Univerza v Ljubljani, Fakulteta za elektrotehniko
Laboratorij za slikovne tehnologije

# Slikovna informatika

## LABORATORIJSKE VAJE

Tomaž Vrtovec

2012

Univerza v Ljubljani, Fakulteta za elektrotehniko
Laboratorij za slikovne tehnologije

# Slikovna informatika

## LABORATORIJSKE VAJE

Tomaž Vrtovec

2012

# Kazalo

Univerza v Ljubljani, Fakulteta za elektrotehniko
Modul A: Slikovna informatika (64135) - Laboratorijske vaje

# Vaja 0: Uvod v Matlab

## Navodila

Uvodna vaja služi spoznavanju osnovnih postopkov za nalaganje, prikazovanje ter shranjevanje slik v okolju Matlab.

1. Dana je barvna slika slika1.bmp. Naložite sliko kot matriko v okolju Matlab s pomočjo funkcije `imread()`, prikažite sliko na zaslon s pomočjo funkcije `imshow()` ter shranite sliko v JPEG formatu s pomočjo funkcije `imwrite()`.

2. Datoteka slika2-660x440-08bit.raw vsebuje sivinsko sliko v obliki surovih podatkov (RAW). Slika velikosti $X \times Y = 660 \times 440$ slikovnih elementov je zapisana z 8 biti na slikovni element (`'uint8'`). Napišite funkcijo za nalaganje poljubne RAW slike:

   ```
   function oImage = loadImage(iPath, iSize, iType),
   ```

   kjer je `iPath` pot do slike (direktorij in ime datoteke), `iSize` vektor velikosti slike (v slikovnih elementih), `iType` pa oblika zapisa (število bitov). Funkcija vrne sliko `oImage` v obliki matrike. Uporabite funkcijo `fread()` v povezavi s funkcijama `fopen()` in `fclose()`.

3. Napišite funkcijo za prikazovanje poljubne slike na zaslon:

   ```
   function displayImage(iImage, iTitle),
   ```

   kjer je `iImage` slika, ki jo želite prikazati, `iTitle` pa naslov prikaznega okna, v katerem je koordinatni sistem prikazane slike. Uporabite funkcijo `image()`, prikaz pa prilagodite zapisu s pomočjo funkcije `colormap()` ter nastavite razmerje koordinatih osi z ukazom `axis image`.

4. Datoteka slika3-540x320-16bit.raw ravno tako vsebuje sivinsko sliko v obliki surovih podatkov (RAW), vendar pa je slika velikosti $X \times Y = 540 \times 320$ slikovnih elementov zapisana s 16 biti na slikovni element (`'uint16'`). Sliko naložite kot matriko ter jo prikažite na zaslon.

5. Napišite funkcijo za shranjevanje poljubne sivinsko slike v RAW format:

   ```
   function saveImage(iImage, iPath, iType),
   ```

   kjer je `iImage` slika, ki jo želite shraniti, `iPath` pot do slike (direktorij in ime datoteke), `iType` pa oblika zapisa (število bitov). Uporabite funkcijo `fwrite()` v povezavi s funkcijama `fopen()` in `fclose()`. Shranite dano 16-bitno sliko v 8-bitnem zapisu.

6. Iz slike slika3-540x320-16bit.raw izluščite območje med vključno slikovnima elementoma 95 in 200 v $x$ smeri ter med vključno slikovnima elementoma 200 in 222 v $y$ smeri. Območje prikažite na zaslon s pomočjo funkcije `displayImage()` ter shranite v RAW format s pomočjo funkcije `saveImage()` in v JPEG format s pomočjo funkcije `imwrite()`.

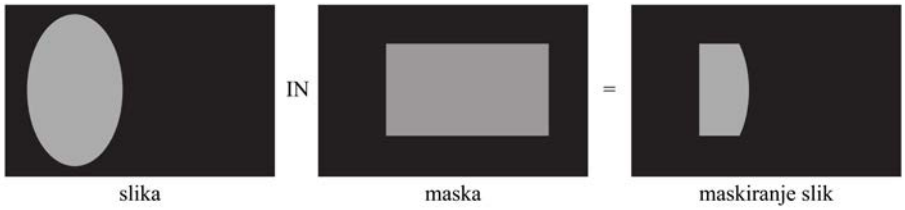
Univerza v Ljubljani, Fakulteta za elektrotehniko
Modul A: Slikovna informatika (64135) - Laboratorijske vaje

# Vaja 1: Sinteza in maskiranje slik

## Navodila

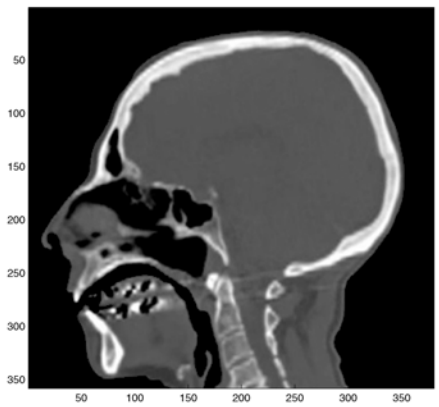
**Sinteza slik** je v splošnem pomenu ustvarjanje slik oziroma združevanje slik v nove slike. Sintezo dveh slik lahko izvedemo z logično ALI operacijo, ki jo v primeru slik definiramo kot seštevanje sivinskih vrednosti enakoležečih slikovnih elementov na obeh slikah (ne-logične več-bitne vrednosti), pri čemer predpostavimo, da sta obe sliki enakih velikosti.


**Maskiranje slik** je postopek pridobivanja področij na sliki, ki so podana s pripadajočo masko. Maska podaja lokacije slikovnih elementov, ki jih želimo na maskirani sliki ohraniti. Maskiranje slike z masko izvedemo z logično IN operacijo, ki jo v primeru slik definiramo kot množenje sivinskih vrednosti enakoležečih slikovnih elementov slike (ne-logične več-bitne vrednosti) in maske (logične bitne vrednosti), pri čemer predpostavimo, da sta slika in maska enakih velikosti. Ustvarjanje mask lahko pojmujemo tudi kot sintezo slike, saj v začetno prazno sliko vpišemo vrednosti maske na izbrane lokacije.

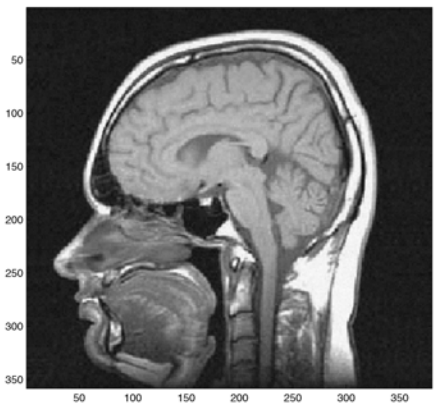



Dani sta sliki head-CT-380x358-08bit.raw in head-MR-380x358-08bit.raw velikosti $X \times Y = 380 \times 358$ slikovnih elementov, ki sta zapisani z 8 biti v obliki surovih podatkov (RAW). Prva slika predstavlja računalniško tomografski (CT), druga slika pa magnetno resonančni (MR) stranski prerez človeške glave.

CT slika




MR slika



1. Napišite funkcijo za sintezo maske v obliki pravokotnika:

   ```
   function oMask = maskRectangle(iMask, iX1, iY1, iX2, iY2, iValue),
   ```

   kjer je `iMask` začetna vhodna maska enake velikosti kot slika, `iX1` in `iY1` sta koordinati zgornjega levega oglišča pravokotnika $(x_1, y_1)$, `iX2` in `iY2` sta koordinati spodnjega desnega oglišča pravokotnika $(x_2, y_2)$, `iValue` pa je vrednost maske. Funkcija vrne masko `oMask` velikosti vhodne maske. Prikažite sliko maske na zaslon, pri čemer za parametre pravokotnika uporabite $(x_1, y_1) = (165, 150)$ in $(x_2, y_2) = (225, 355)$, vrednost maske pa nastavite prikazu primerno.

2. Maskirajte CT sliko z masko v obliki pravokotnika, pri čemer uporabite parametre pravokotnika iz prejšnje točke, vrednost maske pa nastavite maskiranju primerno. Prikažite maskirano sliko na zaslon.

3. Napišite funkcijo za sintezo maske v obliki elipse:

   ```
   function oMask = maskEllipse(iMask, iX, iY, iA, iB, iValue),
   ```

   kjer je `iMask` začetna vhodna maska enake velikosti kot slika, `iX` in `iY` sta koordinati središča elipse $(x, y)$, `iA` in `iB` sta polosi elipse $(a, b)$, `iValue` pa je vrednost maske. Funkcija vrne masko `oMask` velikosti vhodne maske. Pri zaokroževanju koordinat na celoštevilčne vrednosti si pomagajte s funkcijo `round()`. Prikažite sliko maske na zaslon, pri čemer za parametre elipse uporabite $(x, y) = (190, 130)$, $a = 120$ in $b = 100$, vrednost maske pa nastavite prikazu primerno.

4. Maskirajte MR sliko z masko v obliki elipse, pri čemer uporabite parametre elipse iz prejšnje točke, vrednost maske pa nastavite maskiranju primerno. Prikažite maskirano sliko na zaslon.

5. Združite masko v obliki pravokotnika in masko v obliki elipse v novo masko. Uporabite parametre pravokotnika iz točke 1 in parametre elipse iz točke 3, vrednost mask pa nastavite maskiranju primerno. Prikažite dobljeno sliko maske na zaslon. Maskirajte CT in MR sliko z dobljeno masko ter prikažite maskirano sliko.

6. Napišite funkcijo za sintezo maske v obliki šahovnice:

   ```
   function oMask = maskChessboard(iMask, iW, iH, iValue1, iValue2),
   ```

   kjer je `iMask` začetna vhodna maska enake velikosti kot slika, `iW` in `iH` sta širina $(w)$ in višina $(h)$ polja šahovnice, `iValue1` je vrednost maske na diagonalnih poljih šahovnice, `iValue2` pa vrednost maske na antidiagonalnih poljih šahovnice. Funkcija vrne masko `oMask`, ki je enake velikosti kot vhodna maska. Upoštevajte, da se število polj v šahovnici ne vedno izide z velikostjo slike. Prikažite sliko maske na zaslon, pri čemer za parametre šahovnice uporabite $w = 60$ in $h = 40$, vrednost maske pa nastavite prikazu primerno.

7. Maskirajte CT sliko z masko v obliki šahovnice, pri čemer uporabite samo diagonalna polja. Maskirajte MR sliko z masko v obliki šahovnice, pri čemer uporabite samo antidiagonalna polja. Uporabite parametre šahovnice iz prejšnje točke, vrednost maske pa nastavite maskiranju primerno. Prikažite maskirani sliki na zaslon.

8. Združite CT sliko, maskirano z masko v obliki šahovnice diagonalnih polj, ter MR sliko, maskirano z masko v obliki šahovnice antidiagonalnih polj. Prikažite dobljeno sliko na zaslon.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Priložite sliko maske v obliki pravokotnika ter sliko rezultata maskiranja CT slike z masko v obliki pravokotnika. Za parametre pravokotnika uporabite $(x_1, y_1) = (165, 150)$ in $(x_2, y_2) = (225, 355)$.

2. Priložite sliko maske v obliki elipse ter sliko rezultata maskiranja MR slike z masko v obliki elipse. Za parametre elipse uporabite $(x, y) = (190, 130)$, $a = 120$ in $b = 100$.

3. Priložite sliko maske, ki ste jo pridobili z združevanjem maske v obliki pravokotnika in maske v obliki elipse. Za parametre pravokotnika in elipse uporabite enake parametre kot pri predhodnih vprašanjih. Priložite sliko rezultata maskiranja CT slike in sliko rezultata maskiranja MR slike s tako dobljeno združeno masko.

4. Priložite sliko maske v obliki šahovnice diagonalnih in antidiagonalnih polj. Za parametre šahovnice uporabite $w = 60$ in $h = 40$, vrednost maske pa nastavite prikazu primerno.

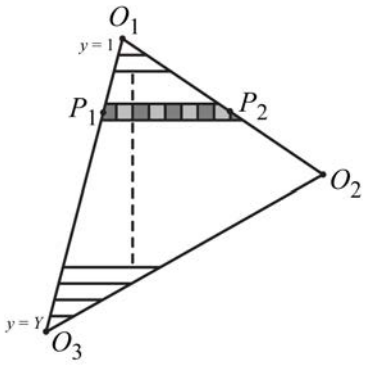
5. Priložite sliko, ki ste jo pridobili z združitvijo CT slike, maskirane z masko v obliki šahovnice diagonalnih polj, ter MR slike, maskirane z masko v obliki šahovnice antidiagonalnih polj. Za parametre šahovnice obakrat uporabite $w = 60$ in $h = 40$.

6. Sinteza maskirane CT in maskirane MR slike v obliki komplementarne šahovnice omogoča prikazovanje razlik med slikama. Na kakšen način bi še lahko prikazali razliko med slikama in na kaj je potrebno paziti pri takem prikazovanju? Obrazložite odgovor.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Splošna oblika maske z ravnimi stranicami je poligon s poljubnim številom oglišč, ki ga lahko predstavimo z množico dotikajočih se trikotnikov. Trikotnik zato predstavlja elementarni gradnik poljubne maske z ravnimi stranicami. Zaradi diskretne oblike predstavitve slik (slikovni elementi) se je za ustvarjanje trikotnikov najbolj uveljavil postopek na osnovi **pregledovanja vrstic** (ang. *scan-line*). Postopek temelji na iskanju presečišč vsake vrstice $y$ trikotnika s stranicami trikotnika, slikovni elementi med obema presečiščema v $x$ smeri pa se nato nastavijo na vrednost maske.




Napišite funkcijo za sintezo maske v obliki trikotnika:

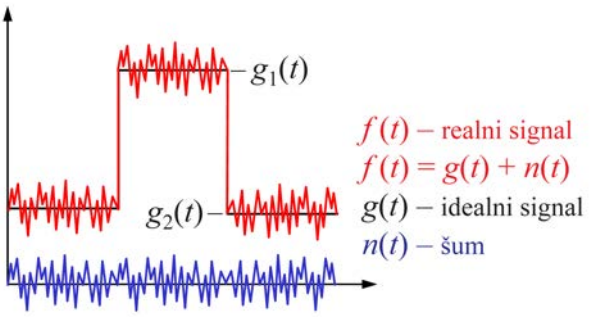
```
function oMask = maskTriangle(iMask, iX, iY, iValue),
```

kjer je `iMask` začetna vhodna maska enake velikosti kot slika, `iX` $= [x_1, x_2, x_3]$ in `iY` $= [y_1, y_2, y_3]$ so koordinate oglišč trikotnika, `iValue` pa je vrednost maske. Funkcija vrne masko `oMask` velikosti vhodne maske. Pri zaokroževanju koordinat na celoštevilčne vrednosti si pomagajte s funkcijo `round()`. Prikažite sliko maske na zaslon, pri čemer za parametre trikotnika uporabite $(x_1, y_1) = (230, 20)$, $(x_2, y_2) = (50, 250)$ in $(x_3, y_3) = (300, 340)$, vrednost maske pa nastavite prikazu primerno.

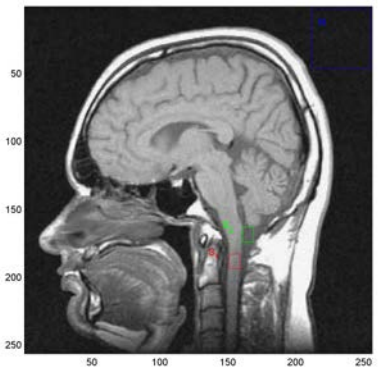
# Vaja 2: Razmerje signal/šum

## Navodila

Razmerje signal/šum (ang. *signal-to-noise ratio*, SNR) je pomembno za vrednotenje relativne jakosti oz. moči signala glede na njegov šum, saj predstavlja merilo za zanesljivost oz. sposobnost zaznavanja prisotnosti ali sprememb veličin, ki jih s signalom opazujemo. Uporabljajo se trije osnovni načini podajanja razmerja signal/šum, in sicer amplitudni ($\mathrm{SNR}_A$), diferencialni ($\mathrm{SNR}_D$) ter močnostni ($\mathrm{SNR}_M$) način. Kljub temu pa podajanje razmerja signal/šum ni standardizirano, ampak je odvisno od vrste signala in šuma ter od namena uporabe.




Dana je slika head-256x256-08bit.raw velikosti $X \times Y = 256 \times 256$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Slika predstavlja stranski prerez človeške glave in je bila pridobljena s tehniko magnetno resonančnega (MR) slikanja.




1. Histogram slike je grafično orodje za prikazovanje frekvenčnih porazdelitev sivinskih vrednosti slikovnih elementov na sliki. Vrednosti na abscisni osi histograma predstavljajo sivinske vrednosti slike (celotno dinamično območje), medtem ko vrednosti na ordinatni osi histograma predstavlja število slikovnih elementov na sliki z izbrano sivinsko vrednostjo. Napišite funkcijo za izračun histograma slike:

   ```
   function [oHist, oLevels] = computeHistogram(iImage),
   ```

   kjer je `iImage` slika, za katero se histogram računa. Funkcija vrne histogram slike `oHist` v obliki vektorja ter pripadajoče dinamično območje sivinskih vrednosti `oLevels` v obliki vektorja enake velikosti. Izbrani sivinski vrednosti na $i$-ti lokaciji v vektorju `oLevels` torej pripada vrednost histograma na $i$-ti lokaciji v vektorju `oHist`.

2. Napišite funkcijo za prikaz histograma slike:

   ```
   function displayHistogram(iHist, iLevels, iTitle),
   ```

   kjer je `iHist` histogram slike, `iLevels` dinamično območje sivinskih vrednosti slike, `iTitle` pa naslov prikaznega okna, v katerem je histogram prikazan. Za prikazovanje histograma uporabite funkcijo `bar()`.

3. V primeru MR slik je odziv (signal) predstavljen s sivinskimi vrednostmi, ki so različne od nič ($s \neq 0$), medtem ko je ozadje slike področje brez odziva in ima v idealnem primeru sivinsko vrednost nič ($s = 0$). Na sliki lahko tako določimo naslednja območja zanimanja:

   - Območje prvega signala ($S_1$) naj predstavlja hrbtenjača, npr. na območju pravokotnika $(x_1, y_1) = (151, 183)$ in $(x_2, y_2) = (158, 193)$.

   - Območje drugega signala ($S_2$) naj predstavlja cerebrospinalna tekočina, npr. na območju pravokotnika $(x_1, y_1) = (161, 163)$ in $(x_2, y_2) = (168, 173)$.

   - Območje šuma na sliki ($N$) naj predstavlja ozadje slike, npr. na območju pravokotnika $(x_1, y_1) = (212, 2)$ in $(x_2, y_2) = (254, 45)$.

   Določite dana območja signalov in šuma na sliki ter prikažite slike in pripadajoče histograme posameznih območij.

4. Izračunajte amplitudno razmerje signal/šum $\mathrm{SNR}_A$ za dana območja signalov in šuma na sliki:

$$\mathrm{SNR}_A = \left.\frac{\mu_1}{\sigma_n}\right|_{\mu_2=0},$$

   kjer je $\mu_1$ povprečna amplituda opazovanega signala ter $\sigma_n$ standardna deviacija šuma. V tem primeru upoštevajte, da je povprečna amplituda signalov, ki jih ne opazujemo, enaka nič ($\mu_2 = 0$).

5. Izračunajte diferencialno razmerje signal/šum $\mathrm{SNR}_D$ za dana območja signalov in šuma na sliki, in sicer na naslednja dva načina:

$$\mathrm{SNR}_D = \frac{\mu_1 - \mu_2}{\sigma_n} \qquad \text{in} \qquad \mathrm{SNR}_D = \frac{|\mu_1 - \mu_2|}{\sqrt{\sigma_1^2 + \sigma_2^2}},$$

   kjer sta $\mu_1$ in $\mu_2$ povprečni amplitudi opazovanih signalov, $\sigma_1$ in $\sigma_2$ standardni deviaciji opazovanih signalov, $\sigma_n$ pa je standardna deviacija šuma.

6. Izračunajte močnostno razmerje signal/šum $\mathrm{SNR}_M$ za dana območja signalov na sliki:

$$\mathrm{SNR}_M = \frac{|\mu_1 - \mu_2|^2}{\sigma_1^2 + \sigma_2^2}$$

   kjer sta $\mu_1$ in $\mu_2$ povprečni amplitudi, $\sigma_1$ in $\sigma_2$ pa standardni deviaciji opazovanih signalov.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Priložite sliko histograma dane slike.

2. Priložite slike območij signalov $S_1$, $S_2$ in $N$ ter pripadajoče slike histogramov.

3. Izračunajte amplitudno ($\mathrm{SNR}_A$), diferencialno ($\mathrm{SNR}_D$) ter močnostno ($\mathrm{SNR}_M$) razmerje signal/šum za dana območja signalov in šuma. Vrednosti podajte tudi v decibelih (dB). Kaj pomenijo te vrednosti, če upoštevamo dana območja računanja?

4. Kako veliko območje je smiselno določiti za področje šuma $N$? Obrazložite odgovor.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Napišite funkcijo, ki sliki doda aditiven Gaussov šum:

```
function [oImage, oNoise] = addNoise(iImage, iStd),
```

kjer je `iImage` slika, kateri se dodaja Gaussov šum, `iStd` pa standardna deviacija dodanega šuma (povprečna amplituda dodanega šuma je $\mu = 0$), ki ga modelirate s pomočjo funkcije `randn()`. Funkcija vrne sliko z dodanim šumom `oImage` ter matriko dodanega šuma `oNoise` (matrika ima enake dimenzije kot slika).

- Opazujte slike, pripadajoče histograme ter razmerja signal/šum pri slikah z dodanim šumom različnih standardnih deviacij.
- Na kaj morate biti pozorni pri prikazovanju slike šuma in pri računanju pripadajočega histograma?
- Kako dodajanje šuma na sliko vpliva na obliko pripadajočega histograma ter na diferencialno razmerje signal/šum $\mathrm{SNR}_D$?

Univerza v Ljubljani, Fakulteta za elektrotehniko
Modul A: Slikovna informatika (64135) - Laboratorijske vaje

# Vaja 3: Interpolacija slik

## Navodila

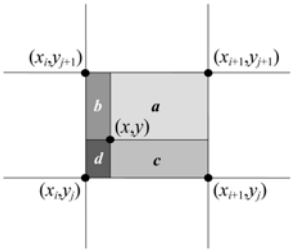
Interpolacija slik je postopek, s katerim sliki povečamo število slikovnih elementov oziroma s katerim lahko določimo sivinsko vrednost v poljubni točki med vzorčnimi točkami (prevzorčenje).

**Interpolacija ničtega reda** (interpolacija po principu najbližji sosed) priredi izbrani točki $(x, y)$ enako sivinsko vrednost kot je sivinska vrednost njej najbližje točke na diskretni mreži vzorčnih točk. **Interpolacija prvega reda** (bilinearna interpolacija v 2D, trilinearna interpolacija v 3D) priredi izbrani točki $(x, y)$ sivinsko vrednost v obliki utežene vsote sivinskih vrednosti štirih njej sosednjih točk na diskretni mreži vzorčnih točk:

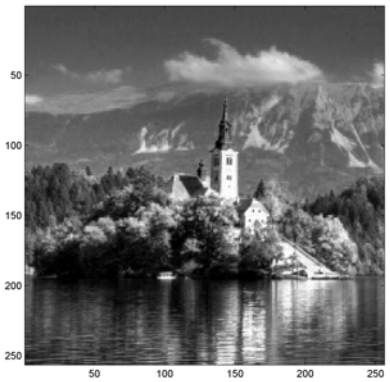
$$f(x,y) = a{\cdot}f(x_i, y_j) + b{\cdot}f(x_{i+1}, y_j) + c{\cdot}f(x_i, y_{j+1}) + d{\cdot}f(x_{i+1}, y_{j+1}),$$

kjer je $f(x, y)$ sivinska vrednost v točki $(x, y)$. Utež posamezne vzorčne točke je sorazmerna deležu ploščine pravokotnika, ki ga določata točka $(x, y)$, v kateri računamo sivinsko vrednost, in točka, ki je njej diagonalno nasprotna. Utež $a$ je npr. enaka:

$$a = \frac{(x_{i+1} - x)(y_{j+1} - y)}{(x_{i+1} - x_i)(y_{j+1} - y_j)}.$$




Dana je slika bled-256x256-08bit.raw velikosti $X \times Y = 256 \times 256$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW).




1. Napišite funkcijo za interpolacijo (prevzorčenje) ničtega reda slike:

   ```
   function oImage = interpolate0Image(iImage, iSize, oSize),
   ```

   kjer je `iImage` slika območja, ki ga interpolirate, `iSize` $= [w, h]$ je velikost tega območja (širina $w$ in višina $h$ v slikovnih elementih), `oSize` $= [w, h]$ pa je velikost interpoliranega območja (širina $w$ in višina $h$ v slikovnih elementih). Funkcija vrne interpolirano sliko oz. območje `oImage` velikosti `oSize`. Za iskanje najbližje točke v diskretni mreži točk uporabite funkcijo `round()`.

2. Napišite funkcijo za interpolacijo (prevzorčenje) prvega reda slike:

```
function oImage = interpolate1Image(iImage, iSize, oSize),
```

kjer je pomen vhodnih in izhodnih parametrov funkcije enak kot pri funkciji za interpolacijo ničtega reda. Za iskanje najbližje spodnje točke v diskretni mreži točk uporabite funkcijo `floor()`.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Za interpolacijo dane slike uporabite naslednje vrednosti: `iCoor` $= [130, 60]$, `iSize` $= [25, 80]$ in `oSize` $= [300, 500]$. Priložite sliko interpolacijskega območja na originalni sliki ter interpolirano sliko, in sicer za interpolacijo ničtega reda in interpolacijo prvega reda. Slike prikažite tako, da bo slika območja interpolacije (velikosti `iSize`) enakih fizičnih dimenzij oz. v dimenzijskem sorazmerju z interpolirano sliko (velikosti `oSize`). Prikaz v dimenzijskem sorazmerju dosežete z uporabo funkcije funkcije `image(gX,gY,I)`, kjer je `I` slika, ki jo prikazukete, `gX` in `gY` pa vektorja položajev slikovnih elementov v $x$ in $y$ smeri.

2. Kaj so prednosti in kaj slabosti interpolacije ničtega reda? Kaj so prednosti in slabosti interpolacije prvega reda? Kaj dosežemo z interpolacijami višjih redov, npr. z interpolacijo drugega reda (bikubična interpolacija)?

3. Naštejte nekaj primerov, kjer potrebujemo učinkovito interpolacijo slik.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Decimacija slik je postopek, s katerim sliki zmanjšamo število slikovnih elementov, tako da jo vzorčimo z nižjo frekvenco. Ob upoštevanju Nyquistovega teorema je potrebno pred vzorčenjem vhodni sliki zmanjšati frekvenčno širino, kar izvedemo z nizkopasovnim digitalnim filtrom. Filtrirano in z 2 decimirano sliko na stopnji $2n$ dobimo kot konvolucijo slike s predhodne stopnje $n$ z jedrom $c(i, j)$ digitalnega filtra velikosti $M \times M$:

$$ {}^{2n}f(x, y) = \sum_{i=-M}^{i=M} \sum_{j=-M}^{j=M} c(i, j) \cdot {}^{n}f(2x - i, 2y - j) \, . $$

Velikost decimirane slike se torej zmanjša glede na stopnjo decimacije. Večstopenjsko decimacijo lahko predstavimo kot zaporedje decimacij z 2 v obliki piramide. Jedro $c(i,j)$ digitalnega filtra, ki ustreza podanim zahtevam, ima v naslednjo obliko:

| | | |
|---|---|---|
| $\frac{1}{16}$ | $\frac{1}{8}$ | $\frac{1}{16}$ |
| $\frac{1}{8}$ | $\frac{1}{4}$ | $\frac{1}{8}$ |
| $\frac{1}{16}$ | $\frac{1}{8}$ | $\frac{1}{16}$ |

$M = 1$

| | | | | |
|---|---|---|---|---|
| $\frac{1}{400}$ | $\frac{1}{80}$ | $\frac{1}{50}$ | $\frac{1}{80}$ | $\frac{1}{400}$ |
| $\frac{1}{80}$ | $\frac{1}{16}$ | $\frac{1}{10}$ | $\frac{1}{16}$ | $\frac{1}{80}$ |
| $\frac{1}{50}$ | $\frac{1}{10}$ | $\frac{4}{25}$ | $\frac{1}{10}$ | $\frac{1}{50}$ |
| $\frac{1}{80}$ | $\frac{1}{16}$ | $\frac{1}{10}$ | $\frac{1}{16}$ | $\frac{1}{80}$ |
| $\frac{1}{400}$ | $\frac{1}{80}$ | $\frac{1}{50}$ | $\frac{1}{80}$ | $\frac{1}{400}$ |

$M = 2$

Napišite funkcijo za decimacijo (podvzorčenje) slike:

```
function oImage = decimateImage(iImage, iKernel, iLevel),
```

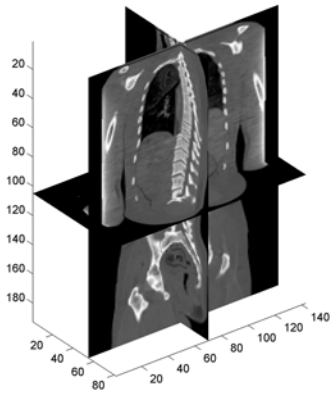
kjer je `iImage` slika, ki jo decimirate, `iKernel` je jedro $c(i,j)$ digitalnega filtra velikosti $M \times M$, `iLevel` pa je celoštevilčna stopnja decimacije. Funkcija vrne decimirano sliko `oImage`. Decimirajte dano sliko z jedrom $c(i,j)$ digitalnega filtra velikosti $M = 1$ in $M = 2$ ter poljubno celoštevilčno stopnjo decimacije (npr. `iLevel` $= 3$).

1. Kaj so prednosti in kaj slabosti decimacije?
2. S kakšnim faktorjem upada velikost slike pri decimaciji? S kakšnim faktorjem upada število podatkov, potrebnih za zapis slike?
3. Kakšne pogoje morajo izpolnjevati koeficienti jedra $c(i,j)$ digitalnega filtra, ki ga uporabljate pri decimaciji slike?
4. Če bi želeli pri decimaciji preprosto vzorčiti vsak drugi slikovni element brez zmanjševanja frekvenčne širine slike, kakšna bi bila v takem primeru oblika in kakšne bi bile vrednosti jedra $c(i,j)$ digitalnega filtra?

# Vaja 4: Prikazovanje slik

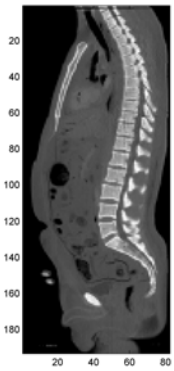
## Navodila

Objekti, ki jih opazujemo v dvodimenzionalnih (2D) slikah, so v splošnem tridimenzionalni (3D) in jih lahko zato boljše analiziramo na podlagi 3D slik. Sliko v 3D predstavimo kot funkcijo $f(x, y, z)$ prostorskih koordinat $x$, $y$ in $z$. Samo prikazovanje 3D slik v posameznih manj dimenzionalnih prostorih pa omogočajo pomožna orodja, kot so npr. prerezi in projekcije.

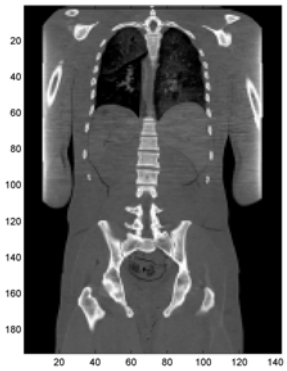


Dana je 3D slika visible-human-143x082x193-08bit.raw velikosti $X \times Y \times Z = 143 \times 82 \times 193$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Slika je zajeta s slikovno tehniko računalniške tomografije (CT) in predstavlja osrednji del človeškega telesa v 3D.

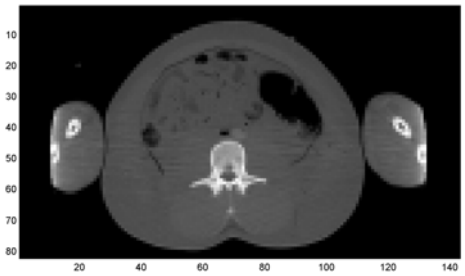
Stranski prerez



Čelni prerez



Prečni prerez



1. Napišite funkcijo za nalaganje 3D slike:

   ```
   function oImage = loadImage3D(iPath, iSize, iType),
   ```

   kjer je `iPath` pot do slike (direktorij in ime datoteke), `iSize` vektor velikosti slike (v slikovnih elementih), `iType` pa oblika zapisa (število bitov). Funkcija vrne naloženo 3D sliko `oImage` v obliki 3D matrike, ki jo sestavite z zaporednim nalaganjem 2D matrik.

2. Na osnovi 3D slike $f(x, y, z)$ lahko določimo osnovne 2D ravninske prereze. Tako je stranski (sagitalni, lateralni) prerez podan z $f_{x=x_c}(y, z) = f(x_c, y, z)$, čelni (koronalni, frontalni) prerez z $f_{y=y_c}(x, z) = f(x, y_c, z)$, prečni (aksialni, transverzalni) prerez pa z $f_{z=z_c}(x, y) = f(x, y, z_c)$. Napišite funkcijo za določanje poljubnega osnovnega ravninskega prereza:

   ```
   function oSection = getCrossSection(iImage, iPlane, iNum),
   ```

   kjer je `iImage` 3D slika, v kateri določate 2D prerez, `iPlane` je ravnina prereza (npr. 'stranska', 'celna' ali 'precna'), `iNum` pa zaporedna številka prereza ($x_c$, $y_c$ ali $z_c$). Funkcija vrne želeni osnovni ravninski prerez `oSection`.

3. Na osnovi 3D slike $f(x,y,z)$ lahko določimo pravokotne 2D projekcije. Tako je stranska projekcija podana z $f_{proj}(y,z) = \underset{k=1\ldots X}{proj}\,(f(k,y,z))$, čelna projekcija z $f_{proj}(x,z) = \underset{k=1\ldots Y}{proj}\,(f(x,k,z))$, prečna projekcija pa z $f_{proj}(x,y) = \underset{k=1\ldots Z}{proj}\,(f(x,y,k))$. Vrsto projekcije določa funkcija $proj(\cdot)$, in sicer lahko izračunamo:

   - projekcijo največjih vrednosti MaxIP oz. MIP (ang. *maximal intensity projection*),
   - projekcijo najmanjših vrednosti MinIP (ang. *minimal intensity projection*),
   - projekcijo povprečnih vrednosti AvgIP (ang. *average intensity projection*),
   - projekcijo vrednosti median MedIP (ang. *median intensity projection*),
   - projekcijo vrednosti standardnih odklonov StdIP (ang. *standard deviation intensity projection*),
   - projekcijo vrednosti varianc VarIP (ang. *variance intensity projection*),
   - ali poljubno projekcijo.

   Napišite funkcijo za določanje poljubne pravokotne projekcije:

   ```
   function oProjection = getOrthogonalProjection(iImage, iPlane, iFunc),
   ```

   kjer je `iImage` 3D slika, v kateri določate 2D projekcijo, `iPlane` je ravnina projekcije ('stranska', 'celna' ali 'precna'), `iFunc` pa vrsta projekcije ('max' za MIP, 'min' za MinIP, 'mean' za AvgIP, 'median' za MedIP, 'std' za StdIP, 'var' za VarIP). Funkcija vrne želeno pravokotno projekcijo `oProjection`.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Priložite slike naslednjih osnovnih ravninskih prerezov 3D slike:
   - stranski prerez za slikovni element $x_c = 76$,
   - čelni prerez za slikovni element $y_c = 31$,
   - prečni prerez za slikovni element $z_c = 102$.

2. Priložite slike stranske, čelne in prečne pravokotne projekcije 3D slike:
   - projekcije največjih vrednosti (MIP),
   - projekcije povprečnih vrednosti (AvgIP).

3. Katere izmed pravokotnih projekcij MIP, AvgIP, MinIP, MedIP, StdIP in VarIP je v primeru prikazovanja CT slik sploh smiselno računati? Obrazložite odgovor.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Upodabljanje površine (ang. *surface rendering*) združuje postopke prikazovanja površine danega 3D objekta oz. strukture. Eden izmed najpogosteje uporabljanih postopkov za upodabljanje površine je ti. izo-površina (ang. *iso-surface*). Sliko je potrebno najprej razgraditi, pri čemer lahko uporabimo namenske postopke za razgradnjo ali pa enostavno upragovljanje (ang. *thresholding*) sivinskih vrednosti. Nato uporabimo funkcijo izo-površine, ki medsebojno poveže vse točke v prostoru, ki imajo enako izo-vrednost (vrednost praga). Površina je končno predstavljena v obliki množice točk površine (ang. *vertices*) ter množice trikotnikov površine (ang. *faces*), ki povezujejo točke površine med sabo. S pomočjo funkcije za upodabljanje površine `surfaceRendering()`, ki temelji na računanju izo-površine, poskusite upodobiti površino v 3D sliki `iImage` pri različnih vrednostih praga sivinskih vrednosti oz. izo-vrednosti `iThreshold`:

```
>> % prikaz izo-povrsine
>> function pHandle = surfaceRendering(iImage, iThreshold)

>>     % izracun izo-povrsine
>>     oPatch = isosurface(iImage, iThreshold);

>>     % izris izo-povrsine
>>     pHandle = patch(oPatch, 'FaceColor', 'red', 'EdgeColor', 'none');

>>     % nastavitev koordinatnih osi
>>     daspect([1 1 1]); axis tight;
>>     set(gca, 'YDir', 'reverse', 'ZDir', 'reverse');

>>     % nastavitev osvetlitve
>>     isonormals(iImage, pHandle); camlight; lighting gouraud;

>>     % nastavitev pogleda
>>     view(217.5, 30);
```

# Vaja 5: Sivinske preslikave slik

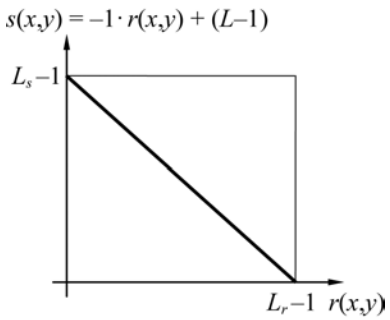
## Navodila

Sivinska preslikava $\mathcal{T}: \mathbb{R} \to \mathbb{R}$ v splošnem preslika vrednosti vseh slikovnih elementov (sivinske vrednosti) iz dinamičnega območja originalne oz. referenčne slike $[0 \dots L_r - 1]$ v dinamično območje preslikane slike $[0 \dots L_s - 1]$.

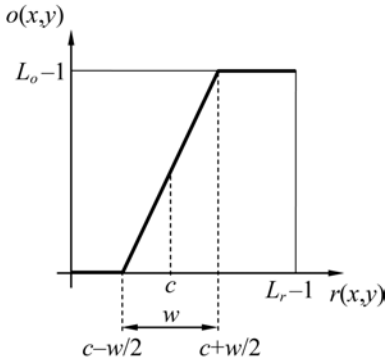
- **Linearno preslikavo** izvedemo tako, da sivinske vrednosti referenčne slike $r(x, y)$ pomnožimo s konstantno vrednostjo $a$ (sprememba kontrasta slike) in zmnožku prištejemo konstantno vrednost $b$ (sprememba svetlosti slike):

$$s(x, y) = a \cdot r(x, y) + b\,.$$

  Rezultat je slika $s(x, y)$, ki ima linearno preslikane sivinske vrednosti.



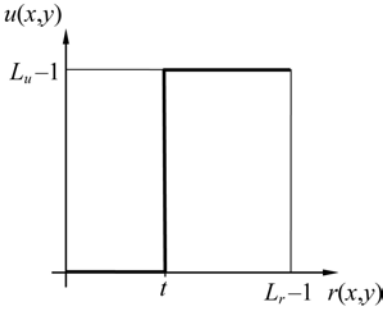

- **Linearno oknenje** izvedemo tako, da na dinamičnem območju referenčne slike definiramo okno s središčem $c$ in širino $w$:
  - vse sivinske vrednosti referenčne slike $r(x, y)$, ki so manjše od spodnje meje okna $c - \frac{w}{2}$, preslikamo v vrednost 0,
  - vse sivinske vrednosti referenčne slike $r(x, y)$, ki so večje od zgornje meje okna $c + \frac{w}{2}$, preslikamo v največjo možno sivinsko vrednost $L_o - 1$,
  - vse sivinske vrednosti referenčne slike $r(x, y)$, ki so znotraj okna, preslikamo z linearno sivinsko preslikavo.




$$o(x, y) = \begin{cases} 0 & \text{pri} \quad r(x, y) < c - \frac{w}{2}\,, \\ \frac{L_o - 1}{w}\left(r(x, y) - \left(c - \frac{w}{2}\right)\right) & \text{pri} \quad c - \frac{w}{2} \le r(x, y) \le c + \frac{w}{2}\,, \\ L_o - 1 & \text{pri} \quad r(x, y) > c + \frac{w}{2}\,. \end{cases}$$

Rezultat je slika $o(x, y)$, ki ima sivinske vrednosti znotraj okna linearno razširjene na celotno dinamično območje $[0 \dots L_o - 1]$.

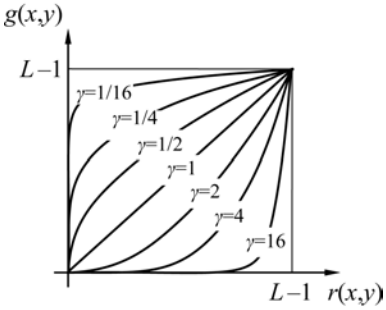
- **Upragovljanje** izvedemo tako, da sivinske vrednosti referenčne slike $r(x,y)$, ki so manjše ali enake izbrani vrednosti praga $t$, preslikamo v vrednost 0, sicer pa v največjo možno sivinsko vrednost $L_u - 1$:




$$u(x,y) = \begin{cases} 0 & \text{pri} \quad r(x,y) \leq t\,, \\ L_u - 1 & \text{pri} \quad r(x,y) > t\,. \end{cases}$$

  Rezultat je upragovljena slika $u(x,y)$, v kateri nastopata natančno dve sivinski vrednosti.

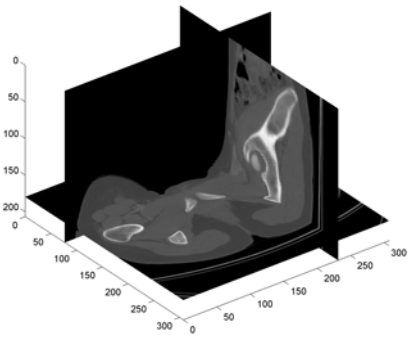
- **Gama preslikava** je nelinearna zvezna preslikava, ki je v primeru enakega dinamičnega območja referenčne in preslikane slike $L_r = L_g = L$ definirana kot:




$$g(x,y) = (L-1)^{1-\gamma} \cdot r^{\gamma}(x,y)\,.$$

  Rezultat je gama preslikana slika $g(x,y)$. Gama preslikava je uporabna takrat, kadar želimo zvezno in nelinearno povečati kontrast svetlejših področij na račun manjšega kontrasta temnejših področij slik ($\gamma > 1$), oziroma obratno ($\gamma < 1$).

Dana je tridimenzionalna (3D) slika človeške medenice pelvis-512x512x70-16bit.raw velikosti $X \times Y \times Z = 512 \times 512 \times 70$ slikovnih elementov, zajeta s slikovno tehniko računalniške tomografije (CT). Velikost slikovnega elementa je enaka $\Delta X \times \Delta Y \times \Delta Z = 0{,}601563 \times 0{,}601563 \times 3{,}0\,\text{mm}$, zapis pa je v obliki surovih podatkov (RAW) s 16 biti na slikovni element. Vrednost vsakega slikovnega elementa predstavlja celo predznačeno število (vrsta podatkov `'int16'`). Čeprav podatki zasedejo 16 bitov, je dinamično območje zapisanih vrednosti 12-bitno.



1. Uporabite funkcijo `loadImage3D()` za nalaganje 3D slike ter funkcijo `getCrossSection()` za pridobivanje osnovnih ravninskih prerezov (Vaja 4: Prikazovanje slik). Dobljene osnovne ravninske prereze prikažite s funkcijo `displayImage()`.

2. Napišite funkcijo za linearno sivinsko preslikavo:

```
function sImage = scaleImage(iImage, iSlope, iIntersection),
```

   kjer je `iImage` slika, ki ji preslikujete sivinske vrednosti, `iSlope` in `iIntersection` pa sta parametra linearne preslikave (parametra premice $a$ in $b$). Funkcija vrne linearno sivinsko preslikano sliko `sImage`. Preizkusite delovanje funkcije na sliki `oImage` za vrednosti `iSlope` $= 1$ ter `iIntersection` $= -1024$.

3. Napišite funkcijo za linearno sivinsko oknenje:

```
function wImage = windowImage(iImage, iCenter, iWidth),
```

kjer je `iImage` slika, ki ji preslikujete sivinske vrednosti, `iCenter` in `iWidth` pa sta parametra linearnega oknenja (središče okna $c$ in širina okna $w$). Funkcija vrne linearno sivinsko oknjeno sliko `wImage`. Preizkusite delovanje funkcije na sliki `sImage` za vrednosti `iCenter` $= 400$ ter `iWidth` $= 2000$, pri čemer upoštevate, da je dinamično območje 8-bitnega monitorja enako $L_o = 2^8$.

4. Napišite funkcijo za sivinsko upragovljanje:

   ```
   function tImage = thresholdImage(iImage, iThreshold),
   ```

   kjer je `iImage` slika, ki ji preslikujete sivinske vrednosti, `iThreshold` pa je parameter upragovljanja (vrednost praga $t$). Funkcija vrne sivinsko upragovljeno sliko `tImage`. Preizkusite delovanje funkcije na sliki `wImage` za vrednost `iThreshold` $= 127$.

5. Napišite funkcijo za gama sivinsko preslikavo:

   ```
   function gImage = gammaImage(iImage, iGamma),
   ```

   kjer je `iImage` slika, ki ji preslikujete sivinske vrednosti, `iGamma` pa je parameter gama preslikave (gama vrednost $\gamma$). Funkcija vrne gama sivinsko preslikano sliko `gImage`. Preizkusite delovanje funkcije na sliki `wImage` za vrednost `iGamma` $= 2$.

6. Prilagodite prikazovanje slik razmerju velikosti slike v milimetrih, kar omogoča funkcija `image(gX, gY, I)`. Vektorja `gX` in `gY` vsebujeta položaje (v milimetrih) vsakega slikovnega elementa vzdolž $x$ in $y$ koordinatne osi slike `I`. Potrebno je torej ustrezno spremeniti funkcijo `displayImage()` (Vaja 3: Interpolacija slik).

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Priložite stranske, čelne in prečne prereze 3D slik pri $x = 103$, $y = 233$ in $z = 52$, in sicer za slike `oImage`, `sImage` ($a = 1$, $b = -1024$), `wImage` ($c = 400$, $w = 2000$), `tImage` ($t = 127$) in `gImage` ($\gamma = 2$). Velikost slik naj bo prilagojena velikosti slikovnega elementa.

2. Zapišite dinamično območje $[0 \ldots L-1]$ za slike `oImage`, `sImage`, `wImage`, `tImage` in `gImage`. Kaj se zgodi pri prikazu slikovnih elementov, ki so izven dinamičnega območja prikaza 8-bitnega monitorja? Obrazložite odgovor.

3. Ali je preslikane slike smiselno uporabljati le za prikazovanje ali tudi za shranjevanje? Obrazložite odgovor.

4. Priložite histograme za 3D slike `oImage`, `sImage`, `wImage`, `tImage` in `gImage`, pri čemer uporabite funkciji `computeHistogram()` in `displayHistogram()` (Vaja 2: Razmerje signal/šum). Kaj se dogaja s histogrami pri sivinskih preslikavah slik? Obrazložite odgovor.

# Vaja 6: Geometrijske preslikave slik

## Navodila

Geometrijske preslikave $\mathcal{T}$: $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ omogočajo preslikavo vseh slikovnih elementov $(x, y)$ dvodimenzionalne (2D) slike na nove lokacije $(u, v)$, pri čemer se sivinske vrednosti slikovnih elementov ohranijo:

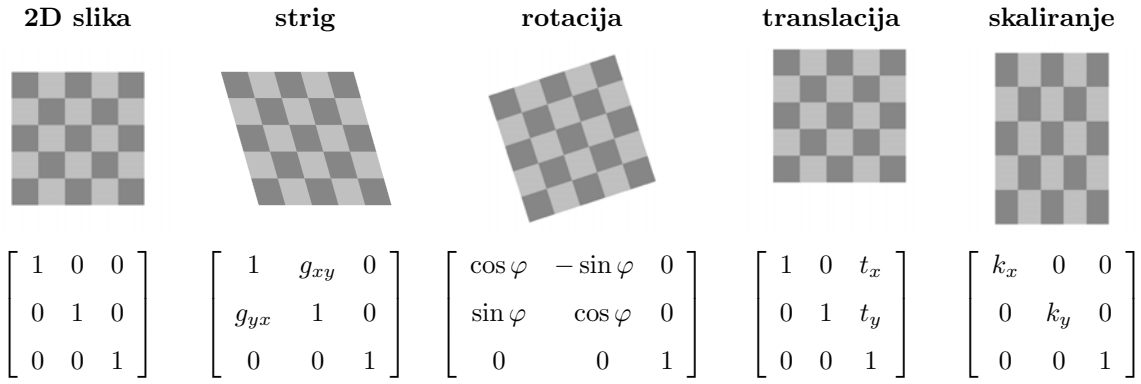
$$(u, v) = \mathcal{T}(x, y)\,.$$

Najbolj splošna linearna geometrijska preslikava je **afina preslikava** (ang. *affine transformation*), ki omogoča poljubno skaliranje, rotacijo, translacijo in strig. Afino preslikavo lahko zapišemo kot produkt vektorja koordinat in transformacijske matrike $\mathbf{T}$. V 2D je transformacijska matrika popolnoma določena s šestimi parametri:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}.$$

Parametra $t_x$ in $t_y$ predstavljata translaciji v $x$ in $y$ smeri, parametri $a_{ij}$ pa skaliranje, rotacijo in strig. Prednost matričnega zapisa afine preslikave je v tem, da jo lahko sestavimo kot zaporedje posameznih elementarnih preslikav, in sicer kot produkt skaliranja ($\mathbf{T}_{\text{skal}}$), translacije ($\mathbf{T}_{\text{trans}}$), rotacije ($\mathbf{T}_{\text{rot}}$) in striga ($\mathbf{T}_{\text{strig}}$):

$$\mathbf{T}_{\text{afina}} = \mathbf{T}_{\text{strig}}\mathbf{T}_{\text{rot}}\mathbf{T}_{\text{trans}}\mathbf{T}_{\text{skal}}\,.$$



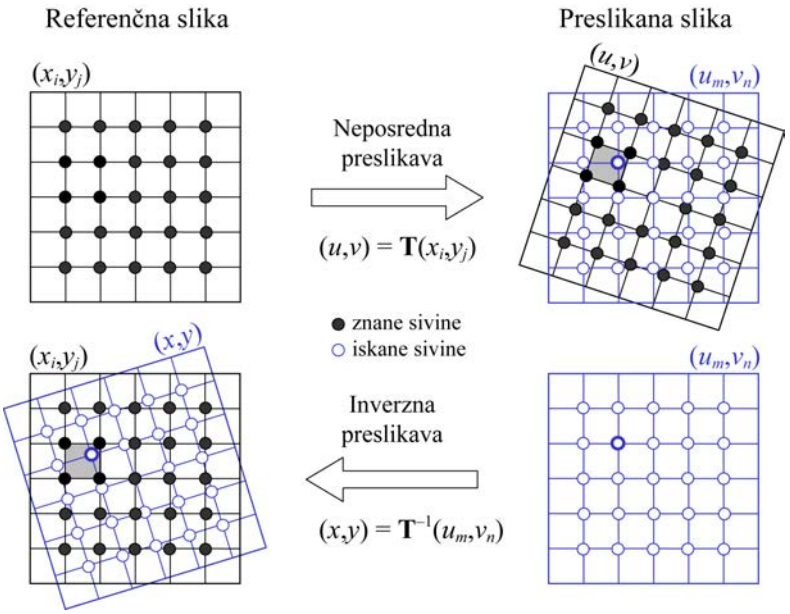

Pri preslikavi se slikovni elementi referenčne slike na lokacijah $(x_i, y_j)$ preslikajo na nove lokacije $(u, v)$:

$$(u, v) = \mathbf{T}(x_i, y_j)\,.$$

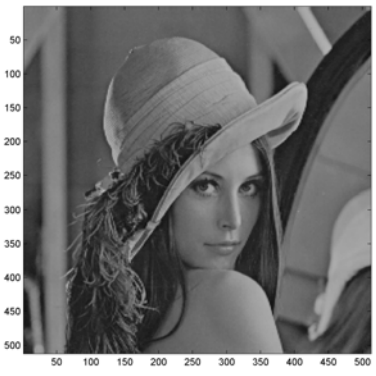
Nove lokacije pa v splošnem ne sovpadajo z diskretno mrežo vzorčnih točk preslikane slike $(u_m, v_n)$. V točkah $(u_m, v_n)$ je zato potrebno določiti sivinske vrednosti z interpolacijo sivinskih vrednosti iz sosednjih preslikanih točk. Takšen pristop z neposredno preslikavo nam postopek interpolacije precej zaplete, saj so sosednje preslikane točke lahko poljubno prostorsko razporejene. Njihova prostorska razporeditev je namreč odvisna od vrste in parametrov preslikave. Zato raje uporabljamo inverzno preslikavo:

$$(x, y) = \mathbf{T}^{-1}(u_m, v_n)\,,$$

s katero lahko za vsako diskretno vzorčno točko preslikane slike $(u_m, v_n)$ določimo njeno pripadajočo lokacijo $(x, y)$ na referenčni sliki, torej lokacijo, iz katere se je diskretna vzorčna točka $(u_m, v_n)$ preslikala in ki se v splošnem nahaja med štirimi vzorčnimi točkami na diskretni mreži referenčne slike $(x_i, y_j)$. Sivinsko vrednost vzorčne točke $(u_m, v_n)$ lahko v tem primeru izračunamo z interpolacijo sivinskih vrednosti v pripadajoči točki $(x, y)$ na referenčni sliki, saj so v tem primeru sosednje točke urejene na diskretni mreži in jih zato lahko hitro in enostavno določimo. Največkrat se zaradi relativne enostavnosti in hitrosti izvedbe za 2D slike uporablja bilinearna interpolacija.




Dana je slika lena-256x512-08bit.raw velikosti $X \times Y = 256 \times 512$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Velikost slikovnega elementa je enaka $\Delta X \times \Delta Y = 2{,}0 \times 1{,}0\,\text{mm}$.




1. Napišite funkcijo, ki za dane vhodne parametre izračuna matriko afine preslikave v 2D:

   ```
   function oMatrix = getAffineMatrix2D(iStrig, iRot, iTrans, iScal),
   ```

   kjer je `iStrig` $= [g_{xy}, g_{yx}]$ vektor striga, `iRot` $= \varphi$ kot rotacije, `iTrans` $= [t_x, t_y]$ vektor premika in `iScal` $= [k_x, k_y]$ vektor faktorjev povečave. Funkcija vrne 2D matriko afine preslikave `oMatrix`.

2. Napišite funkcijo za afino preslikavo točk v 2D:

```
function [oCoorX, oCoorY] = affineTransform2D(iGridX, iGridY, iMatrix),
```

kjer sta `iGridX` $= [\Delta x, 2\Delta x, \ldots, X\Delta x]$ in `iGridY` $= [\Delta y, 2\Delta y, \ldots, Y\Delta y]$ mreži koordinat slikovnih elementov v milimetrih, `iMatrix` $= \mathbf{T}^{-1}$ pa inverzna matrika afine preslikave. Funkcija vrne matriki `oCoorX` in `oCoorY`, ki vsebujeta $x$ oz. $y$ koordinate vsakega preslikanega slikovnega elementa v referenčni sliki v milimetrih. Na primer, če je `oCoorX`$(u, v) = x_c \in \mathbb{R}$ in `oCoorY`$(u, v) = y_c \in \mathbb{R}$, potem se vrednost na lokaciji $(x_c, y_c)$ v referenčni sliki preslika v diskretno lokacijo $(u, v)$.

3. Datoteka imageInterpol.m vsebuje funkcijo za bilinearno interpolacijo sivinskih vrednosti glede na dane koordinate preslikanih točk:

```
function oImage = imageInterpol(iImage, iPixelSize, iCoorX, iCoorY, iBgr),
```

kjer je `iImage` referenčna 2D slika, `iPixelSize` $= [\Delta X, \Delta Y]$ velikost slikovnega elementa v milimetrih, `iCoorX` in `iCoorY` matriki s koordinatami preslikanih točk, `iBgr` pa sivinska vrednost tistih slikovnih elementov, ki ne ležijo znotraj referenčne slike in torej predstavljajo ozadje (ponavadi nastavimo `iBgr` $= 0$, ampak zaradi nazornosti uporabite `iBgr` $= 255$). Funckija vrne interpolirano sliko `oImage`. Uporabite dano funkcijo za interpolacijo sivinskih vrednosti glede na dane koordinate preslikanih točk.

4. Priredite zgornje funkcije tako, da boste upoštevali tudi poljubno izhodišče koordinatnega sistema `iCenter` $= [c_x, c_y]$.

5. Prilagodite prikazovanje slik razmerju velikosti slike v milimetrih, kar omogoča funkcija `image(gX, gY, I)`. Vektorja `gX` in `gY` vsebujeta položaje (v milimetrih) vsakega slikovnega elementa vzdolž $x$ in $y$ koordinatne osi slike `I`. Potrebno je torej ustrezno spremeniti funkcijo `displayImage()` (Vaja 3: Interpolacija slik).

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Rotacijsko in translacijsko preslikavo lahko združimo v togo preslikavo (ang. *rigid transformation*), pri čemer ponavadi opravimo najprej rotacijo in nato translacijo. Zapišite splošno matriko toge preslikave $\mathbf{T}_{\text{toga}}$ v 2D. Kakšne so lastnosti toge preslikave (vzporednost premic, ohranjanje kotov, ohranjanje razdalj)? Namig: Poskusite različne toge preslikave na testni sliki test-125x125-08bit.raw (dimenzija 125×125 slikovnih elementov, velikost slikovnega elementa 1,0×1,0 milimetra, 8-bitni zapis).

2. Togo preslikavo lahko združimo s skaliranjem z vezanim parametrom $k$; $k = k_x = k_y$, ter tako dobimo podobnostno preslikavo (ang. *similarity transformation*). Zapišite splošno matriko podobnostne preslikave $\mathbf{T}_{\text{pod}}$ v 2D. Kakšne so lastnosti podobnostne preslikave

(vzporednost premic, ohranjanje kotov, ohranjanje razdalj)? Namig: Poskusite različne toge preslikave na testni sliki.

3. Kakšne so lastnosti splošne afine preslikave (vzporednost premic, ohranjanje kotov, ohranjanje razdalj)? Namig: Poskusite različne toge preslikave na testni sliki.

4. Priložite izris slike pri strigu z `iStrig` = [0,1, 0,5]. Izhodišče koordinatnega sistema naj bo enkrat v točki $(c_x, c_y) = (0, 0)$, drugič pa v središču slike. Za vrednost ozadja izberite `iBgr` = 255. Izris prilagodite razmerju velikosti slike v milimetrih.

5. Priložite izris slike pri rotaciji z `iRotation` = $-30°$. Izhodišče koordinatnega sistema naj bo enkrat v točki $(c_x, c_y) = (0, 0)$, drugič pa v središču slike. Za vrednost ozadja izberite `iBgr` = 255. Izris prilagodite razmerju velikosti slike v milimetrih.

6. Priložite izris slike pri translaciji z `iTranslation` = $[+20, -30]$ mm. Izhodišče koordinatnega sistema naj bo enkrat v točki $(c_x, c_y) = (0, 0)$, drugič pa v središču slike. Za vrednost ozadja izberite `iBgr` = 255. Izris prilagodite razmerju velikosti slike v milimetrih.

7. Priložite izris slike pri skaliranju z `iScaling` = [0,7, 1,4]. Izhodišče koordinatnega sistema naj bo enkrat v točki $(c_x, c_y) = (0, 0)$, drugič pa v središču slike. Za vrednost ozadja izberite `iBgr` = 255. Izris prilagodite razmerju velikosti slike v milimetrih.

8. Priložite izris slike pri togi transformaciji z `iRotation` = $-30°$ in `iTranslation` = $[+20, -30]$ mm. Izhodišče koordinatnega sistema naj bo enkrat v točki $(c_x, c_y) = (0, 0)$, drugič pa v središču slike. Za vrednost ozadja izberite `iBgr` = 255. Izris prilagodite razmerju velikosti slike v milimetrih.

9. Priložite izris slike pri podobnostni transformaciji z `iRotation` = $-30°$, `iTranslation` = $[+20, -30]$ mm in `iScaling` = [0,7, 1,4]. Izhodišče koordinatnega sistema naj bo enkrat v točki $(c_x, c_y) = (0, 0)$, drugič pa v središču slike. Za vrednost ozadja izberite `iBgr` = 255. Izris prilagodite razmerju velikosti slike v milimetrih.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Afino preslikavo smo definirali kot določeno zaporedje elementarnih preslikav, in sicer kot zaporedje strig-rotacija-translacija-skaliranje:

$$\mathbf{T}_{\text{afina}} = \mathbf{T}_{\text{strig}}\mathbf{T}_{\text{rot}}\mathbf{T}_{\text{trans}}\mathbf{T}_{\text{skal}}\,.$$

Kaj se zgodi, če izberemo neko drugo zaporedje? Obrazložite odgovor.

# Vaja 7: Prostorsko filtriranje slik

## Navodila

Prostorsko filtriranje je definirano kot lokalna preslikava sivinskih vrednosti slike, ki jo izvedemo z operatorjem $\mathcal{T}$ v prostoru slike:

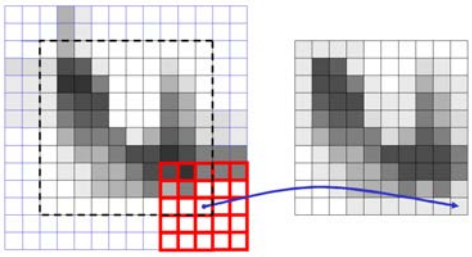
$$g(x,y) = \mathcal{T}\big(f(x,y)\big)\,,$$

kjer je $f(x,y)$ vhodna slika, $g(x,y)$ filtrirana slika in $\mathcal{T}$ lokalni operator (jedro filtra $w(i,j)$), določen na pravokotni diskretni domeni velikosti $M\times N$, pri čemer sta $M$ in $N$ celi lihi števili; $M = 2m+1$ in $N = 2n+1$. Filtrirano sliko $g(x,y)$ dobimo tako, da operator premikamo in računamo njegov odziv v vsaki diskretni točki slike s pomočjo konvolucije.

$$g(x,y) = \sum_{i=-m}^{m}\sum_{j=-n}^{n} w(i,j)\cdot f(x-i,y-j)\,,$$

Kadar so koeficienti filtra $w(i,j)$ simetrični glede na središče, je filtriranje enako korelaciji slike s filtrom.

Problem računanja odziva na robu slike ponavadi rešimo tako, da diskretno domeno slike razširimo v $x$ in $y$ smeri za vrednost $m$ in $n$, in sicer na vsaki strani slike. Sivinske vrednosti nato ekstrapoliramo preko roba slike in tako dosežemo enakomeren prehod sivinskih vrednosti.



**Glajenje slik** najenostavneje izvedemo tako, da izračunamo aritmetično povprečje vseh sivinskih vrednosti slike znotraj domene filtra. Poleg aritmetičnega povprečja se pogosto uporablja tudi uteženo povprečje, ki bolj poudari središče filtra in zato zmanjša učinek glajenja, ter glajenje z Gaussovim jedrom, ki ima koeficiente določene na podlagi Gaussove porazdelitve:

$$w(i,j) = \frac{1}{2\pi\sigma^2}\mathrm{e}^{-\frac{i^2+j^2}{2\sigma^2}}\,.$$

Aritmetično povprečje

$\frac{1}{9}\times$

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

Uteženo povprečje

$\frac{1}{16}\times$

| | | |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

Gaussov filter

| | | |
|---|---|---|
| 0,01 | 0,08 | 0,01 |
| 0,08 | 0,64 | 0,08 |
| 0,01 | 0,08 | 0,01 |

($\sigma = 0{,}5$)

**Ostrenje slik** je analogno prostorskemu odvajanju sivinskih vrednosti. Najenostavnejši operator za drugi odvod je Laplaceov operator, ostrenje slik pa se izvede tako, da se vhodni sliki $f(x,y)$ odšteje drugi odvod slike:

$$g(x,y) = f(x,y) - c\left[\nabla^2 f(x,y)\right] ,$$

kjer konstanta $c$ določa stopnjo ostrenja. Pogosto se uporablja tudi maskiranje neostrih področij, pri čemer vhodni sliki $f(x,y)$ najprej odštejemo njeno zglajeno različico $F(x,y)$. Tako dobljeno sliko maske $m(x,y)$ nato prištejemo vhodni sliki v skladu s stopnjo ostrenja $c$:

$$m\big(f(x,y)\big) = f(x,y) - F\big(f(x,y)\big) \qquad \Longrightarrow \qquad g(x,y) = f(x,y) + c\left[m\big(f(x,y)\big)\right] .$$

Slike lahko ostrimo tudi s pomočjo prvih odvodov oz. gradientov. Vektorska slika gradienta $g(x,y)$ v vsaki točki slike $(x,y)$ kaže v smeri največje spremembe funkcije $f(x,y)$. Velikost oz. amplitudo gradienta lahko izračunamo v vsaki točki slike kot:

$$G(x,y) = \mathrm{amp}\left[\nabla f(x,y)\right] = \sqrt{g_x^2(x,y) + g_y^2(x,y)} .$$

Komponenti gradienta $g_x(x,y)$ in $g_y(x,y)$, ki predstavljata parcialna odvoda slike v $x$ in $y$ smeri, najpogosteje izračunamo s pomočjo Sobelovega operatorja.

Laplaceov operator

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | −8 | 1 |
| 1 | 1 | 1 |

Sobelov operator za $x$ smer

| | | |
|---|---|---|
| −1 | 0 | 1 |
| −2 | 0 | 2 |
| −1 | 0 | 1 |

Sobelov operator za $y$ smer

| | | |
|---|---|---|
| −1 | −2 | −1 |
| 0 | 0 | 0 |
| 1 | 2 | 1 |

Pri **statističnem filtriranju** sivinske vrednosti slikovnih elementov znotraj domene filtra uredimo po velikostnem redu od največje do najmanjše. Najbolj znan statistični filter je medianin filter. Mediana je po definiciji sredinska vrednost v urejenem nizu vrednosti, torej je odziv medianinega filtra velikosti $N \times N$ enak $\frac{N(N+1)}{2}$-ti največji sivinski vrednosti. Poleg mediane lahko določimo tudi filter maksimalne vrednosti in filter minimalne vrednosti.

Dana je slika cameraman-256x256-08bit.raw velikosti $X \times Y = 256 \times 256$, ki je zapisana z 8 biti v obliki surovih podatkov (RAW).

1. Napišite funkcijo za prostorsko filtriranje slike z jedrom:

   ```
   function oImage = kernelFiltering(iImage, iKernel),
   ```

   kjer je `iImage` vhodna slika in `iKernel` jedro filtra dimenzije $N \times N$. Funkcija vrne matriko filtriranih vrednosti `oImage`, ki ima enake dimenzije kot vhodna slika. V splošnem dinamično območje filtriranih vrednosti ne sovpada z dinamičnim območjem slike, ki jo filtriramo, poleg tega pa filtrirane vrednosti niso nujno celoštevilčne. Pri prikazovanju zato spremenimo dinamično območje filtriranih vrednosti na 8 bitov: `oImage = uint8(oImage)`.

2. Uporabite različna jedra filtrov za glajenje slik (aritmetično povprečje, uteženo povprečje, Gaussovo jedro) ter zgladite sliko. Rezultate prikažite na zaslon.

3. Izostrite sliko s postopkom na podlagi Laplaceovega operatorja in z maskiranjem neostrih področij. Za stopnjo ostrenja izberite vrednost $c = 2$. Rezultate prikažite na zaslon.

4. Določite gradiente slike v $x$ in $y$ smeri s pomočjo Sobelovega operatorja. Izračunajte tudi amplitudno sliko gradienta. Rezultate prikažite na zaslon.

5. Napišite funkcijo za statistično filtriranje slike:

   ```
   function oImage = statisticalFiltering(iImage, iLength, iFunc),
   ```

   kjer je `iImage` vhodna slika, `iLength` je dolžina $N$ jedra filtra dimenzije $N \times N$, `iFunc` pa je ime Matlab-ove statistične funkcije, npr. `'median'`. Filtriranje opravite na naslednji način:

   ```
   oValue = feval(iFunc, iVector),
   ```

   kjer je `iVector` vektor sivinskih vrednosti, ki ustrezajo danemu položaju filtra na sliki, `oValue` pa filtrirana vrednost na danem položaju filtra. Funkcija vrne filtrirano sliko `oImage`.

6. Filtrirajte sliko z medianinim filtrom, filtrom maksimalne vrednosti ter filtrom minimalne vrednosti. Rezultate prikažite na zaslon.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Zapišite jedro filtra velikosti $N \times N = 5 \times 5$ za glajenje z aritmetičnim povprečjem, za glajenje z uteženim povprečjem ter za glajenje z Gaussovim jedrom pri $\sigma = 2$.

2. Priložite slike, zglajene aritmetičnim povprečjem ($N \times N = 3 \times 3$), uteženim povprečjem ($N \times N = 3 \times 3$) ter Gaussovim jedrom ($N \times N = 3 \times 3$ in $\sigma = 0,5$). Za kaj se v splošnem uporablja glajenje slik?

3. Priložite slike, izostrene z Laplaceovim operatorjem ter z maskiranjem neostrih področij. Uporabite velikost jedra filtra velikosti $N \times N = 3 \times 3$, standardno deviacijo $\sigma = 2$ in stopnjo ostrenja $c = 2$. Priložite tudi sliko odziva na Laplaceov operator ter sliko maske neostrih področij. Za kaj se v splošnem uporablja ostrenje slik? Kaj je pomanjkljivost ostrenja, ki je opazna tudi na pridobljenih slikah?

4. Priložite slike Sobelovih gradientov v $x$ in $y$ smeri ter amplitudno sliko Sobelovega gradienta. Uporabite velikost jedra filtra velikosti $N \times N = 3 \times 3$. Za kaj se v splošnem uporablja določanje gradientov slik?

5. Priložite slike, filtrirane z medianinim filtrom, filtrom maksimalne vrednosti in filtrom minimalne vrednosti. Uporabite velikost jedra filtra velikosti $N \times N = 3 \times 3$. Za kaj se v splošnem uporablja statistično filtriranje slik? Kaj povzroči uporaba vsakega od navedenih filtrov na sliki? Kakšna je bistvena razlika med filtiranjem z medianinim filtrom in s filtri za glajenje slik?

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

1. Napišite funkcijo za določanje koeficientov jedra filtra arimtetičnega povprečja, filtra uteženega povprečja in Gaussovega filtra:

```
function oKernel = getAritmeticAverageKernel(iLength),
function oKernel = getWeightedAverageKernel(iLength),
function oKernel = getGaussianKernel(iLength, iStd),
```

   kjer je `iLength` poljubna velikost jedra filtra $N$, `iStd` pa standardna deviacija Gaussove porazdelitve. Funkcija vrne jedro pripadajočega filtra `oKernel`. Upoštevajte, da so koeficienti jedra simetrični ter da mora biti vsota vseh koeficientov enaka 1.

2. Pri filtriranju dvodimenzionalnih (2D) slik je jedro poljubnega filtra podano z 2D matriko. Podobno je pri filtriranju tridimenzionalnih (3D) slik jedro poljubnega filtra podano s 3D matriko. Recimo, da za določanje gradienta v $x$ smeri uporabimo naslednjo obliko Sobelovega operatorja:

`SobelX(:,:,1) =`

$$= \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

`SobelX(:,:,2) =`

$$= \begin{bmatrix} -2 & 0 & 2 \\ -4 & 0 & 4 \\ -2 & 0 & 2 \end{bmatrix}$$

`SobelX(:,:,3) =`

$$= \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

   Kakšna je oblika pripadajočih Sobelovih operatorjev za filtriranje 3D slik v $y$ in $z$ smeri?

# Vaja 8: Morfološka obdelava slik

## Navodila

Morfološka obdelava slik zajema tehnike za izluščevanje slikovnih komponent, ki so uporabne za predstavitev in opisovanje oblike področij, npr. robovi, skelet, itn. Z morfološko operacijo v splošnem določimo vrednost izbranega slikovnega elementa izhodne slike na podlagi "pravila", ki ga uporabimo na pripadajočemu slikovnemu elementu ter njemu sosednjim slikovnim elementom vhodne slike. Pravilo določa **strukturni element**, katerega velikost in obliko prilagodimo glede na objekte v sliki, ki jih želimo morfološko obdelati. Predpostavimo, da ima strukturni element obliko pravokotnika velikosti $M \times N$, kjer sta $M$ in $N$ celi lihi števili ($M = 2m + 1,\ N = 2n + 1$), ter da strukturni element zavzema le binarne vrednosti (0 ali 1). Strukturni element $b(i, j)$ lahko torej predstavimo v obliki binarne maske pravokotne oblike $b(i, j) \in \{0, 1\}, \forall i = -m \ldots + m, \forall j = -n \ldots + n$, npr.:

$M \times N = 3 \times 3$

| | | |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

$M \times N = 1 \times 3$

| |
|---|
| 1 |
| 1 |
| 1 |

$M \times N = 5 \times 5$

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |

Osnovni operaciji morfološke obdelave slik sta erozija in dilacija. **Morfološka erozija** sivinske slike $f(x, y)$ s strukturnim elementom $b(i, j)$ je definirana kot:

$$(f \ominus b)(x, y) = \min_{\forall i,j} \Big\{ f(x - i, y - j) \cdot b(i, j) \Big\}.$$

Vrednost slikovnega elementa izhodne slike je torej enaka najmanjši vrednosti v okolici pripadajočega slikovnega elementa vhodne slike. **Morfološka dilacija** sivinske slike $f(x, y)$ s strukturnim elementom $b(i, j)$ je definirana kot:

$$(f \oplus b)(x, y) = \max_{\forall i,j} \Big\{ f(x - i, y - j) \cdot b(i, j) \Big\}.$$

Vrednost slikovnega elementa izhodne slike je torej enaka največji vrednosti v okolici pripadajočega slikovnega elementa vhodne slike. Na podlagi osnovnih morfoloških operacij lahko definiramo še nekatere napredne operacije, med katerimi se najbolj uporabljata morfološko odpiranje in zapiranje. **Morfološko odpiranje** sivinske slike $f(x, y)$ s strukturnim elementom $b(i, j)$ je definirano kot:

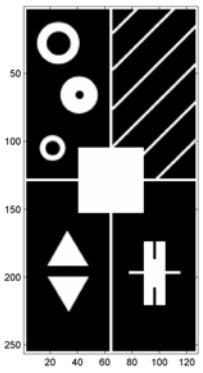
$$(f \circ b)(x, y) = \Big( (f \ominus b)(x, y) \oplus b \Big)(x, y) \quad \text{oz.} \quad f \circ b = (f \ominus b) \oplus b$$

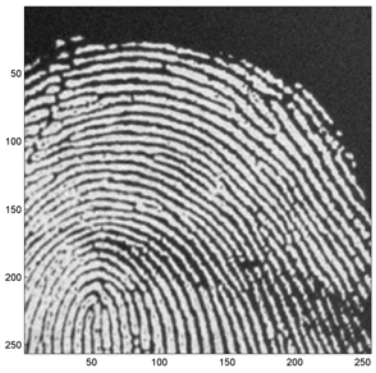
in je torej enako dilaciji z $b(i, j)$ rezultata erozije $f(x, y)$ z $b(i, j)$. **Morfološko zapiranje** sivinske slike $f(x, y)$ s strukturnim elementom $b(i, j)$ je definirano kot:

$$(f \bullet b)(x, y) = \Big( (f \oplus b)(x, y) \ominus b \Big)(x, y) \quad \text{oz.} \quad f \bullet b = (f \oplus b) \ominus b$$

in je torej enako eroziji z $b(i, j)$ rezultata dilacije $f(x, y)$ z $b(i, j)$.

Dana je testna slika test-128x256-08bit.raw velikosti $X \times Y = 128 \times 256$, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Poleg tega je dana tudi realna slika real-256x256-08bit.raw velikosti $X \times Y = 256 \times 256$, ki je ravno tako zapisana z 8 biti v obliki surovih podatkov (RAW).




1. Napišite funkcijo za morfološko erozijo slike:

   ```
   function oImage = morphErosion(iImage, iStruct),
   ```

   kjer je `iImage` vhodna slika, `iStruct` pa binarni pravokotni strukturni element. Funkcija vrne morfološko erodirano sliko `oImage`, ki ima enake dimenzije kot vhodna slika. Domeno vhodne slike razširite glede na velikost strukturnega elementa, tako da bo erozija na robovih slike vrnila ustrezne vrednosti (domeno razširimo z največjo vrednostjo dinamičnega območja sivinskih vrednosti, torej z 255).

2. Napišite funkcijo za morfološko dilacijo slike:

   ```
   function oImage = morphDilation(iImage, iStruct),
   ```

   kjer je `iImage` vhodna slika, `iStruct` pa binarni pravokotni strukturni element. Funkcija vrne morfološko dilatirano sliko `oImage`, ki ima enake dimenzije kot vhodna slika. Domeno vhodne slike razširite glede na velikost strukturnega elementa, tako da bo dilacija na robovih slike vrnila ustrezne vrednosti (domeno razširimo z najmanjšo vrednostjo dinamičnega območja sivinskih vrednosti, torej z 0).

3. Napišite funkcijo za morfološko odpiranje slike:

   ```
   function oImage = morphOpening(iImage, iStruct),
   ```

   kjer je `iImage` vhodna slika, `iStruct` pa binarni pravokotni strukturni element. Funkcija vrne morfološko odprto sliko `oImage`, ki ima enake dimenzije kot vhodna slika. Funkcijo napišite s pomočjo funkcij `morphErosion()` in `morphDilation()`.

4. Napišite funkcijo za morfološko zapiranje slike:

   ```
   function oImage = morphClosing(iImage, iStruct),
   ```

   kjer je `iImage` vhodna slika, `iStruct` pa binarni pravokotni strukturni element. Funkcija vrne morfološko zaprto sliko `oImage`, ki ima enake dimenzije kot vhodna slika. Funkcijo napišite s pomočjo funkcij `morphErosion()` in `morphDilation()`.

5. Preizkusite delovanje morfološke erozije, dilacije, odpiranja in zapiranja na testni in realni sliki, pri čemer uporabite različne strukturne elemente.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Priložite slike, ki ste jih pridobili s pomočjo morfološke erozije, dilacije, odpiranja in zapiranja testne slike na podlagi strukturnega elementa velikosti $5\times5$, ki je podan v navodilih.

2. Priložite slike, ki ste jih pridobili s pomočjo morfološke erozije, dilacije, odpiranja in zapiranja realne slike na podlagi strukturnega elementa velikosti $5\times5$, ki je podan v navodilih.

3. Kakšen je rezultat morfoloških operacij, če uporabimo strukturni element velikosti $3\times1$, ki je podan v navodilih? Kakšen je rezultat, če ta strukturni element zavrtimo za $90^\circ$ in postane torej velikosti $1\times3$? Obrazložite odgovor.

4. Kakšni so učinki morfološke erozije oziroma dilacije? Obrazložite odgovor.

5. Kakšni so učinki morfološkega odpiranja oziroma zapiranja? Obrazložite odgovor.

6. Prikažite razliko med slikama, pridobljenima z morfološko dilacijo in morfološko erozijo, in sicer za testno in za realno sliko. Kaj predstavlja taka slika?

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Ker je dana realna slika sivinska, lahko morfološke postopke in postopek upragovljanja izvedemo na dva načina:

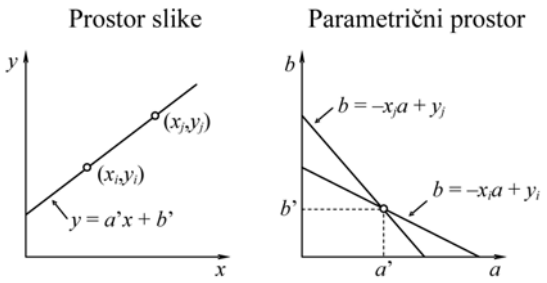
- Sliko najprej upragovimo s poljubnim pragom $t$ ter nato na upragovljeni sliki izvedemo poljubno morfološko operacijo.
- Na sivinski sliki najprej izvedemo poljubno morfološko operacijo ter nato sliko upragovimo s poljubnim pragom $t$.

Primerjajte rezultate, pridobljene na oba načina.

# Vaja 9: Razgradnja slik

## Navodila

Razgradnja ali segmentacija slik (ang. *image segmentation*) združuje postopke, s katerimi sliko razdelimo na osnovna področja oziroma objekte. Med postopke iskanja objektov na sliki spada tudi iskanje premic. Učinkovit postopek iskanja premic na sliki temelji na **Houghovi transformaciji**. Predpostavimo, da imamo binarno sliko robov $f(x, y)$. Skozi poljubno izbrano robno točko $(x_i, y_i)$ lahko potegnemo poljubno število premic oblike:




$$y_i = ax_i + b\,.$$

Isto enačbo lahko zapišemo v prostoru parametrov $(a, b)$, kjer sta koordinati robne točke $x_i$ in $y_i$ parametra premice:

$$b = -x_i a + y_i\,.$$

Za neko drugo robno točko $(x_j, y_j)$ dobimo v prostoru parametrov $(a, b)$ še eno premico oblike:

$$b = -x_j a + y_j\,,$$

ki se v neki točki $(a', b')$ seka s premico, ki pripada točki $(x_i, y_i)$. Parametra $(a', b')$ določata enačbo premice v prostoru slike $(x, y)$, ki poteka skozi točki $(x_i, y_i)$ in $(x_j, y_j)$.

Na ta način bi lahko za vsak par robnih točk v prostoru slike $(x, y)$ izračunali parametra povezujoče premice ter nato v prostoru parametrov $(a, b)$ narisali ustrezno točko $(a', b')$. Nato bi lahko poiskali najbolj pogoste parametre premic v prostoru parametrov $(a, b)$ ter pripadajoče premice v prostoru slike $(x, y)$, ki povezujejo veliko število robnih točk. Problem pri tovrstnem iskanju premic je v tem, da gre pri navpičnih premicah vrednost parametra $a$ in s tem velikost prostora parametrov $(a, b)$ proti neskončnosti. Temu se lahko izognemo tako, da zapišemo premico s pomočjo polarnih koordinat $(r, \varphi)$, in sicer kot:

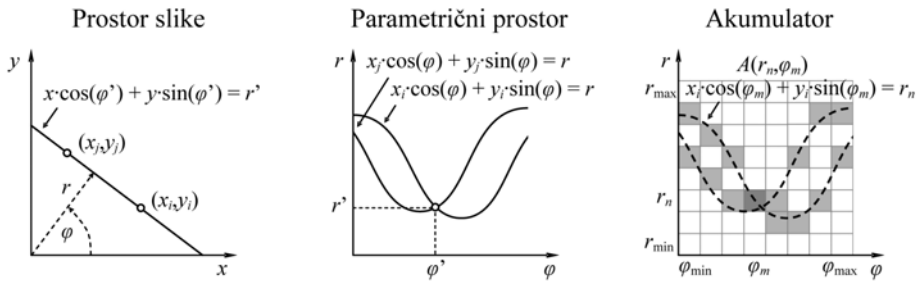
$$x\cos\varphi + y\sin\varphi = r\,.$$

Za navpično premico ($\varphi = 0°$) določa $r$ presečišče premice z $x$ osjo, za vodoravno premico ($\varphi = 90°$) pa določa $r$ presečišče premice z $y$ osjo. V prostoru parametrov $(r, \varphi)$ tako vsaka sinusna krivulja:

$$x_i\cos\varphi + y_i\sin\varphi = r$$

predstavlja množico premic, ki gredo v prostoru slike $(x, y)$ skozi točko $(x_i, y_i)$. Presečišče dveh sinusnih krivulj $(r', \varphi')$ torej določa parametra premice, ki poteka skozi točki $(x_i, y_i)$ in $(x_j, y_j)$ v prostoru slike:

$$x\cos\varphi' + y\sin\varphi' = r'\,.$$




Glavna prednost Houghove transformacije je v tem, da lahko prostor parametrov $(r, \varphi)$ diskretiziramo, tako da dobimo ti. akumulatorske celice. Seveda je potrebno določiti območje parametrov $r$ in $\varphi$ tako, da lahko opišeta poljubne premice skozi vse točke slike, ter ju razdeliti na želeno število akumulatorskih celic:

$$(r_{\min}, r_{\max}) = (0, r_{\text{diag}}) \qquad \text{in} \qquad (\varphi_{\min}, \varphi_{\max}) = (-90^\circ, +180^\circ)\,.$$

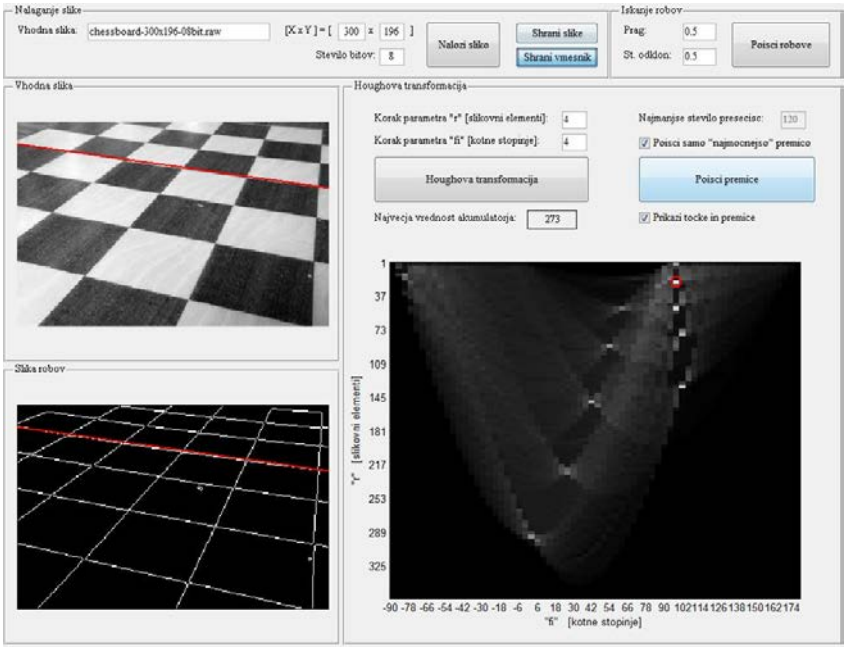
Vrednosti vseh akumulatorskih celic $A(r_n, \varphi_m)$ najprej postavimo na 0, nato pa za vsako robno točko $(x_i, y_i)$ in za vsako možno diskretno vrednost $\varphi_m$ na intervalu $(\varphi_{\min}, \varphi_{\max})$ izračunamo ustrezno vrednost parametra $r$. Vrednost parametra $r$ zaokrožimo na najbližjo diskretno vrednost $r_n$ in vrednost akumulatorske celice $(r_n, \varphi_m)$ povečamo za 1:

$$A(r_n, \varphi_m) \quad \leftarrow \quad A(r_n, \varphi_m) + 1\,.$$

Po končanem postopku vrednost vsake celice akumulatorja $A(r_n, \varphi_m)$ predstavlja število robnih točk, ki v prostoru slike ležijo na premici:

$$x \cos \varphi_m + y \sin \varphi_m = r_n\,.$$

Premice, ki predstavljajo robne točke slike, določimo tako, da poiščemo tiste celice akumulatorja $A(r_n, \varphi_m)$, ki imajo dovolj velike in lokalno največje vrednosti.



Dana je arhivska datoteka vaja09.zip v ZIP formatu. Datoteke, ki se nahajajo v arhivu, shranite v izbrani direktorij ter v Matlabu poženite uporabniški vmesnik vaja09.m. Uporabniški vmesnik omogoča nalaganje vhodne slike, določanje binarne slike robov, izris slike akumulatorja Houghove transformacije, iskanje lokalnih maksimumov v akumulatorju, izris pripadajočih premic in točk ter shranjevanje slik.

V skladu z navodili dopolnite oziroma spremenite obstoječo funkcijo houghTransform2D2P.m za Houghovo transformacijo v dveh dimenzijah (2D) za premice v dvoparametričnem polarnem prostoru $(r, \varphi)$:

```
>>  % Houghova transformacija v 2D za 2 parametra
>>  function [oAcc, rangeR, rangeF] = houghTransform2D2P(iImage, stepR, stepF)
>>      % slepa nastavitev izhodnih vrednosti
>>      oAcc = 0;
>>      rangeR = 1;
>>      rangeF = 1;
```

Vhodne in izhodne spremenljivke so enake:

| | |
|---|---|
| `iImage` | Binarna slika robov, kjer vrednosti različne od nič predstavljajo rob na originalni sliki: $f(x_i, y_i) \neq 0 \;\rightarrow\; (x_i, y_i)$ je robna točka. |
| `stepR` | Korak spremembe parametra $r$: $\Delta r = r_{n+1} - r_n$. |
| `stepF` | Korak spremembe parametra $\varphi$: $\Delta\varphi = \varphi_{m+1} - \varphi_m$. |
| `oAcc` | Akumulator Houghove transformacije: $A(r_n, \varphi_m)$. |
| `rangeR` | Vektor vrednosti parametra $r$: $[r_1, r_2, \ldots, r_n, \ldots, r_{\max}]$. |
| `rangeF` | Vektor vrednosti parametra $\varphi$: $[\varphi_1, \varphi_2, \ldots, \varphi_m, \ldots, \varphi_{\max}]$. |

Delovanje algoritma preizkusite na sliki image1-300x196-08bit.raw velikosti $X \times Y = 300 \times 196$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Poizkusite tudi s sliko image2-300x196-08bit.raw enake velikosti, kateri je bil umetno dodan šum z Gaussovo porazedlitvijo. Pri uporabi detektorja robov boste morali zato za primerno sliko robov ustrezno nastaviti prag ter standardno deviacijo detektorja.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Priložite slike uspešnega iskanja najbolj izrazite premice ($n = 1$) na sliki z dodanim šumom (image2-300x196-08bit.raw). Kaj lahko sklepate o Houghovi transformaciji iz rezultatov tega poskusa? Obrazložite odgovor.
2. Priložite slike uspešnega iskanja najbolj izrazitih premic ($n = 1$) na sliki z dodanim šumom (image2-300x196-08bit.raw). Vrednost najmanjšega števila presečišč nastavite na največjo vrednost, ki omogoča izris čimveč premic, ki jasno razmejujejo iskane objekte. Zapišite izbrane vrednosti parametrov iskanja robov (prag, standardna deviacija) ter izbrano vrednost najmanjšega števila presečišč v akumulatorju.

3. Kaj se zgodi, če zmanjšamo koraka spremembe parametrov $r$ in $\varphi$ na najnižjo vrednost ter iščemo večje število premic? Obrazložite odgovor.

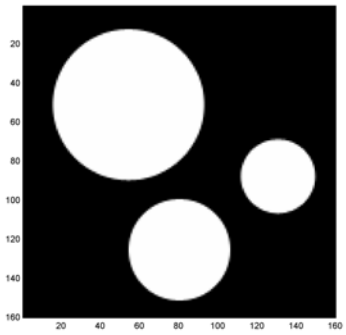
4. Od česa je odvisna natančnost določanja premic s pomočjo Houghove transformacije? Obrazložite odgovor.

5. Kakšne so lastnosti Houghove transformacije? Obrazložite odgovor.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Dana je slika image3-160x160-08bit.raw velikosti $X \times Y = 160 \times 160$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). S pomočjo Houghove transformacije poskusite določiti najbolj izrazito krožnico na sliki.

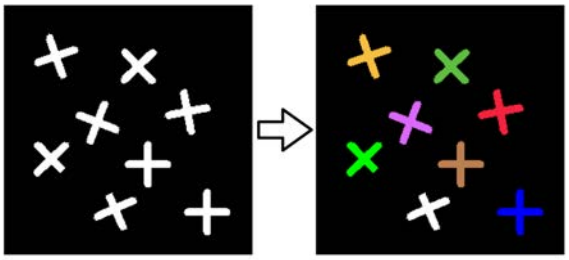



Opomba: Program bo seveda popolnoma ločen od danega uporabniškega vmesnika, ki je prilagojen za problem s premicami.

Univerza v Ljubljani, Fakulteta za elektrotehniko
Modul A: Slikovna informatika (64135) - Laboratorijske vaje

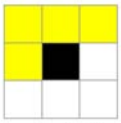
# Vaja 10: Označevanje objektov

## Navodila



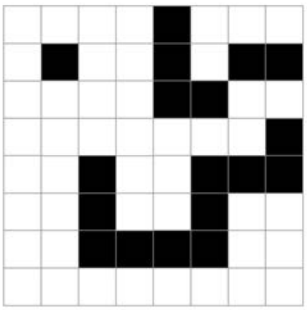
Označevanje objektov v splošnem pomeni, da vsakemu (binarnemu) objektu na sliki priredimo oznako oz. kodo. Take oznake predstavljajo zgoščen zapis objektov in omogočajo lažje opisovanje njihovih zunanjih lastnosti (npr. obseg, usmeritev) in notranjih lastnosti (npr. barva, tekstura). Eden izmed enostavnejših postopkov za označevanje objektov se imenuje **označevanje povezanih komponent** (ang. *connected components labeling*). Postopek temelji na pregledovanju slike po vrsticah in ko naletimo na objekt, mu priredimo oznako glede na štiri sosednje že pregledane točke (leva in zgornje tri točke):

- če so vse štiri sosednje točke še neoznačene, priredimo novo oznako;
- če ima samo ena sosednja točka oznako, priredimo oznako te točke;
- če ima več kot ena sosednja točka oznako, priredimo eno oznako in zabeležimo ekvivalentnost ostalih.

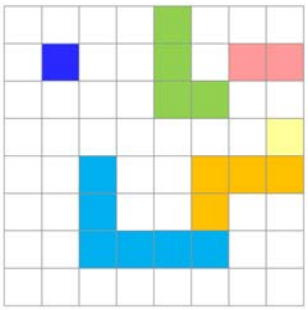


Trenutna točka in sosednje točke

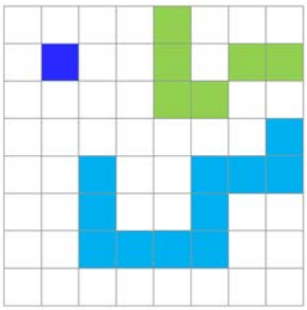
Sledi ponoven pregled slike po vrsticah, v katerem združujemo ekvivalentne oznake v objekte.



Binarna slika z objekti

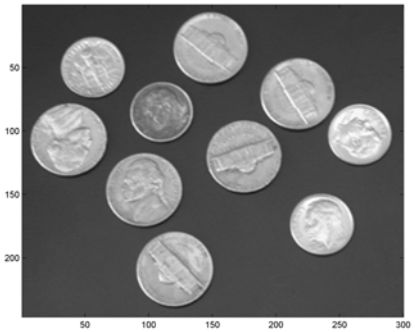


1. prelet in začetne (ekvivalentne) oznake



2. prelet in končne (združene) oznake

Dana je slika coins-300x246-08bit.raw velikosti $X \times Y = 200 \times 100$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Slika predstavlja kovance, ki so brez prekrivanja razporejeni na temnejšem ozadju in predstavljajo objekte na sliki.



1. Originalno sliko najprej upragovite, pri čemer si lahko pomagate s funkcijo `thresholdImage()`, ki ste jo uporabili pri Vaji 5: Sivinske preslikave slik.

2. Napišite funkcijo za označevanje objektov:

   ```
   function oImage = labelImage(iImage),
   ```

   kjer je `iImage` vhodna binarna slika (oz. upragovljena slika), v kateri določate oznake objektov, `oImage` pa je izhodna slika enake velikosti kot vhodna slika, v kateri je vsak objekt predstavljen z unikatno oznako.

   Za določanje ekvivalentnosti oznak uporabite že napisano funkcijo:

   ```
   function oList = equalizeLabels(iList, iVector, iValue),
   ```

   kjer je `iList` seznam oznak (potrebna je ustrezna predhodna inicializacija s prazno matriko), `iVector` je vrstični vektor oznak v sosednjih točkah, `iValue` pa je oznaka iz tega vektorja, ki je bila prirejena opazovanemu slikovnemu elementu. Izhodni seznam `oList` predstavlja novi seznam oznak.

3. Napišite funkcijo za barvno kodiranje oznak:

   ```
   function oImage = encodeColors(iImage, iColors),
   ```

   kjer je `iImage` vhodna slika velikosti $X \times Y$, v kateri določate oznake objektov, `iColors` pa je matrika barv dimenzij $M \times 3$, v kateri vsaka vrstica predstavlja eno barvo in je njena rdeča (R), zelena (G) in modra (B) barvna komponenta zapisana v posameznih stolpcih ($M$ je poljubno število različnih barv). Izhodna slika `oImage` je matrika velikosti $X \times Y \times 3$, vsaka plast matrike (tretja dimenzija) pa predstavlja sliko R, G in B barvne komponente.

4. Prikažite na zaslon originalno sliko, upragovljeno sliko, sliko z označenimi objekti ter sliko z barvno zakodiranimi objekti. Za prikazovanje barvne slike lahko uporabite obstoječo funkcijo `displayImage()`.

## Vprašanja

Poročilo vaje vsebuje lastnoročno (brez uporabe računalnika) zapisane odgovore na naslednja vprašanja, medtem ko so zahtevane slike natisnjene ter priložene poročilu.

1. Za vrednost praga izberite smiselno vrednost ter priložite izrise originalne slike, upragovljene slike, slike z označenimi objekti in slike z barvno zakodiranimi objekti.

2. Koliko je različnih oznak na sliki, če izberete vrednost praga $t = 70$, ter koliko, če izberete vrednost praga $t = 90$.

3. Kako bi za dani problem določili optimalno vrednost praga $t$? Opišite postopek ter napišite program v Matlab-u.

## Dodatek

Odgovore na sledeče probleme ni potrebno prilagati poročilu. Služijo naj v razmislek ter kot vzpodbuda za boljše razumevanje vsebine.

Dana je slika plates-508x266-08bit.raw velikosti $X \times Y = 508 \times 266$ slikovnih elementov, ki je zapisana z 8 biti v obliki surovih podatkov (RAW). Slika predstavlja avtomobilske registrske tablice, potrebno pa je označiti čimveč črk in števk na registraciji.

- Opišite postopek, podajte vrednost uporabljenega praga $t$ ter priložite izrise originalne slike, upragovljene slike, slike z označenimi objekti in slike z barvno zakodiranimi objekti.
- V čem se pristop bistveno razlikuje od prej uporabljenega pristopa? Obrazložite odgovor.



http://lit.fe.uni-lj.si/SI/
http://lit.fe.uni-lj.si/gradivo/SI-LabVaje-slo.pdf