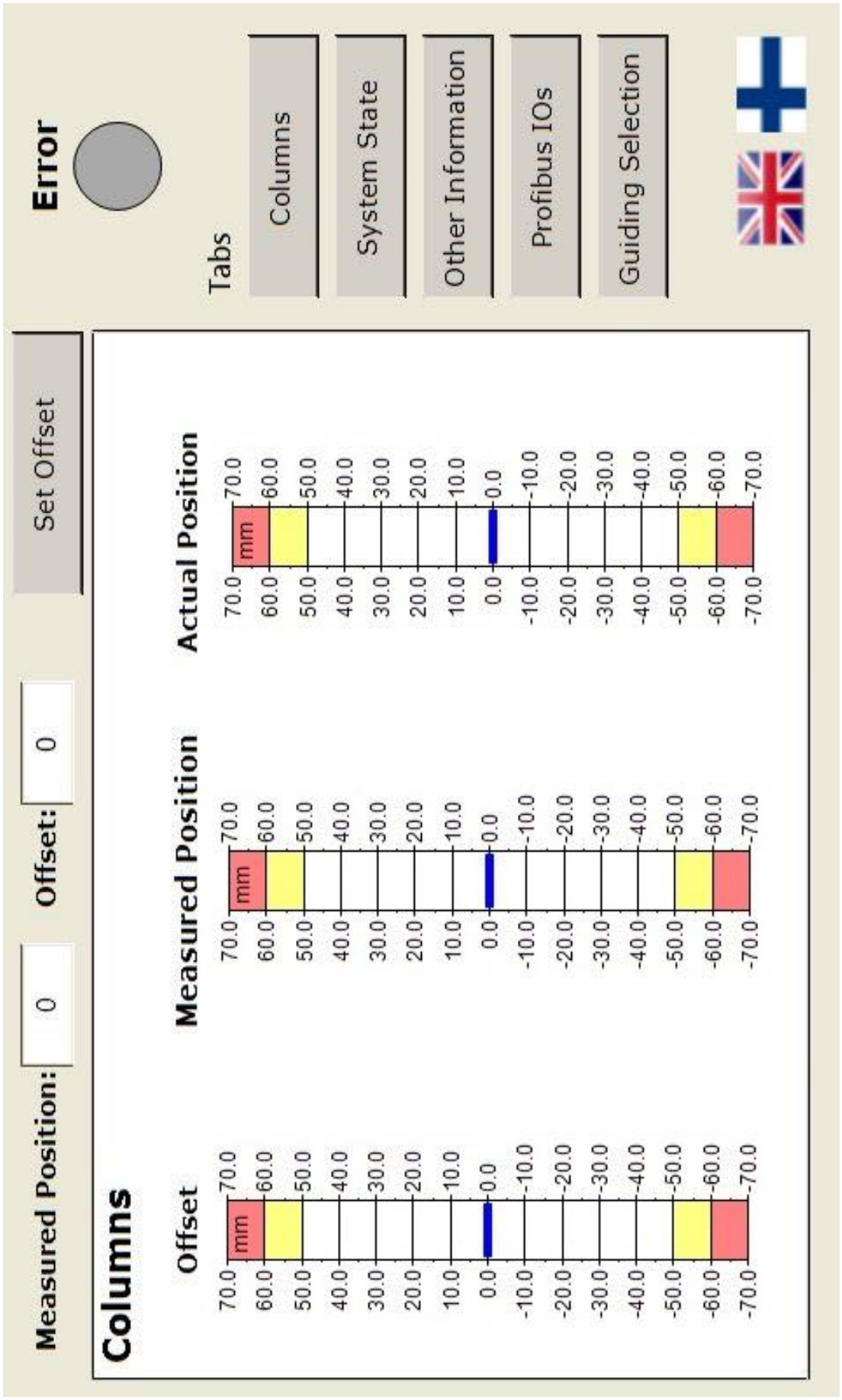
Liite 1. Ensimmäisiä luonnoksia järjestelmän käyttöliittymästä (1)



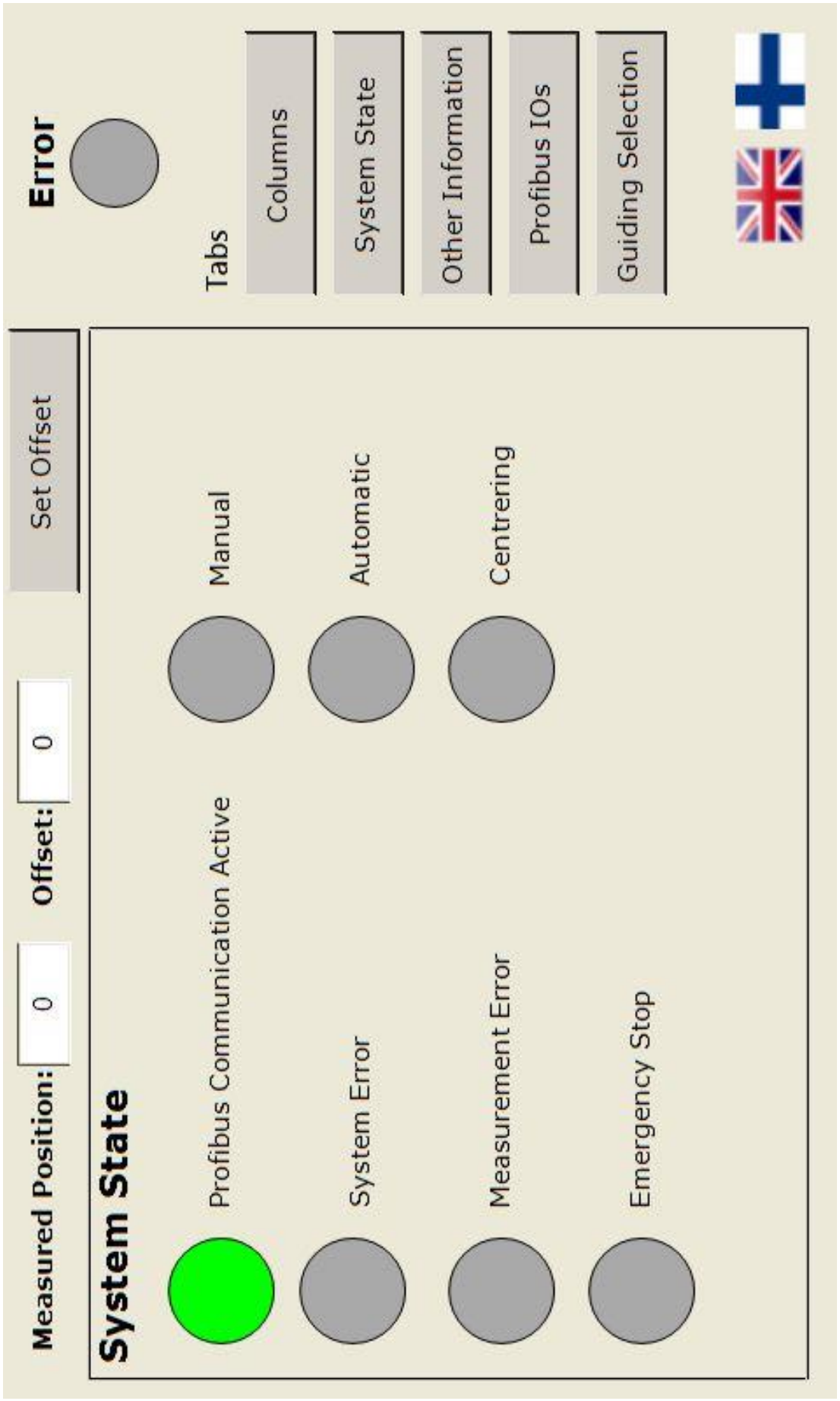
Liite 2. Ensimmäisiä luonnoksia järjestelmän käyttöliittymästä (2)



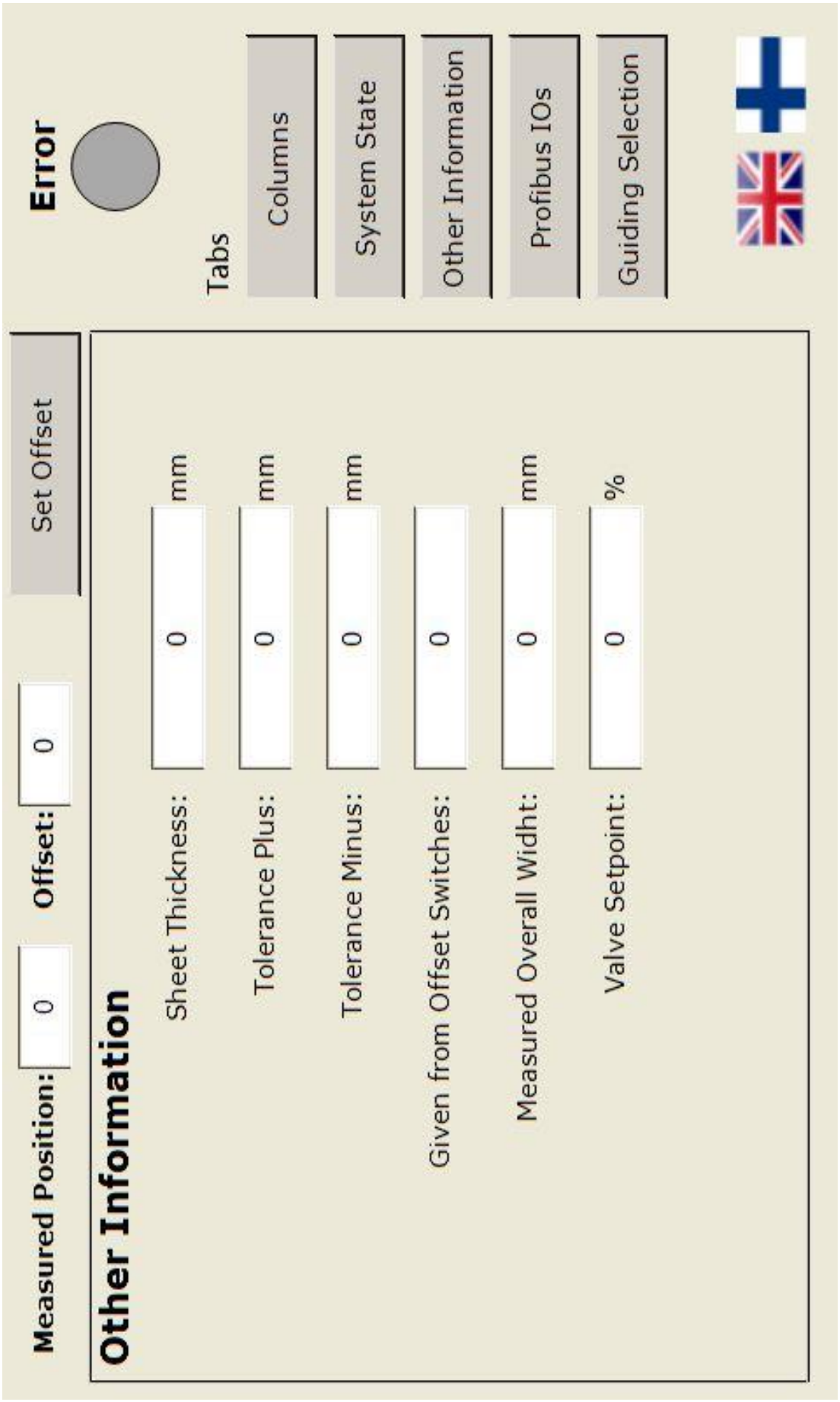
Liite 3. Columns-välilehti



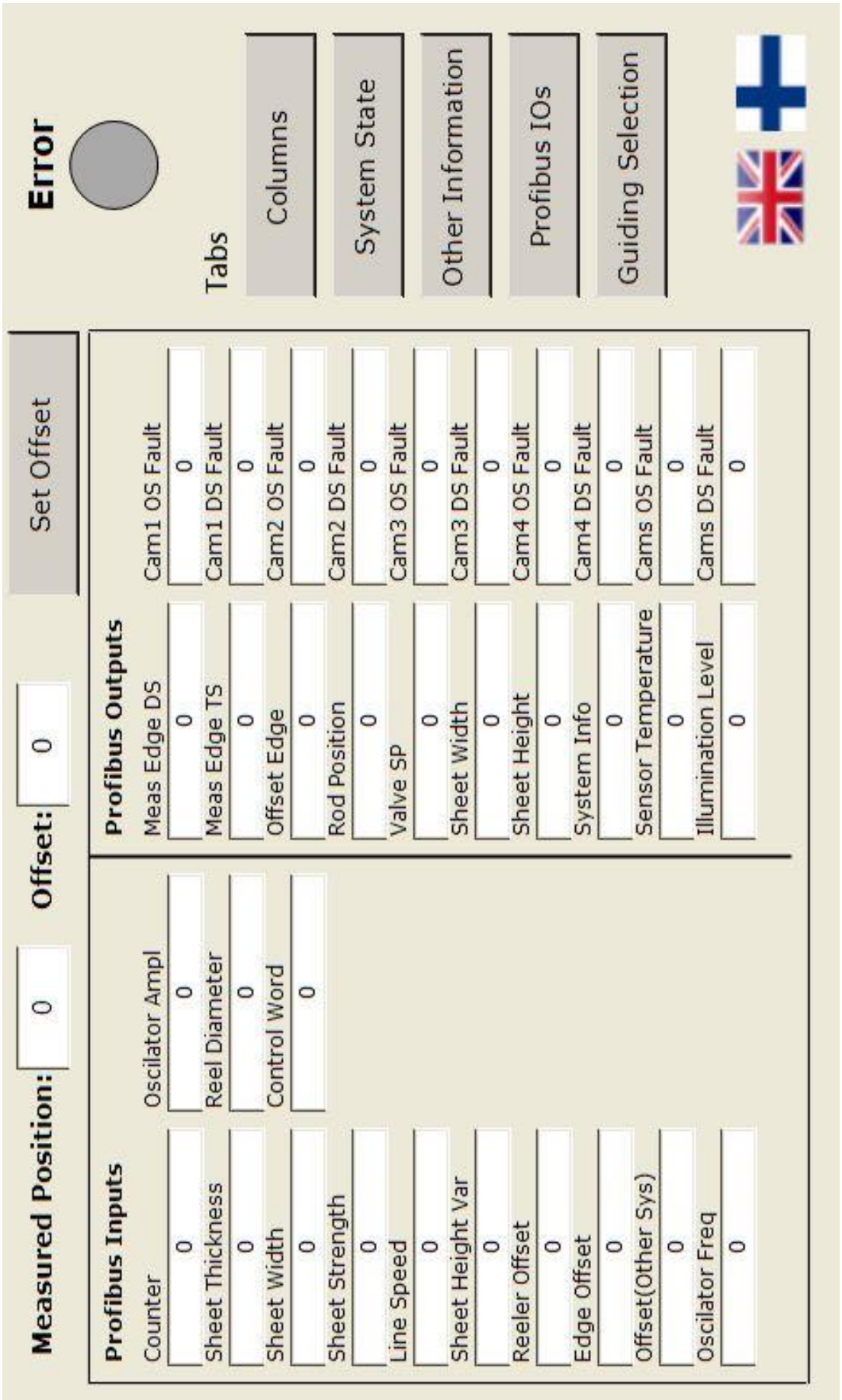
Liite 4. System State -välilehti



Liite 5. Other Information- välilehti



Liite 6. Profibus IOs -välilehti



Liite 7. Guiding Selection -välilehti