VAASAN YLIOPISTO

TEKNILLINEN TIEDEKUNTA

AUTOMAATIOTEKNIIKKA

Janne Koljonen

AUTO1030 Signaalien käsittely

HARJOITUSTYÖOHJE

Sivumäärä: 12

Jätetty tarkastettavaksi: 22.3.2012

Työn tarkastaja Janne Koljonen

SISÄLLYSLUETTELO

SYMBOLI- JA LYHENNELUETTELO

API	Application Programming Interface

1. JOHDANTO

Signaalien käsittely -kurssin suorittaminen koostuu kolmesta osa-alueesta: mikrotenteistä, harjoituksista ja harjoitustyöstä. Arvosana muodostuu yleensä mikrotenttien ja harjoitusten perusteella. Harjoitustyö arvostellaan skaalalla: {hylätty, hyväksytty, kiittäen hyväksytty}. Kiittäen hyväksytty harjoitustyö korottaa kurssin arvosanaa yhdellä.

Ennen harjoitustyötä kurssilla on opittu laajasti signaalinkäsittelyn perusteita. Osaaminen on ollut lähinnä muistamis- ja ymmärtämistasolla. Harjoitustehtävissä on jo jonkin verran sovellettukin osaamista. Harjoitustyön tehtävänä sen sijaan on harjoitella soveltamista, analysoimista, arvioimista ja uuden luomista ongelmalähtöisesti. Harjoitustyössä täytyy perehtyä ongelmaan, osata valita sopivia signaalinkäsittelyn operaatioita, soveltaa niitä ongelmaan, analysoida tuloksia ja arvioida kirjallisesti käytettyä lähestymistapaa ja toteutusta.

Harjoitustyön aihe voi olla mikä tahansa kurssiin liittyvä **käytännön työ**. Kurssin työ ei siis voi olla pelkkä kirjallisuusselvitys. Käytännön osuus on useimmiten ohjelmoimista tai muuta mallintamista, mutta halutessaan se voi suuntautua myös esimerkiksi elektroniikkaan. Ohjelmoimisessa ja mallintamisessa käytetään esimerkiksi Javaa/ImageJ:tä, VHDL:ää tai Matlabia.

Omat aiheet ovat erittäin suositeltavia. Omasta aiheesta tulee keskustella ennakkoon opettajan kanssa. Oma aihe voi liittyä vaikka omaan työpaikkaan, harrastukseen tai johonkin toiseen oppiaineeseen.

2. HARJOITUSTYÖRYHMÄT JA OHJAUS

Harjoitustyön voi tehdä joko yksin tai 2 (joskus 3–4) hengen ryhmissä, mutta kukin tekee **oman yksilöllisen harjoitustyöraportin** ja on siten yksilöllisesti vastuussa oppimisestaan. Ryhmät muodostetaan harjoitustyön ohjauksessa satunnaisesti, jotta osaamisen jakaminen olisi tehokkainta ja kaikki olisivat tasapuolisessa asemassa ryhmien muodostamisessa. Mikäli jollakin on oma harjoitustyöaihe, voi hän koota sitä tekemään oman ryhmän. Harjoitustyön ohjaajana toimii ensisijaisesti Janne Koljonen.

Harjoitustyötä ryhmä tekee pääosin itsenäisesti. Ohjausta annetaan harjoitusryhmäaikoina. Ohjauksen saaminen edellyttää, että on dokumentoinut harjoitustyöraporttiin siihen mennessä tehdyt harjoitustyön osiot.

3. HARJOITUSTYÖN VAIHEET JA AIKATAULU

Tässä luvussa käsitellään yleisesti harjoitustyön vaiheet. Jokaisen työ on kuitenkin yksilöllinen ja voi poiketa tässä esitetystä. Otsikoiden perässä on (ohjauskerta), johon mennessä vaihe tulisi olla tehtynä.

3.1. Aloitus (O1)

Ensimmäisellä ohjauskerralla muodostetaan ryhmät, jaetaan aiheet ja ohjeistetaan harjoitustyön tekemistä. Aiheeseen perehtyminen sekä työnjaosta ja työskentelytavoista sopiminen tulee aloittaa ryhmässä välittömästi.

3.2. Esiselvitys (O2)

Esiselvityksessä perehdytään ongelmaan ja siinä tarvittavaan taustatietoon. Etsi siis tietoa kurssikirjoista, internetistä, jne.

3.3. Määrittely (O2)

Aluksi tulee kirjoittaa kirjallinen määrittely (spesifikaatio) harjoitustyöstä ja siinä toteutettavasta ratkaisusta. Spesifikaatiossa tulee ilmetä ainakin toiminnallisuuksien ja käyttötapausten yleinen kuvaus (esim. vuokaavioin, käyttötapauskaavioin tai sanallisesti) ja halutut/oletetut vasteet/tulokset. Määrittelyä pitäisi voida käyttää testausvaiheen ohjenuorana.

3.4. Suunnittelu (O3)

Suunnittelu perustuu spesifikaatioon. Suunnitteluvaiheessa työ ositetaan rakenteellisesti ja mieluiten myös työryhmän sisällä. Suunnitteluvaiheessa mietitään, millä menetelmillä ongelmaa (harjoitustyötehtävää) yritetään ratkaista ja hahmotellaan, miten ratkaisualgoritmit toteutettaisiin.

3.5. Toteutus (O4)

Toteutetaan suunnitellut osat. Tehdään alustavia testauksia ja palataan tarvittaessa suunnitteluvaiheeseen, tehdään tarvittavat muutokset suunnitelmaan ja palataan taas toteutukseen.

3.6. Testaus ja analyysi (O5)

Varsinaisessa testausvaiheessa tehdään suunnitelmallisesti erilaisia testejä, joilla mitataan toteutuksen ominaisuuksia suhteessa valittuihin muuttujiin.

Mitattavia ominaisuuksia voivat olla esimerkiksi:

- tarkkuus,
- nopeus ja
- käytettävyys.

Testauksen muuttujia puolestaan voivat olla esimerkiksi:

- kohinan määrä,
- erilaiset syötteet (kuvat, tms.) ja
- virheelliset syötteet.

Testauksista pidetään huolellisesti kirjaa. Tulokset tulee analysoida sopivalla tavalla käyttäen esimerkiksi tilastollisia suureita (keskiarvo, keskihajonta, tms.) ja tehdä niistä myös laadullinen arvio. Luonnollisestikaan testituloksia ei saa vääristellä, vaan niin hyvät kuin huonot ominaisuudet raportoidaan ja analysoidaan. Ratkaisun puutteiden syistä ja mahdollisista parannuskeinoista tulee esittää arvio.

4. RAPORTOINTI

Kaikki harjoitustyön vaiheet on dokumentoitava riittävällä tarkkuudella, jotta dokumentin lukija ymmärtää piirin toiminnan, voi varmentua suunnittelun loogisesta virheettömyydestä ja saada oikean kuvan testauksen kattavuudesta. Dokumentointi on osa laatujärjestelmää. Laatujärjestelmä ei vielä takaa, että tulos olisi laadukasta vaan että asioiden tila voidaan todentaa. Samoin on harjoitustyössäkin: harjoitustyö voi olla laadukas, vaikka itse ratkaisu ei toimisikaan hyvin, kunhan suunnittelu, toteutus ja testaus ovat *loogisesti* hyvin tehtyjä ja dokumentoitu hyvin. Tällöin toteutuksen heikkous on selitettävissä väärien oletusten ja hypoteesien tekemisellä.

Dokumentoinnissa voi olla kuvia, vuokaavioita, piirikaavioita, koodinpätkiä (ei pitkiä), RTL viewerin -kuvia, testitulostuksia, viittauksia tausta-aineistoon ja kirjallisuuteen jne. Raportin muoto on vapaa, mutta siitä on löydyttävä normaalin tieteellisen tekstin osat soveltuvin osin (otsikko ja tekijätiedot, johdanto ja teoriapohja, toteutuksen kuvaus, testaus, tulokset, johtopäätökset, lähdeluettelo, liitteet). Lisäksi raportissa tulee olla kansilehti, josta käy ilmi mm. tekijä- ja kurssin tiedot.

Kukin ryhmässä kirjoittaa oman raportin eli on henkilökohtaisessa vastuussa oppimisesta ja kurssin suorittamisesta. Mikäli raportissa käytetään lainauksia/osioita toisen ryhmäläisen raportista, tulee osion tekijään viitata normaalin tieteellisen käytännön mukaisesti, jotta voidaan todeta, mikä on raportin kirjoittajan *kontribuutio*.

Raportoinnin tulee olla reaaliaikaista eli uusien suunnitelmien, toteutusten, yms. valmistuttua niistä tehdään merkinnät raporttiin, joka lopuksi viimeistellään vastaamaan lopullista toteutusta.

Raportin kieliasun tulee olla vähintään hyvä. Kieleksi voi valita suomen, ruotsin tai englannin. Harjoitustyön valmistuttua raportti palautetaan tarkastettavaksi.

Mikäli raportissa on korjattavaa, tulee korjausohjeita noudattaa tai perustella, miksi jotakin ei tarvitsisi korjata. **Mikäli opiskelija toistuvasti jättää korjauskehotukset perusteettomasti huomiotta, harjoitustyö hylätään ja annetaan uusi aihe.**

5. AIHEITA

Ryhmäjaon jälkeen kukin ryhmä saa valita jonkin alla olevista aiheista, kuitenkin siten, että samaa aihetta ei olisi kovin monella ryhmällä. Aiheet on ryhmitelty aihepiirien ja toteutustavan mukaan.

ImageJ:llä tehtävissä harjoitustöissä kannattaa tutustua ImageJ API:in (http://rsb.info.nih.gov/ij/developer/api/index.html), josta löytyy kaikki ImageJ:n valmiit luokat ja metodit. Erityisesti luokka `IJ` on hyödyllinen.

Seuraavassa vielä esimerkki, miten pluginissa voi avata kuvia dialogin kautta, miten käyttäjälle voi näyttää tulostuksia viestidialogilla ja miten käyttäjältä voi kysyä merkkijonoa. Avattava kuva on `ImagePlus`-luokan olio, joka näyttäytyy ruudulla omana kuvaikkunana. Kuvankäsittely varten saadaan `ImageProcessor`-olio `getProcessor()`-metodilla. Harjoitustehtävissä ja kurssikirjassa on esimerkkejä, miten `ImageProcessor`-olioita käytetään kuvankäsittelyssä. Huomaa, että `Opener`-luokka on `ij.io`-pakkauksessa, joka tuodaan käyttöön `import`-komennolla. Viestidialogia tarvitset luultavasti tutkiessasi ohjelman toimintaa korjatessasi mahdollisia virheitä.

```
import ij.*;
import ij.process.*;
import ij.gui.*;
import java.awt.*;
import ij.plugin.filter.*;
import ij.io.*;

public class Opener_ implements PlugInFilter {
	ImagePlus imp;

	public int setup(String arg, ImagePlus imp) {
		this.imp = imp;
		return DOES_ALL;
	}

	public void run(ImageProcessor ip) {
		ImagePlus kuva1= IJ.getImage();
		Opener olio=new Opener();
		String vastaus="k";
		while(vastaus.equals("k"))
		{
			olio.open();
			ImagePlus kuva2=IJ.getImage();
			ImageProcessor ip2=kuva2.getProcessor();
```

```
                IJ.showMessage("Ilmoitus","Näytetään uusi kuva");
                vastaus=IJ.getString("Jatketaanko (k/e)?","k");
            }
            IJ.showMessage("Tiedoitus","Loppu!");
        }
}
```

Aiheet:

Kuvankäsittely PC:llä

1. Focus stacking: syvyysterävyyskuvan luominen yhdistämällä eri fokusetäisyydellä otettuja kuvia. http://lipas.uwasa.fi/~TAU/AUTO1030/slides.php?Mode=NoFrame&File=8100Lab.txt&Page=2&MicroExam=Off.
2. Appelsiinin paleltumisen (pakastin) tunnistaminen: valaistaan UV-valolla, kuvataan fluoresenssia näkyvällä valolla. Kuvien ottaminen ja ImageJ-pluginin tekeminen kuvankäsittelyyn.
3. Kaikki tallessa? -konenäkösovellus. ImageJ-plugin, joka tarkistaa, ovatko esimerkiksi työkalut oikeilla paikoillaan, ennen kuin antaa luvan lähettää lentokone huollosta Kanarian-lennolle.
4. Pienesineiden (ruuvien, tms.) lukumäärän laskeminen. ImageJ-plugin.
5. Perunan tms. tilavuuden estimointi kahdesta eri suunnasta otetun kuvan perusteella. ImageJ-plugin.

Äänenkäsittely PC:llä

6. Äänentunnistus. Esim. ihmisäänen tunnistus vaikka spektrin perusteella: referenssinäytteet ja tunnistettavat testinäytteet. Toteutus esim. Matlabilla.
7. Tinnitussuotimen suunnittelu Matlabilla.

Nopea kuvankäsittely FPGA:lla (DigiPikseli: ks. Koljonen, 2010; Koljonen, Björkqvist, Alander, 2010)

8. Kynnystyksen ja morfologisten suotimien (yksiulotteisten) toteuttaminen FPGA:lla: eroosio, dilataatio, avaus, sulku, reunaviiva. Kynnysarvo ja operaatio valittavissa kytkimillä.
9. FIR-suotimen (yksiulotteinen) toteutus FPGA:lla. Kertolaskun voi korvata logaritmilla, jolloin laskut tulevat yksinkertaisiksi (ks. http://lipas.uwasa.fi/~TAU/AUTO1030/slides.php?Mode=NoFrame&File=1200Conv.txt&Page=15 ja siellä viitattu artikkeli).

LÄHDELUETTELO

Koljonen, Janne (2010). AUTO1010 Digitaalitekniikan perusteet, harjoitustyöohje (syksy 201) – Digipikseli [online]. [Siteerattu 5.4.2011]. Saatavana World Wide Webistä: <URL: http://lipas.uwasa.fi/~jako/digis/HTOhje.pdf>.

Koljonen, Janne, Björkqvist, Mats & Alander, Jarmo T. (2010) Online machine vision for elementary engineering courses, in *Proceedings of the 14th Finnish Artificial Intelligence Conference* (STeP 2010), Pahikkala, Väyrynen, Kortela & Airola (eds.), Espoo (Finland), 17–18 Aug. 2010, pp. 62–69. Finnish Artificial Intelligence Society (FAIS).