Vaja 1: Osnove procesiranja slik in histogrami

Laboratorijske Vaje

19. oktober 2011

Najprej si ustvarite delovno mapo, v katero boste pisali rešitve vseh nalog za predmet. V tej mapi si boste za vsako vajo posebej ustvarili podmapo. Za prvo vajo ustvarite: `vaja1/`. Sem si prenesite in razpakirajte `vaja1.zip` s spletne strani predmeta. Rešitve nalog boste pisali v Matlab/Octave skripte v mapi '`vaja1/`' in jih zagovarjali ob zagovoru nalog. Pri nekaterih nalogah so vprašanja, ki zahtevajo razmislek. Odgovore na ta vprašanja si zabeležite v pisni obliki in jih prinesite na zagovor.

1 Naloga: Osnove procesiranja slik

Namen prve naloge je spoznavanje z osnovno funkcionalnostjo Matlab/Octave in z zapisom slik v matrikah. Ukazi, ki jih boste preizkusili pri tej vaji so: `imread`, `imshow`[1], `imagesc`, `colormap`, `imrotate`. Preizkušanje spodnjih funkcionalnosti si pišete v skripto (npr., '`vaja1_naloga1.m`'), ki jo boste pokazali asistentu na zagovoru. Sledite naslednjim točkam.

(a) Poglejte v *pomoč*, kaj pomenijo zgornji ukazi in kakšna je njihova sintaksa. Primer uporabe pomoči: `help imread`. Četudi uporabljate Octave, ki je precej dobra kopija plačljive različice Matlab, lahko za pomoč pogledate tudi `http://www.mathworks.com/help/techdoc/`. Če uporabljate Matlab, pa imate pomoč tudi integrirano v samem grafičem okolju in jo lahko prikličete preko gumba z vprašajem v orodni vrstici.

(b) Preberite sliko '`data/graf.png`' in jo prikažite v oknu s pomočjo ukazov:

```
A = imread('data/graf.png'); % pozor, slika A bo v 8 bitnem formatu uint8
figure(1); clf ; imagesc( A ); % odpri prostor za sliko, pobriši, prikaži
figure(2); clf ; imshow( A ) ;
```

(c) Na prvi pogled ni bistvene razlike med načinom prikaza slik, če uporabite `imagesc` ali `imshow` – razliko si bomo pogledali v kratkem. Slika, ki ste jo naložili ima tri kanale, RGB, in je spravljena v 3D matriko velikosti $visina \times sirina \times kanali = h \times w \times 3$. Izpišite si na zaslon velikost slike tako, da napišete `[h, w, d]=size(A)`. Spremenite barvno sliko v sivinsko tako, da povprečite kanale in sliko zopet prikažite z dvemi funkcijami za prikaz:

```
A = imread('data/graf.png');
A = double(A) ; % operacija deljenja ni definirana za uint8
% izpisite velikost slike
[w,h,d] = size(A)
% naredite novo prazno matriko
I = zeros(w,h) ;
I = (A(:,:,1) + A(:,:,2) + A(:,:,3))/3.0 ;
figure(1) ; clf ; % odpri prostor za sliko in ga pobriši
```

[1] Možno je, da ta funkcija ni implementirana v okolju Octave.

```
imshow( uint8(I) ) ; % sliko spremeni v 8 bitno
% slika je prikazana v odtenkih privzete barvne mape
```

(d) V zgornjem primeru se je slika izrisala s privzeto barvno mapo. Barvna mapa pomeni način, kako vam Matlab prikaže odtenek sivine. Na primer, temne odtenke lahko prikaže modro, svetle pa rdeče. Poskusite spremeniti barvno mapo v sliki z ukazom `colormap` (za primer uporabe glej Matlabovo pomoč). Poskusite z mapami `jet, bone, gray`.

(e) Izrišite si sivinski profil dvestote vrstice in dvestotega stolpca v sliki. Bodite pozorni, da je prva koordinata v matrikah y (vrstica), druga pa x (stolpec). Bodite tudi pozorni na uporabo znaka :-- ta znak pomeni *'vsi elementi vzdolž te koordinate'*.

```
vrstica200 = A(200,:);
stolpec200 = A(:,200);
figure(1) ; clf ; % zbriše vsebino prvega okna za sliko
% naredi okno za slike razdeljeno v 1 vrstico in 2 stolpca:
subplot(1,2,1) ;
% fokus vrne na prvi razdelek
plot(stolpec200) ; title('Stolpec 200') ;
xlabel('številka vrstice') ; ylabel('odtenek') ;
subplot(1,2,2) ; % fokus na drugi razdelek
plot(vrstica200) ; title('Vrstica 200') ;
xlabel('številka stolpca') ;  ylabel('odtenek') ;
```

(f) Izrežite pravokotno regijo s slike in jo izrišite kot novo sliko. Regijo v prvi sliki bomo označili tako, da tretji kanal (modri) postavimo na nič.

```
A1 = A ;
A1(130:260,240:450,1) = 0 ;
figure(1) ; clf ; subplot(1,2,1) ;
imagesc(uint8(A1));
A2 = A(130:260,240:450) ;
subplot(1,2,2) ;
imagesc(uint8(A2));
axis equal ; axis tight ; % naredi enaka razmerja za
% prikaz po širini in višini
```

(g) Izrišite upragovano binarno sliko, v kateri vrednost 1 označuje elemente, ki imajo v izhodiščni sliki sivinski nivo večji od 150 in spremenite barvno lestvico v črnobelo. Za izhodišče uporabite spodnjo kodo.

```
A = imread('data/graf.png') ;
A = rgb2gray(A) ;
A = double(A) ; % spremeni sliko v sivinsko
% priredimo elementom s sivino > 150 vrednost 1, ostalim 0
A = A > 150 ;
figure(1) ; clf ;
imagesc( uint8(A) ) ;
```

(h) Sedaj naredite primer redukcije števila sivinskih nivojev v sliki. Najprej naložite sliko `graf.png`. Spremenite jo v format `double`. Najvišji možni sivinski nivo je 255, najnižji pa 0. Izračunajte novo sliko tako, da sivinske nivoje originalne slike pomnožite s primernim faktorjem (in nato rezultat zaokrožite) tako, da bo najvišji možni sivinski nivo 16, najnižji pa 0. Sliko si enkrat prikažite z metodo `imagesc` in enkrat z metodo `imshow` – ali opazite razliko v prikazu? Razložite si to razliko glede na način po katerem obe metodi prikazujeta slike (opise najdete v Matlabovi pomoči). Rezultat te naloge posebej napišite v skripto `vaja1_1h.m` .

2 Naloga: Histogrami

V nadaljevanju si bomo ogledali, kako gradimo histograme. Preizkušanje spodnjih funkcionalnosti si pišete v skripte (npr., '`vaja1_naloga2.m`').

(a) Preberite sliko `data/graf.png` in jo spremenite v sivinsko sliko. Spremenite sliko, ki je dimenzij $(N \times M)$ v 1D vektor velikosti $N \times M \times 1$ in izračunajte histogram. Histogram si izrišite za različno število celic in razložite zakaj se oblika histogramov spreminja s številom celic. Za eksperimentiranje lahko izhajate iz spodnje kode (shranite si jo v funkcijo s poljubnim imenom in zaženite).

```
img = double(rgb2gray(imread('data/graf.png')));
[h,w] = size(img) ;
img1D = reshape(img,w*h,1);
figure(1) ; clf ;
bins = 10 ;
h = hist(img1D,bins);
subplot(1,3,1) ; bar(h, 'b') ;
bins = 20 ;
h = hist(img1D,bins);
subplot(1,3,2) ; bar(h, 'b')
bins = 40 ;
h = hist(img1D,bins);
subplot(1,3,3) ; bar(h, 'b')
```

(b) Naredite datoteko `myhist.m` in v njej implementirajte funkcijo `myhist`, kateri podate 2D sivinsko sliko ter število celic, funkcija pa vrne 1D histogram in spodnjo vrednost sivinskega nivoja za vsako celico. Spodnjo kodo si podrobno oglejte in si razložite, kaj počne katera vrstica. Napišite skripto `vaja1_naloga2b.m`, ki naloži sliko `data/graf.png`, jo spremeni v sivinsko, zanjo izračuna histogram s funkcijo `myhist.m` in histogram izriše.

```
function [ mojhist, celice ] = myhist( slika_gray , nbins)
 % preuredi sliko v 1D vektor
 input_vektor = reshape(slika_gray, 1, size(slika_gray, 1)*size(slika_gray,2));
 dolzina = length(input_vektor) ;
 mojhist = zeros(1,nbins) ; % inicializiramo histogram
 % določite mejne vrednosti vhodnega vektorja
 max_val_in = 255 ; % najvišji sivinski nivo
 min_val_in = 0 ; % najmanjši sivinski nivo
 % določite mejne vrednosti indeksov celic histograma
 max_val_out = nbins ; % index najvišje celice
 min_val_out = 1 ; % index najnižje celice
 % preglejte vse vrednosti v vhodnem vektorju in za vsako vrednost povečajte
```

```
% vsebino pripadajoče celice v histogramu
for i = 1 : dolzina
  idx = val2binidx( input_vektor(i), max_val_in, min_val_in,...
                    max_val_out, min_val_out ) ;
  mojhist(idx) = mojhist(idx) + 1 ;
end
% normaliziraj histogram, da je vsota celic (integral) enaka 1
mojhist = mojhist / sum(mojhist) ;

% izračunaj vhodno vrednost za celice histograma
idx = 1 : nbins ;
celice = binidx2val(idx, max_val_in, min_val_in, max_val_out, min_val_out) ;

% ------------------------------------------------- %
function idx = val2binidx(val, max_val_in, min_val_in, max_val_out, min_val_out)
% funkcija transformira vhodno vrednost v index
 skalirni_faktor = (max_val_out - min_val_out) / (max_val_in - min_val_in) ;
 idx =round(val*skalirni_faktor) + min_val_out ;

% ------------------------------------------------- %
function val = binidx2val(idx, max_val_in, min_val_in, max_val_out, min_val_out)
% funkcija transformira index v izhodno vrednost
 skalirni_faktor = (max_val_out - min_val_out) ./ (max_val_in - min_val_in) ;
 val = (idx - min_val_out)./skalirni_faktor ;
```

(c) Preberite sliko `data/graf.png`, ki je barvna RGB slika, in jo spremenite v dvokanalno normalizirano `rg` sliko po enačbi $r = \frac{R}{R+G+B}$, $g = \frac{G}{R+G+B}$. Naredite datoteko `myhistRG.m` v njej pa implementirajte funkcijo `myhistRG()`, kateri za vhod podate sliko z dvema kanaloma (RG sliko) in število celic, funkcija pa vrne 2D histogram. Bodite pozorni na to, kakšna je največja možna vrednost elementa v posameznem kanalu. Ker imate histogram spravljen v 2D matriki, si ga lahko izrišete z ukazom `bar3(moj_2d_histogram)`, kjer je `moj_2d_histogram` 2D matrika s histogramom. Za primer, kako generirati 2D histogram, se lahko zgledujete po spodnji kodi za tvorbo 3D histograma s trokanalne RGB slike. Kot rezultat naloge napišite skripto `vaja1_naloga2c.m`, ki naloži barvno sliko `data/graf.png`, izračuna RG histogram in ga izriše.

```
function h = myhist2RGB(img1,bins)
 % kvantiziraj vhodno sliko
 % (vemo da je najmanjša možna vrednost piksla 0, največja pa 255)
 img1 = double(img1) ; % slika mora biti double!
 img1=floor(img1*(bins)/255)+1;
 %define a 3D histogram with "bins^3" number of entries
 h=zeros(bins,bins,bins);
 %execute the loop for each pixel in the image,
 for i=1:size(img1,1)
    for j=1:size(img1,2)
        % inkrementiraj celico histograma h(R,G,B),
        % ki ustreza vrednosti piksla i,j
        R=img1(i,j,1);
        G=img1(i,j,2);
        B=img1(i,j,3);
        h(R,G,B)=h(R,G,B)+1;
    end
 end
```

```
% normaliziraj histigram, da je njegov integral (vsota celic) enaka 1
h=h/sum(sum(sum(h)));
```

3 Naloga: Razteg in ravnanje 1D histograma

V naslednjih nalogah boste implementirali operaciji *razteg* in *ravnanje* histograma, ki ste ju obravnavali na predavanjih. To vajo pišite v skripto '`vaja1_naloga3.m`'.

(a) Preberite z diska `data/phone2.jpg` (ki je že sivinska slika) in zanjo izrišite histogram z 255 celicami. S histograma lahko opazite, da najnižja sivinska vrednost v sliki ni nič, in najvišja ni 255. Za to sliko želimo izvesti operacijo *razteg histograma*, ki bo sliko popravila tako, da bo njen histogram *raztegnjen preko celotnega spektra sivinskih nivojev*. Po spodnji psevdo kodi implementirajte funkcijo `function nova_slika = razteghist( vhodna_slika )`, ki izvrši operacijo raztega histograma na vhodni sivinski sliki. Za okvirni potek algoritma se zgledujte po prosojnicah z vaj in predavanjih.

```
A = razteghist( vhodna_slika )
 % 1. določite najvišjo in najmanjšo vrednost vhodne slike
 % 2. največjo in najmanjšo vrednost izhodne slike poznate
 % 3. po formuli za razteg izračunajte za vsak slikovni
 %    element njegovo novo sivinsko vrednost
```

Namigi:

- Za vhodno sliko lahko določite najvišjo (`v_max`) in najnižjo (`v_min`) sivinsko vrednost z `max( vhodna_slika(:) )` in `min( vhodna_slika(:) )`.
- Z vsak slikovni element v sliki lahko izračunate novo vrednost po podobni formuli, kot smo jo uporabili v funkciji `val2binidx(val, max_val_in, min_val_in, max_val_out, min_val_out)` v točki (b) prejšnje naloge. Ker se ista operacija z istimi vrednostmi vrši na vseh elementih slike, lahko uporabite `vektorsko` operacijo množenja – izvedete operacijo na celi sliko naenkrat.

(b) Implementirajte še funkcijo `equihist(vhodna_slika)` za *ravnanje* histograma. Za okvirni potek algoritma se zgledujte po prosojnicah z vaj in predavanjih.

(c) Z diska preberite sliko '`data/phone2.jpg`' in izvedite enkrat ravnanje histograma in enkrat razteg histograma. Rezultat predstavite v eni sliki s funkcijo `subplot()`: V prvi vrsti izrišite originalno sliko, sliko po raztegu histograma in sliko po ravnanju histograma. V drugi vrsti izrišite pripadajoče histograme, v tretji vrstici pa izrišite normalizirane kumulativne histograme. Lahko se zgledujete po spodnji kodi za prikaz. Rezultat naloge napišite v skripto `vaja1_naloga3c.m`.

```
function vaja1_naloga3c()
 A = imread('data/phone2.jpg') ;
 A = double(A) ;
 figure(1); clf ;
 subplot(3,3,1) ;
 imshow(uint8(A)) ;
 colormap gray ;
 [ h0, c0 ] = myhist( A , 255) ;
 subplot(3,3,4) ;
 bar(c0,h0) ;
 cs0 = cumsum(h0) ;
 cs0 = cs0 / max(cs0) ;
```

```
subplot(3,3,7) ;
plot(cs0) ;
...
```