

# *Napredna obdelava slik*

*(2. letnik RIT – MAG, l. 2025/2026)*

izr. prof. dr. Božidar Potočnik

Fakulteta za elektrotehniko, računalništvo in informatiko

Kabinet: G2-2N.37 (33)

Govorilne ure: sreda, 9.00 – 11.00

E-mail: bozidar.potocnik@um.si

# **Predvideni študijski rezultati in obveznosti študenta**

- ▷ **Cilji:**
  - ★ Vpeljati študente v napredne tehnike obdelave dvo ali več dimenzionalnih digitalnih slik.
- ▷ **Znanje in razumevanje:**
  - ★ izkazati znanje in razumevanje naprednih postopkov predobdelave in segmentacije slik ter jih analizirati in uporabiti pri načrtovanju rešitev za specifične problemske domene,
  - ★ izbrati ustrezno metodo glede na prepoznan segmentacijski problem,
  - ★ oceniti in ovrednotiti rešitve razvitih tehnik napredne obdelave slik.
- ▷ **ECTS:** 6 točk  $\approx$  180 ur dela
- ▷ **Kontaktne ure:**
  - ★ Predavanja: 30 ur
  - ★ Laboratorijske oz. računalniške vaje: 30 ur
- ▷ **Samostojno delo študenta:**  $\approx$  120 ur
  - ★ Dokončanje laboratorijskih vaj in priprava na pisni izpit

# **Načini ocenjevanja in razpored predavanj**

## ▷ **Načini ocenjevanja:**

- ★ Končno oceno tvorijo:
  - ▷ Opravljene laboratorijske vaje – 50 % oz. 500 točk (minimalno 25 % oz. 250 točk)
  - ▷ Pisni izpit – 50 % oz. 500 točk (minimalno 25 % oz. 250 točk)
- ★ **Ne pozabite se prijavljati na izpitne roke!**

## ▷ **Predviden razpored predavanj:**

- |  |                                       |
|--|---------------------------------------|
| 1. teden: 2.10.                            | 10. teden: 4.12.                      |
| 2. teden: 9.10.                            | 11. teden: 11.12.                     |
| 3. teden: 16.10.                           | 12. teden: 18.12.                     |
| 4. teden: 23.10.                           | 13. teden: 8.1.                       |
| 5. teden: 30.10.                           | 14. teden: 15.1.                      |
| 6. teden: 6.11.                            | 15. teden: <b>22.1. – 2. kolokvij</b> |
| 7. teden: 13.11.                           |                                       |
| 8. teden: <b>pon, 17.11. – 1. kolokvij</b> |                                       |
| 9. teden: <b>pon, 24.11.</b>               |                                       |

# **Vsebina in literatura**

## ▷ **Vsebina:**

1. Konvolucijske nevronske mreže
2. Napredni koncepti zajemanja slik
3. Napredna predobdelava slik
4. Napredna segmentacija
5. Slikovne transformacije

## ▷ **Literatura:**

- ★ Gradivo iz predavanj in zapiski
- ★ Gradivo iz predavanj in zapiski za predmet Uvod v računalniški vid in razpoznavanje vzorcev
- ★ B. Potočnik, Osnove razpoznavanja vzorcev z nevronskimi mrežami, UM-FERI, 2007.
- ★ (Ostali temeljni študijski viri navedeni v učnem načrtu predmeta – spletna stran FERI)

# **Kazalo**

1. Konvolucijske nevronske mreže .....	6
2. Napredni koncepti zajemanja slik .....	34
3. Napredna predobdelava slik .....	61
4. Napredna segmentacija .....	84
5. Slikovne transformacije .....	128

# **1. Konvolucijske nevronske mreže**

---

Umetni nevron in umetne nevronske mreže

Konvolucijske nevronske mreže

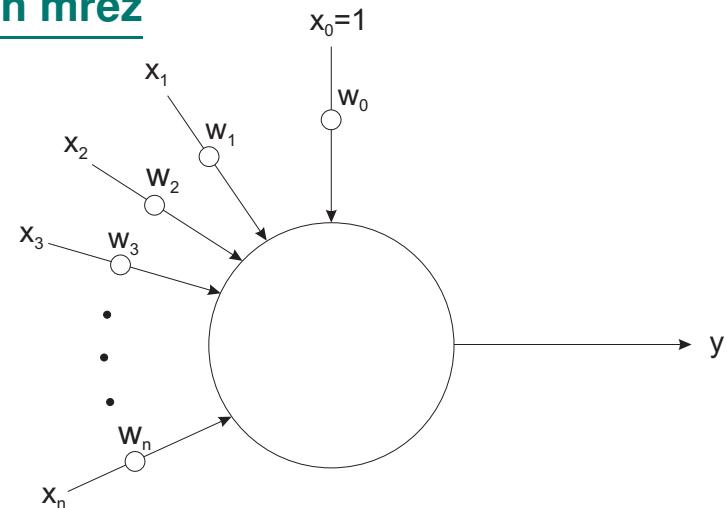
Inženirski pristop k globokim nevronskim mrežam

Aplikacije

# Umetni nevron in umetne nevronske mreže

## Ponovitev: Neuron – osnovni gradnik nevronskega mrež

- ★ Vhodna vlakna (dendrite) modeliramo z vhodi  $x_i$
- ★ Izhodno vlakno (akson) modeliramo z izhodom  $y$
- ★ Prenosno učinkovitost sinaps modeliramo z realnim številom  $w_i$  (tj. utež vhoda)



- ▷ Obtežen vhod v celično telo – seštevek električnih impulzov iz vseh dendritov oz. skalarni produkt med vhodom in utežmi

$$s = \sum_{i=1}^n w_i x_i + b \quad \text{oz.} \quad s = \mathbf{w} \mathbf{x} + b \quad \text{oz.} \quad s = \sum_{i=0}^n w_i x_i \quad \text{oz.} \quad s = \hat{\mathbf{w}} \hat{\mathbf{x}}$$

- ★ utež  $w_0$  določa prag (*bias*), v literaturi pogosto označen kot  $b$
- ★  $\mathbf{w} = [w_1, w_2, \dots, w_n]$  oz.  $\hat{\mathbf{w}} = [w_0, \mathbf{w}]$
- ★  $\mathbf{x} = [x_1, x_2, \dots, x_n]$  oz.  $\hat{\mathbf{x}} = [1, \mathbf{x}]$
- ▷ Izhod  $y$  – monotono naraščajoča funkcija obteženega vhoda  $s$

$$y = f(s)$$

- ★ Funkcija  $f$  – prenosna oz. aktivacijska funkcija nevrona
- ★ Najbolj uporabljane: ReLU, Leaky ReLU, Maxout, tanh in sigmoidna.

# **Umetni nevron in umetne nevronske mreže** (2)

## ReLU (*REctified Linear Unit*)

$$f(s) = \max(0, s)$$

- ▷ Danes prva izbira. Pospešuje hitrost učenja NN, enostavna implementacija.

## Leaky ReLU

$$f(s) = \max(as, s) \quad a > 0, a \in \mathbb{R}, \quad (\text{npr. } a = 0,01)$$

- ▷ Rešuje problem "umiranja" funkcije ReLU za  $s < 0$ .
- ▷ Žal izboljšanje zgolj pri nekaterih aplikacijah.

## Maxout

$$f(\mathbf{x}) = \max(\mathbf{w}_0 \mathbf{x} + b_0, \mathbf{w}_1 \mathbf{x} + b_1)$$

- ▷ Aktivacijska funkcija ni klasične oblike  $f(s)$ , kjer je  $s$  obtežen vhod.
- ▷ Odpravlja slabosti funkcij tipa ReLU, a podvoji število parametrov za nevron!

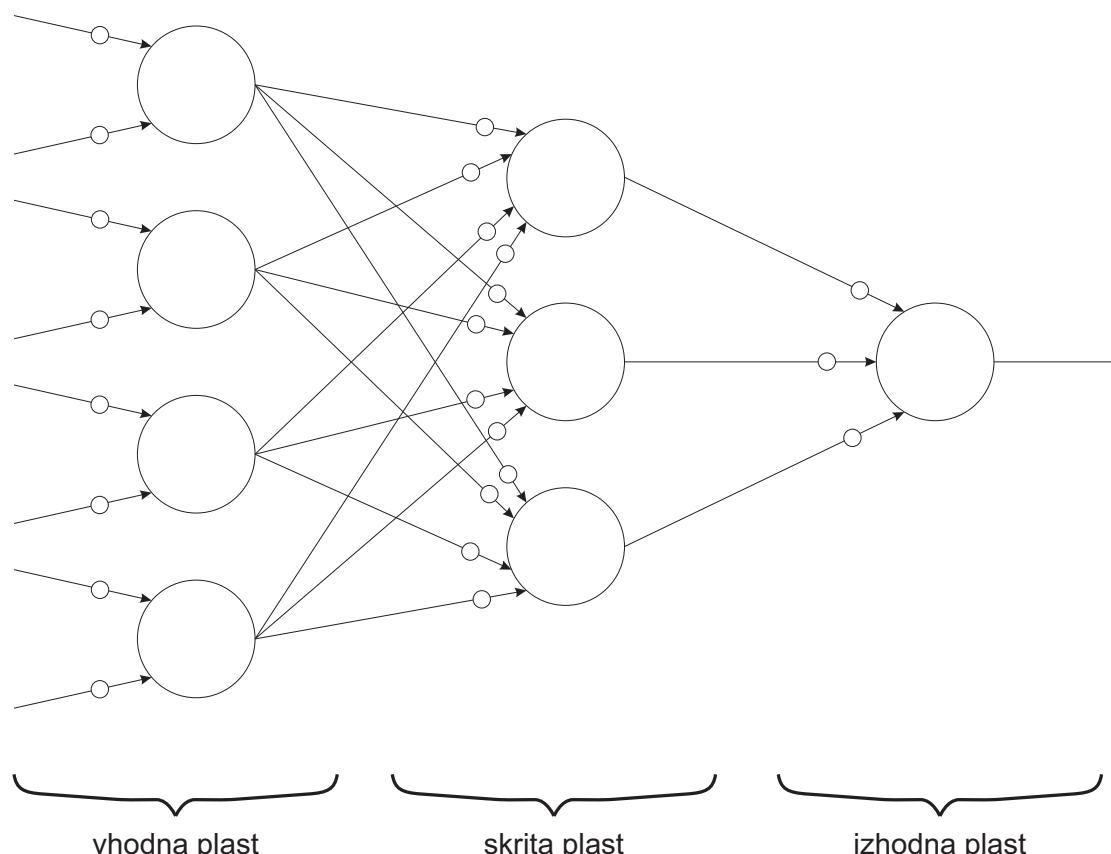
## Sigmoidna (*sigmoid, logistic*)

$$f(s) = 1/(1 + e^{-s})$$

- ▷ Med učenjem potisne gradiente v nasičenje ali jih celo "ubije"; ni dobra izbira za vhodno in skrite plasti NN.
- ▷ Funkcija tanh definirana kot:  $f(s) = 2 \text{ sigmoid}(2s) - 1$

Nevronska mreža (NN) je množica medsebojno povezanih nevronov v graf.

- ▷ **Topologija nevronske mreže** – razmestitev oz. organiziranost nevronov v mreži
- ▷ Nevroni tipično organizirani v **plasti** (*layers*), najpogosteje v **polno povezane plasti** (*fully-connected*).
- ▷ Vrsta NN so **mreže s povezavami naprej** (*feedforward NN*); nevroni so povezani v aciklični graf. Sinonimi: zvezne NN, večplastni zvezni perceptroni.



## **Umetni nevron in umetne nevronske mreže** (4)

- ▷ Zvezne večplastne perceptrone uporabljamo za identificiranje neznane funkcijске relacije  $f$  med dvema množicama  $\mathcal{X}$  in  $\mathcal{Y}$ , pri čemer velja

$$f: \mathcal{X} \longmapsto \mathcal{Y}, \quad \mathcal{X} \subset \mathbb{R}^n \wedge \mathcal{Y} \subset \mathbb{R}^m$$

- ▷ Identificiranje poteka na osnovi učne množice  $\mathcal{U}_M$  – pari: (vhod, želen izhod):

$$\mathcal{U}_M = \{(\mathbf{x}_i, t(\mathbf{x}_i)) \mid \mathbf{x}_i \in \mathcal{X} \wedge t(\mathbf{x}_i) \in \mathcal{Y}\}$$

- ★ Predpostavimo:  $t(\mathbf{x}_i) = f(\mathbf{x}_i)$
- ▷ **Z zveznim večplastnim perceptronom lahko realiziramo neskončno mnogo različnih funkcij  $f_w$  (odvisne od vektorja uteži!).**
  - ★ Topologija NN določa njeno **kapaciteto** (*capacity*) oz. družino funkcij, ki jih z nevronske mreže lahko realiziramo.
  - ★ Znanje skrito v utežeh nevronov. Vsaka utež je en parameter, ki ga moramo naučiti.
  - ★ Včasih: plitke NN
  - ★ Danes: globoke NN z 20 ali več plastmi; ogromno število parametrov

## Učenje nevronske mreže

- ▷ Nadzorovano učenje (*supervised learning*): Učimo z učno množico  $\mathcal{U}_M$
- ▷ Učenje je iterativni proces.
- ▷ Korak učenja (*epoch*) sestoji iz prehoda naprej in prehoda nazaj preko  $\mathcal{U}_M$  (ali "mini batch-a"):

### Prehod naprej

- ★ Po vstavljanju vhoda  $\mathbf{x}_i$ ,  $\mathbf{x}_i \in \mathcal{U}_M$ , dobimo na izhodu mreže  $\mathbf{y}_i$
- ★  $\forall \mathbf{x}_i \in \mathcal{X} \wedge i \in \mathcal{U}_M$  : Izračunamo izgubo (*loss*):

$$L_i = L(\mathbf{x}_i, \mathbf{y}_i, \mathbf{t}(\mathbf{x}_i), \mathbf{w})$$

- ★ Izračunamo strošek (*cost*):

$$J(\mathbf{w}) = E \left( \sum_i L_i \right), \quad E - \text{matematično upanje}$$

### Prehod nazaj

- ★ Določimo gradient stroškovne funkcije,  $\nabla_{\mathbf{w}} J(\mathbf{w})$ , tipično z vzvratnim razširjanjem (*back-propagation*)
- ★ Z učnim algoritmom in izračunanim gradientom  $\nabla_{\mathbf{w}} J(\mathbf{w})$  popravimo vse uteži  $\mathbf{w}$  v NN.

# **Umetni nevron in umetne nevronske mreže** (6)

## ▷ **Stroškovne funkcije**

- ★ Navzkrižna entropija (*cross-entropy*) oz. negativna log-verjetnost (*log-likelihood*), srednja kvadratna napaka (*Mean Squared Error – MSE*), srednja absolutna napaka (*Mean Absolute Error – MAE*)
- ★ MSE in MAE vračata slabe rezultate, če uporabljamo optimizacijo na osnovi gradientov.

## ▷ **Učni algoritmi**

- ★ Iščejo optimum stroškovne funkcije  $J$  v odvisnosti od uteži w nevronske mreže.
- ★ **Osnovni algoritmi** (fiksna stopnja učenja za vsak parameter): Stohastični gradientni sestop in variante, Moment (*Momentum*)
- ★ **Algoritmi z adaptivno stopnjo učenja** (za vsak parameter drugačna stopnja učenja): AdaGrad, RMSProp, Adam (*Adaptive Moments*)
- ★ **Algoritmi na osnovi približkov gradientov drugega reda**: Newtonova metoda, Konjugirani gradienti
- ★ **Izkustven (empiričen) rezultat**: Algoritom RMSProp učinkovit in praktičen optimizacijski algoritmom za globoke NN. Danes prva izbira.

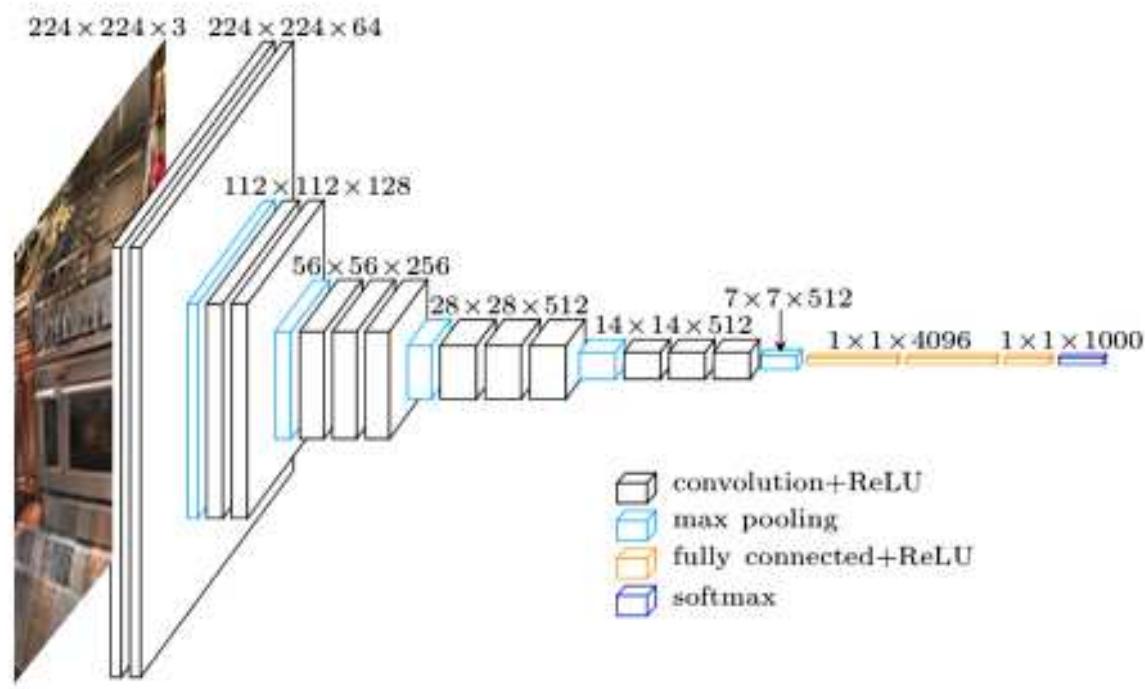
# Konvolucijske nevronske mreže

- ▷ Vrsta NN, namenjena obdelovanju podatkov z **mreži podobno (*grid-like*) topologijo:**
  - ★ Npr.: Časovne serije 1D podatkov, digitalne slike
- ▷ Konvolucijska nevronska mreža (*convolutional NN, CNN*) vedno vključuje vsaj eno **operacijo konvolucije**.
  - ★ **Ponovitev:** filtriranje signalov in slik, filter oz. jedro oz. maska v 2D, implementacija z obračanjem jedra ali s križno korelacijo
- ▷ Dve različni terminologiji za opis strukture CNN:
  1. Mreža sestoji iz nekaj relativno kompleksnih konvolucijskih plasti, vsaka plast pa ima več **faz oz. stopenj (*stage*)**.
    - ★ Konvolucijska plast tipično iz: konvolucijske faze, sledi faza detekcije oz. aktivacije, na koncu je združevalna faza, nato naslednja konvolucijska plast.
  2. Mreža sestoji iz množice preprostih plasti; vsak obdelovalni korak se torej obravnava kot lastna plast.
    - ★ Tipično sosledje: konvolucijska plast, plast detekcije oz. aktivacije, združevalna plast, naslednja plast.
- ▷ Mreža CNN je običajno globoka NN.

# Konvolucijske nevronske mreže

(2)

**Primer:** VGG mreža s 16 plastmi (vir: [2])



# Konvolucijske nevronske mreže

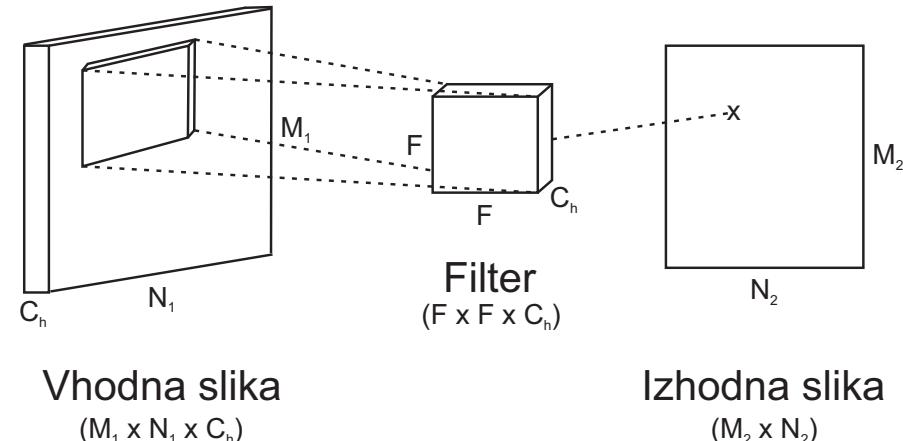
(3)

## Motivacija za konvolucijske nevronske mreže

- ▷ **Problem:** klasične NN zelo slabo skalirajo na "polno" digitalno sliko
  - \* Polno povezan nevron v vstopni plasti ima  $M \times N \times \text{Št\_Kanalov}$  uteži oz. parametrov (+ pragi)
  - \* Eksplozija števila parametrov pri večjem številu nevronov.
  - \* Ogromno število parametrov hitro vodi v nadprileganje (*overfitting*)!
- ▷ CNN eksplizitno predpostavljajo, da so vhodi digitalne slike.
- ▷ Objekti tipično iz preprostih lokalnih značilnic (npr. robov), ki jih grupiramo v kompleksnejše (npr. obrv).
  - \* So na različnih delih slike, četudi sestojijo iz enakih preprostih značilnic.
  - \* **Posledica – Tri pomembne ideje v CNN:**
    1. Redke interakcije oz. redka povezanost (*sparse interactions*) – izhodna enota ni povezana z vsako vhodno enoto plasti (majhna jedra!)
    2. **Deljenje parametrov** – uporaba istega parametra (uteži) za več kot eno funkcijo v modelu (učimo en set parametrov za vse lokacije v sliki!)
    3. "**Ekvivariantna**" predstavitev oz. ekvivariantnost plasti na translacijo – ekvivariantnost: če se vhod spremeni, se izhod spremeni na enak način
      - ▷ Konvolucija ni ekvivariantna na rotacijo in spremembo v skali!

## A. Konvolucijska plast

- ▷ Filtrira vhodno sliko s filtrom.
- ▷ Tipično uporaba **križne korelacije** (skalarni produkt podslike in nezrcaljenega jedra).
- ▷ Jedro premikamo po vrsticah in stolpcih slike s **korakom  $S$**  (*stride*).
- ▷ Zaradi ohranjanja dimenzijs lahko dodamo k sliki **rob  $P$ -tih ničel** (*zero-padding*).
- ▷ **Rezultat:** izhodna slika dimenzijs  $M_2 \times N_2$ , kjer



Vhodna slika  
( $M_1 \times N_1 \times C_h$ )

Izhodna slika  
( $M_2 \times N_2$ )

$$M_2 = (M_1 - F + 2P)/S + 1, \quad N_2 = (N_1 - F + 2P)/S + 1$$

- ▷ **V praksi:** Na sliko apliciramo  $K$  različnih filtrov, zato izhod konvolucijskega sloja dimenzijs

$$M_2 \times N_2 \times K$$

## Konvolucijske nevronske mreže (5)

- ▷ Uteži filtrov niso poznane vnaprej.
- ▷ Določimo jih v fazi učenja CNN.
- ▷ Vsak filter vpeljuje (ideja deljenja parametrov!)

$F \times F \times C_h$  uteži oz. parametrov

- ▷ Skupno potrebujemo za  $K$  filtrov

$(F \times F \times C_h) \times K$  uteži in  $K$  pragov

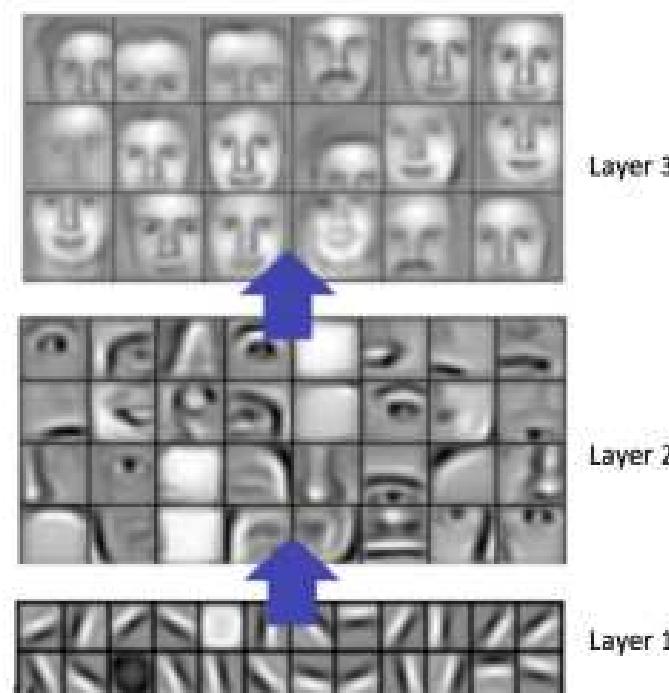
- ▷ Parametri  $F, S, P$  in  $K$  so **hiperparametri** (*hyperparameters*) CNN.
- ▷ **Tipične nastavitev:**
  - ★  $F = 3, S = 1$  in  $P = 1$  – ohranimo dimenzijo vhodne slike
  - ★ Če  $P = 0$  in/ali  $S > 1$  – zmanjšamo dimenzijo vhodne slike
  - ★ **Hiperparametre moramo izbrati tako, da sta  $M_2$  in  $N_2$  celi števili!**
  - ★ Nastavitev parametrov seveda odvisna od problema, ki ga rešujemo!

# Konvolucijske nevronske mreže

(6)

- ▷ V splošnem: Več kot imamo konvolucijskih plasti, bolj kompleksne značilnice se bo zmožna naučiti mreža.
  - \* nizko nivojske značilnice (*low-level features*) – začetne konvolucijske plasti
  - \* srednje nivojske značilnice (*mid-level features*) – vmesne konvolucijske plasti
  - \* visoko nivojske značilnice (*high-level features*) – zaključne konvolucijske plasti

**Primer:** Naučene značilnice s pomočjo konvolucijske mreže globokega zaupanja (*convolutional deep belief network*) (vir: [3])



# Konvolucijske nevronske mreže (7)

## B. Plast detekcije oz. aktivacijska plast

- ▷ Aplicira na vhod eno izmed aktivacijskih funkcij (npr. ReLU).
- ▷ Aktivacijo izvajamo na nivoju posameznega elementa.

## C. Polno povezana plast (*fully-connected layer*)

- ▷ Vsak nevron v tej plasti je povezan z vsemi vhodi (utež za vsak vhod + prag za nevron).
- ▷ Popolnoma identično kot pri zveznih perceptronih.
- ▷ Z ustrezeno izbiro hiperparametrov  $F$ ,  $P$ ,  $S$  in  $K$  lahko vsako polno povezano plast transformiramo v konvolucijsko plast!
  - ★ Pri reševanju določenih problemov lahko s tem pohitrimo izvajanje.

## D. Združevalna plast (*pooling layer*)

- ▷ Tipično jo vstavljamo med zaporedne konvolucijske plasti.
- ▷ Reducira prostorsko dimenzijo (podvzorčenje!), količino parametrov in računanj v mreži, s tem kontroliramo nadprileganje.
  - ★ Če pretiravamo s podvzorčenjem, povzročimo podprileganje (*underfitting*).
- ▷ Operira neodvisno na vsakem kanalu vhoda (dimenzije  $M_1 \times N_1 \times C_h$ ).

# Konvolucijske nevronske mreže

(8)

▷ **Postopek za en kanal:**

- ★ Izberemo velikost filtra  $F \times F$  in korak  $S$  za vhodno sliko (ničelne obrobe ne dodajamo!).
- ★ Nad okolico  $F \times F$  izračunamo vnaprej izbrano funkcijo.
- ★ Izhod je dimenzijs  $M_2 \times N_2$  (za kanal) oz.  $M_2 \times N_2 \times C_h$  (za sliko):

$$M_2 = (M_1 - F)/S + 1, \quad \text{in} \quad N_2 = (N_1 - F)/S + 1.$$

▷ **Združevalne funkcije:** maksimum v okolici (tipično); povprečje;  $L^2$ -norma; obteženo povprečje, glede na oddaljenost od središča

▷ **Nastavitev hiperparametrov v praksi:**

- ★  $F = 2$  in  $S = 2$  (najpogosteje)
  - ★  $F = 3$  in  $S = 2$  (prekrivajoče združevanje)
- ▷ **Združevanje po posameznem kanalu producira invariantnost na translacijo (majhno, znotraj okolice  $F \times F$ ).**
- ★ S pomočjo združevanja preko vseh kanalov se lahko značilnice priučijo invariantnosti tudi na druge funkcije (npr. rotacijo).

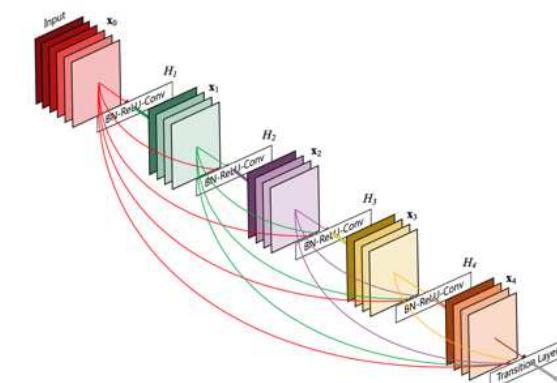
# Konvolucijske nevronske mreže (9)

Primeri uveljavljenih arhitektur iz področja CNN [4, 9]:

- ▷ **Klasične arhitekture:**
  - ★ LeNet (1998)
  - ★ AlexNet (2012) – izboljšava LeNet, ima 60M parametrov!
  - ★ ZFNet (2013) – izboljšava AlexNet
  - ★ VGGNet (2014) – preprosta in homogena struktura, a 140M parametrov
- ▷ **Moderne arhitekture:**
  - ★ GoogLeNet (2014) – "inception" moduli, manj parametrov kot AlexNet
  - ★ ResNet (2015) – blok preostanka (*residual block*), lahko implementira funkcijo identitete
  - ★ DenseNet (2016) – na vhod trenutne plasti so povezani izhodi vseh predhodnih plasti



Arhitektura ResNet (vir: [9])



Arhitektura DenseNet (vir: [9])

# Konvolucijske nevronske mreže

(10)

**Primer:** Plasti mreže VGG in število prostih parametrov (vir: [4, 8])

```
INPUT: [224x224x3]      memory: 224*224*3=150K  weights: 0
CONV3-64: [224x224x64]  memory: 224*224*64=3.2M  weights: (3*3*3)*64 = 1,728
CONV3-64: [224x224x64]  memory: 224*224*64=3.2M  weights: (3*3*64)*64 = 36,864
POOL2: [112x112x64]    memory: 112*112*64=800K  weights: 0
CONV3-128: [112x112x128] memory: 112*112*128=1.6M  weights: (3*3*64)*128 = 73,728
CONV3-128: [112x112x128] memory: 112*112*128=1.6M  weights: (3*3*128)*128 = 147,456
POOL2: [56x56x128]     memory: 56*56*128=400K  weights: 0
CONV3-256: [56x56x256]   memory: 56*56*256=800K  weights: (3*3*128)*256 = 294,912
CONV3-256: [56x56x256]   memory: 56*56*256=800K  weights: (3*3*256)*256 = 589,824
CONV3-256: [56x56x256]   memory: 56*56*256=800K  weights: (3*3*256)*256 = 589,824
POOL2: [28x28x256]     memory: 28*28*256=200K  weights: 0
CONV3-512: [28x28x512]   memory: 28*28*512=400K  weights: (3*3*256)*512 = 1,179,648
CONV3-512: [28x28x512]   memory: 28*28*512=400K  weights: (3*3*512)*512 = 2,359,296
CONV3-512: [28x28x512]   memory: 28*28*512=400K  weights: (3*3*512)*512 = 2,359,296
POOL2: [14x14x512]     memory: 14*14*512=100K  weights: 0
CONV3-512: [14x14x512]   memory: 14*14*512=100K  weights: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512]   memory: 14*14*512=100K  weights: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512]   memory: 14*14*512=100K  weights: (3*3*512)*512 = 2,359,296
POOL2: [7x7x512]        memory: 7*7*512=25K  weights: 0
FC: [1x1x4096]          memory: 4096  weights: 7*7*512*4096 = 102,760,448
FC: [1x1x4096]          memory: 4096  weights: 4096*4096 = 16,777,216
FC: [1x1x1000]          memory: 1000  weights: 4096*1000 = 4,096,000

TOTAL memory: 24M * 4 bytes ~= 93MB / image (only forward! ~*2 for bwd)
TOTAL params: 138M parameters
```

# Inženirski pristop h globokim nevronskim mrežam

## A. Praktični razvojni proces pri delu z globokimi NN [1]:

1. Določiti cilje, metrike napake in ciljne vrednosti teh metrik
  - ▷ Metrike: izguba (ni popolnoma merodajna), točnost (*accuracy*), natančnost (*precision*), priklic (*recall*), krivulja PR, "F-score", pokritost (*coverage*)...
2. Čim prej vzpostaviti delajočo rešitev (*end-to-end pipeline*) in oceniti metrike zmogljivosti
3. Dobro preučiti sistem in določiti ozka grla v zmogljivosti
  - ▷ Npr. diagnosticirati, katere enote deluje slabše kot pričakovano; slabša zmogljivost zaradi pod- ali nadprileganja...
4. Ponavlajoče vnašati inkrementalne spremembe v sistem na osnovi izsledkov diagnosticiranja
  - ▷ Npr. zbirati nove učne podatke, uglasiti hiperparametre, spremeniti algoritme...

## B. Predobdelava podatkov

- ▷ Tipično: ničelno povprečje, normalizacija (deljenje s standardnim odklonom)
- ▷ Redkeje (ne uporabljamo pri CNN!): PCA, redukcija s PCA (obdržimo značilnice z največ variance), beljenje
- ▷ **Pogosta napaka:** Statistike (npr. povprečje) izračunamo na celotni množici podatkov, uporabimo na vseh podmnožicah.
- ▷ **Pravilno:** Statistike izračunamo le na učni množici, uporabimo na vseh podmnožicah!

# Inženirski pristop k globokim nevronskim mrežam (2)

## C. Inicializacija uteži

- ▷ Pogosta napaka: Uteži postavimo na 0.
  - \* Posledica: Ni asimetrije med nevroni, vsi identično spreminjajo uteži med učenjem.
- ▷ Uteži postaviti na majhna naključna števila (npr. Gaussova porazdelitev  $N(0, 1)$ )
  - \* Posledica: majhni gradienti med učenjem, majhna sprememba uteži (ni dobro!).
- ▷ Redka inicializacija
  - \* Uteži vseh nevronov najprej na 0. Nato nevron naključno povežemo z nekaj nevroni na naslednji plasti (do 10). Uteži določimo po  $N(0, 1)$ .
- ▷ Tipično: Kalibrirati varianco z  $1/\sqrt{n}$ 
  - \* Vektor uteži nevrona po naključni inicializaciji množimo z  $1/\sqrt{n}$ , kjer  $n$  število vhodov v nevron.
  - \* **Nevroni z ReLU aktivacijo:** množimo z  $\sqrt{2/n}$

## D. Inicializacija pragov

- ▷ Tipično: Prage postavimo na 0

# Inženirski pristop h globokim nevronskim mrežam (3)

## E. Normalizacija množice (*batch normalization*)

- ▷ V globoke NN se uvaja kot posebna plast pred aktivacijsko plastjo (npr. pred ReLU).
- ▷ Faza učenja: Vsako značilnico normaliziramo kot

$$y_i = \gamma \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta$$

- ★ parametra  $\gamma$  in  $\beta$  učimo
- ★  $\mu_B$  in  $\sigma_B$  povprečje in standardni odklon množice  $B$ ,  $\epsilon$  majhno število  $>0$
- ▷ Faza "inference": Namesto plasti uporabimo zgornjo linearno enačbo, kjer  $\gamma$  in  $\beta$  fiksna,  $\mu$  in  $\sigma$  določimo iz tekočega povprečenja preko vseh množic  $B$ .
- ▷ **V praksi:** NN z normalizacijo množice so bolj robustne na slabo inicializacijo!

## F. Regularizacija

Kontrolira kapaciteto NN in preprečuje nadprileganje.

- ▷  $L^2$  regularizacija (Tikhonova regularizacija, razkroj uteži) – vse uteži imajo majhne vrednosti blizu 0
- ▷  $L^1$  regularizacija – vektor uteži redki in blizu 0, določene uteži postanejo 0

## Inženirski pristop k globokim nevronskim mrežam (4)

- ▷ "Dropout"
  - ★ Zgolj v fazi učenja!
  - ★ Med učenjem obdržimo nevron aktiven z verjetnostjo  $p$  (tipično  $p = 0,5$ )
  - ★ Kombiniramo z drugimi regularizacijskimi tehnikami.
- ▷ Umetno povečanje podatkovne množice (*dataset augmentation*)
  - ★ Ideja: Model strojnega učenja generalizira bolje, če ga učimo na več podatkih.
  - ★ Postopek: S pomočjo različnih tehnik in transformacij generiramo nove lažne podatke in jih dodamo v učno množico.
  - ★ Transformacije: Rotacija, skaliranje, sprememba svetlosti, zamegljenje...
- ▷ Zgodnja prekinitev (*early stopping*)
  - ★ Med učenjem opazujemo napako na validacijski množici.
  - ★ Če se napaka po  $p$  korakih ne zmanjša, prekinemo učenje. Vrnemo parametre (in število učnih korakov), s katerimi dobili najmanjšo napako.
  - ★ Potrebujemo učno in validacijsko množico! Slabost: ne izrabimo vseh podatkov za učenje!
  - ★ Če po zgodnji prekinitvi uporabimo vse podatke (učna + validacijska množica):
    1. učimo od začetka do optimalnega števila korakov
    2. nadaljujemo učenje z optimalnimi parametri

# Inženirski pristop h globokim nevronskim mrežam (5)

## G. Izbera hiperparametrov

### Ročno uglaševanje hiperparametrov

- ▷ Cilj: poiskati takšne hiperparametre, da dobimo najmanjšo napako generalizacije glede na sprejemljiv "strošek" računanja in pomnilnika.

Hiperparameter	Kapaciteta zviša, če	Opombe
Št. skritih enot	Povečamo	Povečamo kapaciteto, a tudi strošek vsake operacije v modelu
Stopnja učenja	Uglašena optimalno	Nepravilna nastavitev pomeni neuspeh optimizacije
Velikost konv. jeder	Povečamo	Več parametrov, a izhod manjše dimenzije (se lahko zmanjša tudi kapaciteta)
Ničelna obroba	Povečamo	Ohranja velikost reprezentacije modela
Koefic. razkroja uteži ( $L^2$ )	Zmanjšamo	Sprosti modelne parametre, da postanejo večji
Stopnja "dropout"	Zmanjšamo	Manj pogosto izločanje enote, večja možnost prileganja učni množici

# Inženirski pristop h globokim nevronskim mrežam (6)

## Avtomatsko izbiranje hiperparametrov

- ▷ Avtomatski algoritmi za optimizacijo hiperparametrov
  - \* Npr. evolucijski in drugi algoritmi po vzoru iz narave
- ▷ Iskanje po mreži (*grid search*)
  - \* Diskretiziramo hiperparametre, izvedemo učenje z eno kombinacijo hiperparametrov, ocenimo napako na validacijski množici. "Optimalni" nabor, kjer napaka najmanjša.
  - \* Smiselno: grobo-fini pristop (v okolici "optimuma" bolj fino razdelimo območje).
- ▷ Naključno iskanje (*random search*)
  - \* Hiperparameter ne diskretiziramo, ampak ga izbiramo po določeni porazdelitvi.
  - \* Smiselno: grobo-fini pristop
  - \* V splošnem učinkovitejši od iskanja po mreži.

# Inženirski pristop h globokim nevronskim mrežam (7)

## H. Privzeti izhodiščni modeli (*default baseline models*)

- ▷ Ideja: Čim prej vzpostaviti delujajočo rešitev (*end-to-end pipeline*) in oceniti metrike zmogljivosti (točka A. Praktični razvojni proces...).
  - ★ Metrike izhodiščnega modela služijo kot "benchmark" (jih želimo izboljšati!).
- ▷ Pристоп izberemo glede na kompleksnost reševanega problema:
  - ★ "AI-complete": Globoko učenje
    - ▷ Npr. prepoznavanje objektov, prepoznavanje govora, strojno prevajanje...
  - ★ Sicer: Preprostejše metode (npr. regresija)

### Nekaj predlogov:

- ▷ Zvezni perceptroni – če vhod vektorji fiksne velikosti
- ▷ Konvolucijske NN – če vhod z znano topološko strukturo (npr. slike)
- ▷ Aktivacijska funkcija – ReLU ali njene pospološitve (npr. Leaky ReLU)
- ▷ Optimizacija – stohastični gradientni sestop (moment + padajoča stopnja učenja)
- ▷ Blaga regularizacija – Zgodnja prekinitve (vedno), "dropout" (smiselno), normalizacija množice (ne v začetne izhodiščne modele)
- ▷ Nadzorovano učenje (za večino problemov)
- ▷ Če že rešitev za soroden problem, uporabimo najboljši model in algoritme od tam

# Inženirski pristop h globokim nevronskim mrežam (8)

## I. Spremljanje uspešnosti učenja, razhroščevanje in drugi napotki

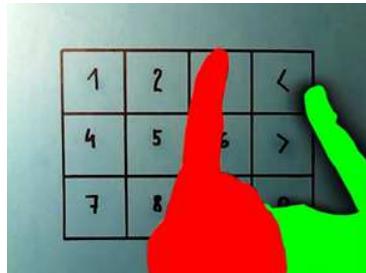
Preveriti oz. opazovati:

- ▷ Gradiente v mreži (uporablja se redkeje)
- ▷ Pričakovano izgubo pri naključnem delovanju (brez regularizacije; predpostavimo, da vse deluje napačno)
- ▷ Nadprilegamo zelo majhno množico podatkov (brez regularizacije) – strošek mora biti 0
- ▷ Izgubo skozi čas (epohe) v odvisnosti od stopnje učenja – mora počasi padati
  - \* Spremenimo stopnjo učenja, velikost "mini-batch"
- ▷ Natančnost na učni in validacijski množici skozi čas (epohe) – razlika med natančnostima kaže velikost nadprileganja
  - \* Velika razlika: večja regularizacija, povečamo množico podatkov
  - \* Majhna razlika: povečamo kapaciteto modela
- ▷ Razmerje norma spremembe uteži proti normi uteži – želimo okrog  $10^{-3}$ 
  - \* Sicer sprememba stopnje učenja
- ▷ Porazdelitev aktivacij / gradientov za posamezno plast – čudne porazdelitve kažejo problem napačne inicializacije
- ▷ Vizualizacijo uteži prve plasti CNN – uteži morajo biti lepe, gladke, čiste, različne
- ▷ Vizualizacijo modela v akciji – rezultati smiselnii?
  - \* Vizualizirati tudi rezultate, kjer se model najbolj moti

# Aplikacije

## Aplikacije konvolucijskih nevronskih mrež

### 1. Segmentacija slik

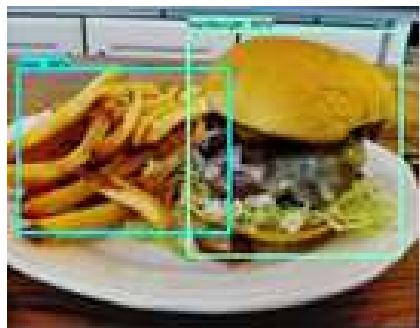


Vir: [10]

### 2. Preiskovanje slikovnih podatkovnih zbirk (*image retrieval*)

### 3. Razvrščanje slik

### 4. Razvrščanje in detekcija oz. lokalizacija



Vir: [11]



Sedenje z verjetnostjo 74 % (vir: [12])

### 5. Luščenje značilnic

## **Dodatna literatura**

- [1] I. Goodfellow, Y. Bengio, A. Courville (2016), Deep learning, The MIT Press, London, VB.
- [2] Abto Software (2018), Kitchen Furniture & Appliances Recognition, <https://www.abtosoftware.com/blog/kitchen-furniture-appliances-recognition-in-photos>
- [3] U. Karn (2016), An Intuitive Explanation of Convolutional Neural Networks, <https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/>
- [4] A. Karpathy, J. Johnson (2018), CS231n Convolutional neural networks for visual recognition, <http://cs231n.github.io/>
- [5] Y. Zhu (2015), All you want to know about CNNs..., <http://www.cs.utoronto.ca/~fidler/teaching/2015/slides/CSC2523/CNN-tutorial.pdf>
- [6] F.-F. Li, J. Johnson, S. Yeung (2018), Lecture 5: Convolutional neural networks, [http://cs231n.stanford.edu/slides/2018/cs231n\\_2018\\_lecture05.pdf](http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf)
- [7] A. Deshpande (2016), The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3), <https://adeshpande3.github.io/adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html>
- [8] Visual geometry group (2014), Very Deep Convolutional Networks for Large-Scale Visual Recognition, [http://www.robots.ox.ac.uk/~vgg/research/very\\_deep/](http://www.robots.ox.ac.uk/~vgg/research/very_deep/)

## **Dodatna literatura**

**(2)**

- [9] J. Jordan (2018), Common architectures in convolutional neural networks,  
<https://www.jeremyjordan.me/convnet-architectures/>
- [10] A. Javornik (2017), Avtomatsko nastavljanje parametrov segmentacijske metode aplikacije virtualna tipkovnica s pomočjo nevronske mreže, diplomsko delo, UM-FERI, Maribor.
- [11] J. Banko (2018), Prepoznavanje jedi iz digitalnih slik s pomočjo konvolucijskih nevronskih mrež, diplomsko delo, UM-FERI, Maribor.
- [12] M. Baketarić (2018), Prepoznavanje aktivnosti osebe iz zaporedja slik s pomočjo konvolucijskih nevronskih mrež, diplomsko delo, UM-FERI, Maribor.

## ***2. Napredni koncepti zajemanja slik***

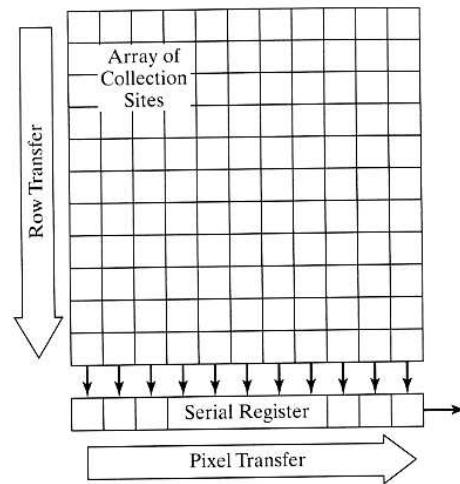
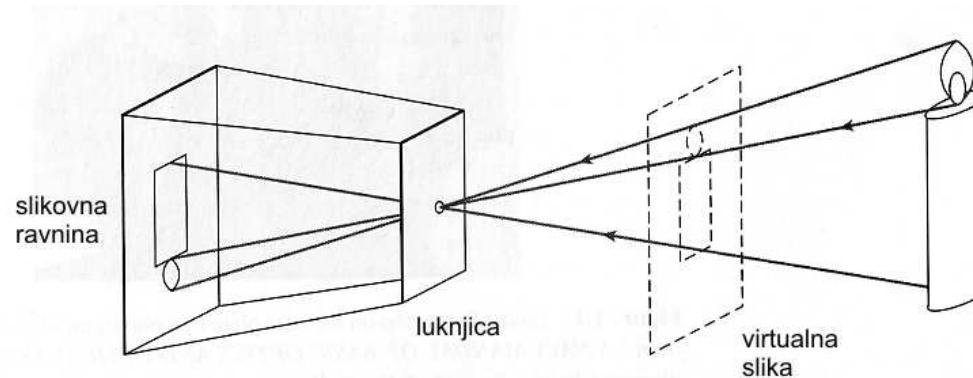
---

Hiperspektralno zajemanje

Polarizirano zajemanje

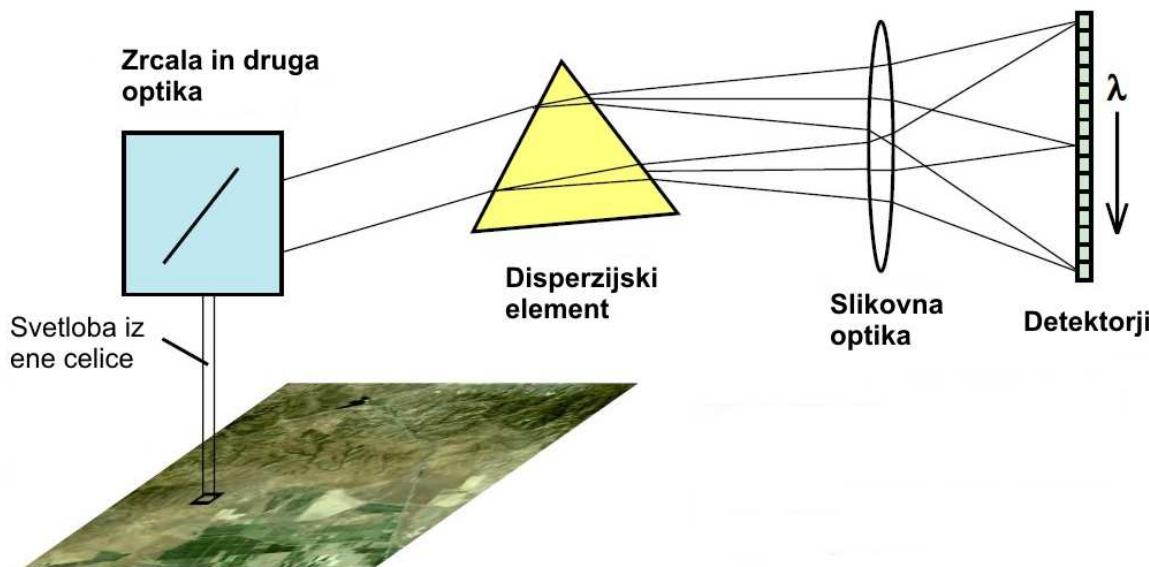
Interferometrija

# Zajemanje slik s kamerami CCD



# Hiperspektralno zajemanje

- ▷ Hiperspektralne slike (*Hyperspectral images*) lahko dobimo s pomočjo kompleksnih senzorjev slikovnih spektrometrov (*imaging spectrometers*)
- ▷ Razvoj teh senzorjev omogočila združitev dveh tehnologij:
  - \* **spektroskopija (spectroscopy)**: študij od materialov (objektov) oddane oz. odbite svetlobe in spremeljanje sprememb v energiji v odvisnosti od valovne dolžine.
  - \* **daljinsko slikanje (remote imaging)** Zemlje in drugih planetov: takšni senzorji zmožni fokusirati in izmeriti odbito svetlobo iz mnogih, blizu ležečih področij na Zemljini površini.

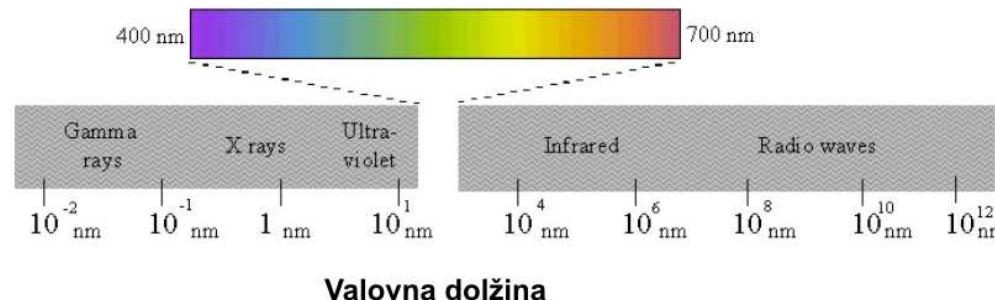


Shematski diagram osnovnih elementov slikovnega spektrometra (vir: [1]).

# Hiperspektralno zajemanje

(2)

- ▷ Elektromagnetno sevanje (npr. svetlobo) opredelimo z valovno dolžino (vir: [2]).



## Razlika med multispektralnim in hiperspektralnim zajemanjem

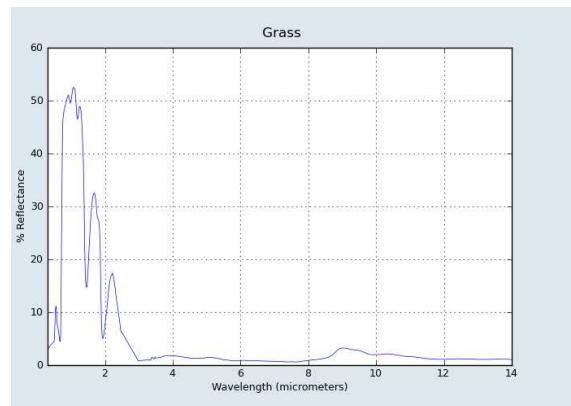
- ▷ Multispektralno zajemanje:
  - ★ merjenje odbitega sevanja s površin zgolj pri nekaj razmiknjenih valovnih dolžinah, pri čemer je opazovan pas valovnih dolžin širok.
- ▷ Hiperspektralno zajemanje:
  - ★ merjenje odbitega sevanja pri množici sosednjih valovnih dolžin, pri čemer je opazovan pas valovnih dolžin ozek.
- ▷ Ni zgolj število pasov valovnih dolžin ključno za hiperspektralnost, ampak predvsem ozko opazovano območje valovnih dolžin (majhen razmik med sosednjimi pasovi)!
- ▷ **Primer:**
  - ★ hiperspektralno – 20 valovnih dolžin, ki so 10 nm narazen
  - ★ multispektralno – 20 valovnih dolžin, ki so 100 nm narazen

# Hiperspektralno zajemanje

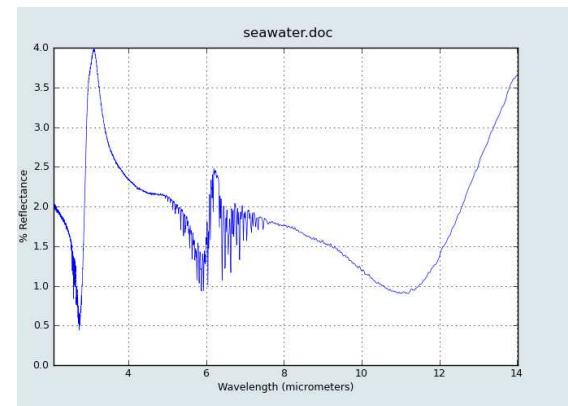
(3)

## Kaj pridobimo s takšnim zajemanjem?

- ▷ Spekter koeficenta odbojnosti (*reflectance*): prikazuje koeficient odbojnosti materiala, izmerjen v določenem pasu valovnih dolžin (analogija velja pri koeficientu absorbcije)
  - \* **Koeficient odbojnosti:** odstotek svetlobe, ki je dosegla material in se od njega odbila
- ▷ Spekter koeficenta odbojnosti in absorbcije, opazovan preko množice valovnih dolžin, lahko enolično identificira opazovane materiale.
- ▷ **Primer:**



Trava



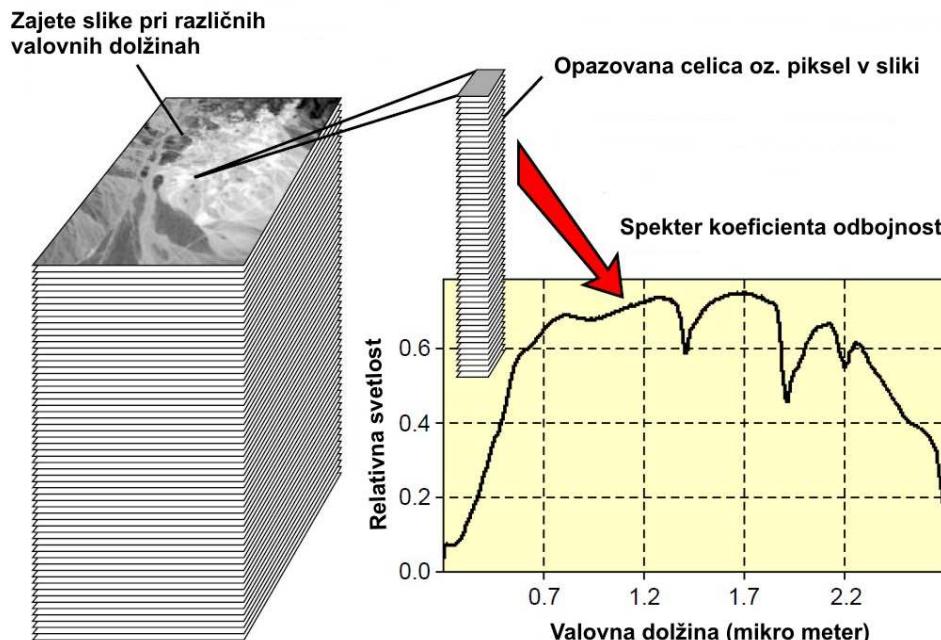
Morje



Borovec

## Hiperspektralno zajemanje (4)

- ▷ Slikovni spektrometer omogoča, da dobimo zvezni spekter koeficienta odbojnosti za vsako slikovno točko.



Vir: [1]

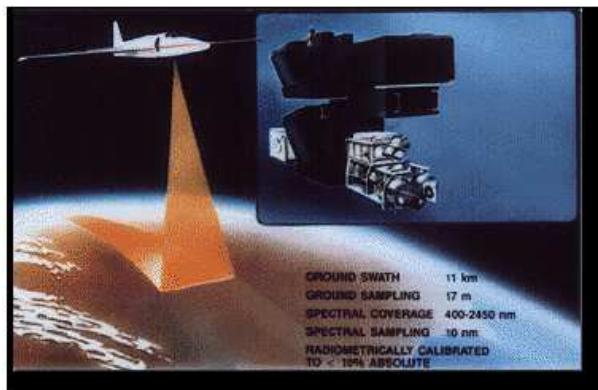
- ▷ Obstajajo knjižnice s spektri koeficientov odbojnosti za naravne in umetno narejene materiale:
  - ★ Knjižnica ASTER (<http://speclib.jpl.nasa.gov>): vsebuje čez 2400 spektrov, pri čemer je pokrit pas valovnih dolžin med 0,4 in 14  $\mu\text{m}$
  - ★ Knjižnica USGS (<http://speclab.cr.usgs.gov/spectral-lib.html>): vsebuje čez 1300 spektrov, pri čemer je pokrit pas valovnih dolžin med 0,2 in 3  $\mu\text{m}$

# Hiperspektralno zajemanje

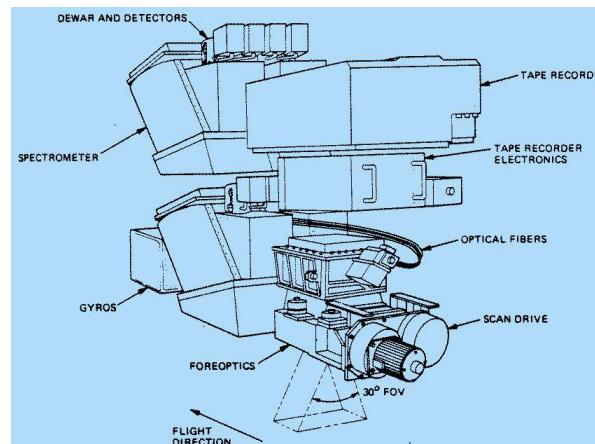
(5)

- ▷ Slikovni hiperspektralni spektrometri lahko nameščeni na:
  - ★ letečo napravo (*airborne sensors*): npr. AVIRIS, AISA, DAIS 2115, ...
  - ★ na satelit (*space-based sensors*): npr. PRISM, PROBA, NEMO, ...

**Primer:** AVIRIS (Airborne Visible Infrared Imaging Spectrometer),  
<http://aviris.jpl.nasa.gov>



Zajem



Inštrumenti

Detajli:

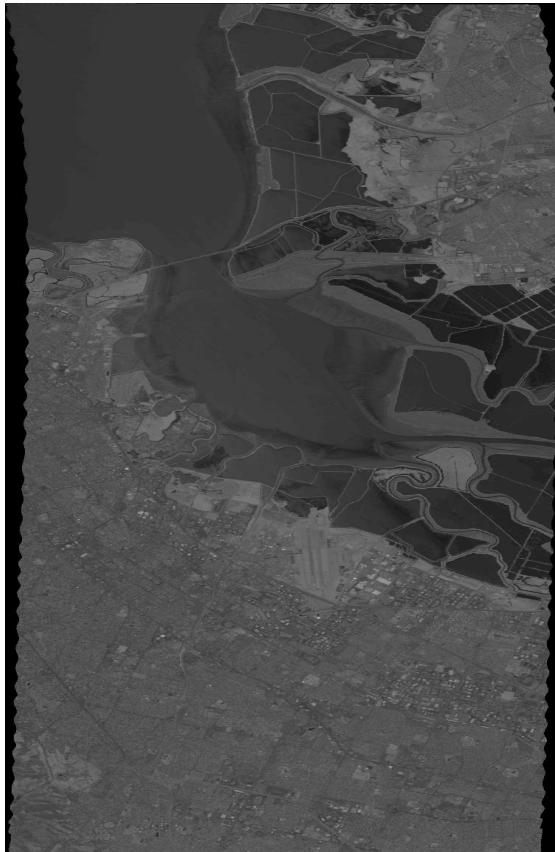
- ▷ Višina zajema: 20 km nad Zemljo (ER-2 jet)
- ▷ Število pasov oz. valovnih dolžin: 224 pasov, razmik med pasovi 10 nm
- ▷ Spektralni obseg: 0,4 do 2,5  $\mu\text{m}$
- ▷ Snemalni pas (*ground swath*): 11 km
- ▷ Ločljivost: 1 piksel ustreza kvadratu  $20 \times 20$  m na Zemlji

# Hiperspektralno zajemanje

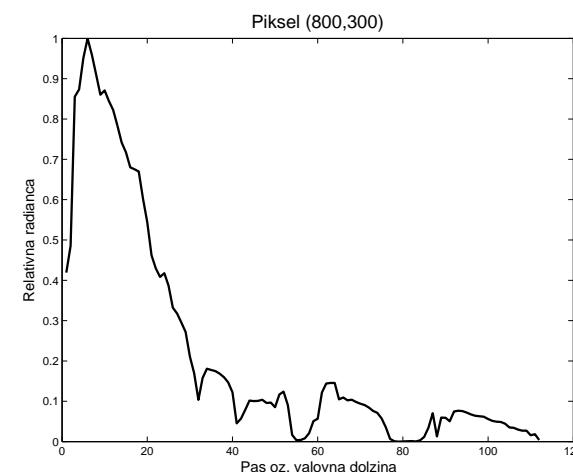
(6)

Letališče Moffett (USA):

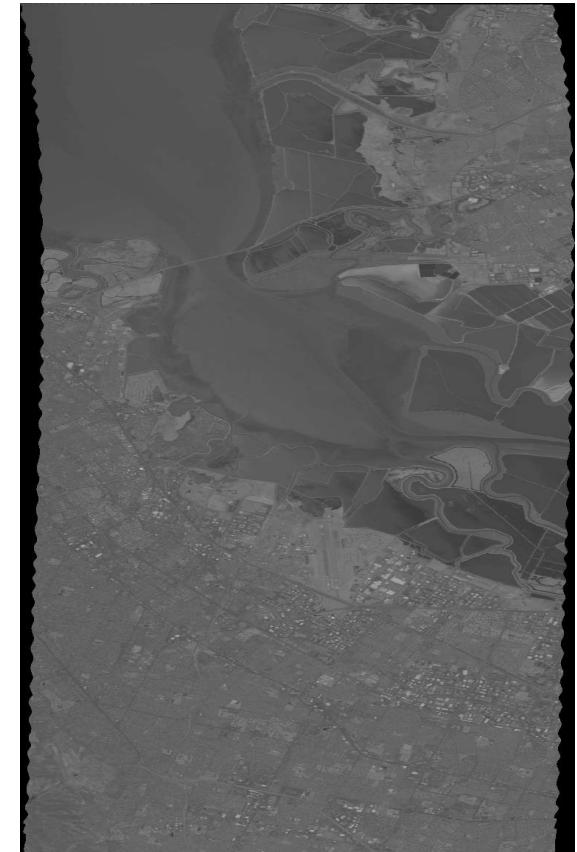
- ▷ Slika velikosti:  $1924 \times 753$  pikslov  $\times 224$  pasov ( $\approx 618$  MB)



Pas 140



Profil za piksel (800,300)



Povprečje skozi vse pasove

## Analiza hiperspektralnih slik

- ▷ Tipično: primerjamo spekter izbranega piksla z referenčnim spektrom (t.i. ciljni spekter)
  - \* ciljni (*target*) spekter lahko dobimo iz knjižnic s spektri odbojnih koeficientov
- ▷ Metode analiziranja:
  - \* Metode celega piksla
  - \* Podpikselne metode (*sub-pixel methods*)

### A. Metode celega piksla

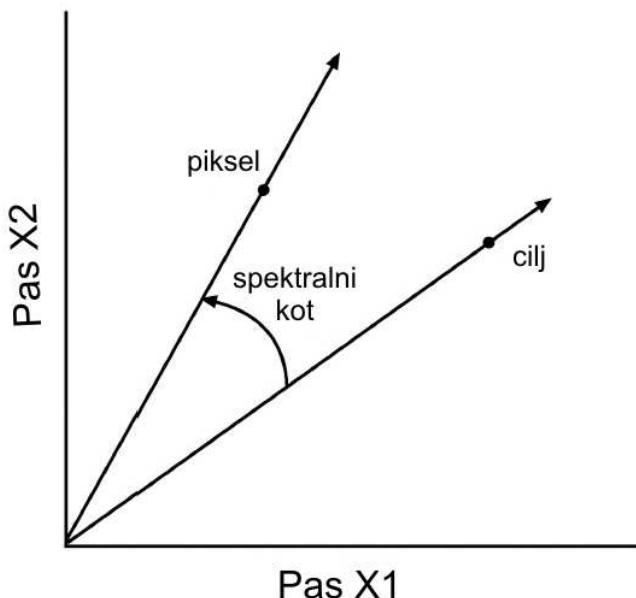
- ▷ Skušajo določiti, če je množica ciljnih materialov prisotnih v izbranem pikslu
  - \* izračun na osnovi spektralne podobnosti med pikslom in ciljnim spektrom (tri)
  - \* metode: najmanjša razdalja, največja verjetnost, preslikava spektralnega kota, prileganje spektralnih značilk...

# Hiperspektralno zajemanje

(8)

## Preslikava spektralnega kota (*spectral angle mapper*)

- ▷ Spekter piksla je točka v  $n$ -dimenzionalnem prostoru (vsaka valovna dolžina ena dimenzija)
- ▷ **Primer:**  $n = 2$



- ▷ Spektralni kot je relativno neodvisen na spremembe v svetlosti
  - ★ to vpliva le na velikost vektorja, ne pa na smer!
- ▷ Treba je izračunati spektralni kot med vsakim pikslom in vsakim ciljem!

# Hiperspektralno zajemanje (9)

## B. Podpikseline metode

- ▷ Gre za izračun deleža prisotnosti ciljnih materialov v vsakem pikslu slike (en piksel lahko vsebuje več različnih materialov).
- ▷ S takšno analizo lahko detektiramo množino ciljev, ki so veliko manjši kot pa je velikost piksla.

### Popolno linearno spektralno razmešanje (*complete linear spectral unmixing*)

- ▷ V sceni poznamo množico spektralno edinstvenih (čistih) materialov (t.i. spektralni končni člani (*endmembers*) za to sceno).
- ▷ Linearno spektralno razmešanje temelji na ideji:
  - ★ utež vsakega končnega člana je direktno proporcionalna ploščini znotraj piksla, ki ga le-ta pokriva.
- ▷ Postopek razmešanja:
  1. Spektri vseh končnih članov v sceni morajo biti znani.
  2. Reši sistem  $n$  linearnih enačb za vsak piksel ( $n$  - število končnih članov,  $e_i$  -  $i$ -ti končni član):

$$\varsigma(\mathbf{p}) = \alpha_1 \varsigma(e_1) + \alpha_2 \varsigma(e_2) + \dots + \alpha_n \varsigma(e_n)$$

- ▷ Za enolično rešitev potrebujemo vsaj  $n$  enačb, kar ni problematično pri hiperspektralnih slikah.

# Hiperspektralno zajemanje

(10)

▷ Rezultat:

- ★ dobimo sliko vsebnosti (*abundance image*) za vsakega končnega člana
- ★ iz te slike odčitamo, kolikšen delež končnega člana je vsebovan v izbranem pikslu

**Primer:** Slike vsebnosti (vir: [1])



Delež vegetacije



Delež vode



Delež tal

## Delno razmešanje

- ▷ detektira prisotnost in količino vsebnosti enega samega ciljnega materiala

# Hiperspektralno zajemanje

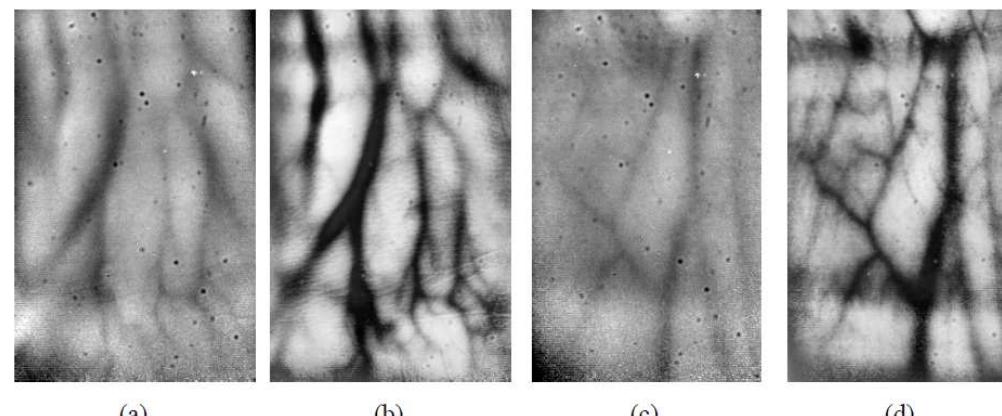
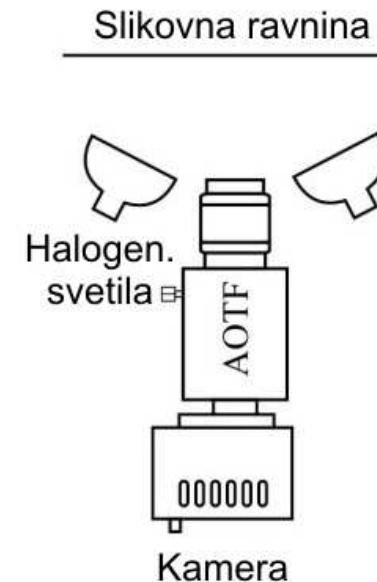
(11)

Hiperspektralne slike ne najdemo samo pri daljinskem snemanju!

**Primer:** Izboljšanje kontrasta slik podkožnih ven [3]

- ▷ Uporabljeni dve prikazani enoti:
  1. Kamera CCD Sony XC55N – pokriva spekter med 550 in 1000 nm
  2. Kamera InGaAs, občutljiva na bližnjo IR svetlubo – pokriva spekter med 900 in 1700 nm
    - ★ razmik med pasovi 10 nm
    - ★ frekvenca zajema: 100 fps
    - ★ 115 slik ob zajemu: 34 iz prve enote in 81 iz druge enote

AOTF – acousto-optical tunable filters



a) originalna slika – zapestje, b) izboljšana slika, c) originalna slika – komolec in d) izboljšana slika

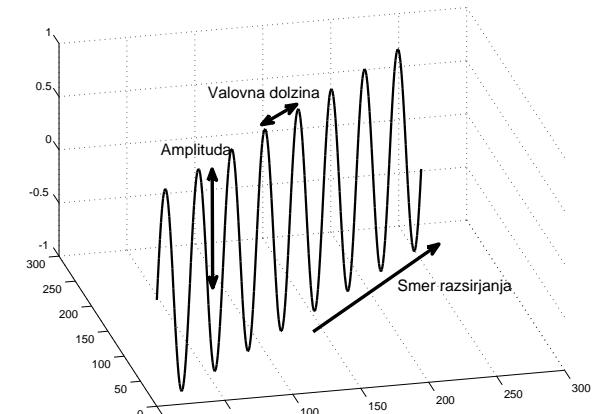
## Aplikacije na osnovi analize hiperspektralnih slik

- ▷ geologija:
  - ★ kartograiranje mineralov
  - ★ detektiranje lastnosti tal (npr. ocenjevanje vlažnosti, slanosti, prisotnosti organskih snovi)
- ▷ vegetacija
  - ★ identificiranje vrst vegetacije
- ▷ vojska:
  - ★ detektiranje vojaških vozil, skritih pod vegetacijo

# Polarizirano zajemanje

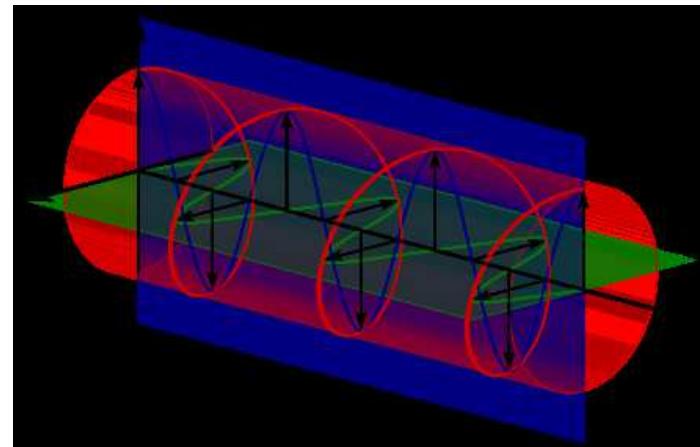
## Osnovni pojmi

- ▷ Polarizacija je splošen deskriptor svetlobe in vsebuje informacijo o objektih, ki odbijajo svetlobo:
  - ★ Tradicionalni senzorji na osnovi intenzitete to informacijo ignorirajo!
- ▷ Trije neodvisni pojmi, s katerimi opišemo svetlobo:
  - ★ valovna dolžina – določa barvo
  - ★ amplituda – določa svetlost oz. intenziteto
  - ★ smer vibriranja – polarizacija
- ▷ Iz polarizacije lahko določimo podrobnosti o izvoru svetlobe in površini, ki svetlobo odbija.
- ▷ Človeško oko ni zmožno vizualizirati polarizacije svetlobe, kar je za živali pogost pojav:
  - ★ čebele – polarizacijo uporabljajo kot kompas
  - ★ določeni insekti (npr. vodni pajek) in kačji pastir – za detektiranje vode in podvodnih predatorjev
  - ★ ligenj – lahko modulira odbojnost svoje kože (za kamuflažo)



## Podrobneje o polarizaciji

- ▷ **Tipi polarizacije** – polarizacija se nanaša na orientacijo električnega polja elektromagnetskoga valovanja
  - \* **linearna polarizacija:**
    - ▷ vektor električnega polja oscilira zgolj v eni ravnini
    - ▷ v naravi: pri odboju svetlobe od površine (npr. odboj od jezera)
  - \* **krožna polarizacija:**
    - ▷ kombinacija dveh linearnih valovanj, pravokotnih ena na drugo, ki sta za  $90^\circ$  iz faze
    - ▷ vektor električnega polja opisuje krožnico skozi čas
- ▷ Polarizacijo svetlobnega valovanja popolnoma popišemo s **Stokesovim vektorjem**, ki sestoji iz 4 komponent:
  - \*  $s_0$  – skupna intenziteta svetlobe
  - \*  $s_1$  – popisuje horizontalno in vertikalno polarizacijo
  - \*  $s_2$  – popisuje komponente pri  $\pm 45^\circ$
  - \*  $s_3$  – popisuje desno in levo krožno polarizacijo



Krožna polarizacija

## Slikovni polarimetri (*imaging polarimeters*)

- ▷ So naprave, ki izmerijo Stokesov vektor za vsako točko v sliki
- ▷ Najdemo jih v različnih izvedbah
  - ★ Časovno sekvenčni (*time sequential*) – spreminja konfiguracijo svojih inštrumentov med meritvami
  - ★ Na osnovi delitve amplitudo – npr. uporaba "beamsplitter"-ja
  - ★ Na osnovi delitve luknjice (fronte valovanja)

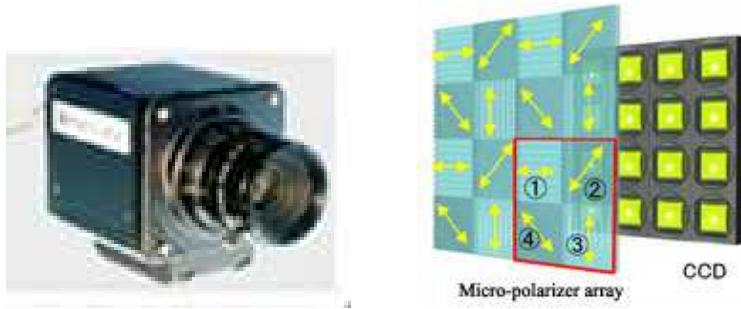
## Polarimeter na osnovi delitve fronte valovanja

- ▷ Fronta valovanja (*wavefront*) je množica točk z isto fazo.
  - ★ Pri razširjanju valovanja v 3D je to ploskev.
- ▷ Ta vrsta polarimetrov deluje na podobni osnovi kot kamere CCD, ki zajemajo barvne slike s pomočjo Bayerjevega filtra.
- ▷ Koncept zajemanja:
  1. Vsak piksel zajema določeno polarizacijsko stanje, ki se razlikuje od stanja soseda.
  2. S pomočjo interpolacije in registracije ocenimo polarizacijo v vsakem pikslu.
- ▷ Robustno zajemanje, omejeni smo le s hitrostjo kamere in s postopkom interpolacije.

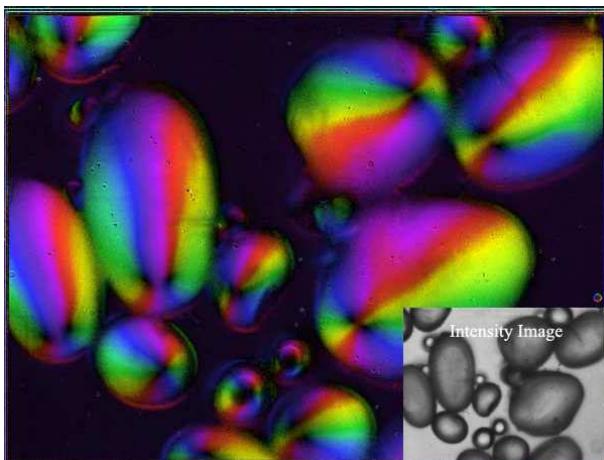
# Polarizirano zajemanje

(4)

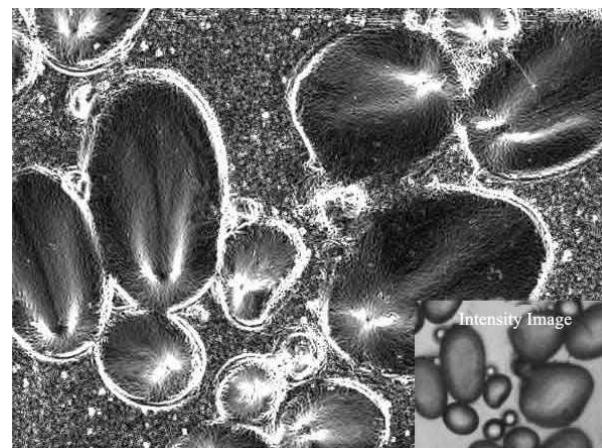
**Primer:** Polarizacijska slikovna kamera PI-100 [4]



- ▷ Tehnične podrobnosti:
  - ★ velikost delovnega območja  $1024 \times 868$  pikslov, hitrost zajema 8 fps, spektralno območje 500–550 nm
- ▷ Primeri slik:



Polarizacija krompirjevega škroba pod mikroskopom.



Slika diferencialnega polarizacijskega azimuta škrobnega zrna. Vidna je interna struktura.

# **Polarizirano zajemanje**

**(5)**

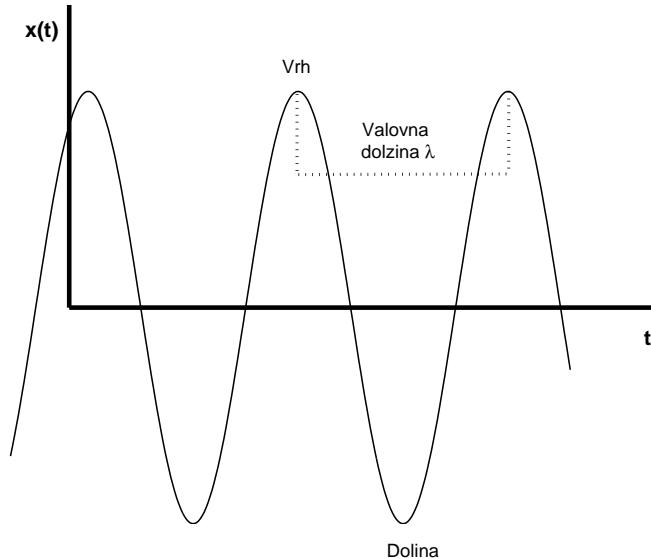
## Aplikacije

- ▷ Identificiranje defektov v proizvodnji letal.
- ▷ "Pogled v vodo":
  - ★ npr. detektiranje oljnih madežev
- ▷ Avtomatsko detektiranje umetnih objektov v naravnih okoljih.
- ▷ Odstranjevanje bleščanja iz slik.

# Interferometrija

## Nujni pojmi za razumevanje interferometrije

- ▷ Svetlobni vir oddaja svetlobno valovanje.
- ▷ Valovi se nenehno razširjajo iz svetlobnega vira v vse smeri
  - \* podobno kot valovanje na jezeru, ko vržemo vanj kamen
- ▷ **Bela svetloba** (npr. sonce) – vsebuje vse valovne dolžine, na katere je človeško oko občutljivo
- ▷ Za interferometrijo je najbolje uporabiti svetlobo ene same valovne dolžine (t.i. monokromatsko svetlobo)
  - \* svetlobni vir mora biti še **koherenoten (*coherent*)** – vsi svetlobni valovi potujejo v fazi
  - \* najbolj primerna je laserska svetloba



## Kaj je interferenca?

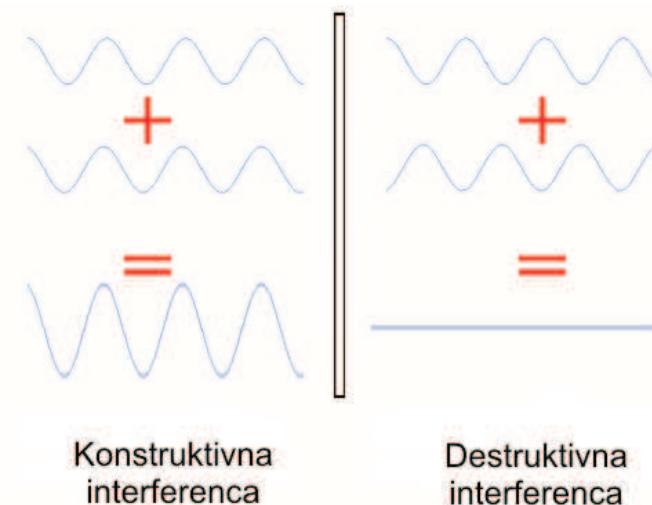
- ▷ Interferenca se pojavi, ko se srečata in prekrijeta dve valovanji, ki nosita energijo:
  - \* namesto dveh valovanj dobimo eno novo
  - \* oblika in velikost novega valovanja je odvisna od originalnih valovanj
  - \* takšnemu združevanju valovanja rečemo tudi **superpozicija (superposition)**

### ▷ Konstruktivna interferenca

- \* dobimo valovanje z isto valovno dolžino in frekvenco
- \* amplituda je večja (višji vrhovi)

### ▷ Destruktivna interferenca

- \* vrhovi v enem valovanju se izničijo z dolinami v drugem valovanju



## Dve ekstremni situaciji:

- ▷ Konstruktivna interferenca – dve identični valovanji sta v fazi
  - \* dobimo novo valovanje identično osnovnemu, le vrhovi so  $2 \times$  večji
- ▷ Destruktivna interferenca – dve identični valovanji sta popolnoma iz faze (sta v antifazi)
  - \* kot rezultat dobimo nič
- ▷ Množica možnosti med obema ekstremoma!

# Interferometrija

(3)

## Interferometer

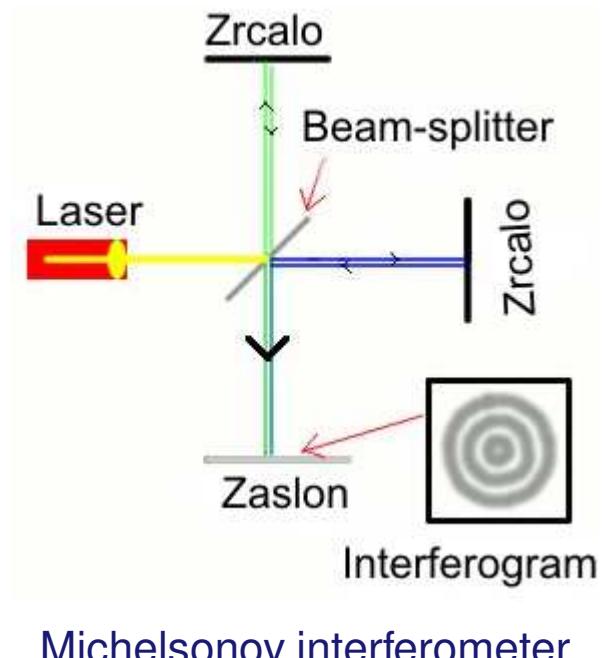
- ▷ Naprava (oz. sistem), ki je zmožna z združevanjem svetlobe različnih virov ustvariti interferogram in ga vizualizirati (oz. posneti s kamero CCD).
- ▷ Interferometer je izredno natančen znanstveni instrument, namenjen merjenju z izredno natančnostjo.

## Kako deluje? Razlaga za dva žarka (oz. valovanja).

- ▷ Uporabimo svetlobni vir (tipično laser) in ga razdelimo v dva žarka.
  - \* "Beam-splitter" – polprosojno steklo ali pol steklo (tj. steklo narahlo obloženo s srebrom)

- ▷ Referenčni žarek
  - \* potuje do zrcala in se odbije do zaslona (kamere CCD)

- ▷ Testni žarek
  - \* sveti na oz. skozi merjenec (objekt), do drugega zrcala, se odbije ter gre skozi "beam-splitter" do zaslona



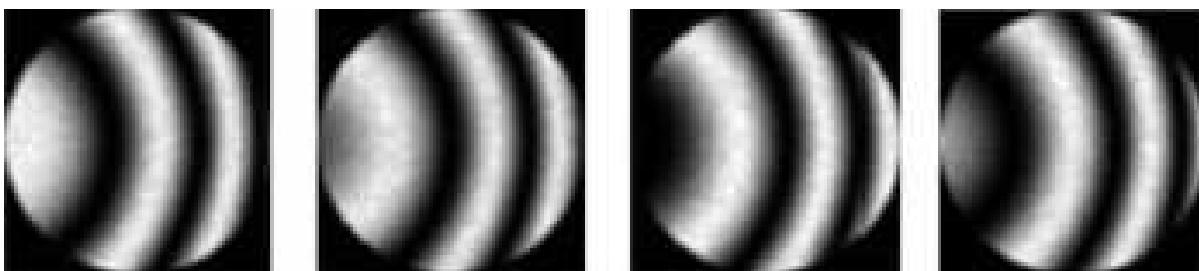
- ▷ Testni žarek potuje malenkost daljšo pot (oz. po drugačni poti) kot prvi žarek.
  - \* Žarka sta iz faze.
  - \* Posledica: pride do interference med žarkoma.

# Interferometrija

(4)

## Interferogram

- ▷ Rezultat interference med žarkoma.
- ▷ Nastane vzorec svetlih in temnih področij, ki ga lahko vizualiziramo.

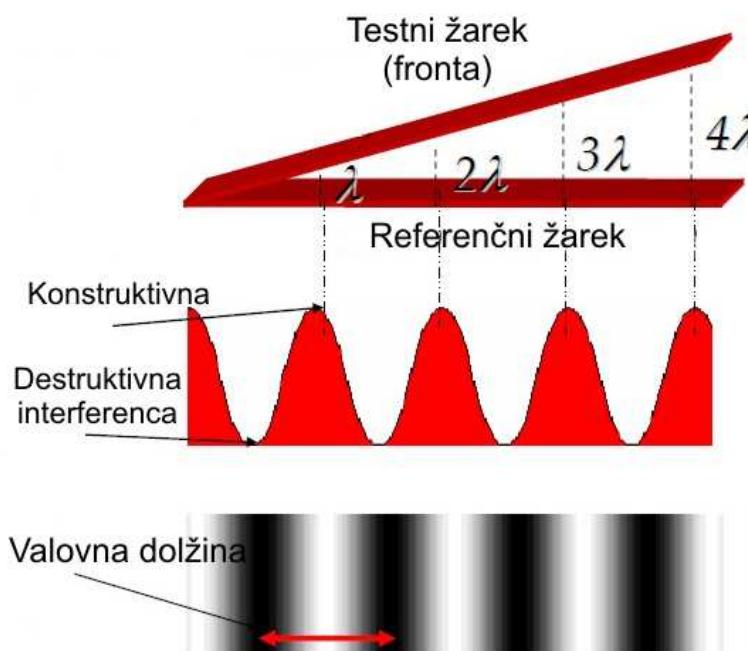


- ▷ Interferogram vsebuje pasove oz. **interferenčne resice (*interference fringes*)**
  - \* temna področja – področja, kjer je prišlo do destruktivne interference
  - \* svetla področja – področja, kjer je prišlo do konstruktivne interference
- ▷ Oblika vzorca interference odvisna od:
  - \* različnosti poti med žarkoma in/ali
  - \* dodatne razdalje, ki jo je prepotoval eden od žarkov.
- ▷ Z opazovanjem in merjenjem resic lahko oba pojava (različnost poti, dodatna razdalja) izračunamo z veliko natančnostjo
  - \* interferometri lahko teoretično merijo razdalje z natančnostjo reda  $\mu\text{m}$

## Oblike interferograma

### A. Žarka z valovno fronto v ravnini (*flat wavefront*)

- ▷ Če sta žarka popolnoma pravokotna oz. je en žarek rahlo nagnjen
  - \* rezultat: Interferogram z ravnimi, paralelnimi svetlo-temnimi pasovi



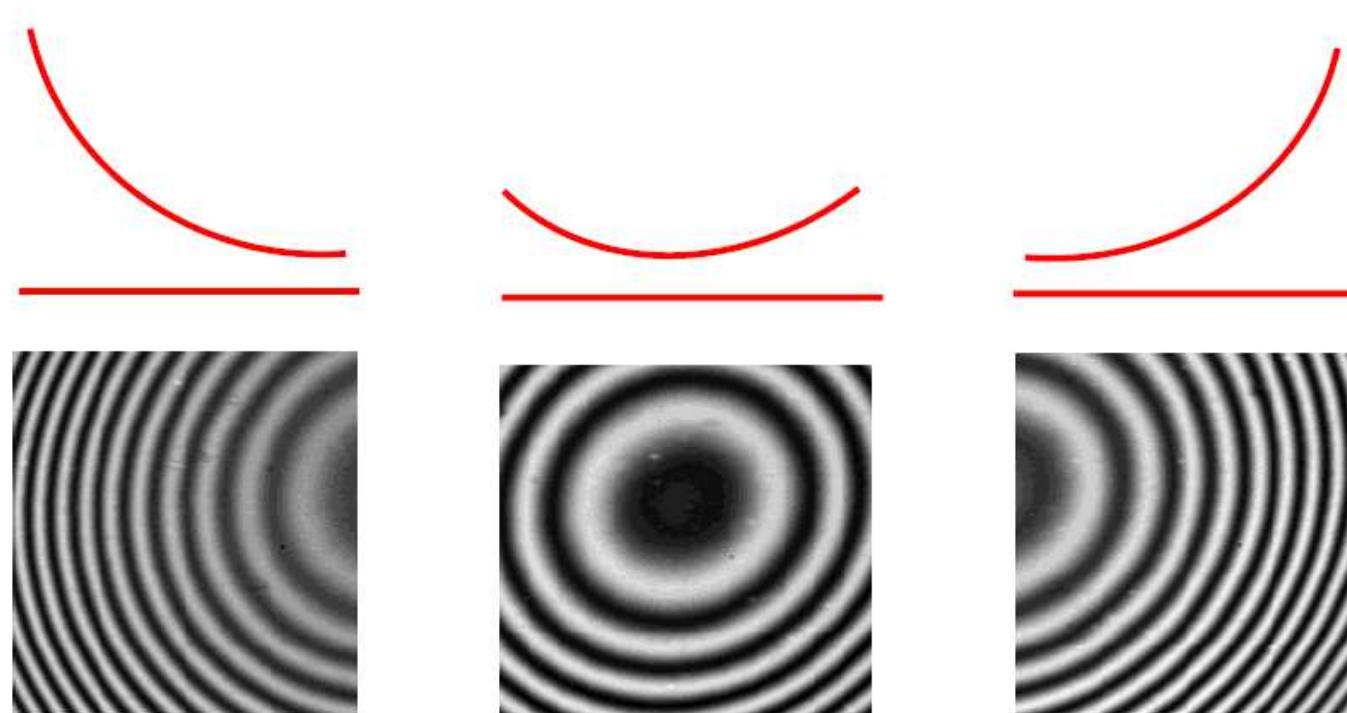
- ▷ Razmik med dvema resicama iste barve je velikosti 1 valovne dolžine.
- ▷ Z nagibanjem enega od obeh žarkov dobimo več resic:
  - \* velikost nagiba določimo kot število pasov v luknjici (zaslonki).

# Interferometrija

(6)

## B. Žarek s sferično valovno fronto, drugi pa s fronto v ravnini

- ▷ Rezultat: Interferogram z ukrivljenimi resicami
- ▷ Če ni nagiba, potem so resice popolnoma krožne.

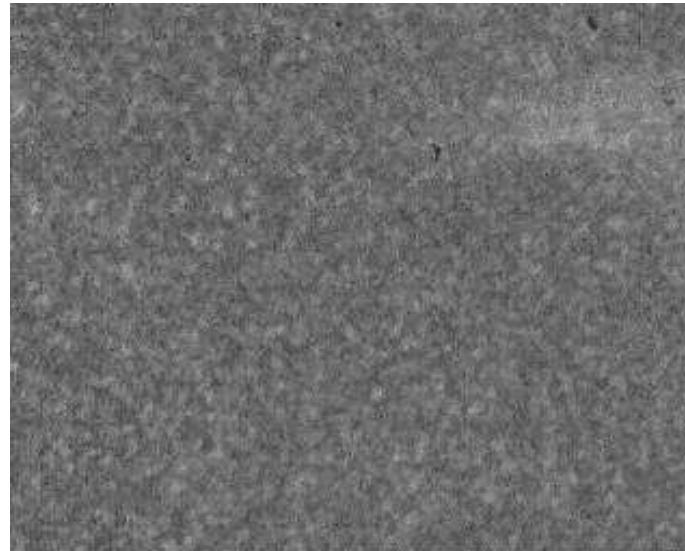


# Interferometrija

(7)

## Kje še najdemo interferometrijo?

- ▷ Radarska interferometrija
- ▷ Mikroskopi na osnovi interferometrije
- ▷ Prototipna aplikacija (Laboratorija SPO in LEOSS): Spremljanje vitalnih življenjskih znakov s pomočjo optičnega vlakna in kamere
  - \* Določanje trenutkov dihanja in srčnih utripov
  - \* Interferenca na osnovi več svetlobnih izvorov (večrodonno optično vlakno)
  - \* Gre za t.i. **pegasto interferometrijo**



## **Dodatna literatura**

- [1] R.B. Smith (2006), Introduction to hyperspectral imaging, MicroImages inc.,  
<http://www.microimages.com/documentation/Tutorials/hyprspec.pdf>
- [2] P. Shippert (2003), Introduction to hyperspectral image analysis,  
<http://spacejournal.ohio.edu/pdf/shippert.pdf>
- [3] J. Katrašnik, M. Burmen, F. Pernuš, B. Likar (2009), Near-infrared imaging of subcutaneous veins, V: Zborniku ROSUS 2009, Maribor, Slovenija, str. 101–109.
- [4] Polarization imaging camera PI-100,  
[http://www.photonic-lattice.com/en/Pol\\_Camera0.html](http://www.photonic-lattice.com/en/Pol_Camera0.html)
- [5] A. G. Andreou, Z. K. Kalayjian(2002), Polarization imaging: Principles and integrated polarimeters, IEEE Sensors journal, vol. 2, št. 6, str. 566–576.
- [6] L. Bigué, P. Ambs, A. Jaulin, A. Foulonneau, L. Gendre (2008), Dynamic polarimetric imaging: Overview and implementation using liquid crystal cells, Journal of Physics: Conference Series, vol. 139, št. 1, str. 012007.
- [7] How interferometers work: a simple introduction to interferometry,  
<http://www.explainthatstuff.com/howinterferometerswork.html>
- [8] Interferometry basics,  
[http://www.cyanogen.com/help/quickfringe/...QUICK\\_FRINGE.htm#QUICK\\_FRINGEInterferometry\\_Basics.htm](http://www.cyanogen.com/help/quickfringe/...QUICK_FRINGE.htm#QUICK_FRINGEInterferometry_Basics.htm)

### ***3. Napredna predobdelava slik***

---

Odstranjevanje pegastega šuma

Barvna vztrajnost

# **Odstranjevanje pegastega šuma**

- ▷ Odstranjevanje/zmanjševanje šuma (*noise removal/reduction*) iz slik še vedno aktualen raziskovalni izziv:
  - ★ postopki vnašajo artefakte v slike
  - ★ zameglij oz. zapacajo (*blur*) slike
- ▷ **Šum razdelimo v dve skupini:**
  - ★ šum, neodvisen od slikovnih podatkov
  - ★ šum, odvisen od slikovnih podatkov

## **A. Šum, neodvisen od slikovnih podatkov**

- ▷ Modeliramo ga v obliki modela aditivnega šuma

$$I(\mathbf{p}) = X(\mathbf{p}) + N(\mathbf{p})$$

	I	–	posneta oz. šumna digitalna slika
	X	–	brezšumna oz. želena digitalna slika
	N	–	šum
	$\mathbf{p} = [i, j]$	–	piksel

- ▷ Pri obravnavanju te vrste šuma običajno sprejmemo določene predpostavke:
  - ★ šum ima ničelno povprečje
  - ★ varianca šuma je  $\sigma_N^2$
  - ★ šum je enakomerno porazdeljen po celotni frekvenčni domeni (**t.i. bel oz. Gaussov šum**)

# **Odstranjevanje pegastega šuma**

(2)

## **B. Šum, odvisen od slikovnih podatkov**

- ▷ Modeliramo ga v obliki multiplikativnega oz. nelinearnega modela šuma

$$I(\mathbf{p}) = X(\mathbf{p})N(\mathbf{p})$$

## **Statistične mere za merjenje vpliva šuma**

- ▷ Uporabljene oznake:
  - ★  $I$  – originalna oz. šumna slika,  $\hat{X}$  – slika z odstranjenim šumom
  - ★  $\sigma$  – standardni odklon,  $\mu$  – povprečje v določenem oknu oz. celotni sliki
- ▷ **MSE (mean squared error):**

$$MSE = \frac{1}{MN} \sum_i \sum_j (I(\mathbf{p}) - \hat{X}(\mathbf{p}))^2$$

- ▷ **RMSE (root mean squared error):**

$$RMSE = \sqrt{MSE}$$

- ▷ **PSNR (peak signal-to-noise ratio):**

$$PSNR = 20 \log_{10} \left( \frac{\max_{\mathbf{p}} (I(\mathbf{p}))}{RMSE} \right) \quad [\text{dB}]$$

# Odstranjevanje pegastega šuma

(3)

- ▷ SNR (*signal-to-noise ratio*):

$$SNR = 10 \log_{10} \frac{\sigma_I^2}{\sigma_{\hat{X}}^2} \quad [\text{dB}]$$

$$SNR = \frac{\mu_I}{\sigma_I}$$

- ▷ S/MSE (*signal-to-mean-square error ratio*):

$$S/MSE = 10 \log_{10} \frac{\sum_i \sum_j \hat{X}(\mathbf{p})^2}{\sum_i \sum_j (\mathbf{l}(\mathbf{p}) - \hat{X}(\mathbf{p}))^2} \quad [\text{dB}]$$

**Primer:**



MSE = 902,7 ; SNR = 2,0 dB

PSNR = 18,6 dB

MSE = 3074,3 ; SNR = 1,5 dB

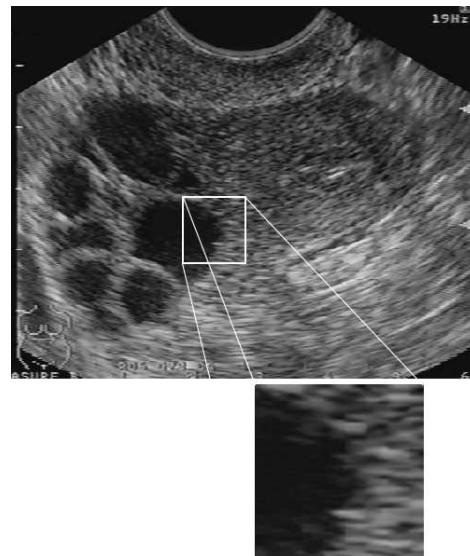
PSNR = 13,3 dB

# Odstranjevanje pegastega šuma

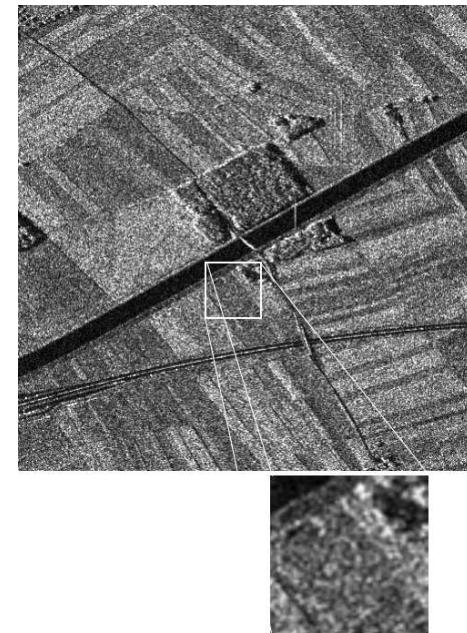
(4)

## Pegasti šum (*speckle noise*)

- ▷ Šum, odvisen od slikovnih podatkov: multiplikativni šum.
- ▷ Artefakt, ki ga povzroči interferenca energije iz naključno porazdeljenega raztresanja (*scattering*).
- ▷ Pojavlja se pri vsakem koherentnem slikanju (npr. ultrazvočnih slikovnih materialih, radarskem slikanju).
  - \* sliko pri koherentnem slikanju dobimo kot seštevek odbitih signalov (valovanj) od razsipovalcev (*scatterers*).
- ▷ Sliki daje karakterističen zrnat izgled.



medicinska ultrazvočna slika



slika SAR

# **Odstranjevanje pegastega šuma**

**(5)**

Kako nastane?

1. Znotraj ene ločljivostne celice množica elementarnih razsipovalcev odbije vpadno valovanje proti senzorju.
  2. Nazaj razsipano koherentno valovanje z različnimi fazami je izpostavljeno konstruktivni ali destruktivni interferenci na naključen način.
- ▷ Pegasti šum se pojavi, ko se valovanje razsipa od površine, katere hrapavost je reda valovne dolžine valovanja, ki zatem povzroči interferenco.
- ▷ **Statistične lastnosti tega šuma so odvisne od gostote in prostorske razporeditve razsipovalcev:**
- \* pega ima naključno in deterministično naravo – razsipovalci so namreč porazdeljeni naključno oz. koherentno na površini (tkivu)
  - \* **tekstura opazovane pege (*speckle*) ne ustrezajo spodaj ležečemu objektu**
    - ▷ lokalna svetlost vzorca pa kljub temu odseva lokalno ehogeničnost spodaj ležečih razsipovalcev

# **Odstranjevanje pegastega šuma**

(6)

## **Metode za odstranjevanje/zmanjševanje pegastega šuma**

- ▷ Primerna metoda za odstranjevanje te vrste šuma mora:
  - ★ izboljšati razmerje SNR
  - ★ ohraniti robove in linije v slikah

Uporabljene oznake (prvi del je prikazan na str. 62)

- ▷ Predpostavka: šum ima povprečje 1 ( $\mu_N = 1$ )

$$ENL = \left( \frac{\mu_I}{\sigma_I} \right)^2 = \frac{1}{\sigma_N} = L$$

- ▷ ENL (*equivalent number of looks*) – za ocenjevanje nivoja pegastega šuma v slikah SAR

### **A. Adaptivni mediani filter (*adaptive neighbourhood median filter*)**

Postopek:

1. Definiramo velikost večje maske (okna, jedra)  $H_1$  in manjše maske  $H_2$ .
2. Vse piksle, ki so manjši od praga  $T$ , filtriramo z medianim filtrom z masko  $H_1$ , preostale piksle pa z medianim filtrom z masko  $H_2$ .
3. Korak 2 lahko ponovimo 2×.

# ***Odstranjevanje pegastega šuma*** (7)

## **B. Kuanov filter**

- ▷ Temelji na kriteriju minimuma povprečne kvadrirane napake (*MMSE – minimum square error*).
- ▷ Predpostavlja naslednji multiplikativen model šuma:

$$I(\mathbf{p}) = X(\mathbf{p}) + (N(\mathbf{p}) - 1)X(\mathbf{p})$$

- ▷ Tipična uporabljena predpostavka:
  - ★ šum ima povprečje enako 1 (*unit-mean noise assumption*)
- ▷ Postopek:

$$\hat{X}(\mathbf{p}) = \mu_I + \frac{\sigma_X^2 (I(\mathbf{p}) - \mu_I)}{\sigma_X^2 + (\mu_I + \sigma_X^2) / L} \quad (1)$$

kjer

$$\sigma_X^2 = \frac{L\sigma_I^2 - \mu_I^2}{L + 1}$$

- ▷ V patološki situaciji, kadar je  $\sigma_X^2 < 0$ , določimo  $\hat{X}(\mathbf{p}) = \mu_I$ .

## **C. Leejev (MMSE) filter**

- ▷ Posebna oblika Kuanovega filtra, kjer iz enačbe (1) odstranimo člen  $\sigma_X^2 / L$ .

# **Odstranjevanje pegastega šuma** (8)

## **D. Filter gama**

- ▷ Je filter MAP (*maximum a posteriori*), ki temelji na Bayesovi statistični analizi slike.
- ▷ Predpostavka:
  - ★ odbojnost radarskih žarkov, kakor tudi pegasti šum sta porazdeljena po porazdelitvi gama. Superpozicija obeh porazdelitev vrne porazdelitev K.
- ▷ Postopek:

$$\hat{X}(\mathbf{p}) = \frac{(\alpha - L - 1) \mu_l + \sqrt{\mu_l (\alpha - L - 1)^2 + 4\alpha L \mu_l I(\mathbf{p})}}{2\alpha}$$

kjer

$$\alpha = \frac{L + 1}{L (\sigma_l / \mu_l)^2 - 1}$$

# **Odstranjevanje pegastega šuma** (9)

## E. Frostov filter

- ▷ Je adaptivni Wienerjev filter, ki opravi konvolucijo vrednosti piksov znotraj okna (maske) fiksne velikosti z eksponentnim enotnim odzivom  $h$ , definiranim kot

$$h = e^{-KC_I(\mathbf{t}_0)\|\mathbf{t}\|}, \quad C_I = \frac{\sigma_I}{\mu_I},$$

- ▷ kjer je  $K$  parameter filtra (npr.  $K = 1$ ),  $\mathbf{t}_0$  predstavlja lokacijo obdelovanega piksla in  $\|\mathbf{t}\|$  je razdalja, merjena od piksla  $\mathbf{t}_0$  (oz. od središčnega piksla maske).
- ▷ Za vsak piksel v sliki dejansko določimo svojo masko!

## F. Oddyjev filter

- ▷ Neke vrste nizko sito, ki spreminja velikost okna glede na lokalne statistike.
- ▷ Postopek:

- \*  $\hat{X}(\mathbf{p}) = \mu_I$ , če  $m < \alpha\mu_I$
- \*  $\hat{X}(\mathbf{p}) = \frac{\sum_k \sum_l W_{kl} I(k,l)}{\sum_k \sum_l W_{kl}}$ , sicer

kjer

- \*  $W_{kl} = 1$ , če  $|I(k,l) - I(\mathbf{p})| \leq m$
- \*  $W_{kl} = 0$ , sicer

# **Odstranjevanje pegastega šuma** (10)

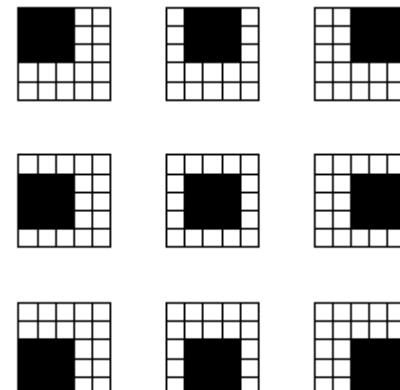
- ▷  $\hat{X}$  ocenujemo lokalno, v oknu velikosti  $3 \times 3$  piksle:

$$m = \frac{1}{8} \sum_k \sum_l |I(k, l) - I(p)|$$

- ▷  $W$  ima vlogo adaptivne binarne maske.
- ▷  $\alpha$  je parameter fitra (npr.  $\alpha = 1$ )

## **G. Filter AFS (*adaptive filter on surfaces*)**

- ▷ Je filter z adaptivno masko, ki uporablja koncept vrednost l.e.s. (*local emerging surface*).
- ▷ Vrednost l.e.s. je ploščina ploskve slikovnega grafa, definiranega znotraj maske oz. okna.
- ▷ Postopek:
  1. Vrednost l.e.s. izračunamo za 9 prikazanih binarnih mask velikosti  $5 \times 5$  pikslov.
  2. Izberemo masko z minimalno vrednostjo l.e.s.
  3. Filtriramo z nizkim sitom z izbrano masko.
  4. Dobljeno vrednost priredimo središčnemu pikslu okna, velikosti  $5 \times 5$  pikslov.



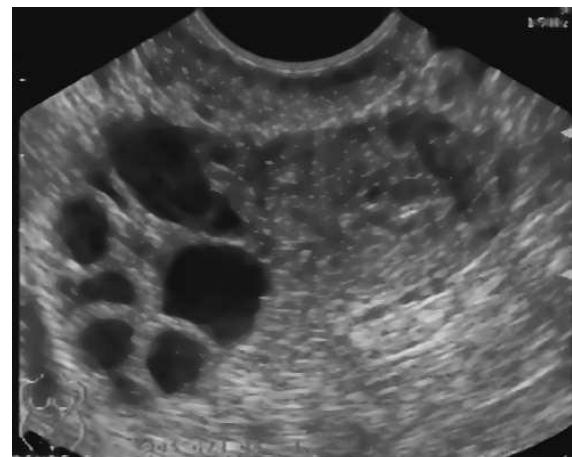
## H. Drugi pristopi

- ▷ Odstranjevanje šuma z valčno transformacijo.
- ▷ Odstranjevanje šuma z nevronskimi mrežami (NN)
  - \* Možen pristop:
    - ▷ Definiramo 3 slojni zvezni perceptron –  $K \times L$  nevronov v 1. plasti,  $K \times L$  nevronov v 2. plasti in 1 nevron v izhodni plasti
    - ▷ Vhod: seznam vrednosti pikslov iz maske, velikosti  $K \times L$  pikslov
    - ▷ Izhod: filtrirana vrednost za središčni piksel maske
    - ▷ **Potrebno je učenje!**

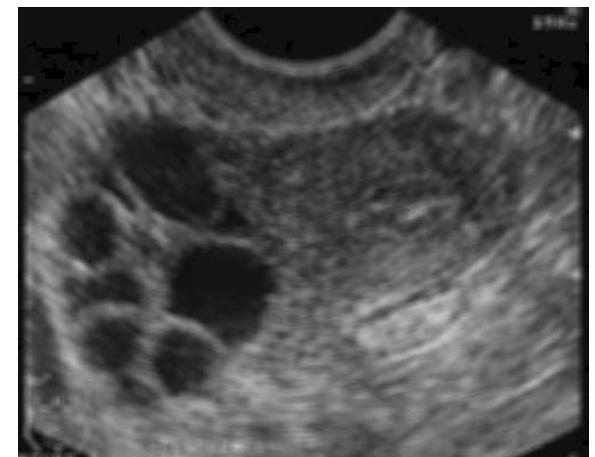
**Primer:**



original



Adaptivni median ( $2 \times 2$ )  
 $H_1 = 9 \times 9$  in  $H_2 = 3 \times 3$   
PSNR = 25,45 dB



Kuanov filter  
maska  $7 \times 7$   
PSNR = 23,19 dB

## Barvna vztrajnost

- ▷ Barvna kalibracija ali pravilne barve barvna vztrajnost (ang. colour constancy) omogoča konsistentno dojemanje barve predmetov pod različnimi osvetlitvami.



Ali so na sliki pravilne barve?

- ▷ Barvna vztrajnost dobro razvita pri ljudeh, opazili pa jo tudi pri zlatih ribicah in čebelah.

### Primer:

- ▷ Opazovanje predmeta pri sončni svetlobi:
  - ★ sončna svetloba je intenzivna, saj ima barvno temperaturo okrog 6000 K.
- ▷ Enak predmet opazujemo v stanovanju pod nitno žarnico (barvna temperatura med 2700 K do 3300 K):
  - ★ praktično ne opazimo spremembe v barvi
- ▷ Enak učinek pod svetlogo varčnih žarnic (barvna temperatura okrog 5500 K)
- ▷ V ekstremnih pogojih pa tudi barvna vztrajnost odpove (npr. v nekaterih nočnih klubih, kjer uporabljajo barvne filtre).

## Vrste algoritmov za barvno vztrajnost

- ▷ Razvitih že mnogo algoritmov barvne vztrajnosti:
  - ★ najpogosteje uporabljene predpostavke: določen tip osvetlitve, karakteristika tipa površin, pomen povprečne barve v sliki ipd.
- ▷ Ločimo dve večji skupini algoritmov:
  - ★ algoritmi, ki predpostavljajo uniformno osvetlitev scene
  - ★ algoritmi, ki predpostavljajo več izvorov svetlobe in neuniformno osvetlitev

### A. Algoritmi, ki predpostavljajo uniformno osvetlitev scene

- ▷ Predpostavka uniformne osvetlitve predvideva sceno z enim virom svetlobe, brez vidnih senc in brez sprememb v jakosti osvetlitve v prostoru.
- ▷ V primeru izpolnjevanja te predpostavke so zelo uspešni.
- ▷ Takšno osvetlitev je težavno pripraviti že v laboratoriju, v nekontroliranem okolju te omejitve skoraj nikoli ne izpolnimo:
  - ★ kljub temu se uporabljajo tudi za uravnavanje bele barve v naravnih scenah.

### B. Algoritmi, ki predpostavljajo več izvorov svetlobe in neuniformno osvetlitev

- ▷ So splošnejši, a še vedno ne univerzalni.
- ▷ Pogosta predpostavka: počasno spremenjanje oz. nizek gradient osvetlitve.
- ▷ Ob tem se predpostavijo še hitre spremembe oz. visok gradient odbojnosti predmetov in geometrijskega faktorja scene (tj. vpliv prostorskih razmerij med virom svetlobe, odbojno površino in senzorjem).

## Model za formiranje barvnih vrednosti piksla

- ▷ Barvna vrednost piksla je signal treh različnih vrst senzorjev, kjer se vsak senzor odziva na določen del vidnega spektra svetlobe.
  - \* Odziv senzorja označimo s funkcijo spektralnega odziva  $S_v(\lambda)$ , kjer indeks  $v$  označuje senzor oz. barvni kanal (tj. R, G ali B).
  - \* Svetlobo, na senzorju na lokaciji piksla  $\mathbf{p}$ , označimo s funkcijo  $E(\lambda, \mathbf{p})$ .
  - \* **Vrednost signala oz. barvnega kanala piksla  $C_v(\mathbf{p})$  modeliramo kot:**

$$C_v(\mathbf{p}) = \int_{\lambda} S_v(\lambda) E(\lambda, \mathbf{p}) d\lambda$$

- \* Funkcija  $E(\lambda, \mathbf{p})$  odvisna od
  - ▷ spektra svetlobe  $L(\lambda, \mathbf{p})$ , s katero je osvetljena scena,
  - ▷ odbojnosti predmetov  $R(\lambda, \mathbf{p})$  v sceni in
  - ▷ vpadnega kota ter tipa odboja svetlobe od predmetov.
- ▷ **Enačbo poenostavimo, če predpostavimo, da**
  1. prisoten le razpršen odboj svetlobe – tj. za vse površine velja Lambertov kosinusni zakon.
  2. senzor se odziva samo na valovno dolžino  $\lambda_v$  – funkciji  $R$  in  $L$  sta odvisni od vrste senzorja:

$$C_v(\mathbf{p}) = G(\mathbf{p}) R_v(\mathbf{p}) L_v(\mathbf{p})$$

- \* Funkcija  $G(\mathbf{p})$  predstavlja geometrijski faktor scene.

## A. Algoritem WPR (*white patch retinex*)

- ▷ Preprost algoritem, ki sliko popravlja po posameznih barvnih kanalih, in sicer neodvisno enega od drugega.
- ▷ Uporabljene predpostavke:
  1. osvetlitev scene je uniformna:
    - ★ osvetlitev tedaj neodvisna od položaja piksla.
  2. v sceni je prisoten predmet bele barve, ki odbija vso osvetlitev.
    - ★ takšen predmet ima maksimalno barvno vrednost, vsebuje pa tudi popolno informacijo o osvetlitvi, kar modeliramo kot:

$$L = \max(C(\mathbf{p}))$$

- ▷ Novo vrednost piksla,  $O(\mathbf{p})$ , neodvisno od osvetlitve scene, določimo kot:

$$O(\mathbf{p}) = G(\mathbf{p})R(\mathbf{p}) = \frac{C(\mathbf{p})}{L}$$

## B. Algoritem GWA (*gray world assumption*)

- ▷ Predpostavka:
  - ★ v sceni predmeti različnih barv, pri čemer so vse barve enako zastopane.
  - ★ osvetlitev določimo iz "povprečne barve" v sliki, z izračunanim povprečjem nato uravnnavamo belo barvo.
- ▷ Postopek:

$$O(\mathbf{p}) = \frac{C(\mathbf{p})}{\bar{C}\varphi}$$

kjer

- ★  $\bar{C}$  – povprečna vrednost barvnega kanala
- ★  $\varphi$  – poljubno nastavljiv faktor povečave (npr.  $\varphi = 2$ )

# Barvna vztrajnost

(6)

## C. Algoritem CCN (*comprehensive colour normalization*)

- ▷ Iterativen algoritem, kjer zaporedoma izvajamo dva koraka:
  1. Odstranjevanje geometrijskega faktorja s pomočjo barvnega normiranja:

$$O'(\mathbf{p}) = \frac{C(\mathbf{p})}{C_R(\mathbf{p}) + C_G(\mathbf{p}) + C_B(\mathbf{p})}$$

2. Skaliranje vrednosti z algoritmom GWA:

$$O(\mathbf{p}) = \frac{O'(\mathbf{p})}{\bar{C}\varphi}$$

3. V naslednji iteraciji:  $C(\mathbf{p}) \leftarrow O(\mathbf{p})$
  4. Zaključni pogoj (konvergenca):
    - ★ vse vhodne vrednosti pikslov v trenutni iteraciji,  $C(\mathbf{p})$ , enake izhodnim vrednostim  $O(\mathbf{p})$
- ▷ **Z barvno normalizacijo izgubimo informacijo o intenzivnosti za posamezen piksel:**
    - ★ Če to informacijo potrebujemo, potem uporabimo intenzivnost izvorne slike in jo vstavimo v obdelano sliko (npr. komponento  $Y$  originalne slike v prostoru YCbCr priredimo komponenti  $Y$  obdelani sliki).

## D. Algoritem LSAC (*local space average colour*)

- ▷ Temelji na algoritmu GWA.
- ▷ Predpostavka: Imamo počasi spremenljajočo se osvetlitev.
- ▷ Osvetlitev izločimo iz slike z 2-D nizkim filtrom.
- ▷ Postopek:

$$O(\mathbf{p}) = \frac{C(\mathbf{p})}{L(\mathbf{p})} = \frac{C(\mathbf{p})}{\bar{C}(\mathbf{p})\varphi}$$

- \*  $\bar{C}(\mathbf{p})$  – funkcija, ki iz slike izloči visoke frekvence (tj. počasi spremenljajočo se osvetlitev):

$$\bar{C}(\mathbf{p}) = C(\mathbf{p}) * K(\mathbf{p})$$

- \*  $K(\mathbf{p})$  – jedro 2D nizkega sita (npr. Gaussovo ali eksponentno jedro)

**Primer:** Gaussovo jedro

$$K(\mathbf{p}) = K(x, y) = a^{-1} e^{-\frac{x^2+y^2}{2\rho^2}}$$

- ▷  $a$  – ojačanje filtra
- ▷ Optimalna vrednost parametra  $\rho$  se določi iz dimenzije slike kot

$$\rho = 0,093 \max(M, N)$$

## E. Algoritem RSR (*random spray retinex*)

- ▷ Predpostavka:
  - ★ v sliki so prisotne bele površine, ki odbijajo celoten spekter osvetlitve (tj. vsebujejo popoln podatek o osvetlitvi).
  - ★ osvetlitev je počasi spreminjača se.
- ▷ Barvno vrednost piksla popravljamo s pomočjo  $P$  razpršenih množic ( $p = 1, \dots, P$ ), vsaka izmed njih pa vsebuje  $K$  pikslov.
- ▷  $k$ -ti piksel iz  $p$ -te razpršene množice,  $\mathbf{p}_{p,k} = [i_{p,k}, j_{p,k}]$ , določimo kot

$$\begin{aligned} i_{p,k} &= i + r\eta \cos(\mu) \\ j_{p,k} &= j + r\eta \sin(\mu) \end{aligned}$$

kjer  $\eta \in U(0, 1)$ ,  $\mu \in U(0, 2\pi)$ ,  $\mathbf{p} = [i, j]$  in  $r$  je radij razpršene množice.

- ▷ Postopek barvnega izravnavanja:

$$O(\mathbf{p}) = \frac{1}{P} \sum_{p=1}^P \frac{C(\mathbf{p})}{\max_k(C(\mathbf{p}_{p,k}))}$$

- ▷ Prosti parametri:
  - ★ radij razpršene množice pikslov,  $r$  (optimalno: velikost diagonale slike)
  - ★ število razpršenih množic na piksel,  $P$  (npr.  $P = 20$ )
  - ★ število pikslov v razpršeni množici,  $K$  (npr.  $K = 500$ )
  - ★ Optimalna nastavitev parametrov odvisna od vsebine slike!

## Barvni kalibrator

- ▷ Barvni kalibrator oz. barvna tablica (*colour checker*) – referenčni predmet na digitalni sliki z znano barvo.
- ▷ Uporaba:
  - \* Za barvno kalibracijo (npr. fotografiranje), tudi za ocenjevanje uspešnosti barvnega popravljanja.
- ▷ Izdelava: tablica z barvnimi polji, kjer imajo barve specifične lastnosti za širok nabor osvetlitev (skoraj neodvisne od osvetlitve)

- ▷ Kalibrator GretagMacbeth oz. X-Rite
  - \* Velikosti:  $216 \times 279$  oz.  $57 \times 82$  mm.
  - \* 1. in 2. vrstica: naravne barve in barve, ki jih je težko reproducirati.
  - \* 3. vrstica: osnovne barve seštevalnega prostora RGB in odštevalnega modela CMY.
  - \* 4. vrstica: sivinska lestvica.



- ▷ Kalibrator QP201
  - \* Velikosti:  $142 \times 40$  mm.
  - \* Barve: Nima jasno razdeljenih skupin barv (razen sivinske lestvice)



# Barvna vztrajnost

(10)

**Primer:** Povzeto po [2]



Originalna slika



Algoritem WPR



Algoritem GWA



Algoritem CCN



Algoritem LSAC



Algoritem RSR

- ▷ Merjenje uspešnosti:
  - \* s pomočjo mer kot MSE, RMSE...

## **Dodatna literatura**

- [1] L. Gagnon, A. Jouan (1997), Speckle Filtering of SAR Images - A Comparative Study Between Complex-Wavelet-Based and Standard Filters, V: Zborniku SPIE, vol. 3169, San Diego, ZDA, str. 80-91.
- [2] M. Šavc (2011), Primerjava algoritmov za barvno kalibriranje digitalnih slik, Diplomsko delo, UM-FERI, Maribor, Slovenija.
- [3] M. Sonka, V. Hlavac, R. Boyle (1994), Image processing, analysis and machine vision, Chapman & Hall Computing, London, VB.

## ***4. Napredna segmentacija***

---

Pospoljena Houghova transformacija

Modeli aktivnih kontur – kače

Analiza neodvisnih komponent

## **Posplošena Houghova transformacija**

- ▷ Pomembne informacije o objektu v sliki so pogostokrat skrite v obliki njegove meje (*boundary*):
  - ★ notranja in zunanja meja objekta (glej zapiske predavanj URVRV, poglavje 4)
- ▷ Houghova transformacija (HT) je metoda za detektiranje krivulj, pri čemer izkorišča dualnost med točkami krivulje in parametri te iste krivulje.
- ▷ Obstaja HT za detektiranje:
  - ★ analitičnih krivulj in
  - ★ neanalitičnih krivulj
- ▷ HT se opira na informacijo o:
  - ★ **robnih pikslih**
    - ▷ uporaba zgolj informacije o jakosti robnega piksla oz. gradiента
  - ★ **celotni sliki robov**
    - ▷ rob je vektorska spremenljivka;  $\text{rob} = [\text{jakost roba}, \text{smer gradienta}]$
    - ▷ uporabljata se obe informaciji
- ▷ Pojmi: slika robov (*edge image*), robni piksel (*edge pixel*), jakost gradienta, smer gradienta (glej zapiske predavanj URVRV, poglavje 3).

# Posplošena Houghova transformacija

(2)

## A. HT za analitične krivulje

- ▷ Mejo iskanega objekta lahko predstavimo z analitično krivuljo.
- ▷ Splošno analitično krivuljo zapišemo v obliki:

$$\mathcal{F}(\mathbf{x}, \mathbf{a}) = 0$$

kjer

- \*  $\mathbf{x}$  – slikovna točka oz. piksel
- \*  $\mathbf{a}$  – vektor parametrov krivulje

## 1. HT za iskanje linij oz. preamic

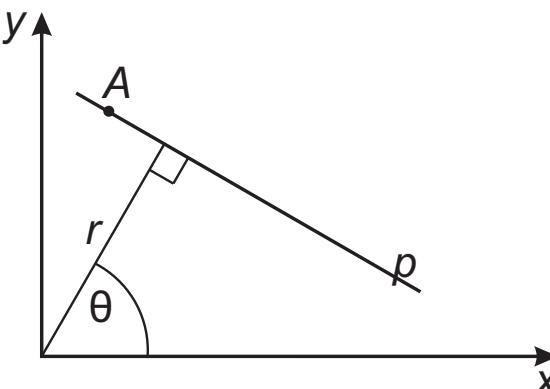
- ▷ Povzetek, podrobnosti v zapiskih predavanj URVRV, poglavje 4

- ▷ Klasična enačba premice:

$$y = kx + n$$

- ▷ Normalna oz. parametrična oblika enačbe:

$$x \cos \theta + y \sin \theta = r$$



- ▷ HT preslika točke iz kartezijiske 2D-ravnine (tj.  $x, y$ ) v parametrični prostor modela (tj.  $r, \theta$ )

- \* primer: točke premice  $p$  se preslikajo v isto točko v Houghovem prostoru

## **Posplošena Houghova transformacija**

(3)

- ▷ HT apliciramo na binarni sliki (npr. binarizirani sliki robov)
- ▷ Za piksele v sliki pa seveda ne vemo, kakšna(e) premica(e) poteka(jo) skozi njih (če sploh!):
  - \* skozi posamezen piksel lahko torej teoretično potekajo premice z vsemi možnimi smermi  $\theta$
- ▷ Piksel  $p = [i, j]$  se s HT preslika v sinusno krivuljo:
  - \* Pri znani orientaciji, tj.  $\theta$ , izračunamo oddaljenost  $r$  kot:

$$r = i \cos \theta + j \sin \theta \quad (2)$$

- \* izračun naredimo pri vseh možnih orientacijah, rezultat pa shranimo v akumulatorsko polje  $A(\theta, r)$ 
  - ▷ Houghov parametrični prostor in izračunane vrednosti moramo diskretizirati!

### **Postopek:**

1. Inicializiraj akumulator  $A$ , tj.  $A(\theta, r) = 0$
2. FOR vsak robni piksel
  - FOR  $\theta = \min_\theta : \Delta\theta : \max_\theta$
  - Izračunaj enačbo (2)
  - $A(\theta, r) = A(\theta, r) + \Delta A$
3. Poišči lokalne maksimume v akumulatorju  $A$ , kar ustreza položajem premic v sliki.

# Posplošena Houghova transformacija

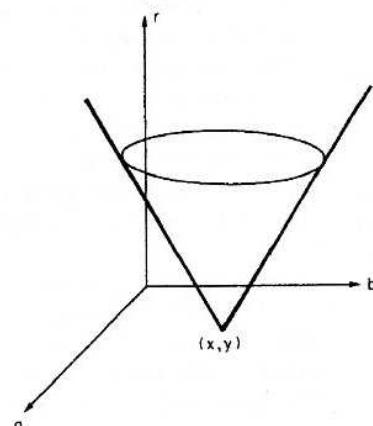
(4)

## 2. HT za iskanje krožnic, izključno na osnovi robnih piksov

- ▷ Klasična enačba krožnice:

$$(x - a)^2 + (y - b)^2 = r^2$$

- ▷ Postavimo si naslednje vprašanje:
  - \* Če naj bi piksel  $\mathbf{p} = [x, y]$  ležal na krožnici, kaj je možen prostor parametrov  $(a, b, r)$  za to krožnico (v parametričnem prostoru)?



Odgovor: Pravilni krožni stožec (vir: [1]).

- ▷ Če v sliki dejansko obstaja krožnica, potem se bo množica takšnih pravilnih krožnih stožcev sekala v skupni točki.
- ▷ Potrebujemo 3D akumulator A, tj.  $A(a, b, r)$ .

# **Posplošena Houghova transformacija**

**(5)**

## **Postopek:**

1. Inicializiraj akumulator  $A$ , tj.  $A(a, b, r) = 0$

2. FOR vsak robni piksel

    FOR  $a = \min_a : \Delta a : \max_a$

        FOR  $b = \min_b : \Delta b : \max_b$

$$r = \sqrt{(x - a)^2 + (y - b)^2}$$

$$A(a, b, r) = A(a, b, r) + \Delta A$$

3. Poišči lokalne maksimume v akumulatorju  $A$ , kar ustreza položajem krožnic v sliki.

## **DN:**

▷ HT za iskanje elips, izključno na osnovi robnih pikslov

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1$$

▷ HT za iskanje elips, izključno na osnovi robnih pikslov, pri čemer je glavna os elipse rotirana za kot  $\varphi$  glede na os  $x$

$$x(t) = x_0 + a \cos t \cos \varphi - b \sin t \sin \varphi$$

$$y(t) = y_0 + a \cos t \sin \varphi + b \sin t \cos \varphi$$

# **Posplošena Houghova transformacija** (6)

## **3. HT za analitične krivulje v sivinskih slikah, na osnovi vseh informacij iz slike robov**

---

- ▷ Iz originalne slike moramo izračunati kompletno sliko robov:
  - ★ jakost oz. velikost gradienta (na tej osnovi določamo robne piksele!)
  - ★ smer gradienta  $\psi$
- ▷ V HT uporabljamo tako originalno enačbo krivulje

$$\mathcal{F}(\mathbf{x}, \mathbf{a}) = 0 \quad (3)$$

kakor tudi njen odvod

$$\frac{d\mathcal{F}}{dx}(\mathbf{x}, \mathbf{a}) = 0 \quad (4)$$

- ▷ V odvodu dobimo člen  $\frac{dy}{dx}$ , ki je poznan, saj

$$\frac{dy}{dx} = \operatorname{tg}[\psi(\mathbf{x}) - \frac{\pi}{2}]$$

### **Postopek:**

1. Inicializiraj akumulator  $A$ , tj.  $A(\mathbf{a}) = 0$
2. FOR vsak robni piksel

Izračunaj vse parametre  $\mathbf{a}$ , za katere veljata enačbi (3) in (4)

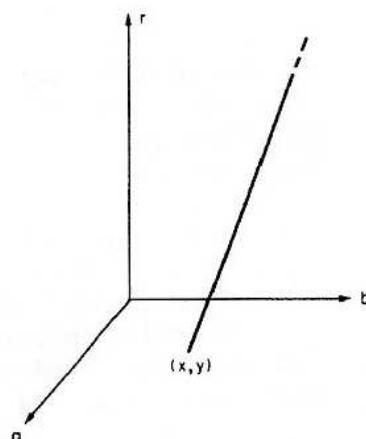
$$A(\mathbf{a}) = A(\mathbf{a}) + \Delta A$$

3. Poišči lokalne maksimume v akumulatorju  $A$ .

## **Posplošena Houghova transformacija**

(7)

- ▷ Če v HT uporabljamo še informacijo o smeri gradiента, se računska zahtevnost bistveno zmanjša.
- ▷ **Primer:**
  - \* Pri iskanju krožnic s tem postopkom, se število prostih parametrov zmanjša na 1.
  - \* Prostor možnih parametrov se iz stožca zreducira na premico.



Prostor možnih parametrov pri iskanju krožnic s pomočjo informacije o smeri gradienta.

# **Posplošena Houghova transformacija** (8)

## **4. HT za iskanje elips v sivinskih slikah, s pomočjo informacije o smeri gradienta**

---

▷ **Začetna predpostavka:**

- ★ elipsa orientirana tako, da je njena glavna os vzporedna z  $x$ -osjo (v nadaljevanju bomo dodali še rotacijo!)

▷ **Enačba elipse:**

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1 \quad (5)$$

▷ Vpeljimo  $X = x - x_0$  in  $Y = y - y_0$  ter odvajamo po  $X$ . Dobimo

$$\frac{2X}{a^2} + \frac{2Y}{b^2} \frac{dY}{dX} = 0 \quad (6)$$

▷ Vrednost  $\frac{dY}{dX}$  je poznana iz informacije o robnem pikslu:

- ★ označimo:

$$\frac{dY}{dX} = \xi$$

▷ Iz enačb (5) in (6) izračunamo

$$Y = \pm \frac{b^2}{\sqrt{1 + \frac{a^2}{b^2} \xi^2}} \quad X = \pm \frac{a^2}{\sqrt{1 + \frac{b^2}{a^2} \xi^2}} \quad (7)$$

## **Posplošena Houghova transformacija** (9)

- ▷ Torej: Če poznamo  $a, b, x, y$  in  $\frac{dY}{dX}$  potem določimo  $x_0$  in  $y_0$  kot

$$y_0 = y \pm \frac{b^2}{\sqrt{1 + \frac{a^2}{b^2} \xi^2}} \quad x_0 = x \pm \frac{a^2}{\sqrt{1 + \frac{b^2}{a^2} \xi^2}} \quad (8)$$

- ★ Dobimo štiri rešitve v štirih kvadrantih.
  - ★ Pravilen kvadrant izberemo na osnovi testiranja predznaka izrazov  $dY$  in  $dX$
- ▷ Upoštevajmo še rotacijo elipse, tj. parameter  $\varphi$
- ★ Parameter  $\xi$  se sedaj določi kot

$$\xi = \operatorname{tg}(\psi - \varphi - \frac{\pi}{2}) \quad (9)$$

- ★ Z izračunanim  $\xi$  določimo  $X$  in  $Y$
- ★ Na koncu dobljeno točko  $(X, Y)$  rotiramo za kot  $\varphi$ , s čimer dobimo središče elipse  $(x_0, y_0)$

# **Posplošena Houghova transformacija**

**(10)**

## **Postopek:**

1. Inicializiraj akumulator  $A$ , tj.  $A(x_0, y_0, \varphi, a, b) = 0$
2. FOR vsak robni piksel

FOR  $a = \min_a : \Delta a : \max_a$

FOR  $b = \min_b : \Delta b : \max_b$

FOR  $\varphi = \min_\varphi : \Delta \varphi : \max_\varphi$

Izračunaj enačbo (9).

Določi  $X$  in  $Y$  po enačbi (7), pri čemer določi predznak na osnovi opazovanja  $dY$  in  $dX$ .

Rotiraj  $X$  in  $Y$  za kot  $\varphi$ .

$$x_0 = x + X$$

$$y_0 = y + Y$$

$$A(x_0, y_0, \varphi, a, b) = A(x_0, y_0, \varphi, a, b) + \Delta A$$

3. Poišči lokalne maksimume v akumulatorju  $A$ , kar ustreza položajem elips v sliki.

# **Posplošena Houghova transformacija** (11)

## **B. Posplošitev HT za neanalitične krivulje**

- ▷ Za opis oblike neanalitične krivulje uporabljamo naslednji vektor parametrov

$$\mathbf{a} = [\mathbf{y}, \mathbf{s}, \varphi]$$

kjer

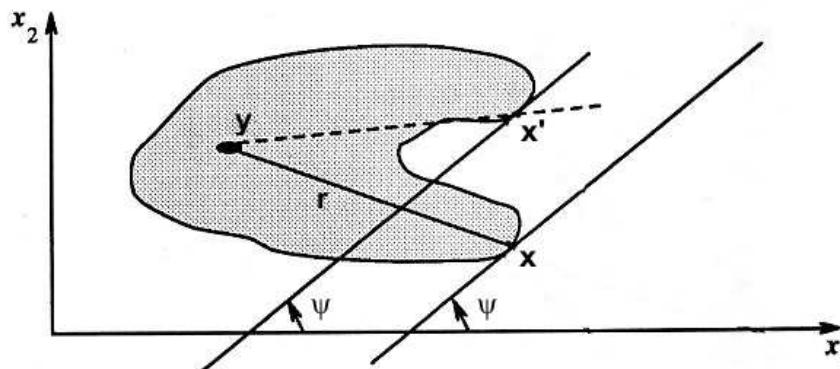
- ★  $\mathbf{y} = [x_0, y_0]$  – referenčno izhodišče za obliko (krivuljo)
- ★  $\varphi$  – orientacija krivulje
- ★  $\mathbf{s} = [s_x, s_y]$  – ortogonalna skalirna faktorja (v praksi  $s_x = s_y$ )
- ★ Položaj referenčnega izhodišča  $\mathbf{y}$  opišemo v obliki tabele možnih orientacij robnih pikslov (tj. R-tabela)
- ▷ **Ključna stvar pri pospološitvi HT je uporaba informacije o smeri gradienta:**
  - ★ ne le, da pohitri algoritom, ampak tudi bistveno izboljša natančnost!

# Posplošena Houghova transformacija

(12)

## 1. R-tabela

- R-tabelo skonstruiramo v učni fazi za vsako iskano obliko.



## Postopek:

- Izberi referenčno točko  $y$  znotraj iskane oblike.
- FOR vsako točko  $x$  iz meje

Izračunaj smer gradienta  $\psi(x)$ .

$$\mathbf{r} = \mathbf{y} - \mathbf{x}$$

Shrani  $\mathbf{r}$  v R-tabelo v odvisnosti od  $\psi(\mathbf{x})$ .

## Primer: Oblika R-tabele [1]

$i$	$\psi_i$	$R_{\psi_i}$
0	0	$\{\mathbf{r}   \mathbf{y} - \mathbf{r} = \mathbf{x} \wedge \psi(\mathbf{x}) = 0 \wedge \mathbf{x} \text{ je na meji}\}$
1	$\Delta\psi$	$\{\mathbf{r}   \mathbf{y} - \mathbf{r} = \mathbf{x} \wedge \psi(\mathbf{x}) = \Delta\psi \wedge \mathbf{x} \text{ je na meji}\}$
2	$2\Delta\psi$	$\{\mathbf{r}   \mathbf{y} - \mathbf{r} = \mathbf{x} \wedge \psi(\mathbf{x}) = 2\Delta\psi \wedge \mathbf{x} \text{ je na meji}\}$
...	...	...

# **Posplošena Houghova transformacija**

(13)

## Lastnosti R-tabele

Za iskano obliko poznamo njeno R-tabelo, tj.  $R(\psi)$ :

- ▷ Kakšna je nova R-tabela,  $R'(\psi)$ , pri znani transformaciji?
- 1. Skaliranje oblike (objekta) za faktor  $s$

$$R'(\psi) = sR(\psi) \quad (10)$$

- 2. Rotiranje oblike (objekta) za kot  $\varphi$

$$R'(\psi) = \text{Rotacija}\{R[(\psi - \varphi) \bmod 2\pi], \varphi\} \quad (11)$$

- ▷ vse indekse najprej inkrementiramo za  $(-\varphi \bmod 2\pi)$
- ▷ dobljene vektorje  $\mathbf{r}$  pa rotiramo za kot  $\varphi$
- 3. Sprememba referenčne točke iz  $\mathbf{y}$  v  $\mathbf{y}'$

$$R'(\psi) = R(\psi) + (\mathbf{y} - \mathbf{y}')$$

# **Posplošena Houghova transformacija**

(14)

## **2. Iskanje poljubnih oblik**

1. Inicializiraj akumulator  $A$ , tj.  $A(y, s, \varphi) = 0$

2. FOR vsak robni piksel  $x$

    FOR  $s = \min_s : \Delta s : \max_s$

        FOR  $\varphi = \min_\varphi : \Delta\varphi : \max_\varphi$

            Določi  $R'$  po enačbi (10).

            Vstavi  $R'$  v enačbo (11) in določi  $R''$ .

            FOR  $r \in R''(\psi(x))$

$A(x + r, s, \varphi) = A(x + r, s, \varphi) + \Delta A$

3. Poišči lokalne maksimume v akumulatorju  $A$ .

▷ Učinkovite implementacije posplošene HT in njene variante najdemo v [2].

## **C. Inkrementalne strategije in kompenziranje napak**

### **1. Inkrementalne strategije**

- ▷ Če v vsaki iteraciji inkrementiramo akumulator za 1, potem je vsebina akumulatorja proporcionalna obsegu iskane oblike:
  - \* takšna strategija preferira iskanje oblik, kjer je velik del objekta (krivulje) možno detektirati
- ▷ Če želimo detektirati krajše, zelo pomembne dele objekta (npr. v primeru delnega zakrivanja objektov), potem so primernejše alternativne inkrementalne strategije:

$$A(\mathbf{a}) = A(\mathbf{a}) + |\text{grad}(\mathbf{x})|$$

ali pa kombinacija obojega

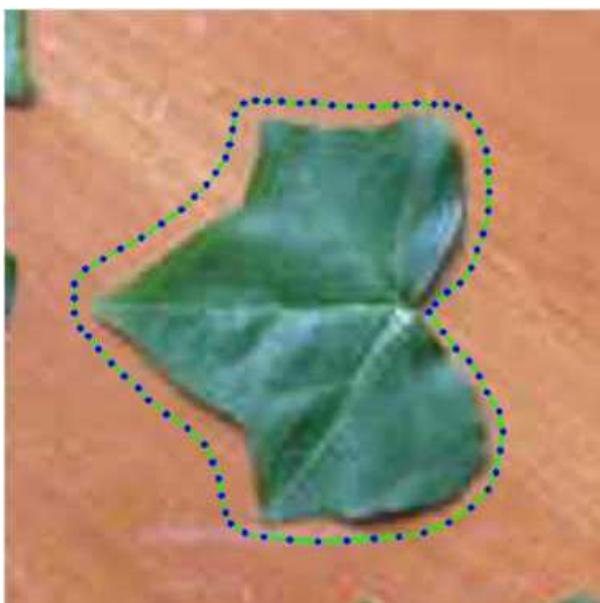
$$A(\mathbf{a}) = A(\mathbf{a}) + |\text{grad}(\mathbf{x})| + \text{konstanta}$$

### **2. Detektiranje lokalnih maksimumov**

- ▷ Pri detektiranju maksimumov v akumulatorju A se lahko pojavijo problemi:
  - \* mnogi faktorji vplivajo na natančnost izračunanega vektorja  $\mathbf{a}$ 
    - ▷ **Posledica:** Ob idealni točki  $\mathbf{a}$  inkrementiramo v akumulatorju še množico lokacij v neposredni okolini te točke!
- ▷ Pokazalo se je [1], da z glajenjem akumulatorja odpravimo nenatančnosti pri izračunu smeri gradiента in vektorja  $\mathbf{r}$ .

## Modeli aktivnih kontur – kače

- ▷ Kača je spremenljiv zlepek (*deformable spline*) na osnovi minimizacije energije, na katero vplivajo omejitve in slikovne sile, ki jo povlečejo proti meji objekta.
- ▷ Kače so poseben primer splošne tehnike **ujemanja spremenljivega modela** (*matching of deformable model*) na sliko na osnovi minimizacije energije.



Začetna kača iz 78 kontrolnih točk.



Končno stanje kače po 128 iteracijah (vir: [4]).

## A. Ideja osnovnega postopka kač

- ▷ Osnovni model kač je **kontroliran nepretrgan zlepak** (*controlled continuity spline*) pod vplivom slikovnih sil in sil zunanjih omejitev (*external constraint forces*):
  - \* **interne sile zlepka** (*internal spline forces*) vnašajo omejitev gladkosti za posamezne odseke (*piecewise smoothness constraint*)
  - \* **slikovne sile** potisnejo kačo proti pomembnim (*salient*) značilnicam:
    - ▷ npr. linijam, robovom, vnaprej pripravljenim konturam
  - \* **sile zunanjih omejitev** potegnejo kačo proti želenim lokalnim minimumom:
    - ▷ npr. lahko jih posreduje uporabnik ali pa so rezultat višje nivojskih interpretacij
- ▷ Položaj kače predstavimo parametrično kot

$$\mathbf{v}(s) = [x(s), y(s)]$$

- \* kjer je parameter  $s$  proporcionalen dolžini krožnega loka, tj.  $s \in [0, 1]$
- ▷ **Energijski funkcional kače,  $E_{\text{snake}}$ , definiramo kot**

$$E_{\text{snake}} = \int_0^1 \left( E_{\text{int}}(\mathbf{v}(s)) + E_{\text{image}}(\mathbf{v}(s)) + E_{\text{con}}(\mathbf{v}(s)) \right) ds \quad (12)$$

- \*  $E_{\text{int}}$  – interna energija zlepka zaradi upogibanja
- \*  $E_{\text{image}}$  – slikovne sile
- \*  $E_{\text{con}}$  – sile zunanjih omejitev

# Modeli aktivnih kontur – kače

(3)

## B. Interna energija

- ▷ Definirana je kot

$$E_{int} = \frac{1}{2} \left( \alpha(s) |\mathbf{v}_s(s)|^2 + \beta(s) |\mathbf{v}_{ss}(s)|^2 \right)$$

kjer

- ★  $\mathbf{v}_s$  – prvi odvod  $\mathbf{v}$  po  $s$  (tj. člen 1. reda)
- ★  $\mathbf{v}_{ss}$  – drugi odvod  $\mathbf{v}$  po  $s$  (tj. člen 2. reda)
- ★  $\alpha$  in  $\beta$  – uteži

- ▷ Člen 1. reda povzroči, da se kača obnaša kot membrana.
- ▷ Člen 2. reda povzroči, da se kača obnaša kot tanek metalni papir (*thin-plate*).
- ▷ Uteži  $\alpha(s)$  in  $\beta(s)$  kontrolirata pomembnost obeh členov:
  - ★ velikokrat:  $\alpha(s) = \alpha$  in  $\beta(s) = \beta$
  - ★ če  $\beta(s) = 0$  pri določenem vozlišču (točki), potem dovolimo, da tam kača ustvari vogal (*corner*).

## C. Slikovne sile

- ▷ To je energijski funkcional, ki priteguje kačo k pomembnim značilnostim v sliki.
- ▷ Skupna slikovna energija je definirana kot utežena kombinacija treh energijskih funkcionalov:

$$E_{image} = w_{line} E_{line} + w_{edge} E_{edge} + w_{term} E_{term} \quad (13)$$

kjer

- ★  $E_{line}$  – energijski funkcional, ki pritegne kačo k linijam
- ★  $E_{edge}$  – energijski funkcional, ki pritegne kačo k robovom
- ★  $E_{term}$  – energijski funkcional, ki pritegne kačo h zaključkom
- ★  $w_{line}, w_{edge}$  in  $w_{term}$  – uteži

### 1. Linijski funkcional

$$E_{line} = I(x, y)$$

- ▷ V odvisnosti od predznaka  $w_{line}$  bomo kačo pritegnili bodisi k svetlim oz. temnim linijam v sliki.

# Modeli aktivnih kontur – kače

(5)

## 2. Robni funkcional

$$E_{edge} = -|\nabla I(x, y)|^2$$

- ▷ V tem primeru kačo pritegujejo konture z velikimi slikovnimi gradienti.

## 3. Funkcional zaključkov

- ▷ Za iskanje zaključkov linijskih segmentov in oglišč se uporablja ukrivljenost (*curvature*) v nivojskih linijah v rahlo zglajeni sliki  $C$ .
- ▷ Zglajena slika je definirana kot

$$C(x, y) = G_\sigma(x, y) * I(x, y)$$

kjer je  $G_\sigma$  Gaussov filter.

- ▷ **Energijski funkcional je definiran kot**

$$E_{term} = \frac{C_{yy}C_x^2 - 2C_{xy}C_xC_y + C_{xx}C_y^2}{(C_x^2 + C_y^2)^{\frac{3}{2}}} \quad (14)$$

kjer so  $C_x$ ,  $C_y$ ,  $C_{xx}$ ,  $C_{yy}$  in  $C_{xy}$  ustrezni parcialni odvodi

- ▷ S kombiniranjem funkcionalov  $E_{edge}$  in  $E_{term}$  lahko kreiramo kače, ki jih pritegnejo robovi oz. zaključki.

## D. Postopek minimizacije energijskega funkcionala kače

- ▷ Postopek je iterativna tehnika, ki uporablja redke matrike, pri čemer je časovna zahtevnost reda  $\mathcal{O}(n)$ .
- ▷ V vsaki iteraciji učinkovito izvedemo implicitni Eulerjev korak v odvisnosti od interne energije ter eksplicitni Eulerjev korak v odvisnosti od slikovne energije in energije zunanjih omejitev.
- ▷ Najprej energijo kače (enačba (12)) diskretiziramo

$$E_{\text{snake}} = \sum_{i=1}^n E_{\text{int}}(i) + E_{\text{ext}}(i)$$

- ★ parcialne odvode določimo s pomočjo razlik
- ★  $i$ -to vozlišče kače  $\mathbf{v}$  pa zapišemo v vektorski notaciji kot

$$\mathbf{v}_i = [x_i, y_i] = [x(ih), y(ih)]$$

- ▷ Energijski funkcional  $E_{\text{int}}$  sedaj zapišemo kot

$$E_{\text{int}}(i) = \alpha_i \frac{|\mathbf{v}_i - \mathbf{v}_{i-1}|^2}{2h^2} + \beta_i \frac{|\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}|^2}{2h^4}$$

kjer  $\mathbf{v}_0 = \mathbf{v}_n$

## Modeli aktivnih kontur – kače (7)

▷ Naj bo  $\mathcal{F}_x(i) = \frac{\partial E_{ext}}{\partial x_i}$  in  $\mathcal{F}_y(i) = \frac{\partial E_{ext}}{\partial y_i}$ .

▷ Pripadajoča Eulerjeva enačba je

$$\begin{aligned} \alpha_i(\mathbf{v}_i - \mathbf{v}_{i-1}) - \alpha_{i+1}(\mathbf{v}_{i+1} - \mathbf{v}_i) + \beta_{i-1}(\mathbf{v}_{i-2} - 2\mathbf{v}_{i-1} + \mathbf{v}_i) \\ - 2\beta_i(\mathbf{v}_{i-1} - 2\mathbf{v}_i + \mathbf{v}_{i+1}) + \beta_{i+1}(\mathbf{v}_i - 2\mathbf{v}_{i+1} + \mathbf{v}_{i+2}) \\ + (\mathcal{F}_x(i), \mathcal{F}_y(i)) = 0 \end{aligned}$$

▷ Kar zapišemo v matrični notaciji kot

$$\begin{aligned} \mathbf{Ax} + \mathcal{F}_x(\mathbf{x}, \mathbf{y}) &= 0 \\ \mathbf{Ay} + \mathcal{F}_y(\mathbf{x}, \mathbf{y}) &= 0 \end{aligned} \tag{15}$$

kjer sta  $\mathbf{x}$  in  $\mathbf{y}$   $n$ -dimenzionalna vektorja  $x$  oz.  $y$  koordinat vozlišč kače.

## Modeli aktivnih kontur – kače

(8)

- ▷ Matrika A je pentadiagonalna matrika, velikosti  $n \times n$ , oblike

$$A = \begin{bmatrix} c & b & a & 0 & 0 & 0 & 0 & \dots & 0 & a & b \\ b & c & b & a & 0 & 0 & 0 & \dots & 0 & a & a \\ a & b & c & b & a & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & a & b & c & b & a & 0 & \dots & 0 & 0 & 0 \\ & & & & & & & \vdots & & & \\ & & & & & & & \vdots & & & \\ 0 & & & & & & & \dots & a & b & c & b & a \\ a & 0 & & & & & & \dots & 0 & a & b & c & b \\ b & a & 0 & & & & & \dots & 0 & 0 & a & b & c \end{bmatrix}$$

kjer

- ★  $a = \beta$
- ★  $b = -(\alpha + 4\beta)$
- ★  $c = 2\alpha + 6\beta$
- ★ Zgoraj navedeno velja ob predpostavki, da  $\alpha_i = \alpha$  in  $\beta_i = \beta$

## Modeli aktivnih kontur – kače (9)

- ▷ Enačbo (15) rešimo kot

$$\begin{aligned}\mathbf{x}_t &= (\mathbf{A} + \gamma \mathbf{I})^{-1}(\gamma \mathbf{x}_{t-1} - \mathcal{F}_x(\mathbf{x}_{t-1}, \mathbf{y}_{t-1})) \\ \mathbf{y}_t &= (\mathbf{A} + \gamma \mathbf{I})^{-1}(\gamma \mathbf{y}_{t-1} - \mathcal{F}_y(\mathbf{x}_{t-1}, \mathbf{y}_{t-1}))\end{aligned}\quad (16)$$

kjer

\*  $\gamma$  – velikost koraka (pogosto  $\gamma = 1$ )

- ▷ Enačba (16) definira spremembo položaja vozlišč kače iz iteracije  $t - 1 \rightarrow t$

### E. Algoritem kač

1. Vnesi začetni približek (krivuljo) iskanega objekta.
2. Enakomerno razporedi  $n$  vozlišč po krivulji.
3. Določi  $E_{ext}$  po enačbi (13).
4. Izračunaj  $\mathcal{F}_x = \frac{\partial E_{ext}}{\partial x}$  in  $\mathcal{F}_y = \frac{\partial E_{ext}}{\partial y}$ .
5. Določi matriko  $\mathbf{A}$ .
6. WHILE (ni izpolnjen konvergenčni pogoj)
  - Določi nov položaj vozlišč kače v iteraciji  $t$  po enačbi (16).
  - $t = t + 1;$
  - Enakomerno razporedi  $n$  vozlišč po krivulji.

## Opombe:

- ▷ V algoritmu imamo vse skozi opravka z  $n$  vozlišči:
  - ★ Če želimo število vozlišč spremeniti, moramo matriko A ponovno določiti!
- ▷ Konvergenčni pogoj:
  - ★ število iteracij, absolutni premik vozlišč  $< T$  itd.
- ▷ Če želimo v posamezni iteraciji prirediti zgolj izbranemu vozlišču določene omejitve, potem moramo prav tako ustreznno ažurirati matriko A.

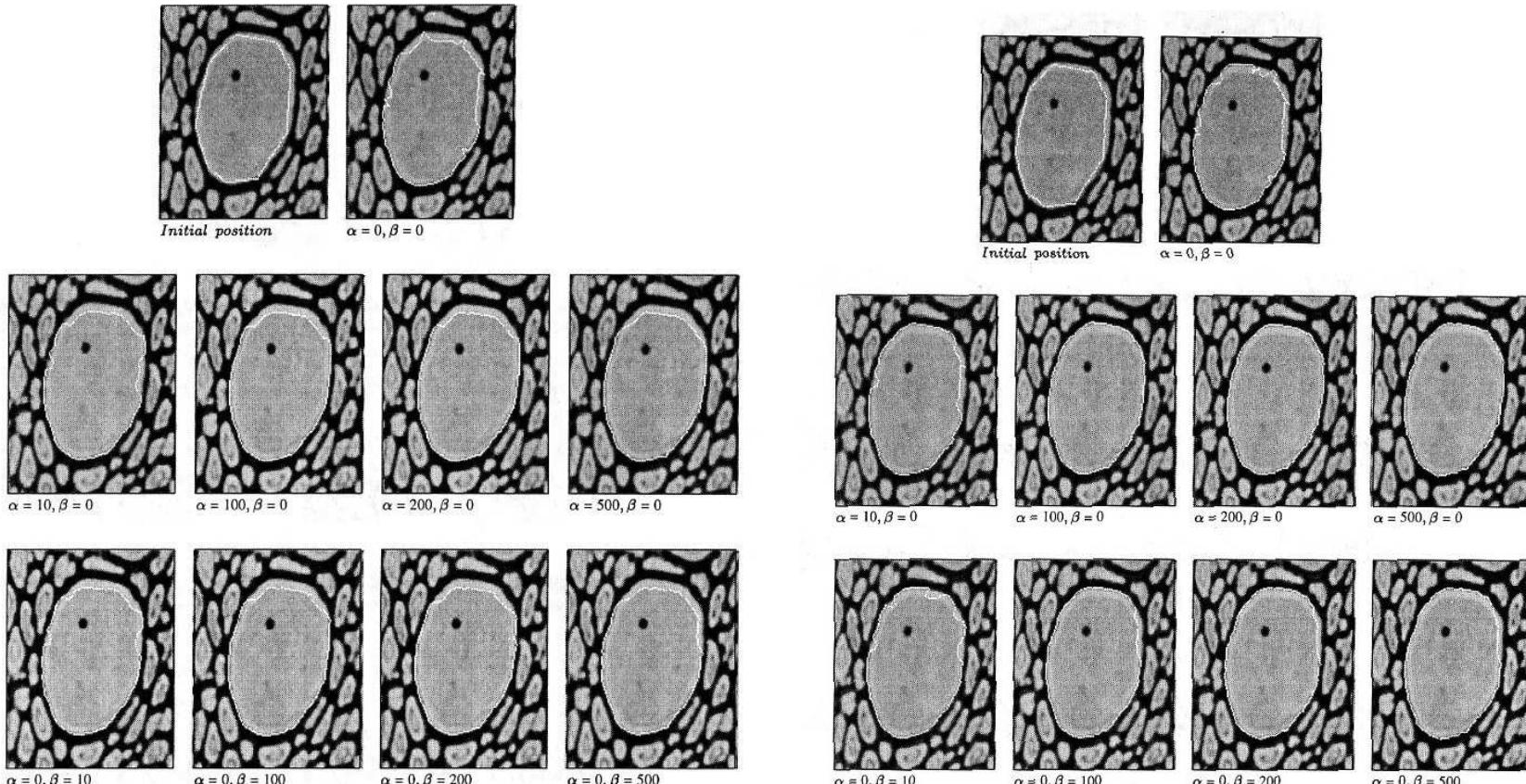
## Problemi:

- ▷ Inicializacija začetnega položaja:
  - ★ ročni posegi, rezultat drugih metod...
- ▷ Določitev ustreznih vrednosti za parametre ( $\alpha, \beta, w_{line}$  in  $w_{edge}...$ )
  - ★ Parametri imajo ogromen vpliv na končni rezultat!

# Modeli aktivnih kontur – kače

(11)

**Primer:** Segmentiranje mikroskopske slike lesa (vir: [5])



- ▷ Začetni položaj 1
- ▷ Variiranje parametrov  $\alpha$  in  $\beta$
- ▷ Pravilne segmentacije ni bilo možno doseči!
- ▷ Začetni položaj 2
- ▷ Variiranje parametrov  $\alpha$  in  $\beta$
- ▷ Dosegli pravilno segmentacijo!

## **Modeli aktivnih kontur – kače**

**(12)**

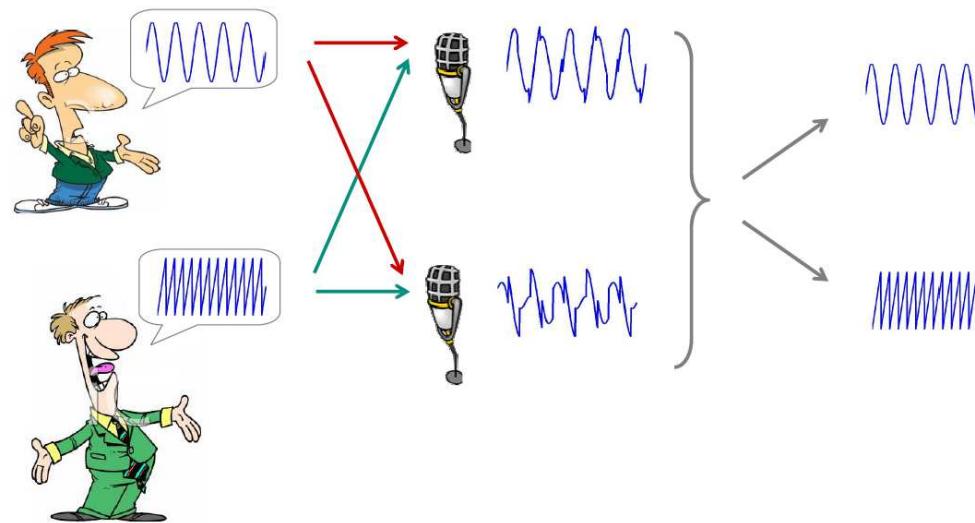
- ▷ Možne naprednejše aplikacije:
  - ★ sledenje počasnemu gibanju objekta skozi zaporedje slik
  - ★ iskanje ujemanja v stereo slikah
    - ▷ Predlagan je dodatni energijski funkcional:

$$E_{stereo} = (\mathbf{v}^L(s) - \mathbf{v}^R(s))^2$$

- ▷ Variante kač [4]:
  - ★ kače GVF (*gradient vector flow*)
  - ★ kače baloni (*balloon snake*)
  - ★ difuzijske kače (*diffusion snakes*)

# Analiza neodvisnih komponent

- ▷ Analiza neodvisnih komponent (*independent component analysis – ICA*) je splošno namenska metoda obdelave signalov in analize podatkov:
  - ★ v osnovi je bila razvita za reševanje problemov, ki so sorodni **problemu zabave s koktajli**
- ▷ **Zabava s koktajli (cocktail-party problem):**
  - ★ Problem: V sobi imamo več govorcev in več mikrofonov.
  - ★ Cilj: Izluščiti glas individualnega govorca od preostalih glasov, na osnovi vseh signalov, posnetih z mikrofoni.



Vir: [6]

- ▷ Metoda ICA izvaja linearno projekcijo na neodvisne komponente.

# **Analiza neodvisnih komponent** (2)

## **A. Matematična formulacija metode**

- ▷ Opazujemo  $n$  linearnih mešanic,  $x_1(t), x_2(t), \dots, x_n(t)$ , ki sestojijo iz  $n$  neodvisnih komponent  $s_i(t)$ :

$$x_j(t) = a_{j1}s_1(t) + a_{j2}s_2(t) + \cdots + a_{jn}s_n(t) \quad \forall j$$

- ▷ V nadaljevanju indeks  $t$  izpustimo ter predpostavimo, da je vsaka mešanica  $x_j$  in vsaka neodvisna komponenta  $s_k$  naključna spremenljivka
  - \*  $x_j(t)$  – opazovana vrednost, npr. signal na mikrofonu, v problemu zabave s koktajli, je vzorec te naključne spremenljivke
- ▷ Predpostavka: spremenljivke mešanice, kakor tudi neodvisne komponente imajo ničelno povprečje (*zero mean*)
  - \* Če to ne drži, lahko od opazovanih spremenljivk vedno odštejemo povprečje vseh vzorcev, s čimer ga centriramo in naredimo model z ničelnim povprečjem.

## Analiza neodvisnih komponent (3)

- ▷ Mešalni model (*mixing model*) zapišemo v matrični notaciji:

$$\underbrace{\mathbf{X}}_{n \times N} = \underbrace{\mathbf{A}}_{n \times n} \underbrace{\mathbf{S}}_{n \times N} \quad (17)$$

kjer

- ★  $n$  – število opazovanih spremenljivk in neodvisnih komponent
- ★  $N$  – število vzorcev (v spremenljivki)
- ★  $\mathbf{A}$  – mešalna matrika
- ★ teorijo je možno razviti tudi v primeru, kadar se števili opazovanih spremenljivk,  $n$ , in število neodvisnih komponent,  $p$ , razlikujeta (tipično:  $n > p$ )

- ▷ Model, podan z enačbo (17), se imenuje analiza neodvisnih komponent oz. model ICA.
- ▷ Model ICA je generativen model: opisuje, kako se opazovani podatki generirajo v procesu mešanja neodvisnih komponent  $s_i$ .
- ★ neodvisne komponente so latentne oz. prikrite spremenljivke (*latent variables*), ki jih ni možno direktno opazovati.
  - ★ mešalna matrika  $\mathbf{A}$  ni znana.

## Analiza neodvisnih komponent

(4)

- ▷ Rešujemo naslednji problem:
  - ★ na osnovi opazovanja linearnih mešanic  $X$ , ocenujemo tako mešalno matriko  $A$ , kot neodvisne komponente  $S$
- ▷ Brez vpeljave določenih predpostavk ta problem ni rešljiv!
- ▷ Običajno sprejmemo naslednje predpostavke:
  1. komponente  $s_i$  so statistično neodvisne
  2. neodvisne komponente ne smejo imeti Gaussovo porazdelitev
  3. neznana mešalna matrika je kvadratna (to omejitev je možno tudi omiliti)
- ▷ Po ocenitvi matrike  $A$  najprej izračunamo njen inverz  $W$ , neodvisne komponente pa določimo kot

$$S = WX \tag{18}$$

- ▷ Metoda ICA je tesno povezana z **metodo slepega ločevanja izvorov oz. signalov (blind source separation – BSS)**:
  - ★ "izvor" – originalen signal oz. neodvisna komponenta
  - ★ "slepo" – vemo zelo malo (lahko tudi nič) o mešalni matriki in naredimo majhne predpostavke o izvornih signalih
  - ★ Je najbolj pogosto uporabljena metoda za izvajanje slepega ločevanja izvorov.

# **Analiza neodvisnih komponent**

**(5)**

## **B. Nejasnosti oz. nedoločljivosti v metodi ICA**

1. Ne moremo določiti variance (energije) neodvisnih komponent.
  - ▷ V praksi: Neodvisne komponente so naključne spremenljivke, za katere predpostavimo, da imajo varianco enako 1.
    - ★ Ta omejitev se nato upošteva pri adaptiranju matrike A.
  - ▷ **Kljud zgorjni predpostavki ostane še nejasnost predznaka v rešitvi:**
    - ★ neodvisne komponente lahko pomnožimo z -1, ne da bi vplivali na model
2. Ne moremo določiti točnega vrstnega reda neodvisnih komponent:
  - ▷ vsako od neodvisnih komponent lahko obravnavamo kot prvo!

## **Ocenjevanje neodvisnosti oz. princip ocenitve ICA**

- ▷ Ključ k ocenjevanju modela ICA je "ne Gaussov" (*nongaussianity*).
  - ★ **Vemo:** brez predpostavke o ne Gaussovi porazdelitvi ocenjevanje niti ni možno.

## Mere za merjenje "ne Gaussov"

- ▷ Naj bo naključna spremenljivka  $y$  z ničelnim povprečjem ter varianco enako 1.

### 1. Kurtozis

$$kurt(y) = E\{y^4\} - 3(E\{y^2\})^2 \stackrel{\text{pred.}}{=} E\{y^4\} - 3$$

- ▷ Za Gaussove naključne spremenljivke je kurtozis enak 0, za večino ostalih pa je različen od 0.
  - ★ Kurtozis ni dovolj robustna mera – zelo občutljiva na osamelce (*outliers*).

### 2. Negentropija

- ▷ Negentropija (*negentropy*) temelji na informacijsko-teoretični vrednosti (diferencialne) entropije.
- ▷ Entropijo naključne spremenljivke interpretiramo kot stopnjo informacije, ki jo daje opazovanje spremenljivke.
  - ★ Bolj "naključna", tj. nepredvidljiva in nestrukturirana je spremenljivka, večja je entropija.
  - ★ **Rezultat informacijske teorije [7]: Gaussova spremenljivka ima največjo entropijo med vsemi naključnimi spremenljivkami z isto varianco.**
- ▷ Negentropija je vedno ne negativna in je 0 tedaj in le tedaj, ko ima  $y$  Gausovo porazdelitev.

## **Analiza neodvisnih komponent**

(7)

- ▷ Določiti negentropijo je računsko izredno zahtevno, zato v praksi raje uporabljamo njene približke.
- ▷ Primer približka negentropije,  $J(y)$  [7]:

$$J(y) = [E\{\mathcal{G}(y)\} - E\{\mathcal{G}(v)\}]^2$$

kjer

- ★  $\mathcal{G}$  – poljubna nekvadratična funkcija
- ★  $v$  – Gaussova spremenljivka z ničelnim povprečjem in varianco enako 1

- ▷ Zelo primerni sta se izkazali naslednji funkciji  $\mathcal{G}$ :

$$\mathcal{G}(u) = \frac{1}{a} \log \cosh au$$

$$\mathcal{G}(u) = -e^{-\frac{u^2}{2}}$$

kjer

- ★  $a$  – konstanta iz intervala  $[1, 2]$

## C. Predobdelava za metodo ICA

- ▷ Cilj predobdelave: Narediti problem ocenitve ICA preprostejši in boljše pogojen.

### 1. Centriranje

- ▷ Centriramo opazovane vrednosti v  $X$ :
  - \* odštejemo povprečni vektor, tj.  $\mathbf{m} = E\{X\}$
  - \* dejansko od vsake vrstice matrike  $X$  odštejemo povprečno vrednost
    - ▷ po centriranju  $X$  vsebuje spremenljivke z ničelnim povprečjem
    - ▷ tudi originali v  $S$  postanejo spremenljivke z ničelnim povprečjem
- ▷ Po ocenitvi mešalne matrike  $A$  s centriranimi podatki, prištejemo k centriranim ocenam v  $S$  še povprečen vektor  $A^{-1}\mathbf{m}$ .

### 2. Beljenje (*whitening*)

- ▷ Centrirane opazovane vektorje nato še linearно transformiramo tako, da postanejo "beli":
  - \* tj. njihove komponente so nekorelirane, njihova varianca pa je 1.
  - \* Kovariančna matrika transformiranih podatkov je enaka identiteti:
    - ▷ kovariančno matriko določimo tako, da izračunamo kovarianco med vsakim parom signalov oz. vrstic matrike  $X$  (Matlab: `cov(X', 1)`).

## **Analiza neodvisnih komponent**

(9)

- ▷ Beljenje izvedemo z dekompozicijo na lastne vrednosti kovariančne matrike netransformiranih podatkov:

$$E\{\mathbf{x}\mathbf{x}^T\} = \mathbf{E}\mathbf{D}\mathbf{E}^T$$

kjer

- ★ E – ortogonalna matrika lastnih vektorjev
- ★ D – diagonalna matrika pripadajočih lastnih vrednosti

- ▷ Beljenje opravimo z množenjem opazovanih vektorjev X s faktorjem  $\mathbf{D}^{-\frac{1}{2}}\mathbf{E}^T$ , tj.

$$\tilde{\mathbf{X}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{E}^T\mathbf{X}$$

- ▷ Po analizi ICA pa opravimo še t.i. razbeljenje (*dewhitening*) dobljene mešalne matrike s faktorjem  $\mathbf{E}\mathbf{D}^{\frac{1}{2}}$ , tj.

$$\mathbf{A} = \mathbf{E}\mathbf{D}^{\frac{1}{2}}\tilde{\mathbf{A}}$$

## D. Algoritem ICA

### 1. Postopek za en stolpec inverzne mešalne matrike W

1. Naključno določi stolpec  $w$  (dimenzije  $n \times 1$ ).
2. Izračunaj

$$w^+ = E\{X\mathcal{G}'(X^T w)\} - E\{\mathcal{G}''(X^T w)\}w$$

3.  $w = w^+ / \|w^+\|$
4. IF  $((w^T w - 1) < \epsilon)$  Konec; ELSE Korak 2

Opombe:

- ▷ Podatki morajo biti centrirani in razbeljeni!
- ▷  $\mathcal{G}'$  in  $\mathcal{G}''$  sta prvi in drugi odvod funkcije  $\mathcal{G}$ , s katero ocenujemo negentropijo.
- ▷  $\epsilon$  – majhno pozitivno število
- ▷  $E\{X\mathcal{G}'(X^T w)\}$  izračunamo:
  - ★ najprej določimo  $X\mathcal{G}'(X^T w)$ , s čimer dobimo stolpec dimenzije  $n \times 1$
  - ★ vsak element stolpca zatem delimo s številom vzorcev (tj. z  $N$ )

# **Analiza neodvisnih komponent** (11)

## **2. Postopek za celotno inverzno mešalne matrike $\tilde{W}$**

1. FOR vsak stolpec matrike  $\tilde{W}$

Izvedi postopek za en stolpec (točka 1).

2. Dekoreliraj vrstice v matriki  $\tilde{W}$ , npr. kot

$$\tilde{W} = \tilde{W} / \sqrt{\|\tilde{W}\tilde{W}^T\|}$$

REPEAT

$$\tilde{W} = \frac{3}{2}\tilde{W} - \frac{1}{2}\tilde{W}\tilde{W}^T\tilde{W}$$

UNTIL rezultat ne konvergira

3. Določi originalno mešalno matriko A z razbeljenjem:

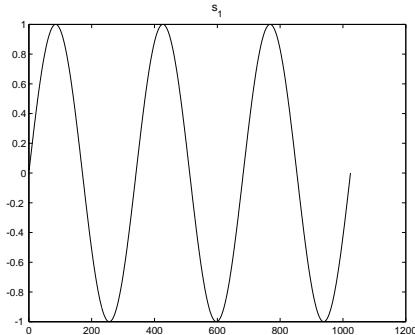
$$A = ED^{\frac{1}{2}}\tilde{W}^{-1}$$

Opombe:

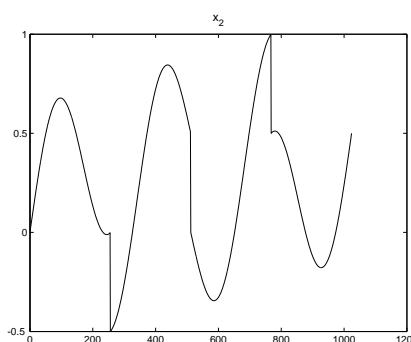
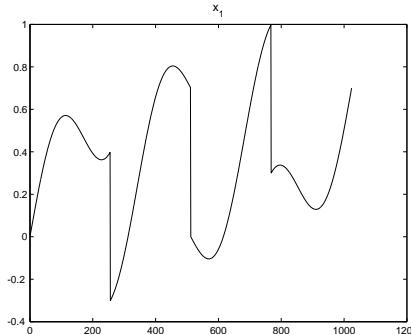
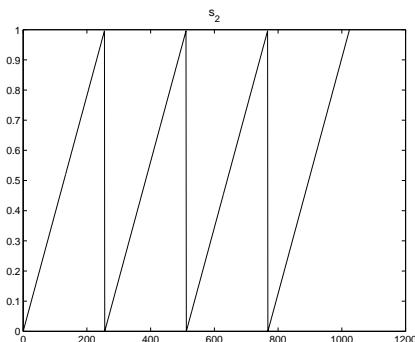
- ▷ Matrika  $\tilde{W}$ , ki jo določimo z zgornjim postopkom je inverzna mešalna matrika za predobdelane podatke!

## E. Aplikacije metode ICA

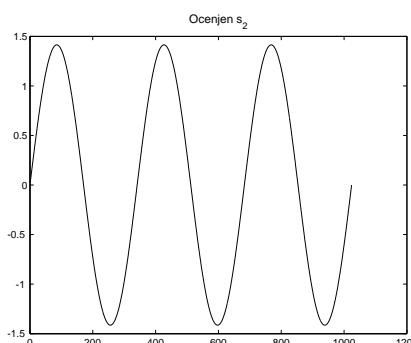
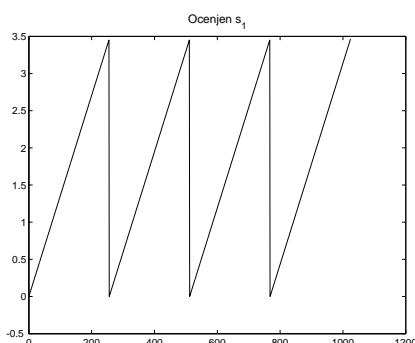
### 1. Problem zabave s koktajli



$$A = \begin{bmatrix} 0,3 & 0,7 \\ 0,5 & 0,5 \end{bmatrix}$$



↓ ICA



# Analiza neodvisnih komponent

(13)

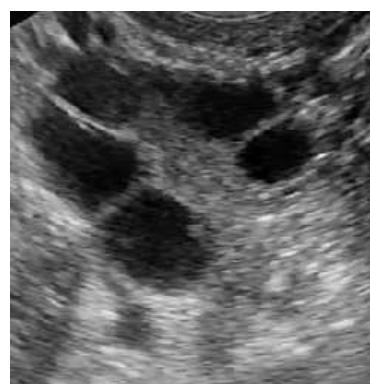
## 2. Ločevanje mešanice slik



$$A = \begin{bmatrix} 0,35 & 0,65 \\ 0,6 & 0,4 \end{bmatrix}$$

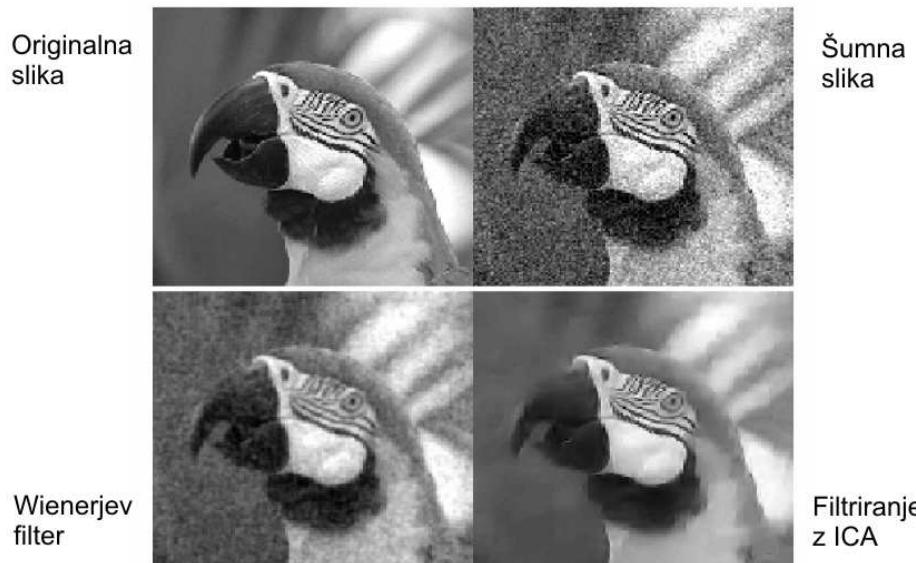


↓ ICA



prikaz: skaliranje na  
interval [0, 255]

## 3. Odstranjevanje šuma



Vir: [6, 7]

▷ **Ideja:**

1. Na brezšumni množici podatkov določi transformacijo  $W$ .
2. Oceni množico parametrov.
3. Izvedi transformacijo  $W$  na šumnih podatkih.
4. Nad dobljenim rezultatom apliciraj operator krčenja (*shrinkage operator*).
5. Rezultat na koncu transformiraj s  $W^T$  v prostor originalnih spremenljivk.

▷ Različice metode ICA za odstranjevanje šuma so opisane v [8].

## 4. Določanje značilnic: Primer razpoznavanja obrazov [9]



Originali.

$\Downarrow$  ICA



Baza neodvisnih slik.

---


$$\text{Face Image} = b_1 \text{ Component 1} + b_2 \text{ Component 2} + \dots + b_n \text{ Component n}$$

- ▷ Vektor značilnic  $\mathbf{b} = [b_1, b_2, \dots, b_n]$  – uporabimo za razpoznavanje

## **Dodatna literatura**

- [1] D.H. Ballard (1981), Generalizing the Hough transform to detect arbitrary shapes, Pattern recognition, vol. 13, str. 111–122.
- [2] A.A. Kassim, T. Tan, K.H. Tan (1999), A comparative study of efficient generalised Hough transform techniques, Image and vision computing, vol. 17, str. 737–748.
- [3] M. Kass, A. Witkin, D. Terzopoulos (1987), Snakes: Active contour models, International journal of computer vision, vol. 1, št. 4, str. 321–331.
- [4] N.P. Tiilikainen, A comparative study of active contour snakes,  
<ftp://ftp.diku.dk/diku/image/publications/nikolas.070901.pdf>
- [5] A. Klemenčič, S. Kovačič, A. Leonardis (1995), Redundant initialization and selection of active contour models for image segmentation, V: Zborniku 19 konference OAGM, Maribor, Slovenija, str. 240–248.
- [6] M. Journee (2008), Matlab project: Independent component analysis,  
[http://www.inma.ucl.ac.be/absil/Grenoble2008/Grenoble\\_Matlab\\_project.pdf](http://www.inma.ucl.ac.be/absil/Grenoble2008/Grenoble_Matlab_project.pdf)
- [7] A. Hyvarinen, E. Oja (2000), Independent Component Analysis: Algorithms and applications, Neural networks, vol. 13, št. 4–5, str. 411–430.
- [8] P. Anjali, S. Ajay, S.D. Sapre (2010), A review on natural image denoising using independent component analysis (ica) technique, Advances in computational research, vol. 2, št. 1, str. 6–14.
- [9] A. Rajgarhia (2007), Face detection using independent component analysis,  
<http://cs229.stanford.edu/proj2007/Rajgarhia-FaceDetectionUsingICA.pdf>

## ***5. Slikovne transformacije***

---

2D Fourierova transformacija

2D Valčna transformacija

# 2D Fourierova transformacija

## A. Ponovitev: 1D signali

- ▷ Fourierova ugotovitev:
  - ★ vsak signal je možno predstaviti kot vsoto sinusov, ki imajo različne amplitude, frekvence in fazne zamike
  - ★ dejansko gre za zapis signala v novem vektorskem prostoru, kjer so bazni vektorji sinus
- ▷ Fourierova transformacija (FT) transformira signal iz časovnega prostora v frekvenčni prostor!
  - ★ Časovna predstavitev (prostor):
    - ▷ neodvisna spremenljivka  $t$  (čas)
    - ▷  $x = x(t)$
  - ★ Frekvenčna predstavitev (prostor):
    - ▷ neodvisna spremenljivka  $\omega$  (krožna frekvenca;  $\omega = 2\pi f$ )
    - ▷  $\mathbb{X} = \mathbb{X}(\omega)$
    - ▷ vsak signal je popolnoma opredeljen z naborom trojic  $(A, f, \varphi)$

## 2D Fourierova transformacija (2)

### 1D zvezna FT

- ▷ Transformacijski par:

$$\mathbb{X}(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-i\omega t} dt$$

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} \mathbb{X}(\omega)e^{i\omega t} d\omega$$

kjer

- ★  $i = \sqrt{-1}$  – imaginarna enota
- ★  $e^{-i\omega t} = \cos(\omega t) - i \sin(\omega t)$

- ▷ **Oznaka za Fourierovo transformacijo:  $\mathcal{F}[\cdot]$  oz.  $\mathcal{F}^{-1}[\cdot]$ .**

### 1D diskretna FT

- ▷ Transformacijski par:

$$\mathbb{X}(k) = \sum_{n=0}^{N-1} x(n)e^{-i\frac{2\pi}{N}kn}, \quad k = 0, 1, \dots, N-1$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} \mathbb{X}(k)e^{i\frac{2\pi}{N}kn}, \quad n = 0, 1, \dots, N-1$$

## 2D Fourierova transformacija (3)

### B. 2D Zvezna FT

- ▷ Gre za posplošitev FT za 2D signale (tj. slike).
- ▷ Zvezna slika  $\mathcal{I}$  je funkcija dveh neodvisnih spremenljivk (slikovnih koordinat):

$$\mathcal{I} = \mathcal{I}(x, y)$$

- ▷ FT opravi dekompozicijo slikovne funkcije s pomočjo linearne kombinacije na ortonormalne funkcije (t.i. harmonične funkcije).
- ▷ Transformacijski par:

$$\mathcal{F}[\mathcal{I}(x, y)] = \mathbb{I}(u, v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \mathcal{I}(x, y) e^{-i2\pi(xu+yv)} dx dy$$

$$\mathcal{F}^{-1}[\mathbb{I}(u, v)] = \mathcal{I}(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \mathbb{I}(u, v) e^{i2\pi(xu+yv)} du dv$$

kjer

- ★  $u$  in  $v$  – prostorski frekvenci (*spatial frequency*)
- ★  $e^{i2\pi(xu+yv)}$  – preprost periodični vzorec

## 2D Fourierova transformacija (4)

### C. 2D Diskretna FT

▷ Transformacijski par:

$$\mathbb{I}(k_1, k_2) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I(m, n) e^{-i2\pi(\frac{mk_1}{M} + \frac{nk_2}{N})} \quad (19)$$

$$k_1 = 0, 1, \dots, M - 1$$

$$k_2 = 0, 1, \dots, N - 1$$

$$I(m, n) = \sum_{k_1=0}^{M-1} \sum_{k_2=0}^{N-1} \mathbb{I}(k_1, k_2) e^{i2\pi(\frac{mk_1}{M} + \frac{nk_2}{N})}$$

$$m = 0, 1, \dots, M - 1$$

$$n = 0, 1, \dots, N - 1$$

## 2D Fourierova transformacija (5)

### Možna implementacija 2D FT

- ▷ Enačbo (19) zapišimo kot

$$\mathbb{I}(k_1, k_2) = \frac{1}{M} \sum_{m=0}^{M-1} \left( \frac{1}{N} \sum_{n=0}^{N-1} \mathbb{I}(m, n) e^{-\frac{i2\pi nk_2}{N}} \right) e^{-\frac{i2\pi mk_1}{M}}$$

$$k_1 = 0, 1, \dots, M - 1$$

$$k_2 = 0, 1, \dots, N - 1$$

- ▷ **Opazimo:**

- ★ izraz v oklepaju ustreza 1D FT  $m$ -te vrstice
- ★ ta izračun lahko opravimo s hitro Fourierovo transformacijo – FFT (ob predpostavki, da je  $N = 2^v$ )
- ★ vsako vrstico zamenjamo z njeno Fourierovo transformiranko
- ★ zatem pa opravimo še 1D DFT nad vsakim stolpcem

## 2D Fourierova transformacija (6)

### D. Interpretacija 2D DFT

- ▷ Fourierova transformacija realne funkcije (kar slikovna funkcija vsekakor je!) je kompleksna funkcija:

$$\mathbb{I}(k_1, k_2) = \operatorname{Re}(k_1, k_2) + i\operatorname{Im}(k_1, k_2)$$

kjer

- ★  $\operatorname{Re}(k_1, k_2)$  – realni del in  $\operatorname{Im}(k_1, k_2)$  – imaginarni del

- ▷ Na področju obdelave slik ima interpretacija slikovnega Fourierovega spektra zelo pomembno vlogo.
- ▷ **Frekvenčni spekter (*frequency spectrum*)**
  - ★ za enak pomen najdemo še izraze: **amplitudni spekter, funkcija velikosti**

$$|\mathbb{I}(k_1, k_2)| = \sqrt{\operatorname{Re}^2(k_1, k_2) + \operatorname{Im}^2(k_1, k_2)}$$

- ▷ **Fazni spekter (*phase spectrum*)**

$$\varphi(k_1, k_2) = \operatorname{arctg} \left( \frac{\operatorname{Im}(k_1, k_2)}{\operatorname{Re}(k_1, k_2)} \right)$$

- ▷ **Močnostni spekter (*power spectrum*)** oz. spektralna gostota

$$P(k_1, k_2) = |\mathbb{I}(k_1, k_2)|^2 = \operatorname{Re}^2(k_1, k_2) + \operatorname{Im}^2(k_1, k_2)$$

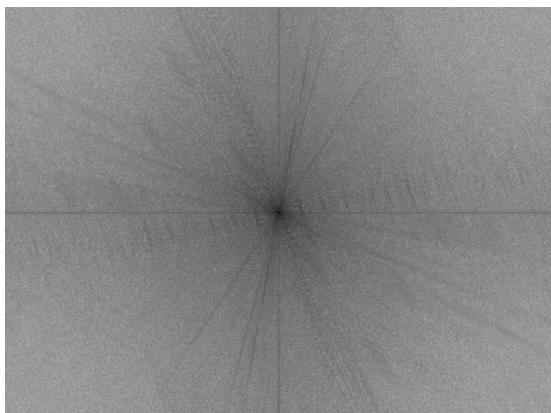
## 2D Fourierova transformacija

(7)

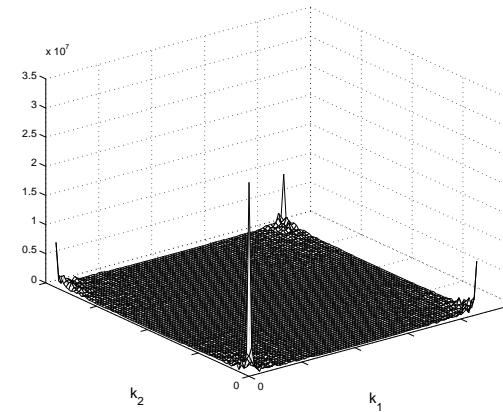
Primer: Transformacija slike v Fourierov prostor



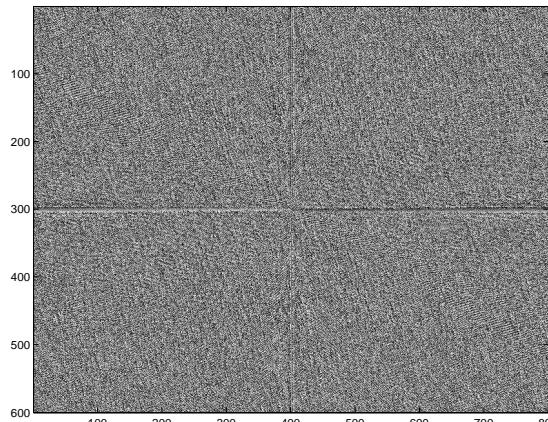
Original –  
 $600 \times 800$  piksov



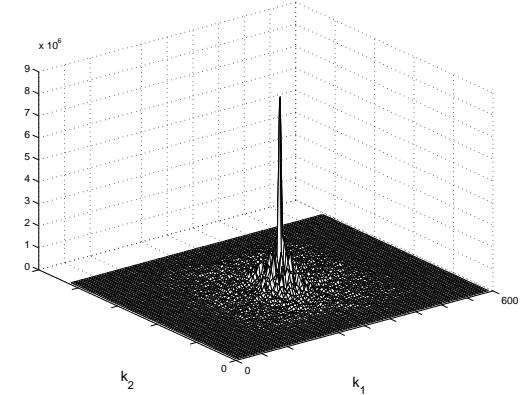
Frekv. spekter kot intenziteta  
(log., skal. in neg. spekter)



Frekvenčni spekter –  
izhodišče v  $(0,0)$



Fazni spekter



Frekvenčni spekter –  
izhodišče v  $(300,400)$

- Uporabljene funkcije: `fft2()`, `fftshift()`, `abs()`, `angle()`, `log()`, `mesh()`

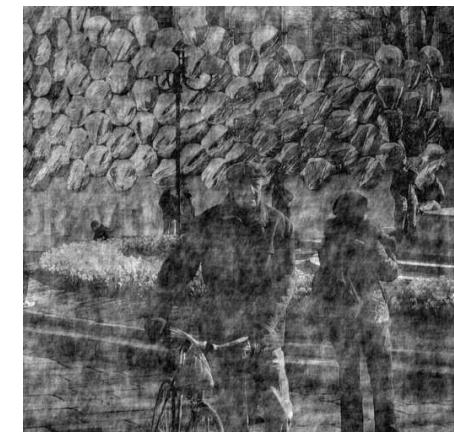
## 2D Fourierova transformacija (8)

**Kaj nosi več informacije o strukturi slike: amplituda ali faza? (vir: [3])**

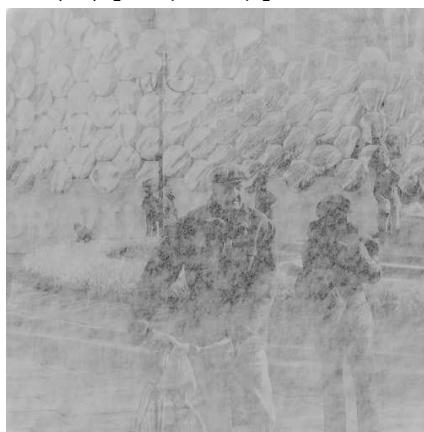
- ▷ Fourierova transformiranka v polarni obliki:

$$\mathbb{I}(k_1, k_2) = |\mathbb{I}(k_1, k_2)| e^{i\varphi(k_1, k_2)}$$

- ★ faza – relativni položaj periodične strukture v sliki.



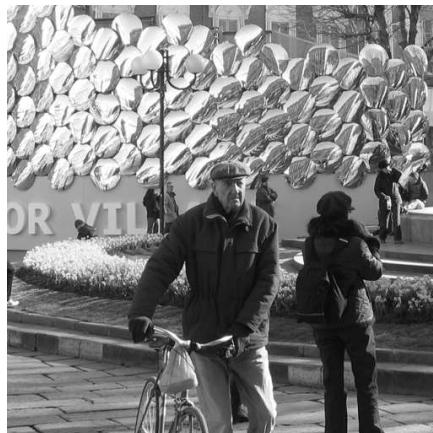
Slika 1;  $[\mathbb{I}(k_1, k_2), \varphi(k_1, k_2)]$      $|\mathbb{I}'| = |\mathbb{I}| [U(0, 1)]^{1/2}; \varphi' = \varphi$      $|\mathbb{I}'| = |\mathbb{I}| [U(0, 1)]^2; \varphi' = \varphi$



$$\varphi' = \varphi [U(0, 1)]^{1/2}; |\mathbb{I}'| = |\mathbb{I}| \quad \varphi' = \varphi [U(0, 1)]^2; |\mathbb{I}'| = |\mathbb{I}|$$

## 2D Fourierova transformacija

(9)



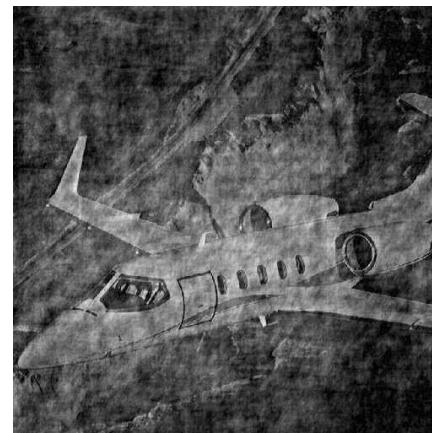
Slika 1



Slika 2



Amplituda od slike 2  
faza od slike 1



Amplituda od slike 1  
faza od slike 2

▷ **Ugotovitev:**

- ★ Faza FT nosi bistveno informacijo o strukturi slike.
- ★ **Informacija o fazi je ključna za človekovo percepцијо!**

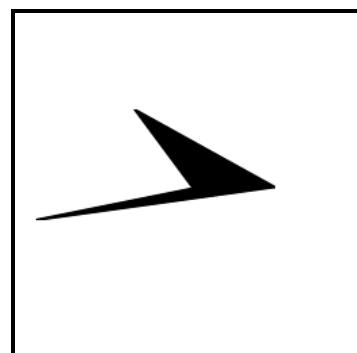
## E. Lastnosti 2D DFT [1]

### 1. Linearnost

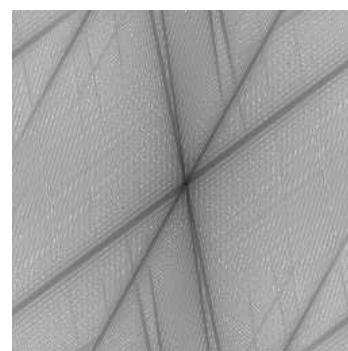
$$\mathcal{F}[a\mathbb{I}_1(m, n) + b\mathbb{I}_2(m, n)] = a\mathbb{I}_1(k_1, k_2) + b\mathbb{I}_2(k_1, k_2)$$

### 2. Premik v slikovni ravnini

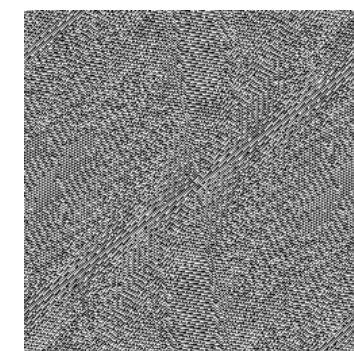
$$\mathcal{F}[\mathbb{I}(m - a, n - b)] = \mathbb{I}(k_1, k_2)e^{-i2\pi(\frac{ak_1}{M} + \frac{bk_2}{N})}$$



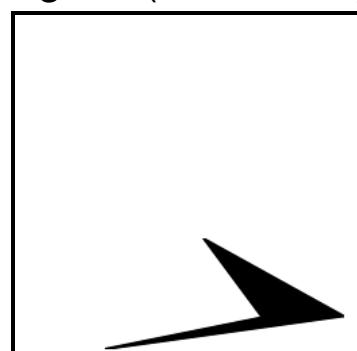
Original ( $256 \times 256$ )



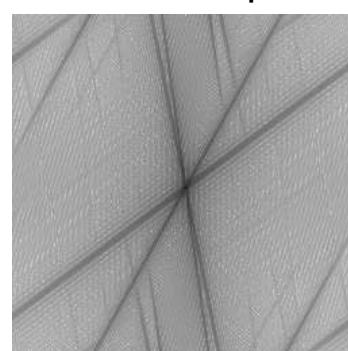
Frekvenčni spekter



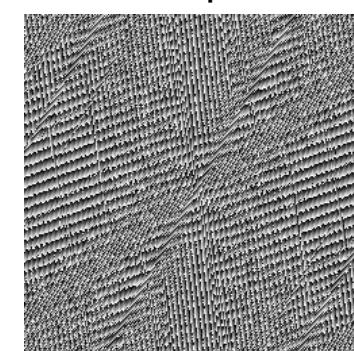
Fazni spekter



Po translaciji



Frekvenčni spekter



Fazni spekter

# 2D Fourierova transformacija

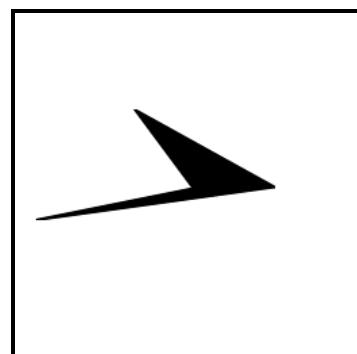
(11)

## 3. Premik v frekvenčni domeni

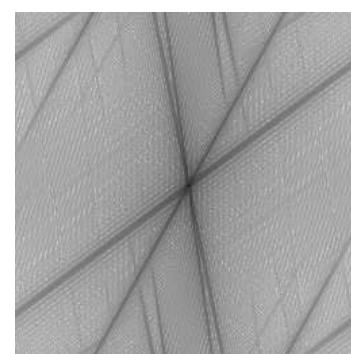
$$\mathbb{I}(k_1 - a, k_2 - b) = \mathcal{F} \left[ I(m, n) e^{i2\pi(\frac{am}{M} + \frac{bn}{N})} \right]$$

## 4. Skaliranje

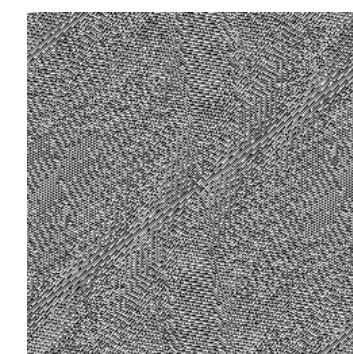
$$\mathcal{F}[I(am, bn)] = \frac{1}{|ab|} \mathbb{I}\left(\frac{k_1}{a}, \frac{k_2}{b}\right)$$



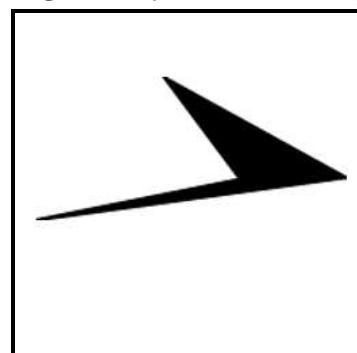
Original ( $256 \times 256$ )



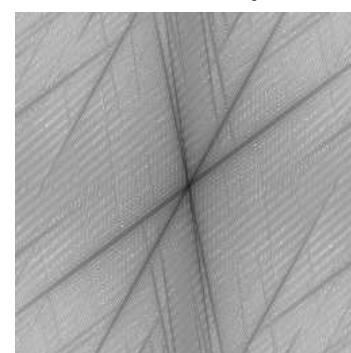
Frekvenčni spekter



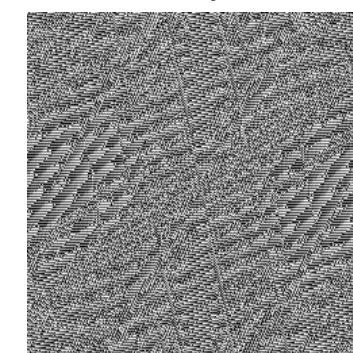
Fazni spekter



Po skaliranju



Frekvenčni spekter



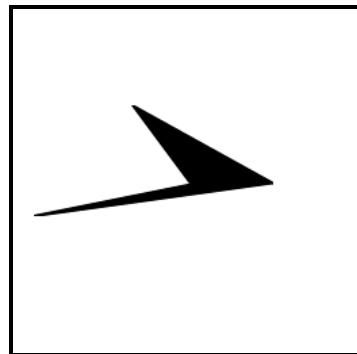
Fazni spekter

# 2D Fourierova transformacija

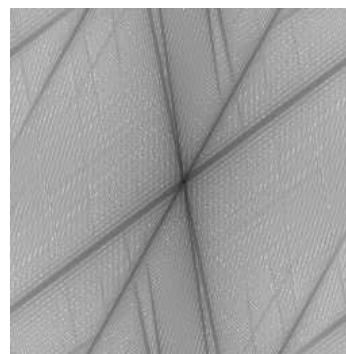
(12)

## 5. Rotacija

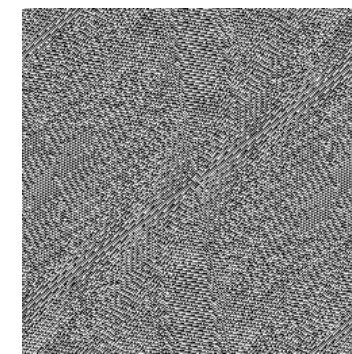
$$\mathcal{F}[\mathbf{I}(m \cos \theta + n \sin \theta, -m \sin \theta + n \cos \theta)] = \mathbf{I}(k_1 \cos \theta + k_2 \sin \theta, -k_1 \sin \theta + k_2 \cos \theta)$$



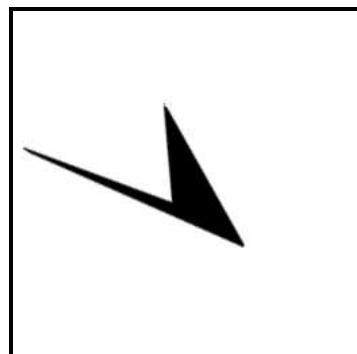
Original (256 × 256)



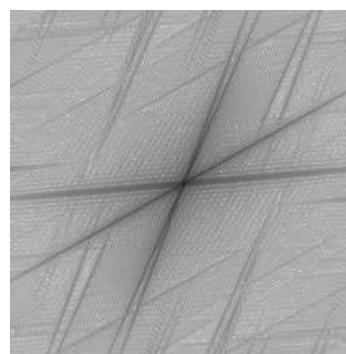
Frekvenčni spekter



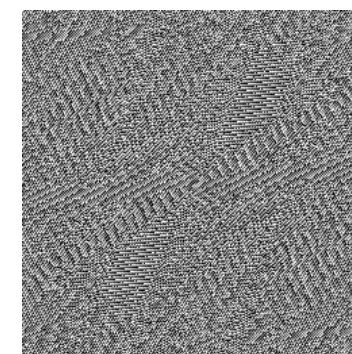
Fazni spekter



Po rotaciji



Frekvenčni spekter



Fazni spekter

# **2D Fourierova transformacija** (13)

## **6. Periodičnost**

- ▷ Predpostavimo, da je slikovna funkcija periodična, tj.

$$\mathbf{I}(-m, n) = \mathbf{I}(M - m, n) \quad \text{oz.} \quad \mathbf{I}(m, -n) = \mathbf{I}(m, N - n)$$

v splošnem

$$\mathbf{I}(aM + m, bN + n) = \mathbf{I}(m, n)$$

- ▷ potem je tudi Fourierova transformiranka periodična

$$\mathbb{I}(aM + k_1, bN + k_2) = \mathbb{I}(k_1, k_2)$$

## **7. Konvolucija**

- ▷ V diskretnem prostoru:

$$\mathbf{G}(m, n) = \mathbf{H}(m, n) * \mathbf{I}(m, n) = \mathbf{I}(m, n) * \mathbf{H}(m, n) = \sum_{a=0}^{M-1} \sum_{b=0}^{N-1} \mathbf{I}(a, b) \mathbf{H}(m-a, n-b)$$

- ▷ V frekvenčnem diskretnem prostoru:

$$\mathbb{G}(k_1, k_2) = \mathbb{I}(k_1, k_2) \mathbb{H}(k_1, k_2)$$

\* Gre za množenje po elementih in ne za matrično množenje!

# 2D Fourierova transformacija

(14)

Primer:



$I(1944 \times 2592)$

$$\frac{1}{121} \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$$

$H(11 \times 11)$

$$\downarrow$$
  
$$\boxed{H}^{11}_{11} \quad \emptyset$$

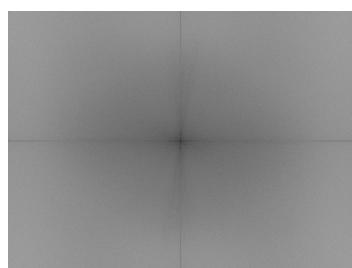
$H(1944 \times 2592)$

Konvolucija  
(diskretni  
prostor)

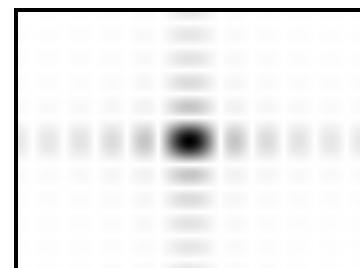


$G(1944 \times 2592)$

$\Downarrow \mathcal{F}$

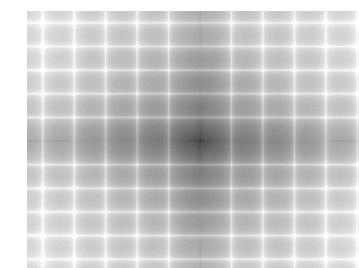


$|\mathbb{I}(k_1, k_2)|$



$|\mathbb{H}(k_1, k_2)|$

Konvolucija  
(frekvenčni  
prostor)



$|\mathbb{G}(k_1, k_2)|$

$\Updownarrow \mathcal{F}^{-1}$

# 2D Fourierova transformacija

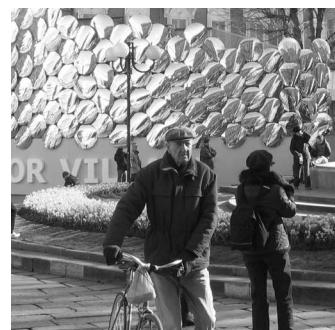
(15)

## F. Aplikacije 2D DFT

### 1. Filtriranje slik

- ▷ Področja uporabe: odstranjevanje različnih vrst šuma, poudarjanje določenih lastnosti slik (npr. robov), druge operacije predobdelave slik...

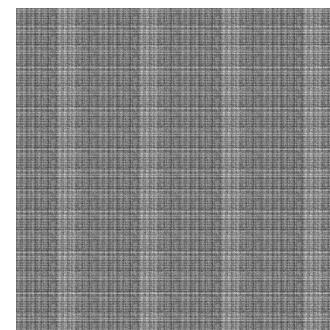
#### Primer: Pomen frekvence



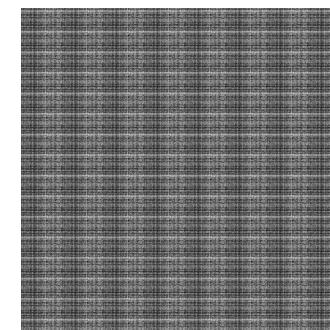
Original



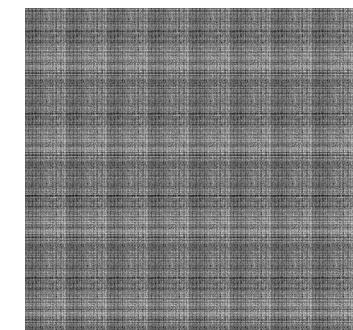
Enosmerna k. –  $\tilde{I}$



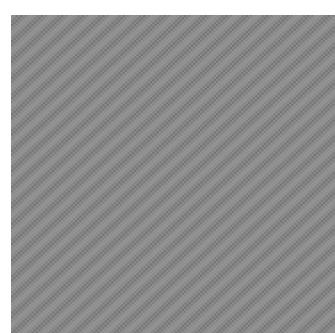
(1 frekvenca)



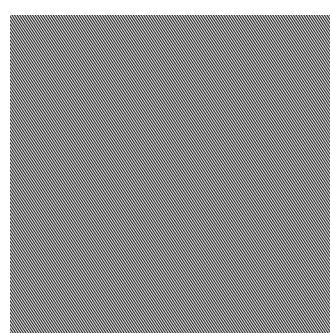
(1)



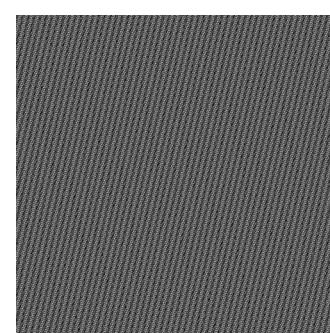
(1)



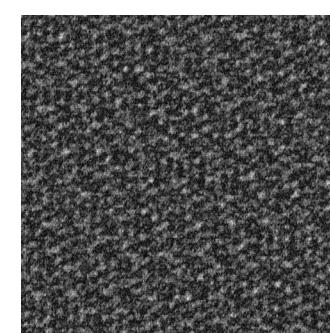
(2)



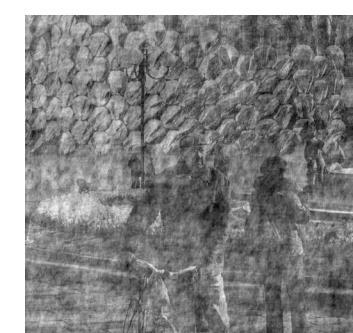
(2)



(10)



(997)



(114224)

## 2D Fourierova transformacija

(16)

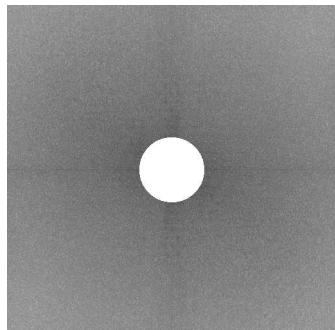
Primer: Različni filtri



Original



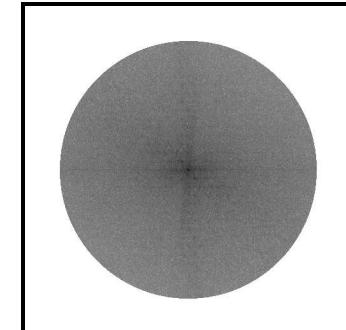
Nizko sito



Frek. spekter



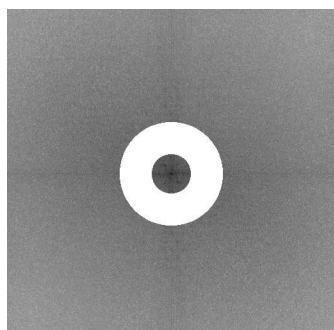
Visoko sito



Frek. spekter



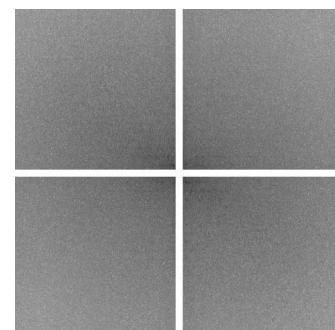
Pasovno p. sito



Frek. spekter



Križ



Frek. spekter

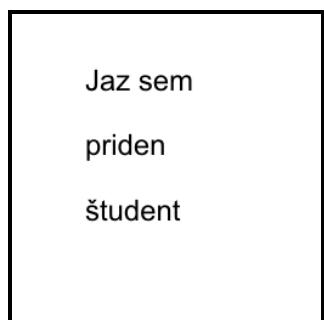
# 2D Fourierova transformacija

(17)

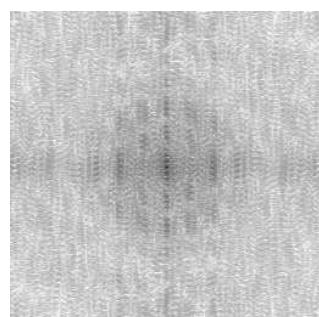
## 2. Ujemanje (iskanje) šablon in (križna) korelacija [2]

- ▷ Ujemanje oz. iskanje šablon v sliki lahko enostavno izvedemo tudi v frekvenčnem prostoru (ujemanje šablon v diskretnem prostoru, glej zapiske predavanj URVRV, poglavje 4)
- ▷ **Postopek:**
  1.  $\mathbb{I}(k_1, k_2) = \mathcal{F}[\mathbf{I}(m, n)]$  in  $\mathbb{H}(k_1, k_2) = \mathcal{F}[\mathbf{H}(m, n)]$
  2.  $\mathbf{R}(m, n) = \mathcal{F}^{-1}[\bar{\mathbb{I}}(k_1, k_2)\mathbb{H}(k_1, k_2)]$
  3. (Lokalni) maksimum v  $R$  določa položaj šablone  $H$  v sliki  $I$ .
    - \* oznaka  $\bar{\mathbb{I}}$  označuje konjugirano kompleksno število

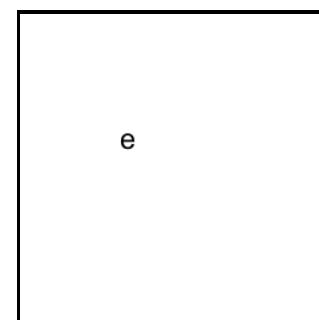
### Primer:



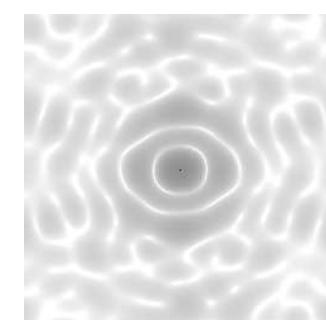
Original



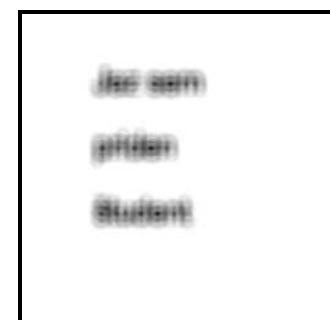
Frek. spekter



Šablon



Frek. spekter



Po korelaciji

- ▷ Ideja uporabna tudi pri računanju parametrov toge poravnave!

## 3. Merjenje orientacije in razmika periodične strukture v sliki [2]

- ▷ Naj v sliki obstaja periodična struktura, potem
  - \* vrh v močnostnem spektru ustreza iskani periodični strukturi
  - \* vrh je za določen radij oddaljen od izhodišča (0,0)
    - ▷ ta radij ustreza razmiku med ponovitvami periodične strukture
    - ▷ smer ustreza orientaciji periodične strukture

## 4. Popravljanje zamegljenih slik oz. obnavljanje slik (*image restoration*)

- ▷ Gre za popravljanje defektov med zajemom slik:
  - \* kamera ni bila ustrezeno fokusirana (*out-of-focus*)
  - \* zameglitev slike zaradi gibanja

## 5. Drugo

- ▷ izločanje značilk, kompresija slik ipd.

## 2D Valčna transformacija

### Zakaj potrebujemo valčno transformacijo (*wavelet transformation – WT*)?

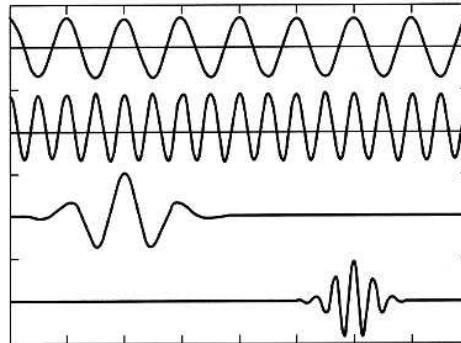
- ▷ FT ni primerna za analizo signalov, ki imajo časovno-spremenljivo frekvenco (**nestacionarni signali**)
  - \* zgolj informacija, če določena frekvenčna komponenta obstaja ali ne
  - \* ta informacija je neodvisna od tega, kje v času se komponenta pojavi
- ▷ Če potrebujemo časovno lokalizacijo spektralnih komponent, potem moramo aplicirati transformacijo, ki vrne časovno-frekvenčno predstavitev signala (*time-frequency representation*).
- ▷ Pri kratko-časovni FT (*short term Fourier transform*) signal razdelimo na kratke odseke, kjer bi naj signal bil stacionaren
  - \* za razdelitev signala uporabljamo okensko funkcijo
  - \* **Posledica:**
    - ▷ ne moremo točno vedeti, katere spektralne komponente obstajajo ob določenih časovnih trenutkih (princip nedoločenosti)
    - ▷ poznamo zgolj časovne intervale, v katerih obstaja določen pas frekvenč (problem ločljivosti)
  - \* **Velja:**
    - ▷ okno neskončne dolžine  $\Rightarrow$  FT
    - ▷ ozko okno  $\Rightarrow$  dobra časovna ločljivost ter slaba frekvenčna ločljivost
    - ▷ široko okno  $\Rightarrow$  slaba časovna ločljivost ter dobra frekvenčna ločljivost

## 2D Valčna transformacija

(2)

### A. Valčna transformacija

- ▷ Je transformacija, ki ima bazne funkcije omejenega trajanja:
  - \* pri FT se sinusi razširjajo v neskončnost
  - \* pri WT pa bazne funkcije variirajo v položaju, kakor tudi v frekvenci:
    - ▷ so valovi omejenega trajanja, imenovani tudi valčki (*wavelets*)



Valova in valčka (vir: [4])

- ▷ STFT daje fiksno ločljivost pri vseh časih, med tem ko WT daje spremenljivo ločljivost, in sicer:
  - \* višje frekvence so boljše razložene v času
  - \* nižje frekvence pa so boljše razložene v frekvenci

## 2D Valčna transformacija (3)

### B. 1D Zvezna valčna transformacija

- ▷ 1D zvezna valčna transformacija (CWT) signala  $x(t)$  v odvisnosti od valčka  $\psi(t)$  je definirana kot

$$W_x(a, b) = \langle x, \psi_{a,b} \rangle = \int_{-\infty}^{\infty} x(t) \psi_{a,b}(t) dt$$

- ▷ kjer

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) \quad (20)$$

pri čemer sta  $a$  in  $b$  realni števili in  $a > 0$ .

- ★ spremenljivka  $a$  odraža merilo (*scale*) oz. dolžino izbrane bazne funkcije.
- ★ spremenljivka  $b$  pa specificira pomik bazne funkcije vzdolž osi  $t$ .
- ★ **Na nek način velja:  $a$  ustreza frekvenci,  $b$  pa času**
  - ▷ s translacijo valčka v času je signal lokaliziran v času, s spremenjanjem  $a$  pa je lokaliziran v merilu (frekvenci).
- ▷ Vidimo: Koeficiente WT dobimo kot notranji produkt funkcije  $x$  z vsako bazno funkcijo.
- ▷ Množico valčnih baznih funkcij  $\{\psi_{a,b}(t)\}$  tvorimo s translacijo in skaliranjem osnovnega valčka  $\psi(t)$ .

## 2D Valčna transformacija

(4)

### Merilo

- ▷ Nizke frekvence (tj. visoka merila) ustrezano globalni informaciji o signalu
- ▷ Visoke frekvence (tj. nizka merila) ustrezano detajlnejši informaciji o skritem vzorcu v signalu (le-ta običajno traja kratek čas)
- ▷ V enačbi (20) je člen  $\frac{1}{\sqrt{a}}$ :
  - \* uporablja se za normalizacijo energije, tj. vsak transformirani signal bo pri vsakem merilu imel enako energijo

### Osnovni oz. prototipni valček $\psi(t)$ :

- ▷ katerikoli pasovno-prepustni (*bandpass*) filterski impulzni odziv z ničelnim povprečjem
- ▷ s povečevanjem frekvence mora padati dovolj hitro k 0

### Inverzna 1D zvezna valčna transformacija:

$$x(t) = \frac{1}{C_\psi} \int_0^\infty \int_{-\infty}^\infty W_x(a, b) \psi_{a,b}(t) db \frac{da}{a^2}$$

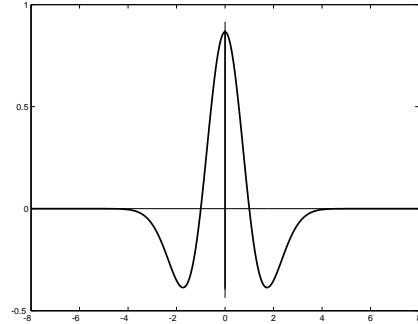
kjer

$$C_\psi = \int_{-\infty}^\infty \frac{|\psi(t)|}{|t|} dt$$

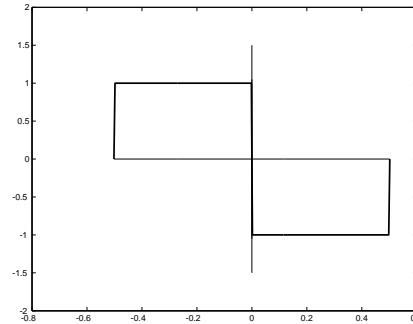
## 2D Valčna transformacija

(5)

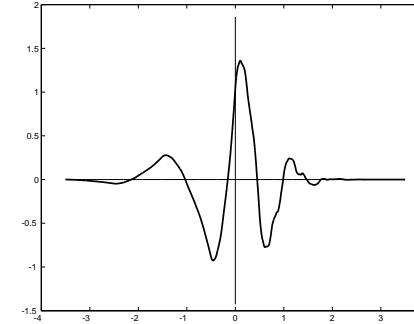
### Primer: Prototipni valčki



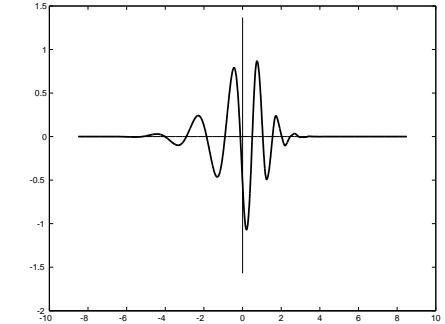
Mehiški klobuk



Haar



Daubechies (4)



Daubechies (9)

- ▷ Definicija CWT nakazuje:
  - ★ valčna analiza je mera podobnosti med baznimi funkcijami (valčki) in signalom
  - ★ gre za podobnost v smislu podobne frekvenčne vsebine
  - ★ koeficienti CWT ustrezajo bližini (podobnosti) signala k valčku pri določenem merilu.
- ▷ Kadar računamo CWT z diskretnimi računalniki, potem parametra  $a$  in  $b$  inkrementiramo po majhnem koraku:
  - ★ to ustreza diskretizaciji časovno-merilne ravnine
  - ★ **Diskretizirana oz. vzorčena 1D CWT ni diskretna WT!**

## B. 1D Diskretna valčna transformacija

- ▷ Transformacijski par:

$$\begin{aligned} c_{j,k} &= \sum_n x(n) \psi_{j,k}(n) & j &= 0, 1, \dots, \log_2(N) - 1 \\ x(n) &= \sum_j \sum_k c_{j,k} \psi_{j,k}(n) & k &= 0, 1, \dots, 2^j - 1 \end{aligned} \quad (21)$$

- ▷  $\psi_{j,k}(n)$  – množica ortonormalnih valčkov

$$\psi_{j,k}(n) = 2^{j/2} \psi(2^j n - k)$$

- ▷  $\psi(n)$  – osnovni valček

$$\psi(n) = \sum_k h_1(k) \phi(2n - k)$$

- ▷  $h_1(k)$  – enotin odziv diskretnega visokega sita

$$h_1(k) = (-1)^k h_0(-k + 1)$$

- ▷  $h_0$  – enotin odziv diskretnega nizkega sita (mora izpolnjevati določene pogoje!)
- ▷  $\phi(n)$  – skalirna funkcija, ki jo določimo iz  $h_0$  kot

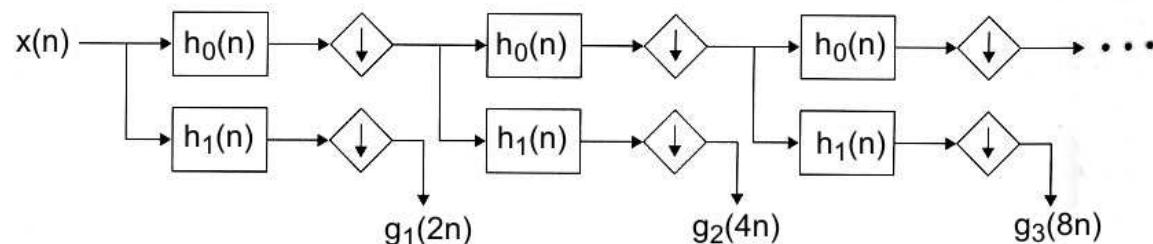
$$\phi(n) = \sum_k h_0(k) \phi(2n - k)$$

## **2D Valčna transformacija**

(7)

## Algoritme FWT (*fast wavelet transform herringbone algorithm*)

- ▷ DWT lahko implementiramo direktno (enacba (21)) ali s hitro valcno transformacijo FWT.
  - ▷ Dolzina signala  $x(n)$  mora biti potenca stevila 2, tj.  $N = 2^{\vartheta}$ .



## Algoritmem FWT (vir: [4])

## Razlaga za 1. nivo dekompozicije:

- ▷ Signal  $x(n)$  spustimo skozi nizko sito  $h_0$ , ki ima lomno frekvenco na polovici celotnega frekvenčnega pasu signala ( $f_{max}$  – maksimalna frekvenca v signalu).
  - ▷ Signal  $x(n)$  pa spustimo tudi skozi visoko sito  $h_1$ , ki ima lomno frekvenco prav tako na polovici celotnega frekvenčnega pasu signala.
  - ▷ Zaradi Shannonovega teorema o vzorčenju (Nyquistova frekvenca vzorčenja!) lahko oba dobljena signala podvzorčimo oz. decimiramo s faktorjem 2 (tj. izpustimo vsak drugi vzorec).

## 2D Valčna transformacija (8)

- ▷ Rezultat tega postopka pri visokem situ  $h_1$  vrne  $\frac{N}{2}$  koeficientov DWT na 1. nivoju:
  - \* ti koeficienti predstavljajo signal v frekvenčnem območju  $[\frac{f_{max}}{2}, f_{max}]$
- ▷ Rezultat tega postopka pri nizkem situ  $h_0$  pa vrne signal iz  $\frac{N}{2}$  vzorcev, ki predstavljajo signal v območju  $[0, \frac{f_{max}}{2}]$ :
  - \* ta podsignal posredujemo v 2. nivo dekompozicije, kjer opisani postopek ponovimo (filtriranje, podvzorčenje, dobimo  $\frac{N}{4}$  koeficientov DWT na 2. nivoju, ki predstavljajo signal v območju  $[\frac{f_{max}}{4}, \frac{f_{max}}{2}]$  ter podsignal iz  $\frac{N}{4}$  vzorcev iz pasu  $[0, \frac{f_{max}}{4}]$ , ki ga posredujemo v 3. nivo)
- ▷ Postopek zaključimo po  $\vartheta = \log_2(N)$  nivojih dekompozicije.
- ▷ Rezultat algoritma FWT je  $N$  koeficientov DWT, pri čemer koeficiente zlagamo od najvišjega dekompozicijskega nivoja proti najnižnjemu.
- ▷ Po vsakem nivoju dekompozicije:
  - \* časovna ločljivost se razpolovi – le polovica vzorcev še okarakterizira celoten signal,
  - \* frekvenčna ločljivost se podvoji – pas frekvenc zaobjema le še polovico prejšnjega pasu, s čimer se nejasnost v frekvenci zreducira za polovico.
- ▷ Na podoben način pridemo do IFWT, kjer namesto podvzorčenja uporabimo nadvzorčenje (*upsampling*), dobljena podsignala pa seštejemo ter ju pošljemo na naslednji nivo rekonstrukcije.

## 2D Valčna transformacija (9)

### D. 2D Zvezna valčna transformacija

- ▷ Zvezna slika  $\mathcal{I}(x, y)$  – funkcija dveh neodvisnih spremenljivk
- ▷ 2D zvezna valčna transformacija (2D CWT) je definirana kot

$$W_x(a, b_x, b_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathcal{I}(x, y) \psi_{a, b_x, b_y}(x, y) dx dy$$

kjer

- ★  $b_x$  in  $b_y$  specificirata translacijo v dveh dimenzijah
- ★  $\psi_{a, b_x, b_y}(x, y)$  je 2D valček definiran kot

$$\psi_{a, b_x, b_y}(x, y) = \frac{1}{|a|} \psi\left(\frac{x - b_x}{a}, \frac{y - b_y}{a}\right)$$

- ▷ Obstaja ekstaktna formula za izračun inverzne 2D CWT [4].

## 2D Valčna transformacija (10)

### E. 2D Diskretna valčna transformacija

- ▷ Idejo 1D DWT enostavno posplošimo v 2D.
- ▷ 2D skalirna funkcija mora biti ločljiva (*separable*), tj. produkt dveh 1D skalirnih funkcij:

$$\phi(x, y) = \phi(x)\phi(y)$$

- ▷ Če je  $\psi(x)$  pripadajoč valček, potem tvorimo tri 2D osnovne valčke:

$$\psi^{(1)}(x, y) = \phi(x)\psi(y), \quad \psi^{(2)}(x, y) = \psi(x)\phi(y), \quad \psi^{(3)}(x, y) = \psi(x)\psi(y)$$

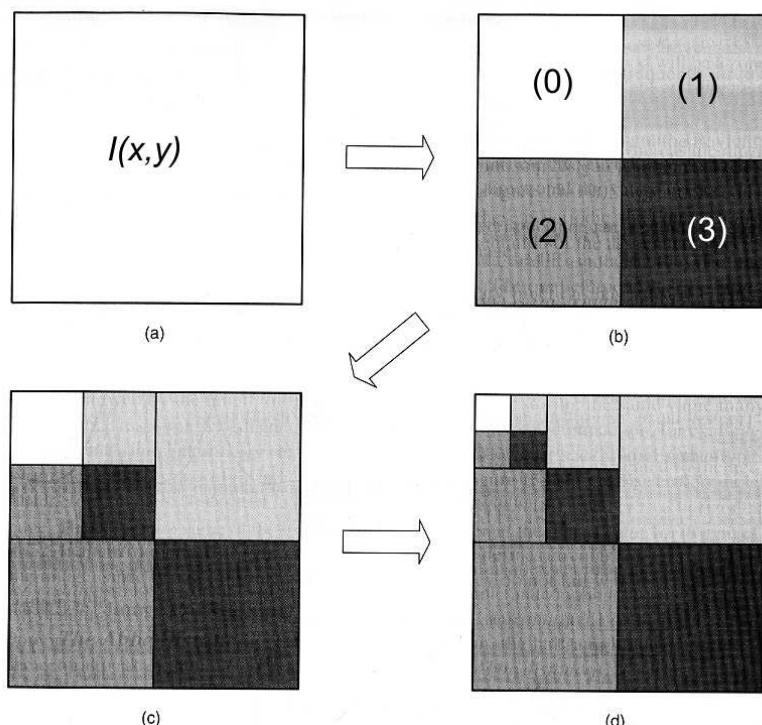
★ nadpisana številka v oklepaju pomeni indeks in ne potence!

- ▷ Zgornji trije valčki tvorijo temelj za 2D DWT – z njimi tvorimo ortonormalno bazo:

$$\{\psi_{j,m,n}^{(l)}\} = \{2^j \psi^{(l)}(x - 2^j m, y - 2^j n)\} \quad j \geq 0, l = 1, 2, 3, m, n \in \mathbb{N}$$

## Algoritem dekompozicije slike

- ▷ Sliko na vsakem nivoju transformacije razčlenimo na štiri podslike, velikosti četrtine originalne slike:
  - ★ Vsako podsliko dobimo z notranjim produktom slike in ene od valčnih baznih slik, čemur sledi podvzorčenje v smeri  $x$  in  $y$  s faktorjem 2.



2D DWT: a) original, b) prvi, c) drugi  
in tretji nivo (vir: [4]).

- ▷ Za prvi nivo (tj.  $j = 1$ ) dobimo:

$$\begin{aligned}
 I_2^{(0)}(m, n) &= \langle I_1(x, y), \phi(x - 2m, y - 2n) \rangle \\
 I_2^{(1)}(m, n) &= \langle I_1(x, y), \psi^{(1)}(x - 2m, y - 2n) \rangle \\
 I_2^{(2)}(m, n) &= \langle I_1(x, y), \psi^{(2)}(x - 2m, y - 2n) \rangle \\
 I_2^{(3)}(m, n) &= \langle I_1(x, y), \psi^{(3)}(x - 2m, y - 2n) \rangle
 \end{aligned}$$

## 2D Valčna transformacija

(12)

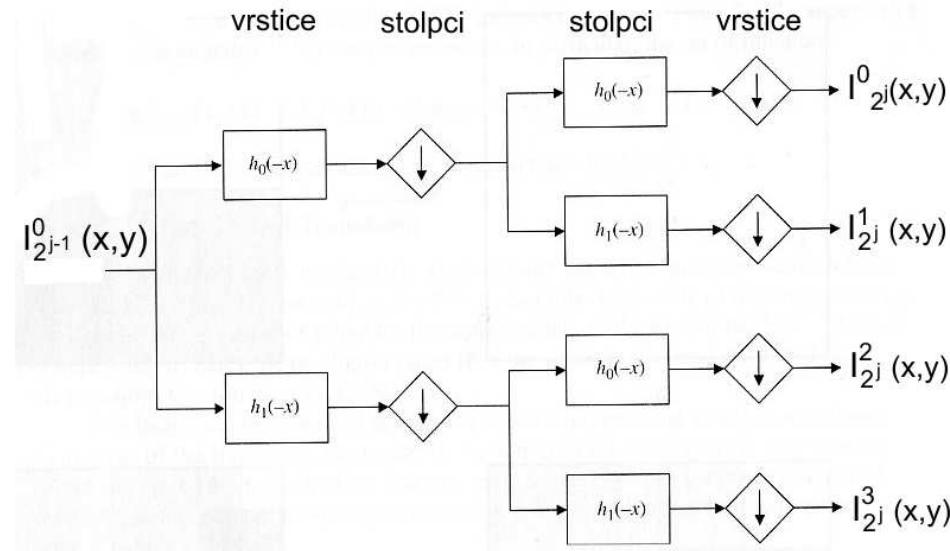
- ▷ V naslednjih nivojih, kjer  $j > 1$ , sliko  $I_{2^j}^{(0)}(x, y)$  razčlenimo na enak način:
  - \* kot rezultat dobimo štiri manjše podslike pri merilu  $2^{j+1}$ .
- ▷ Pri vsakem merilu velja:
  - \*  $I_{2^j}^{(0)}(x, y)$  – vsebuje informacije o nizkih frekvencah iz prejšnjega nivoja
  - \*  $I_{2^j}^{(1)}(x, y)$  – vsebuje informacije o horizontalnih robovih
  - \*  $I_{2^j}^{(2)}(x, y)$  – vsebuje informacije o vertikalnih robovih
  - \*  $I_{2^j}^{(3)}(x, y)$  – vsebuje informacije o diagonalnih robovih
- ▷ Postopek dekompozicije se za sliko, velikosti  $N \times N$  pikslov, zaključi v  $J$  korakih, kjer  $J \leq \log_2(N)$ .

### Hitra implementacija 2D DWT:

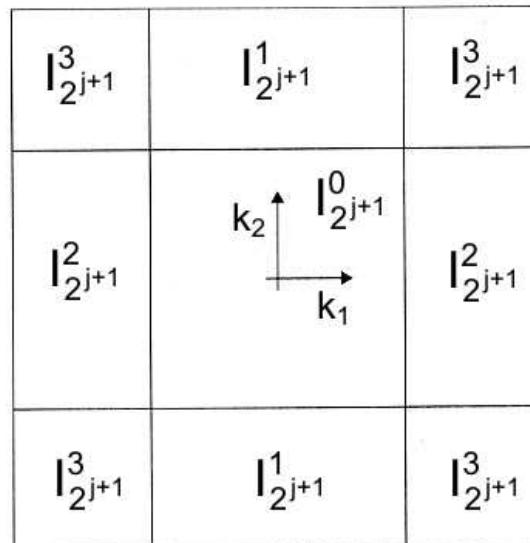
- ▷ skalirne in valčne funkcije so ločljive
- ▷ dekompozicijo implementiramo z 1D konvolucijo po vrsticah oz. stolpcih ter podvzorčenjem po stolpcih oz. vrsticah (detajli v [4])
- ▷ Obstaja tudi hiter postopek za inverzno 2D DWT, s katero rekonstruiramo originalno sliko z majhno degradacijo [4].

## 2D Valčna transformacija

(13)



Dekompozicijski korak DWT pri hitri implementaciji (vir: [4]).



Dekompozicija DWT v frekvenčni domeni (vir: [4]).

# 2D Valčna transformacija

(14)

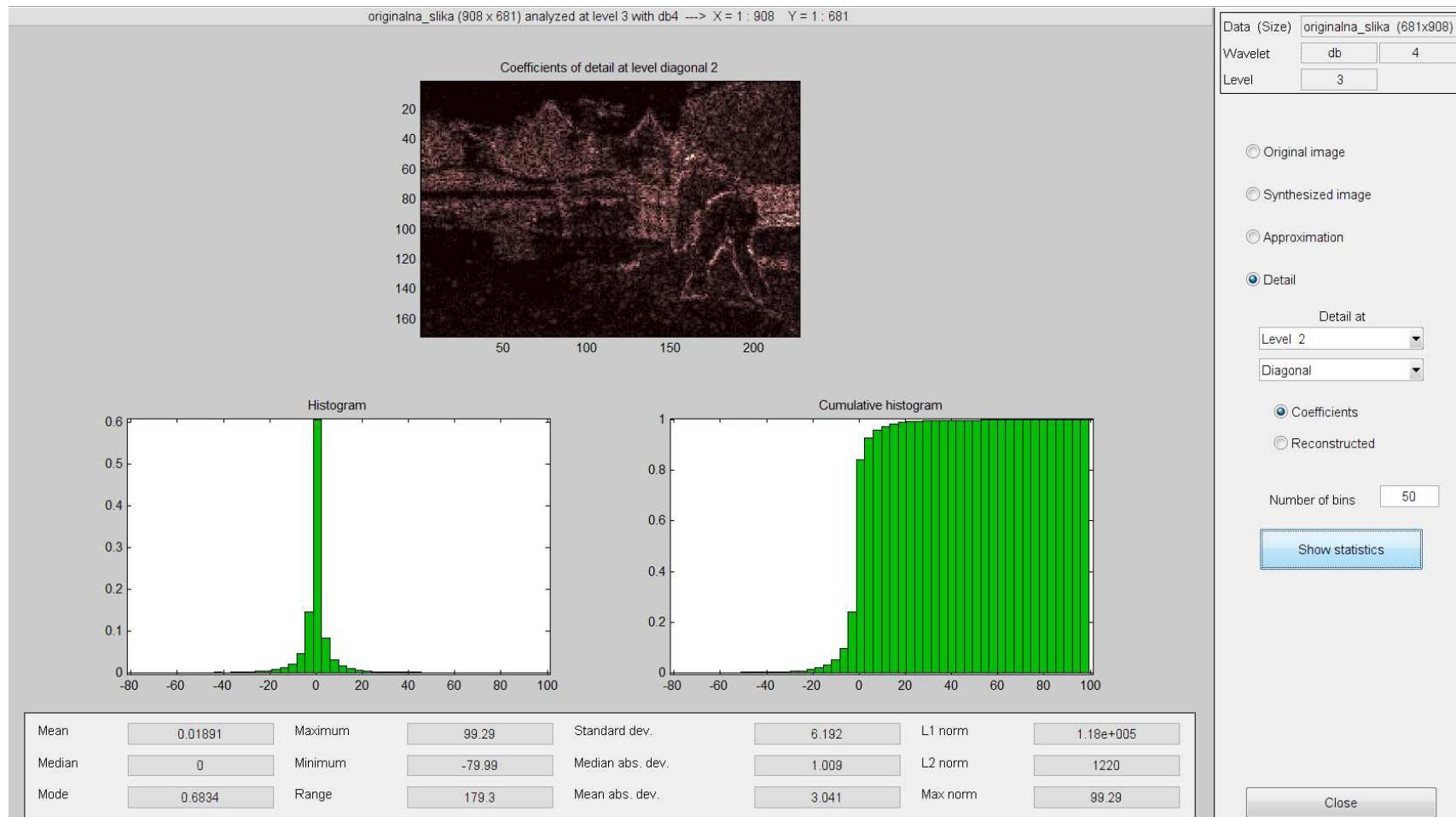
## Zgled: Dekompozicija slike



Originalna slika velikosti  $681 \times 908$  pikslov. Uporabljen Daubechiesov valček, reda 4. Trije nivoji dekompozicije.

## 2D Valčna transformacija

(15)



Podrobnosti o diagonalnih robovih na drugem nivoju dekompozicije.

## **F. Aplikacije 2D DWT**

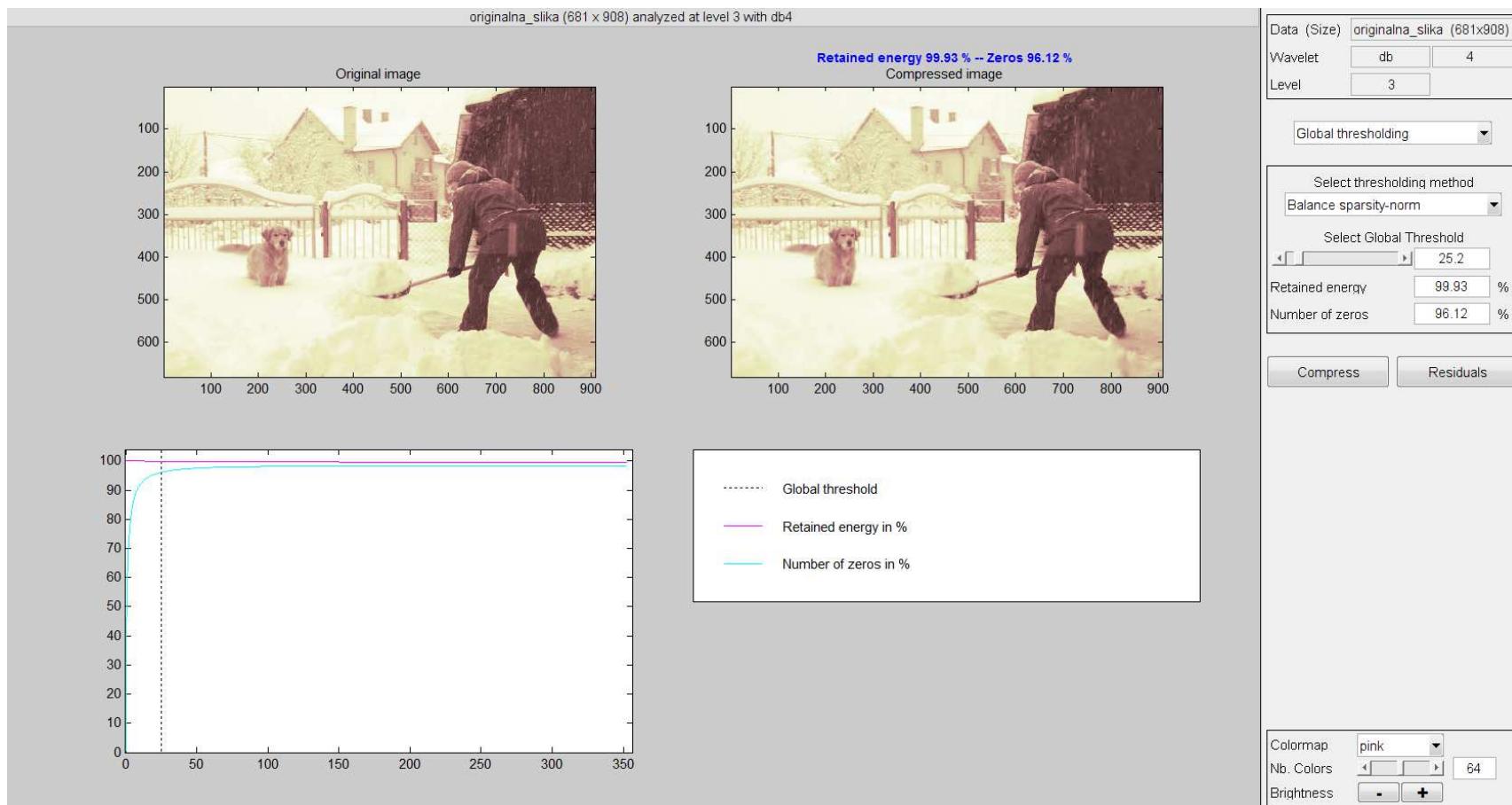
### **1. Kompresija slik**

- ▷ Histogram originalne sivinske slike ima običajno poljubno obliko.
- ▷ Histogrami podslik z detajli o robovih, dobljenih z 2D DWT, pa so tipično unimodalni in simetrični okrog 0:
  - \* To dejstvo močno poenostavi analizo statističnih lastnosti slike.
- ▷ Večina koeficientov DWT ima majhne vrednosti, zato jih lahko postavimo na 0!
- ▷ **Ideja kompresije:**
  1. eliminiramo koeficiente z zelo majhnimi vrednostmi ter
  2. izvedemo rekonstrukcijo
  - \* Ideja je implementirana v JPEG 2000.

## 2D Valčna transformacija

(17)

### Zgled: Kompresija slike



Originalna slika velikosti  $681 \times 908$  piksov. Uporabljen Daubechiesov valček, reda 4 in trije nivoji dekompozicije. Kompresija izvedena z globalnim pragom. Velikost komprimirane slike je 83 % originalne.

### **2. Izboljšanje (filtriranje) slik**

- ▷ Velika podobnost linearemu filtriranju v Fourierovi frekvenčni domeni.
- ▷ **Postopek:**
  - \* S spreminjanjem valčnih koeficientov selektivno poudarimo zanimive komponente v sliki (na račun neželenih).

### **3. Luščenje značilk**

### **4. Detektiranje skritih nezveznosti v signalu (sliki)**

- ▷ Skrite nezveznosti opazimo v koeficientih DWT v višjih nivojih.

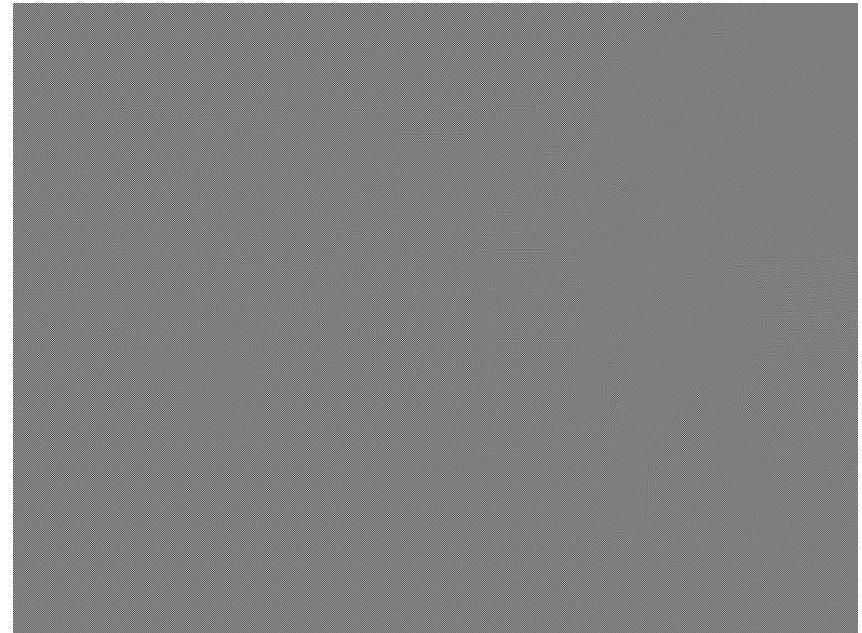
## 2D Valčna transformacija

(19)

Zgled: Izboljšava slik



Originalna slika



Pri rekonstrukciji zamenjali diagonalne koeficiente  
in koeficiente aproksimacije na 1. nivoju

### 5. Zlitje slik

- ▷ Pri postopku zlitja slik (*image fusion*) združujemo dve ali več poravnanih slik istega objekta (scene):
  - \* Cilj: lažje interpretiranje takšne zlite slike, kot pa vsako posamezno originalno
- ▷ Slike združujemo v domeni valčne transformacije:
  - \* združujemo na nivoju istoležnih koordinat
  - \* uporabimo lahko različne funkcije zlitja slik
  - \* tipične funkcije: maksimalna, minimalna oz. naključna vrednost koeficientov, koeficiente vzamemo od izbrane slike ipd.
  - \* nad tako dobljenimi koeficienti izvedemo inverzno 2D DWT:
    - ▷ s tem rekonstruiramo zlito sliko
- ▷ **Tipična področja uporabe:**
  - \* medicina – zlitje slik različnih modalitet iste opazovane telesne strukture (npr. slik CT in MRI)
  - \* interpretiranje multispektralnih slik

# 2D Valčna transformacija

(21)

## Zgled: Zlitje slik



## 2D Valčna transformacija

(22)



Slika 1



Slika 2



Zlita slika

## **Dodatna literatura**

- [1] M. Sonka, V. Hlavac, R. Boyle (1994), Image processing, analyis and machine vision, Chapman & Hall Computing, London, VB.
- [2] J.C. Russ (1995), The image processing handbook, 2. izdaja, CRC press, Boca Raton, ZDA.
- [3] B. Jahne (1993), Digital image processing: Concepts, algorithms, and scientific applications, 2. izdaja, Springer-Verlag, Heidelberg, Nemčija.
- [4] K.R. Castleman (1996), Digital image processing, Prentice Hall, New Jersey, ZDA,
- [5] R. Polikar (1996), The wavelet tutorial,  
<http://users.rowan.edu/~polikar/WAVELETS/WTpart1.html>