4. LABORATORIJSKA VAJA (do 7 točk + do 3 točke za poročilo)

Namen vaje je seznaniti študente z različnimi postopki obdelave digitalnih slik, ki jih nato aplicirajo na problem detekcije obrazov. Študenti se najprej seznanijo s predvidenim postopkom detekcije (glej prilogo), ki ga je potrebno udejanjiti, ter algoritmi, postopki ter orodji, ki jih pri tem uporabljajo. Študenti se v okviru prve vaje seznanijo z lastnostmi, karakteristikami in načinom uporabe:

- barvnih filtrov,
- morfoloških operacij (dilatacije, erozije, zapiranja, …),
- konvolucijskih filtrov (glajenja, …),
- postopkov iskanja robov z gradientnimi operatorji (Sobel, …),
- upragovljanja,
- logičnih operacij med slikami,
- geometrijskih operacij (skaliranje, …), in
- posplošene Hough-ove transformacije.

Poleg zgoraj naštetih postopkov se študenti seznanijo še s frekvenčno predstavitvijo slike ter histogrami slik.

Za izvedbo vaje pridejo v poštev Matlab funkcije, ki so del originalne Matlab zbirke programskih orodij »*Image Processing Toolbox*« ter nekaj dodatnih funkcij, skript in programov, ki so jih pripravili sodelavci laboratorija LUKS in so dosegljivi preko spletnih strani predmeta (arhiv na: http://luks.fe.uni-lj.si/sl/studij/GST/vaje/vaja4b.html).

Za uporabo dodatnih Matlab funkcij je potrebno pred pričetkom dela v Matlab okolju dodati poti do direktorijev, kamor je bil prenesen in razširjen arhiv *Vaja4_predloge.rar*. To se izvede bodisi z uporabo ukaza

```
addpath('ustrezna_pot');
```

bodisi s spremembo delovnega direktorija Matlab-a na direktorij, kjer se nahajajo dodatne funkcije.

Osnovo za delo študentov predstavlja skripta `vaja4_predloga.m`, znotraj katere se nahaja ogrodje programske kode za izvedbo postopka detekcije obraza. Skripta `vaja4_predloga.m` izvaja klice funkcij, ki pa jih morajo študenti udejanjiti sami. Vsaka takšna funkcija je znotraj skripte jasno označena s komentarjem `%IMPLEMETIRAJ`. Primer takšnega funkcijskega klica je predstavljen spodaj:

```
%% Morfološka obdelava binarne maske koze
Skin_corrected = clean_skin_mask(Skin);      % IMPLEMENTIRAJ
```

Pri izvedbi funkcij si študenti pomagajo z ogrodjem, ki je vključeno v arhivu *Vaja4_predloge.rar* in določa zahtevani vmesnik (angl. Application Programming Interface - API). Za funkcijo iz prejšnjega primera je API zapisan v glavi datoteke `clean_skin_mask.m` in ga je mogoče prebrati direktno iz Matlaba z ukazom

```
help clean_skin_mask
```

ki vrne:

```
Funkcija popravi binarno masko podrocja koze tako, da zapolni luknje in
podrocje nekoliko razsiri

  Vhod:
      Skin ... binarna slika s prvo oceno maske podrocja koze (velikosti axb)
  Izhod:
      Skin_corrected ... popravljena binarna maska podrocja koze (velikosti
                         axb) brez lukenj; slikovni elementi, ki ustrezajo
                         barvi koze imajo vrednost 1, ostali vrednost 0

  Avtor: Vitomir Štruc
  Datum: 25.4.2012
  Copyright (c) 2012 FE UL
```

Kot testne podatke študenti uporabljajo izbrane slike z zbirke XM2VTS, ki jih lahko prenesejo z naslednje povezave:

http://luks.fe.uni-lj.si/sl/studij/GST/pub/data/

Vsak par študentov pri preizkušanju svojih rešitev uporabi eno, od asistenta določeno sliko obraza.

Naloga 1 (1 točka)

Spišite Matlab funkcijo `skin_color_filter_RGB`, ki s pomočjo posebnega »barvnega« filtra iz slike izloči vse slikovne elemente, ki ne ustrezajo barvi kože. Filter naj kot vhod vzame barvno sliko zapisano v RGB barvnem prostoru (v obliki 3D matrike velikosti *a* x *b* x *3*, kjer sta *a* in *b* višina in širina slike, in je tretja dimenzija barvni prostor) in na izhodu vrne binarno sliko, na kateri beli slikovni elementi z vrednostjo 1 ustrezajo slikovnim elementom z barvo kože in naj črni slikovni elementi z vrednostjo 0 ustrezajo vsem ostalim slikovnim elementom. Pri pisanju funkcije uporabite filter, ki upošteva naslednje pogoje:

$$R>95,\ G>40,\ B>20,\ \max\{R,G,B\}-\min\{R,G,B\}>15,\ |R-G|>15,\ R>G,\ R>B$$

Pri implementaciji si pomagajte z naslednjim zgledom, ki udejanja preprostejši filter z zgolj dvema pogojema:

```
%% Inicializacija izhoda - POZOR: nepreizkušena funkcija
Skin = [];

%% Zacetne operacije
[a,b,c] = size(X);
Skin = zeros(a,b);

%% Filter
R=X(:,:,1);
G=X(:,:,2);
B=X(:,:,3);

for i=1:a
    for j=1:b
      if(R(i,j)>95 & G(i,j)>40)
        Skin(i,j)=1;
      else
        Skin(i,j)=0;
      end
    end
end

%Zgornji filter lahko udejanjimo v eni vrstici
%Skin = R>95 & G>40
```

Opomba: V poročilu podajte in komentirajte celotno programsko kodo spisane funkcije `skin_color_filter_RGB`. Prikažite vhodno in izhodno sliko funkcije (ukaz: imshow), podajte in komentirajte frekvenčni spekter in histogram binarne slike, ki predstavlja izhod udejanjene funkcije ter ga primerjajte s spektrom in histogramom (sive) vhodne slike. Prikažite tudi posamezne barvne komponente vhodne slike.

POZOR: Funkciji `show_histogram` in `show_log_mag_spectrum` prikažeta histogram in spekter sive različice vhodne slike.

Naloga 2 (1 točka)

Spišite Matlab funkcijo `clean_skin_mask`, ki kot vhod vzame binarno sliko področja kože, ki predstavlja izhod funkcije `skin_color_filter_RGB`, to binarno področje obdela z zaporedjem morfoloških operacij:

- polnjenja lukenj ter
- dilatacije

in na izhodu vrne očiščeno področje kože brez lukenj. Pri dilataciji uporabite kvadratni (angl. square) strukturni element velikost 5 slikovnih elementov.

Pri udejanjanju funkcije si pomagajte z Matlab-ovimi funkcijami `imfill`, `strel`, in `imdilate`. Delovanje navedenih funkcij preverite s pomočjo ukaza `help`:

```
help imfill
help strel
help imdilate
```

Opomba: V poročilu prikažite vhodno in izhodno sliko ter komentirajte učinek morfološke obdelave. Podajte in komentirajte še celotno programsko kodo spisane funkcije.

Naloga 3 (1 točka)

Spišite Matlab funkcijo `rgbimage2greyimage`, ki kot vhodni podatek vzame izvirno vhodno RGB sliko obraza in na izhodu vrne sivo sliko. Funkcijo udejanjite s pomočjo Matlab funkcije `mean`, pri čemer upoštevajte, da sivi nivoji sive slike predstavljajo povprečno svetilnost vseh treh RGB barvnih komponent. Pri pisanju funkcije ne pozabite, da mora rezultat funkcije (t.j., izhodna siva slika) predstavljati sliko z elementi tipa `uint8`. Rezultat povprečenja je torej potrebno pretvoriti na naslednji način:

```
uint8(mean(…))
```

Opomba: V poročilu prikažite vhodno in izhodno sliko funkcije in podajte kodo spisane funkcije.

Naloga 4 (1 točka)

Spišite Matlab funkcijo `smooth_grey_image`, ki kot vhodni podatek vzame sivo sliko obraza in na njej izvede glajenje z Gaussovim filtrom velikosti 7x7 slikovnih elementov in standardno deviacijo σ=2. Funkcija naj na izhodu vrne glajeno različico vhodne sive slike.

Pri udejanjanju vaše funkcije uporabite Matlab funkcijo `fspecial` za izgradnjo Gaussovega filtra ter ukaz `imfilter` za izvedbo filtriranja. Problem glajenja robnih elementov slike rešite s pomočjo podvajanja (angl. *replication*) robnih slikovnih elementov.

Način uporabe potrebnih funkcij lahko preverite z naslednjimi ukazi

```
help fspecial
help imfilter
```

Opomba: V poročilu podajte in komentirajte spisano programsko kodo. Prikažite vhodno in izhodno sliko ter njuna frekvenčna spektra. Podajte vaša opažanja in morebitne komentarje v zvezi z razlikami v frekvenčnih spektrih obeh slik.

Naloga 5 (1 točka)

Spišite Matlab funkcijo `gradient_approximation`, ki kot vhodni podatek vzame glajeno različico sive slike obraza, s pomočjo Sobelovega operatorja izvede aproksimacijo odvoda slike v horizontalni in vertikalni smeri ter na izhodu vrne amplitudo odvoda. Pri udejanjanju vaše funkcije uporabite Matlab funkcijo `fspecial` za izgradnjo vertikalne Sobelove maske ter funkcijo `imfilter` za izvedbo filtriranja. Problem filtriranja robnih elementov slike rešite s pomočjo podvajanja (angl. *replication*) robnih slikovnih elementov.

Upoštevajte, da lahko Sobelov filter (oz. masko) za aproksimacijo odvoda v horizontalni smeri, pridelate s transponiranjem vertikalnega filtra. Pri izvedbi si pomagajte z naslednjimi kodnimi izseki:

```
%% Izgradnja filtra in filtriranje
hv = fspecial( . . . );                    % generiraj vertikalno masko
hh = hv';                                  % generiraj horizontalno masko
Gx = (imfilter(double(SmoothGrey),hv,'replicate')); % odvod v x smeri
Gy = . . .                                 % odvod v y smeri
Edges = sqrt(Gx.^2+Gy.^2);                 % rezultat
```

Opomba: V poročilu podajte komentirano spisano programsko kodo funkcije in vhodno ter izhodno sliko funkcije. Prikažite frekvenčna spektra obeh slik in podajte vaša opažanja. Enako storite še za histograme slik.

Naloga 6 (1 točka)

Spišite Matlab funkcijo `threshold_image`, ki kot vhodni podatek vzame sliko robov dobljenih s Sobelovim operatorjem, jo upragovi in na izhodu vrne upragovljeno binarno sliko. Funkcija torej vsem slikovnim elementom, ki so nad vrednostjo praga, priredi vrednost ena in vsem slikovnim elementom, katerih sivi nivo je pod vrednostjo praga priredi vrednost nič.

Opomba: V poročilu podajte komentirano programsko kodo funkcije in vhodno ter izhodno sliko funkcije. Prikažite histograma obeh slik in podajte vaša opažanja.

Naloga 7 (1 točka)

Spišite Matlab funkciji `mask_edges` in `im_resize`, kjer prva funkcija izvede množenje elementov očiščene binarne maske področja kože in upragovljenje slike robov in na izhodu vrne produkt obeh slik. Druga funkcija kot vhodni podatek vzame produkt slik in ga zmanjša na 0.3 njegove velikosti. Prvo funkcijo izvedite s pomočjo matričnega operatorja `'.*'`, ki pomnoži istoležne elemente slike in drugo s pomočjo Matlab funkcije `imresize` s privzeto metodo interpolacije.

Na koncu zaženite vse programske komponente skupaj s eliptično Hough-ovo transformacijo in si oglejte rezultate.

Opomba: V poročilu podajte komentirano programsko kodo funkcij in vhodne ter izhodne slike. Prikažite tudi končni rezultat celotne verige procesiranja, ki ga opravi skripta `vaja4_predloga` s pomočjo vaših funkcij.

PRILOGA

Na spodnji sliki je prikazan shematski potek postopka detekcije obraza, ki ga je v okviru vaje potrebno udejanjiti.