Uvod v računalniški vid in razpoznavanje vzorcev

(3. letnik RIT – UN, l. 2024/2025)

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:**
  - Seznaniti študente s temeljnimi znanji s področij računalniškega vida in razpoznavanja vzorcev ter vpeljati nevronske mreže kot primerno orodje za obe področji.
- **Znanje in razumevanje:**
  - izkazati znanje in razumevanje osnovnih postopkov računalniškega vida, razvrščanja vzorcev in uporabe nevronskih mrež,
  - načrtati in sestaviti preproste razpoznavalne sisteme, ki temeljijo na slikovnem materialu,
  - oceniti in ovrednotiti ter uglasiti razpoznavalne sisteme oziroma njegove posamezne dele.
- **ECTS:** 6 točk $\approx$ 180 ur dela
- **Kontaktne ure:**
  - Predavanja: 30 ur
  - Laboratorijske oz. računalniške vaje: 45 ur
- **Samostojno delo študenta:** $\approx$ 105 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)
      - **Pozor: Vsak kolokvij minimalno 35 % razpisanih točk!**
  - **Ne pozabite se prijavljati na izpitne roke!**

- **Predviden razpored predavanj:**

1. teden: 1.10.
2. teden: 8.10.
3. teden: 15.10.
4. teden: 22.10.
5. teden: 29.10.
6. teden: 5.11.
7. teden: 12.11.
8. **teden: 19.11. – 1. kolokvij**
9. teden: 26.11.
10. teden: 3.12.
11. teden: 10.12.
12. teden: 17.12.
13. teden: 24.12.
14. teden: 7.1.
15. **teden: 14.1. – 2. kolokvij**

Vsebina in literatura

- **Vsebina:**
  1. Uvod v računalniški vid
  2. Model kamere in zajemanje slik
  3. Predobdelava slik
  4. Segmentacija oz. razčlenjevanje slik
  5. Zaporedje slik in sledenje gibanja
  6. Geometrija več pogledov
  7. Uvod v razpoznavanje vzorcev
  8. Osnovni pojmi o nevronskih mrežah
  9. Zvezni večplastni perceptroni
  10. Samoorganizirajoče se nevronske mreže

- **Literatura:**
  - Gradivo iz predavanj in zapiski
  - B. Potočnik, Osnove razpoznavanja vzorcev z nevronskimi mrežami, UM-FERI, 2007.
  - Zapiski predavanj za predmet DOSIS
  - (Ostali temeljni študijski viri navedeni v učnem načrtu predmeta – spletna stran FERI)

Kazalo

1. Uvod v računalniški vid

Osnovni pojmi

Uporabljene oznake

Operacije nad slikami

Osnovni pojmi

- Zvezno 2D-sliko matematično modeliramo kot zvezno funkcijo dveh neodvisnih spremenljivk:

$$\mathsf{I} = \mathcal{I}(x, y) \qquad x, y \in \mathbb{R}$$

  - x in y sta prostorski koordinati
  - Slikovna funkcija je definirana v vsaki točki ravnine.
- Vrednost slikovne funkcije lahko pomeni:
  - svetlost (sivino), barvo, temperaturo, porazdelitev tlaka, oddaljenost od opazovalca...
- Digitalna oz. diskretna 2D-slika je diskretizirana zvezna 2D-slika
  - vrednost slikovne funkcije obstaja samo v diskretnih prostorskih trenutkih oz. lokacijah:

$$\mathsf{I} = \mathcal{I}(i, j) \qquad i, j \in \mathbb{N}$$

- Digitalno sliko običajno predstavimo kot 2D-mrežo (matriko):
  - M vrstic in N stolpcev
  - osnovni gradnik slike je piksel $\mathbf{p}$ (*pixel*) – npr. $\mathbf{p} = [i, j]$ in je določen z indeksom vrstice i in stolpca j

Osnovni pojmi *(2)*

- Vrednost diskretne slikovne funkcije v pikslu $\mathbf{p}$ označimo kot

$$\mathsf{I}(\mathbf{p}) \quad \text{oz.} \quad \mathsf{I}(i,j)$$

- Vrednost piksla popišemo z 8 biti (možno tudi s 16, pri barvnih slikah pa potrebujemo $3 \cdot 8$ bitov)
  - ⋆ v naših predavanjih se bomo pretežno ukvarjali z 8-bitnimi sivinskimi slikami (v redkih izjemah z barvnimi)
- Vrednost piksla (tj. sivina oz. svetlost v tem pikslu) leži na intervalu $[0, Q]$
  - ⋆ pri 8 bitih je $Q = 2^8 - 1 = 255$
- Sivinske slike vizualiziramo tako, da vrednost piksla pomeni en vstop v barvno paleto (v primeru sivinskih slik je to paleta sivin)
  - ⋆ vrednost 0 - črna barva
  - ⋆ vrednost 127 - 50 % siva barva
  - ⋆ vrednost 255 - bela barva

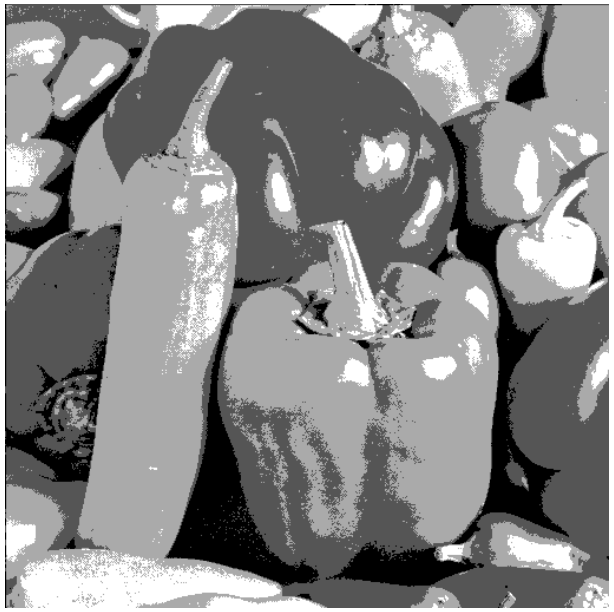
Osnovni pojmi *(3)*

- **Kontrastna ločljivost slike:** število različnih nivojev sivin (tj. sivin) v sliki
  - maksimalna kontrastna ločljivost v sliki je $Q + 1$ (npr. 256 pri 8-bitnih slikah)
  - pri manjši kontrastni ločljivosti se del informacije iz slike izgubi (detajli se zlijejo z večjimi področji)

- **Prostorska ločljivost slike:** definira relacijo med piksli in merskimi enotami realnega sveta
  - podana v enoti piksel na enoto dolžine (npr. 5 pikslov/cm)
  - pri opazovanju scene je prostorska ločljivost tesno povezana z velikostjo slike
  - nujno potrebna za interpretacijo dolžin na slikah
  - pri manjši prostorski ločljivosti se del informacije iz slike izgubi (detajli se zlijejo z večjimi področji)
  - manjša prostorska ločljivost pomeni tudi manjšo velikost slik

Primer: Kontrastna in prostorska ločljivost

originalna slika (512 x 512, 228 sivin)

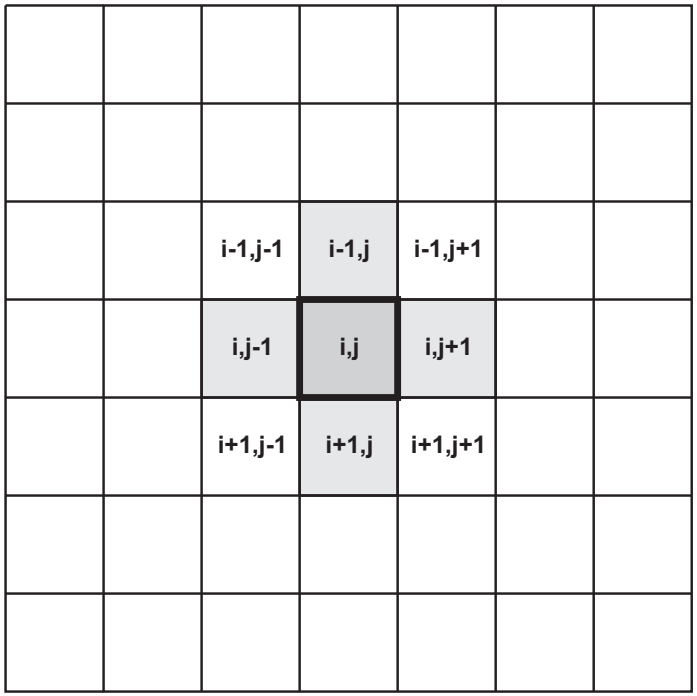


4 sivine

8 x manjša prostorska ločljivost (64 x 64)

Osnovni pojmi *(4)*

- Pikslu $\mathbf{p} = [i, j]$ lahko definiramo v sliki I njegove sosede (tj. sosednje piksle) na dva načina:
  - ⋆ v 4-sosedstvu
  - ⋆ v 8-sosedstvu



Uporabljene oznake

- Konsistentnost uporabljenih oznak skozi predavanja.
- Vektorji – majhne oz. velike črke, zapisane krepko (odvisno od opazovanega prostora). Primer: vektor $\mathbf{b} = [b_0, b_1, \ldots, b_{n-1}]$, vektor $\mathbf{A}$
- Matrike – velike črke, zapisane v sanserifni pisavi. Primer: matrika A
- Slika – I
- Zaporedje 2D-slik – $\mathfrak{I}$
- Velikost slike (št. vrstic in stolpcev) – $M\mathrm{x}N$
- Piksel – $\mathbf{p} = [i, j]$
- Točke, premice in ravnine v prostoru – črke, zapisane kurzivno: Primer: ravnina $\mathit{\Pi}$
- Funkcije – velike črke, zapisane v kaligrafski pisavi: Primer: funkcija $\mathcal{F}()$
- Standardni odklon – σ
- Povprečje – $\bar{p}$
- Absolutna vrednost – $|p|$

Operacije nad slikami

- Operacije nad slikami opravljamo na nivoju istoležnih pikslov.
- Seštevanje/odštevanje slik:

$$\mathsf{C} = \mathsf{A} \pm \mathsf{B} \;\Rightarrow\; \mathsf{C}(i,j) = \mathsf{A}(i,j) \pm \mathsf{B}(i,j)$$

- Množenje/deljenje slik:

$$\mathsf{C} = \mathsf{A} \cdot / \, \mathsf{B} \;\Rightarrow\; \mathsf{C}(i,j) = \mathsf{A}(i,j) \cdot / \, \mathsf{B}(i,j)$$

- Množenje s skalarjem:

$$\mathsf{C} = \alpha\mathsf{A} \;\Rightarrow\; \mathsf{C}(i,j) = \alpha\mathsf{A}(i,j)$$

- Kvadriranje slik:

$$\mathsf{C} = \mathsf{A}^2 \;\Rightarrow\; \mathsf{C}(i,j) = \mathsf{A}(i,j)^2$$

- Če bomo potrebovali matrične operacije, bomo na to posebej opozorili (npr. množenje matrik, transponiranje, inverz).

2. Model kamere in zajemanje slik

Zajemanje slik s kamerami CCD

Modeli kamer

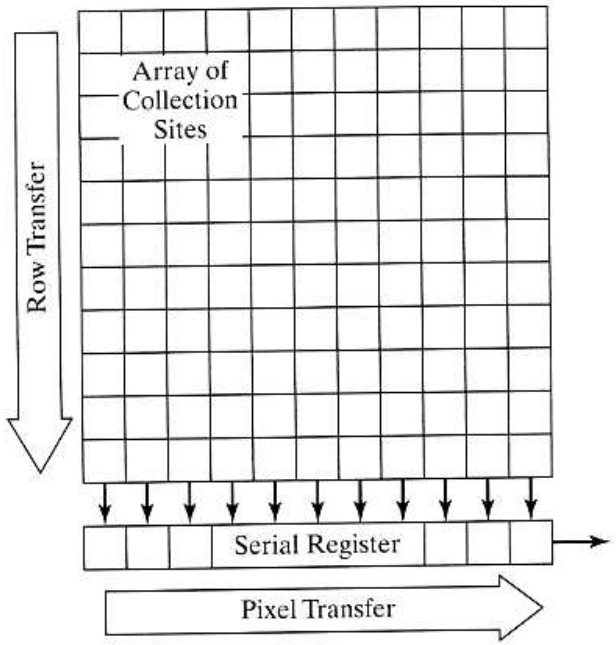
Parametri kamere

Geometrijsko kalibriranje kamer

Zajemanje slik s kamerami CCD

- Kamere CCD (*charge-coupled-device*) vsebujejo senzor CCD
- Senzor je običajno pravokotne oblike in sestoji iz množice polj (elementov)
- Vsako polje je občutljivo na svetlobo, ki pade nanjo:
  - lahko meri količino svetlobne energije, ki pade nanjo
  - količina svetlobne energije se v vsakem polju zbira skozi določeno časovno obdobje

- Shranjen naboj v posameznih poljih celotne vrstice se nato prenese v register (začnemo pri najnižji vrstici).
- Slika se prebere iz senzorja CCD vrstico za vrstico, pri čemer se celotna vrstica vzporedno posreduje v serijski izhodni register.
- Register posreduje prejet naboj polja (enega na enkrat) v izhodni ojačevalnik, ki zgenerira ustrezen analogni oz. zvezni signal.
- Pred pošiljanjem naslednje vrstice se pošlje kontrolni signal.

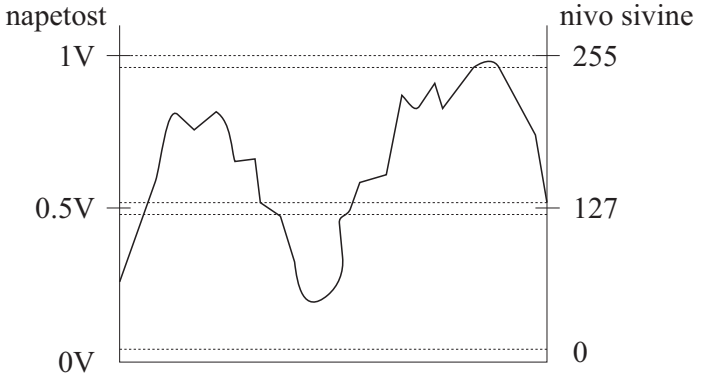


Zajemanje slik s kamerami CCD (2)

- Opisani proces se ponovi za vsako vrstico CCD-naprave oz. dokler ni prebrana celotna slika.
- Branje oz. formiranje slike, po tem postopku, se lahko ponovi večkrat na sekundo (npr. 30 fps), lahko pa traja tudi več ur (npr. v astronomiji).

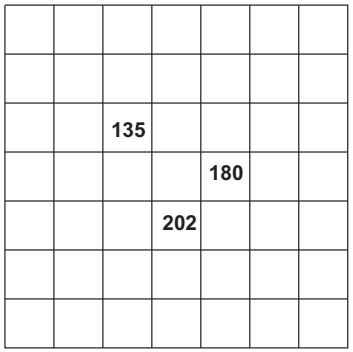
Kako pa dobimo digitalno sliko?

- ustvarjeni analogni video signal se posreduje slikovnemu digitalizatorju (*frame grabber*)
- izvrši se analogno/digitalna (A/D) pretvorba – signal se pretvori v nivo sivine:
  - npr. v trenutku T se vrednost signala 0,5 V pretvori v sivino 127
- digitaliziramo oz. vzorčimo tipično z ločljivostjo 8 bitov:
  - pomeni, da dobimo $2^8 = 256$ različnih sivin
  - pomeni, da se območje napetosti razdeli na 256 razredov, pri čemer je posamezen razred velik $\triangle = \frac{U_{max} - U_{min}}{256}$
  - vsem napetostim znotraj istega razreda priredimo enako sivino



Zajemanje slik s kamerami CCD *(3)*

- ▷ Vzorčimo ob točno določenih časovnih trenutkih.
- ▷ Časovni trenutek eksaktno določa položaj piksla znotraj matrike (slike).
- ▷ Rezultat vzorčenja analognega video signala – digitalna slika.



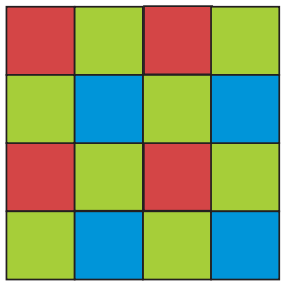
Kako pridemo do barvnih digitalnih slik?

- ▷ Barva sestoji iz treh komponent (npr. rdeče, zelene in modre komponente pri barvnem modelu RGB)
- ▷ Različni načini zajemanja barvni slik (”single-shot”, ”multi-shot”).

1. Način ”single-shot”

a) Nad senzor položimo Bayerjev filter, ki prepušča le določen barvni spekter.

- ⋆ po zajemu uporabimo postopek Bayerjevega demoziciranja in dobimo digitalno barvno sliko
- ⋆ gre za interpoliranje podatkov na osnovi vrednosti sosednjih pikslov



Zajemanje slik s kamerami CCD *(4)*

b) V napravi imamo tri senzorje:
   - ▷ vsak senzor namenjen za eno od osnovnih barvnih komponent
   - ▷ vstopno svetlobo pošljemo na stekleno prizmo (*beam splitter*), ki svetlobo razbije in odbije na posamezen senzor

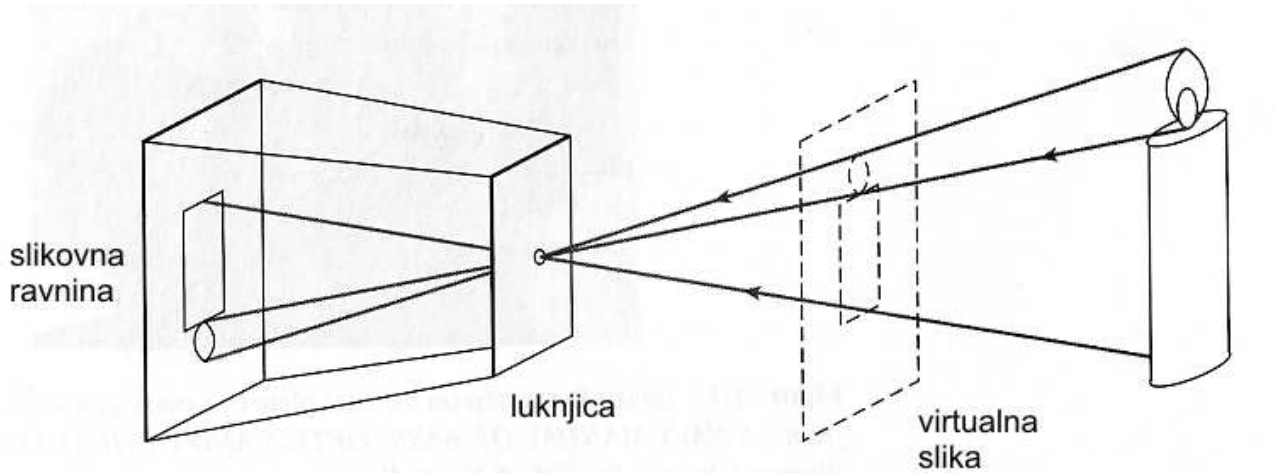
2. Pristop ”multi-shot”

- ⋆ Senzor 3x zaporedoma (v zelo kratkem časovnem intervalu) izpostavimo svetlobi (sceni).
- ⋆ Pri vsakem odpiranju zaslonke se pred senzor položi filter, ki prepušča le določeno barvno komponento.

Modeli kamer

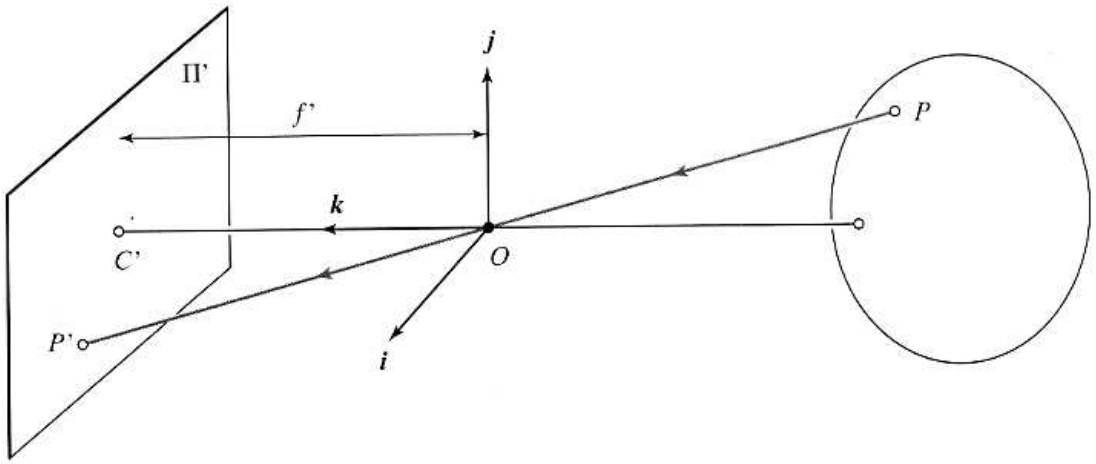
- Kamera na luknjico (*pinhole camera*)
- Kamera s tankimi lečami
- Kamera z debelimi lečami

A. Kamera na luknjico (*pinhole camera*)



- Velikost objektov na slikovni ravnini je odvisna od njihove oddaljenosti od luknjice.
- Perspektivni efekt: oddaljeni predmeti se zdijo manjši od predmetov, ki so bližje luknjici.

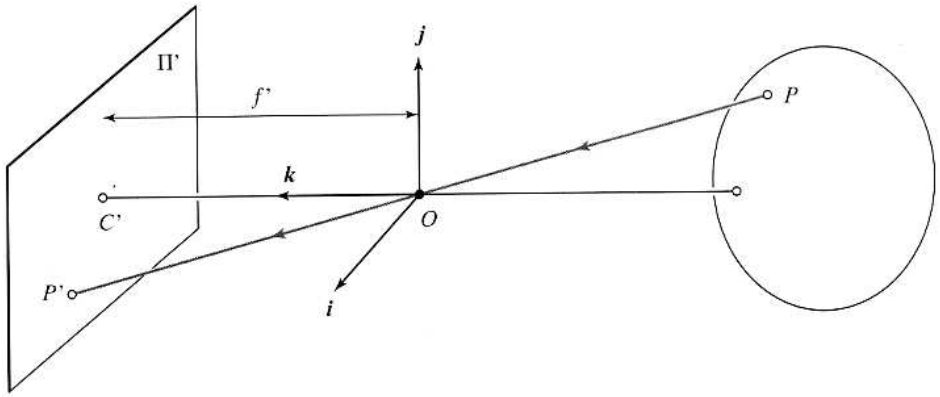
Modeli kamer *(2)*



- Kameri na luknjico priredimo koordinatni sistem:
  - O – izhodišče je v luknjici
  - Π' – slikovna ravnina
  - $\vec{i}$ in $\vec{j}$ – bazna vektorja za ravnino, ki je paralelna ravnini Π'
  - f' – oddaljenost ravnine Π' od luknjice v smeri vektorja $\vec{k}$ (merjeno pozitivno)
  - C' – slikovni center (središče)
    - leži na presečišču optične osi in ravnine Π'
  - optična os je pravokotna na ravnino Π' in gre skozi luknjico
  - Točko C' lahko uporabimo za izhodišče slikovne ravnine
    - pomembna za kalibracijo kamer

Modeli kamer *(3)*

Kako določimo za točko v sceni njen položaj na sliki?



- Točka P – točka iz scene:

$$\mathbf{P} = [x, y, z]$$

- Točka P' – slika točke P na slikovni ravnini

$$\mathbf{P}' = [x', y', z']$$

- Točka P' leži na slikovni ravnini, zato $z' = f'$.
- Koordinati x' in y' določimo iz opazovanja geometrije:

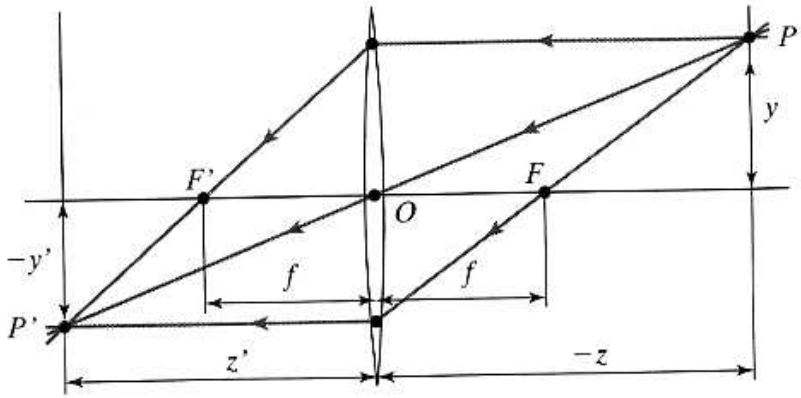
$$x' = f' \frac{x}{z} \quad \text{in} \quad y' = f' \frac{y}{z}$$

Modeli kamer *(4)*

B. Kamera s tankimi lečami

- Perspektivna projekcija pri luknjici je le približek geometrije slikovnega procesa.
- Večina kamer je opremljenih z lečami. Dva razloga:
  1. potrebne so za zbiranje svetlobe, saj bi sicer posamezen žarek dosegel vsako točko v slikovni ravnini pri idealni projekciji skozi luknjico
  2. realne luknjice imajo končno velikost. Večja kot je luknjica, svetlejša je slika, a hkrati bolj zamegljena (*blurry*), in obratno. Drugi razlog je torej, da sliko obdržimo čim bolj v ostrem fokusu, pri čemer zbiramo svetlobo iz širšega fokusa.

Modeli kamer *(5)*



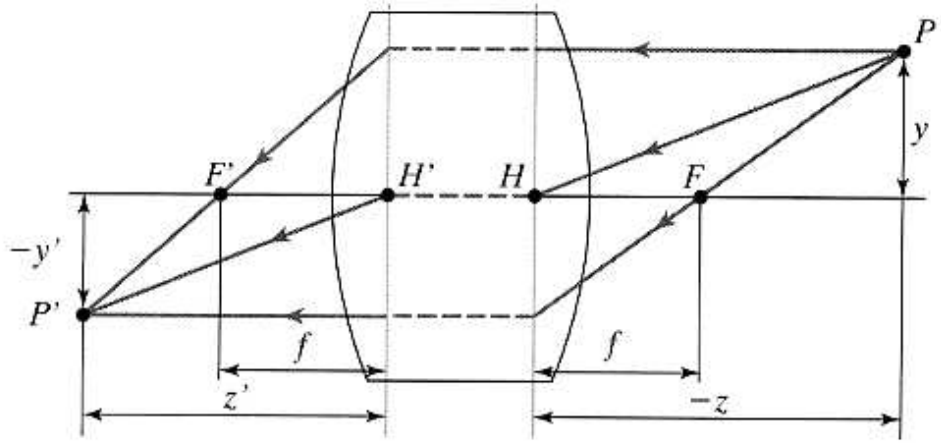
- Velja:

$$\frac{1}{z'} - \frac{1}{z} = \frac{1}{f} \tag{1}$$

- f – goriščna razdalja leče
- Žarki skozi središče O se ne lomijo.
- Žarki, ki so paralelni z optično osjo se fokusirajo v goriščno točko F'.
- Točki F in F' – goriščni točki oz. goriščnici.
- Pozor: Položaj točke P' je enak kot pri perspektivni projekciji kamere z luknjico.
- FOV (*field of view*) kamere – delež scene, ki se dejansko projicira na retino (mrežnico) kamere:
  - ⋆ odvisen od goriščne razdalje in efektivnega področja retine (tj. ploščine filma v kameri oz. ploščine senzorja CCD)

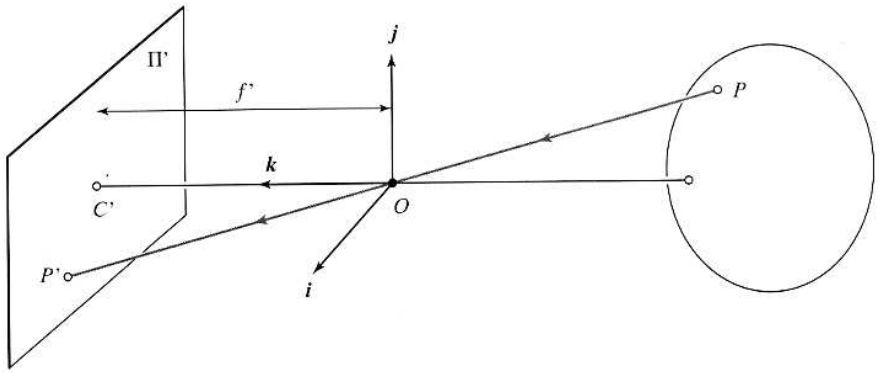
Modeli kamer *(6)*

C. Kamera z debelimi lečami



- ▷ Bolj realističen model optičnega sistema.
- ▷ Točki H in H' sta ”glavni točki” leč
- ▷ Enačba (1) velja.
- ▷ Preproste leče so podvržene številnim popačenjem:
  - ⋆ npr. sferično popačenje, kromatično popačenje
- ▷ Popačenja minimiziramo, če sestavimo več preprostih leč z dobro izbranimi oblikami in karakteristikami:
  - ⋆ sestavljene leče lahko še vedno modeliramo z enačbami, veljavnimi za debele leče

Parametri kamere

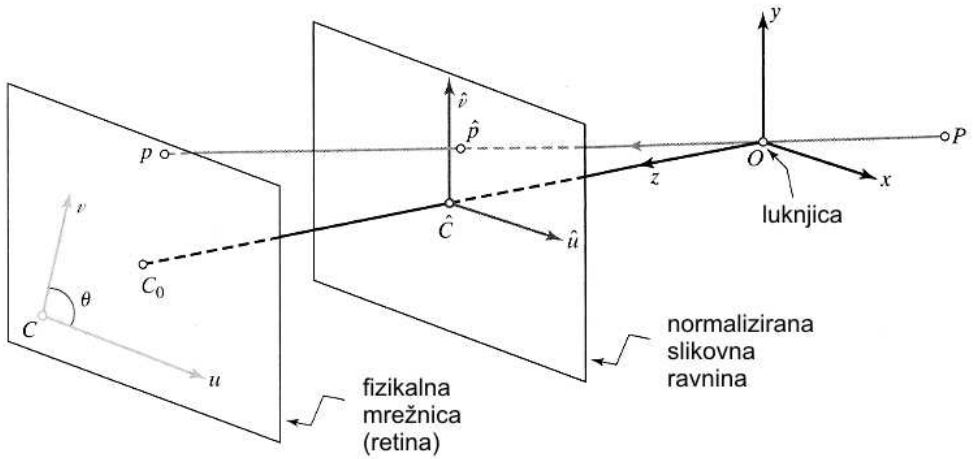


$$x' = f' \frac{x}{z} \quad \text{in} \quad y' = f' \frac{y}{z}$$

- Enačbi veljavni, če razdalje merimo v koordinatnem sistemu kamere, slikovne koordinate pa imajo izhodišče v glavni točki, kjer os simetrije kamere prebije retino (tj. točka C').
- V praksi sta realni (*world*) koordinatni sistem in koordinatni sistem kamere povezana preko fizikalnih parametrov, kot so goriščna razdalja leč, velikosti pikslov, položaja glavne točke, položaja ter orientacije kamere.
- Ločimo dve vrsti parametrov kamere:
  - a) notranje (*intrinsic*) in b) zunanje (*extrinsic*)
- **Notranji parametri** spravijo v relacijo koordinatni sistem kamere in idealiziran koordinatni sistem (glej sliko).
- **Zunanji parametri** spravijo v relacijo koordinatni sistem kamere s fiksnim realnim koordinatnim sistemom:
  - dejansko specificirajo položaj in orientacijo kamere v prostoru

Parametri kamere (2)

A. Notranji parametri kamere



- Notranji parametri kamere:
  - α, β, u_0, v_0 in θ.

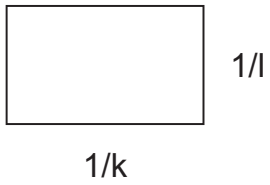
- Velja:

$$\mathbf{p} = \frac{1}{z}\mathrm{M}\mathbf{P}, \qquad \mathrm{M}_{3\mathrm{x}4} = [\mathrm{K}, \mathbf{0}], \qquad \mathrm{K} = \begin{bmatrix} \alpha & -\alpha\cot\theta & u_0 \\ 0 & \frac{\beta}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

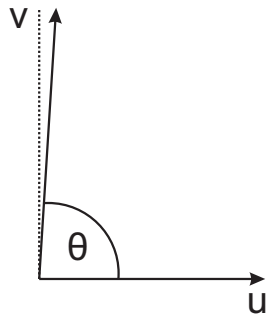
- Koordinatni sistem kamere je v točki C.
- M – perspektivna projekcijska matrika
- K – matrika notranjih parametrov
- Točka $\mathbf{p}$ je izražena v pikslih – $\mathbf{p} = [u, v, 1]^{\mathrm{T}}$
- Točka $\mathbf{P}$ je izražena v homogenih koordinatah koordinatnega sistema kamere – $\mathbf{P} = [x, y, z, 1]^{\mathrm{T}}$

Parametri kamere (3)

- $\alpha = kf$ in $\beta = lf$, pri čemer sta izražena v enotah piksel:
  - k in l sta podana v enoti piksel/meter in določata velikost piksla



- Parametra u_0 in v_0 določata premik izhodišča koordinatnega sistema kamere (tj. točka C) v točko C_0 (tj. točka, kjer optična os prebada slikovno ravnino).
- Kot med obema osema koordinatnega sistema, zaradi nenatančnosti pri izdelavi kamer, ni popolnoma enak $\frac{\pi}{2}$ oz. 90^o (je pa zelo blizu).



- Normalizirana slikovna ravnina – ravnina, ki je oddaljena za 1 enoto od luknjice.
- Fizikalna mrežnica je na oddaljenosti f od luknjice in vzporedna normalizirani slikovni ravnini.

Parametri kamere *(4)*

B. Zunanji parametri kamere

- Smiselni, kadar realni koordinatni sistem (W) in koordinatni sistem kamere (C) nista enaka.
- Potrebna je transformacija enega koordinatnega sistema v drugega:

$$\mathbf{p} = \frac{1}{z}\mathrm{M}\mathbf{P}, \quad \text{vendar sedaj} \quad \mathrm{M}_{3\mathrm{x}4} = \mathrm{K}[\mathrm{R}, \mathbf{t}] \tag{2}$$

- $\mathrm{R}_{3\mathrm{x}3}$ – rotacijska matrika, ki opiše realni koordinatni sistem (W) v koordinatnem sistemu kamere.
- $\mathbf{t}$ – stolpični vektor 3x1; tj. translacijski vektor, ki opiše izhodišče realnega koordinatnega sistema v koordinatah kamere.
- $\mathbf{P}$ – zapisana v homogenih koordinatah realnega koordinatnega sistema.
- Enačbo (2) lahko zapišemo tudi kot

$$u = \frac{\mathbf{m_1 P}}{\mathbf{m_3 P}} \quad \text{in} \quad v = \frac{\mathbf{m_2 P}}{\mathbf{m_3 P}} \tag{3}$$

  - ⋆ $\mathbf{m_i}$ so vrstice matrike M.

Parametri kamere *(5)*

- 6 zunanjih parametrov:
  - 3 koti, ki definirajo rotacijo, ter 3 koordinate vektorja translacije $\mathbf{t}$
- Vsako rotacijo lahko podamo kot kompozicijo rotacij okoli treh osi (Eulerjev teorem):

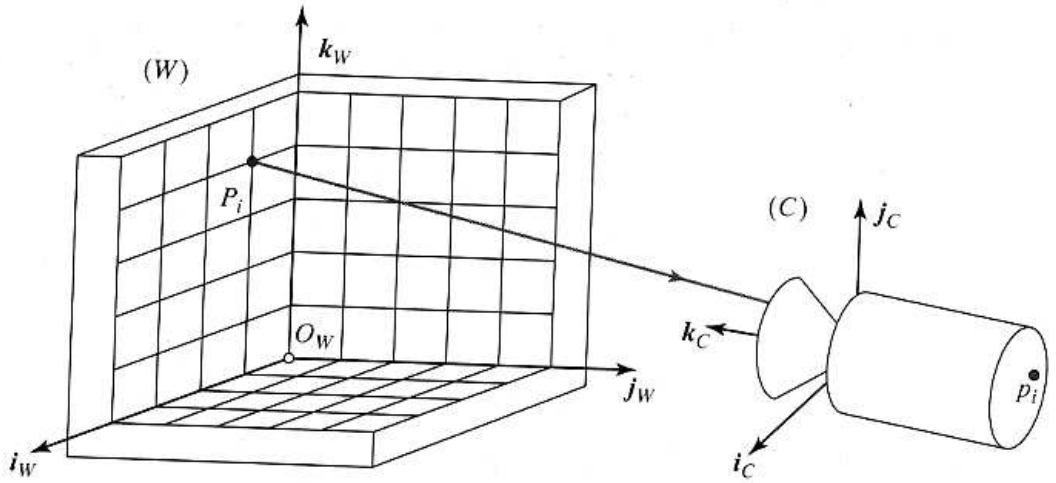
$$\mathsf{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix}, \; \mathsf{R}_y = \begin{bmatrix} \cos\beta & 0 & -\sin\beta \\ 0 & 1 & 0 \\ \sin\beta & 0 & \cos\beta \end{bmatrix},$$

$$\mathsf{R}_z = \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

  - koti so merjeni v proti urni smeri.

Geometrijsko kalibriranje kamer

- Geometrijsko kalibriranje kamer – proces ocenjevanja notranjih in zunanjih parametrov kamere



- Predpostavka: kamera opazuje množico točk (linij ipd.) z znanim položajem v fiksnem realnem koordinatnem sistemu (W).
- Tedaj kalibracijo modeliramo kot optimizacijski proces, kjer minimiziramo razhajanje med položajem opazovanih točk (značilnic) in njihovim teoretičnim položajem (ki je napovedan z enačbami perspektivne projekcije):
  - minimiziramo glede na notranje in zunanje parametre kamere

Geometrijsko kalibriranje kamer (2)

A. Linearni postopek kalibriranja

- Poznamo točne položaje za n točk, in sicer:
  - $\mathbf{P_i}$ – poznamo njegove točne homogene koordinate ($i = 1, \ldots, n$)
  - $\mathbf{p_i}$ – položaj točk $\mathbf{P_i}$ v sliki; $\mathbf{p_i} = [u_i, v_i]$
  - točke $\mathbf{p_i}$ lahko določimo avtomatsko ali pa ročno
- Če preuredimo enačbo (3), dobimo:

$$\begin{aligned}(\mathbf{m_1} - u_i\mathbf{m}_3)\mathbf{P_i} &= 0\\ (\mathbf{m_2} - v_i\mathbf{m}_3)\mathbf{P_i} &= 0\end{aligned}$$

  - za 1 točko torej dobimo 2 enačbi
- Za n točk dobimo $2n$ enačb, ki jih zložimo v matriko P in vektor $\mathbf{m}$:

$$\mathrm{P}_{2n\mathrm{x}12} = \begin{bmatrix} \mathbf{P_1}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -u_1\mathbf{P_1}^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & \mathbf{P_1}^{\mathrm{T}} & -v_1\mathbf{P_1}^{\mathrm{T}} \\ \vdots & \vdots & \vdots \\ \mathbf{P_n}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & -u_n\mathbf{P_n}^{\mathrm{T}} \\ \mathbf{0}^{\mathrm{T}} & \mathbf{P_n}^{\mathrm{T}} & -v_n\mathbf{P_n}^{\mathrm{T}} \end{bmatrix} \quad \text{in} \quad \mathbf{m}_{12\mathrm{x}1} = \begin{bmatrix} \mathbf{m_1}^{\mathrm{T}} \\ \mathbf{m_2}^{\mathrm{T}} \\ \mathbf{m_3}^{\mathrm{T}} \end{bmatrix}$$

Geometrijsko kalibriranje kamer (3)

- Rešiti je treba homogen sistem enačb:

$$\mathrm{P}\mathbf{m} = \mathbf{0}_{2n\mathrm{x}1},$$

  - ⋆ kjer je matrika P znana, vektor $\mathbf{m}$ pa je neznan (iskan).
  - ⋆ Naj opomnimo, da matriko M s parametri kamere, dobimo iz izračunanega vektorja $\mathbf{m}$ kot:

$$\mathrm{M} = \begin{bmatrix} \mathbf{m_1} \\ \mathbf{m_2} \\ \mathbf{m_3} \end{bmatrix}$$

- Rešitev dobimo, kadar je $n \geq 6$:
  - ⋆ pri $n = 6$ dobimo enolično rešitev, sicer minimiziramo izraz $|\mathrm{P}\mathbf{m}|^2$
- Iz projekcijske matrike M lahko določimo notranje (5) in zunanje (6) parametre kamere:
  - ⋆ obstajajo eksaktne formule
- Pozor: pri izbiranju točk $\mathbf{P_i}$ je treba paziti, da vse točke ne ležijo na isti ravnini!
- Zgoraj navedeno velja za idealne leče:
  - ⋆ v praksi se srečamo z raznimi popačenji, ki jih moramo pri kalibraciji upoštevati.
- Ko je kamera kalibrirana, potem lahko izvajamo natančno 3D merjenje položaja točk v digitalnih slikah.

3. Predobdelava slik

Spreminjanje kontrasta

Lokalne operacije oz. filtriranje slik

Odstranjevanje šuma z lokalnimi operatorji

Detektorji robov

Cilj

- Izvaja se na najnižjem nivoju abstrakcije slike (tj. nivoju pikslov).

- Cilj predobdelave:
  - popravljanje določenih degradacij v sliki (npr. neenakomerna osvetlitev scene, slab kontrast, šum, geometrijske deformacije)
  - poudarjanje določenih značilnosti slik, ki so pomembne za nadaljno obdelavo (npr. poudarjanje robov/oglišč)

- Večjo učinkovitost dosežemo, če imamo vnaprejšnje znanje o sliki in tipu degradacije v sliki.

Spreminjanje kontrasta

- Najpreprostejše spreminjanje kontrasta je oblike:

$$g' = \mathcal{F}(g)$$

  - g – stara sivina piksla $\mathbf{p}$ oz. $g = \mathsf{I}(\mathbf{p})$
  - g' – nova sivina piksla $\mathbf{p}$
  - $\mathcal{F}$ – poljubna funkcija
- Funkcijo $\mathcal{F}$ imenujemo tudi točkovna operacija oz. operacija piksel-v-piksel
  - v splošnem ne obstaja inverz funkcije $\mathcal{F}$!

A. Linearizacija sivin

$$g' = \frac{Q}{max - min}(g - min)$$

- max in min – največja in najmanjša sivina, ki jo najdemo v sliki I
- Q – teoretično maksimalna sivina po želeni transformaciji (npr. 15, 255, 4095)
- Linearizacija enakomerno raztegne sivine skozi celoten pas sivin (tj. na območje $[0, 255]$).

Spreminjanje kontrasta (2)

B. Histogram sivin

- Daje preprost pregled nad porazdelitvijo sivin v sliki.
- Je vektor, ki ima toliko elementov, kot je teoretično maksimalna kontrastna ločljivost v sliki (tj. $Q + 1$):

$$\mathbf{h} = [h_0, h_1, \ldots, h_Q]$$

  - h_i – število pikslov v sliki, ki imajo sivino enako i
- Histograme prikazujemo kot stolpične grafe.
- Če je v histogramu sivin težišče (večina pikslov) na levi strani (tj. pri nizkih sivinah), potem je slika najverjetneje pretemna (in obratno).
- V histogram sivin ni zajeta prostorska informacija (tj. položaj piksla):
  - vsebinsko popolnoma različni sliki lahko imata enak histogram sivin

C. Izenačitev histograma

- Je postopek za avtomatsko spreminjanje kontrasta.
- Izboljša kontrast za sivine, ki so blizu maksimumov v histogramu, ter zmanjša kontrast za sivine blizu minimumov.

Spreminjanje kontrasta *(3)*

- Postopek:

$$g' = \frac{Q}{MN} \sum_{i=0}^{g} h_i$$

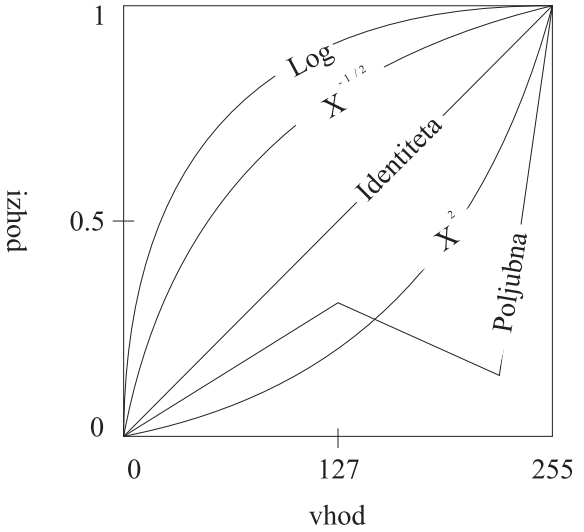
  - MN – število pikslov v sliki, g' – nova sivina, h_i – i-ti element histograma

- Zgolj, če so vse sivine v sliki zastopane z istim deležem, je postopek izenačitve histograma enak linearizaciji!

D. Prenosne funkcije sivin

- Spreminjanje kontrasta definiramo s funkcijo v grafu

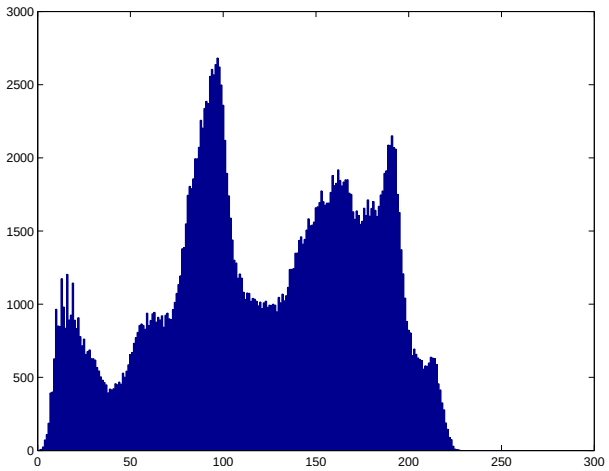
  - na abcisi so vhodne vrednosti oz. sivine g pred spreminjanjem
  - na ordinati so izhodne vrednosti na intervalu $[0, 1]$
  - novo sivino g' dobimo tako, da izhodno vrednost pomnožimo z Q
  - prenosne funkcije so lahko analitične (npr. $g' = \frac{1}{16}\sqrt{g}$) ali pa numerične



Spreminjanje kontrasta *(4)*

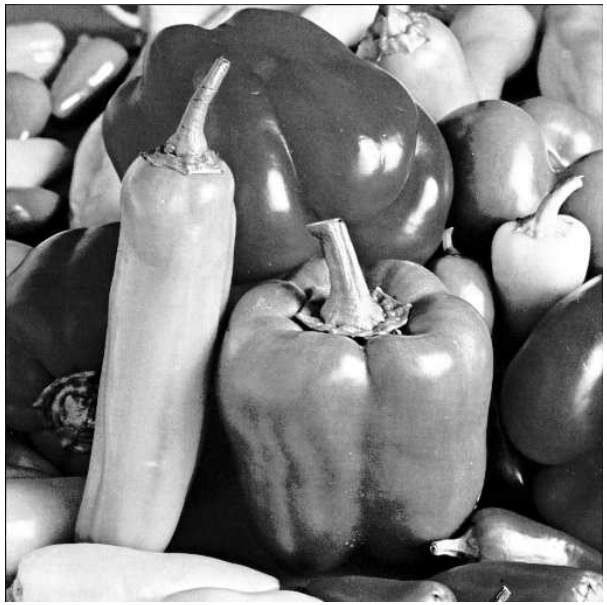
- Graf prenosne funkcije sivin uporabljamo na naslednji način:
  - izberemo vhodno vrednost
  - poiščemo oz. odčitamo izhodno vrednost funkcije
  - izhodno vrednost pomnožimo z Q
- Izhodne vrednosti, ki so nad simetralo, pomenijo, da smo sliko posvetlili (in obratno).

originalna slika (512 x 512, 228 sivin)

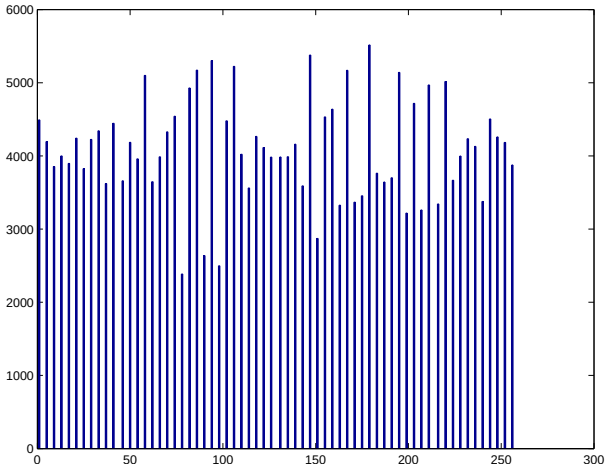


histogram sivin

Spreminjanje kontrasta *(5)*

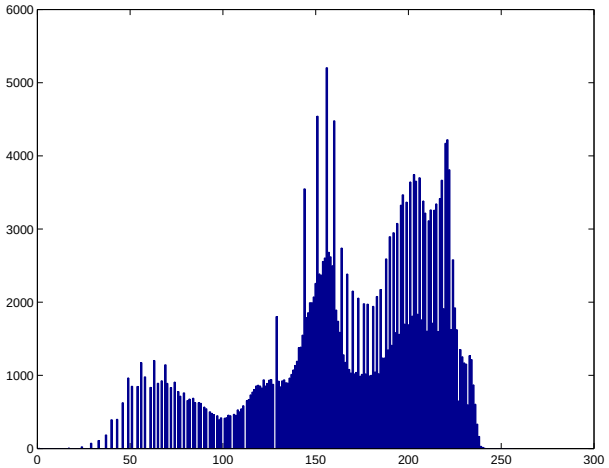


izenačitev histograma



histogram sivin

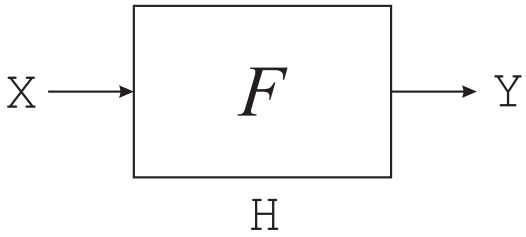
$g' = 1/16\sqrt{g}$



histogram sivin

Lokalne operacije oz. filtriranje slik

- Filter je sistem.



- Obnašanje diskretnega sistema je popolnoma opredeljeno z enotinim odzivom H (ob pogoju, da je sistem linearen in pomično neodvisen):
  - velja za večino realnih sistemov
- Filtriranje slik shematsko opredelimo kot:
  - novo vrednost piksla $\mathbf{p}$, tj. $\mathsf{I}'(\mathbf{p})$, določimo kot funkcijo $\mathcal{F}$, ki operira nad pikslom $\mathbf{p}$ in njegovo okolico $\mathbb{O}$:

$$\mathsf{I}'(\mathbf{p}) = \mathcal{F}(\mathbb{O}_{\mathbf{p}})$$

  - obnašanje (lastnosti) funkcije $\mathcal{F}$ je opredeljeno z enotinim odzivom H

Lokalne operacije oz. filtriranje slik *(2)*

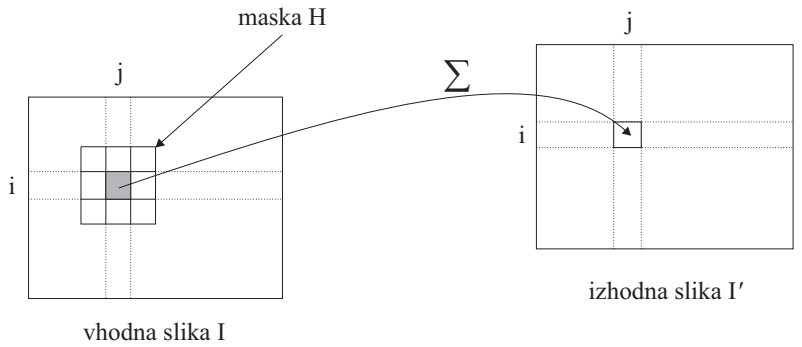
Kako filtriramo sliko?

1. Določimo obnašanje filtra
   (a) izberemo velikost maske H
      - ▷ s tem definiramo okolico piksla $\mathbb{O}$
      - ▷ maska je majhna podslika, običajno pravokotne (kvadratne) oblike in lihih dimenzij – npr. maska 3x3 piksle, 5x7 pikslov
      - ▷ najdemo tudi maske drugih oblik (npr. oblike križa, diskretiziran krog)

   (b) določimo uteži znotraj maske H
2. Izračunamo konvolucijo med masko H in sliko I:

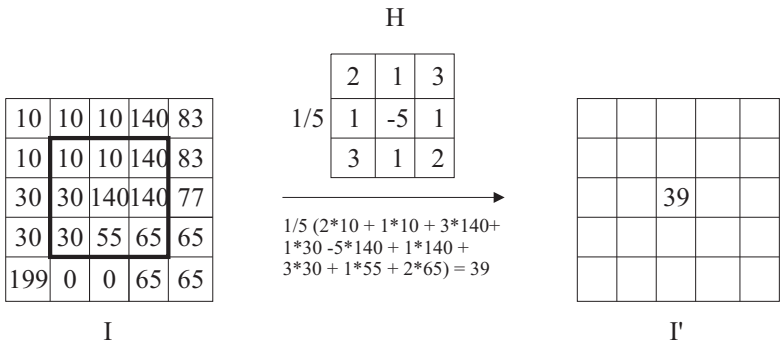
$$I' = I * H, \qquad * \; - \text{ operacija konvolucije}$$

- ▷ postopek za piksel $\mathbf{p}$:
  (a) masko H položimo na sliko I tako, da je sredina maske poravnana s pikslom $\mathbf{p}$
  (b) izračunamo produkt med vsakim elementom maske in pripadajočim pikslom slike, ki se nahaja direktno pod tem elementom maske
  (c) produkte seštejemo ter dobimo novo oz. filtrirano vrednost piksla, tj. $I'(\mathbf{p})$



Lokalne operacije oz. filtriranje slik *(3)*

- pozor: pri filtriranju moramo nove vrednosti shranjevati v novo sliko (pri točkovnih operacijah to ni bilo potrebno)



- celotno sliko filtriramo tako, da opisani postopek ponovimo za vsak piksel v sliki
  - ⋆ običajno začnemo v levem zgornjem kotu slike in nadaljujemo od leve proti desni ter od zgoraj navzdol

Problemi:

- Piksli ob robu slike – del maske je zunaj slike I. Možne rešitve:
  - ⋆ takšnih pikslov ne upoštevamo $\Rightarrow$ velikost izhodne slike se zmanjša za velikost maske
  - ⋆ vrednost manjkajočih pikslov določimo kot 0 ali pa z ustrezno ekstrapolacijo sosednjih pikslov
- Po filtriranju so vrednosti pikslov realna števila in zunaj intervala $[0, 255]$
  - ⋆ možna rešitev: linearizacija + zaokroževanje

Odstranjevanje šuma z lokalnimi operatorji

- ▷ Imenujemo jih tudi operatorji glajenja (*smoothing operators*).
- ▷ Njihova slabost: zapacajo ostre robove objektov v slikah.

A. Nizko sito (*low-pass filter*)

- ▷ Prepušča nizke frekvence v slikah:
  - ⋆ to so velika področja s približno konstantnimi sivinami
- ▷ Visoke frekvence zaduši oz. odstrani iz slike:
  - ⋆ to so majhna področja nenadnih sprememb oz. detajlov
- ▷ Je lokalni operator, s katerim izračunamo povprečje v okolici piksla.
- ▷ Maska vsebuje same 1 in je pomnožena z obratno vrednostjo števila enic v maski.

B. Visoko sito (*high-pass filter*)

- ▷ Ima obratno funkcionalnost kot nizko sito:
  - ⋆ visoke frekvence prepušča, nizke frekvence pa duši

Primer: nizko in visoko sito 3x3 piksle

$$\mathsf{H}_{\text{nizko}} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \mathsf{H}_{\text{visoko}} = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Odstranjevanje šuma z lokalnimi operatorji (2)

C. Gaussov filter

- Masko dobimo iz Gaussovega jedra

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

- (i,j)-ti element maske, velikosti $(2k+1)\mathrm{x}(2k+1)$ pikslov, dobimo kot

$$\mathsf{H}(i,j) = \frac{1}{2\pi\sigma^2} e^{-\frac{(i-k-1)^2+(j-k-1)^2}{2\sigma^2}}$$

- Primer:

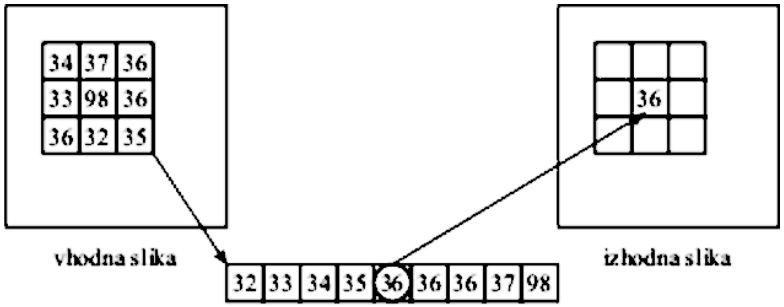
$$\mathsf{H} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \text{in} \quad \mathsf{H} = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- Če je standardni odklon σ zelo majhen (< 1 piksel), potem glajenje praktično nima učinka.
- Če je σ velik, potem bomo odstranili veliko šuma, hkrati pa bomo izbrisali veliko detajlov v sliki (veliko zamegljenje slike)
  - ⋆ pri velikih σ je tudi velikost maske velika

Odstranjevanje šuma z lokalnimi operatorji (3)

D. Mediani filter

- Ideja: trenutni piksel zamenjamo z vrednostjo mediane iz njegove okolice.



- Filter je primeren za odpravljanje impulznega šuma (tj. šum, kjer je deformiranih nekaj izoliranih pikslov)
  - če je deformiranih več pikslov v okolici, potem moramo vzeti mediani filter večjih dimenzij kot 3x3 piksle

- Slabost teh filtrov: poškodujejo tanke linije in ostre robove v sliki
  - problem delno rešimo z ustreznejšo izbiro okolice (npr. okolica oblike križa je primerna za ohranjanje vertikalnih in horizontalnih linij)
- Spadajo v skupino filtrov, ki sortirajo/rangirajo vrednosti pikslov in nato izberejo novo vrednost (*rank value filters*)
  - primer: minimalni in maksimalni filter

Odstranjevanje šuma z lokalnimi operatorji (4)

originalna slika

nizko sito 11x11 pikslov

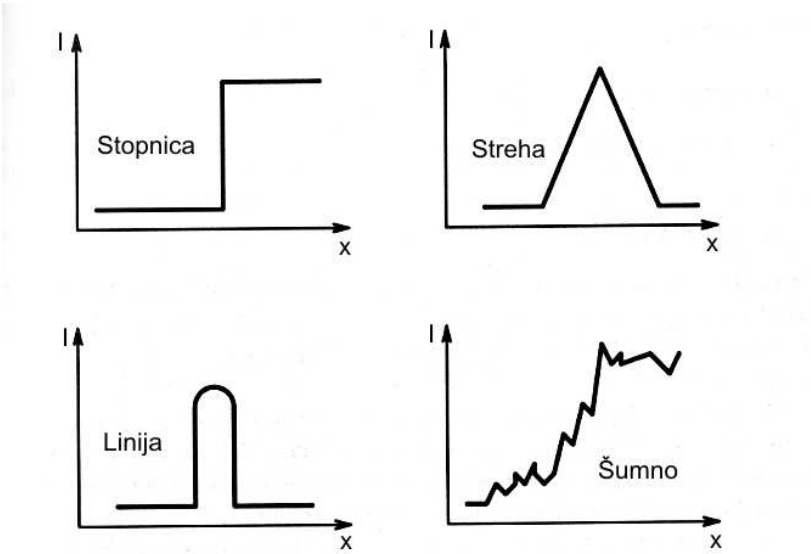
visoko sito

Gaussov filter 11x11 pikslov ($\sigma = 2$)

Detektorji robov

- Uporabljajo se za ugotavljanje nenadnih sprememb v funkciji intenzivnosti (slikovni funkciji):
  - ⋆ pravi robovi so piksli, kjer se ta funkcija nenadno in močno spremeni
- Slikovna funkcija $\mathcal{I}$ je funkcija dveh spremenljivk (dveh koordinat v ravnini).
- Spremembo slikovne funkcije lahko popišemo z gradientom, ki kaže v smeri največje rasti slikovne funkcije.
- Rob je lastnost, ki je pridružena vsakemu pikslu v sliki:
  - ⋆ določi se iz slikovne funkcije in okolice piksla
- Rob je vektorska spremenljivka, sestavljena iz dveh komponent:
  1. velikost (jakost) roba – tj. velikost (jakost) gradienta
  2. smer roba – smer gradienta minus 90^o (če smer 0^o kaže proti vzhodu)

- Tipični profili robov, ki jih najdemo v slikah vzdolž smeri gradienta



Detektorji robov *(2)*

- Za zvezno slikovno funkcijo $\mathcal{I}(x, y)$ določimo velikost in smer gradienta kot

$$|\operatorname{grad} \mathcal{I}(x,y)| = \sqrt{\left(\frac{\partial \mathcal{I}}{\partial x}\right)^2 + \left(\frac{\partial \mathcal{I}}{\partial y}\right)^2} \quad - \text{velikost gradienta}$$

$$\psi = \arctan\left(\frac{\partial \mathcal{I}}{\partial y} / \frac{\partial \mathcal{I}}{\partial x}\right) \quad - \text{smer gradienta}$$

- Digitalne slike so diskretne, zato moramo parcialne odvode aproksimirati z razlikami:

$$\begin{aligned}
\frac{\partial \mathsf{I}(i,j)}{\partial x} &\approx \frac{\mathsf{I}(i,j) - \mathsf{I}(i-\triangle x, j)}{\triangle x} && - \text{razlika nazaj} \\
&\approx \frac{\mathsf{I}(i+\triangle x, j) - \mathsf{I}(i,j)}{\triangle x} && - \text{razlika naprej} \\
&\approx \frac{\mathsf{I}(i+\triangle x, j) - \mathsf{I}(i-\triangle x, j)}{2\triangle x} && - \text{simetrična razlika}
\end{aligned}$$

  - $\triangle x$ običajno enak 1
  - podobno zapišemo parcialni odvod po y in 2. odvode
- Detektorji robov implementirajo zgoraj zapisano teorijo.
- Če sliko I obdelamo z detektorjem robov, dobimo kot rezultat sliko robov E (*edge image*):
  - vsak piksel v tej sliki ima dve komponenti: velikost roba in smer roba

Detektorji robov *(3)*

A. Laplaceov operator

- Približek za 2. odvod.
- V sliki robov dobimo le velikost gradienta (smeri ne!).
- Konvolucijska maska za 3x3 okolico:

$$\mathsf{H} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

B. Sobelov operator

- Ima osem konvolucijskih mask, za nas pomembni dve:
  - za poudarjanje horizontalnih (maska H_1) in vertikalnih (maska H_2) robov

$$\mathsf{H}_1 = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad \mathsf{H}_2 = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

- Če $\mathsf{Y} = \mathsf{I} * \mathsf{H}_1$ in $\mathsf{X} = \mathsf{I} * \mathsf{H}_2$, potem je:
  - velikost roba: $\sqrt{X^2 + Y^2}$ oz. $|X| + |Y|$
  - smer roba: $\arctan\left(\frac{Y}{X}\right)$
- Pravi robovi so tam, kjer je velikost roba velika!

Detektorji robov *(4)*

originalna slika

Sobelov operator (H_1)

Sobelov operator (H_1 in H_2)

Laplaceov operator

4. Segmentacija slik

Pragovna operacija

Pragovna operacija na sliki robov

Ujemanje šablon

- Segmentacija oz. razčlenjevanje slik je proces, ki sliko razdeli (razčleni) na posamezne dele (tj. regije), ki imajo močno korelacijo z objekti/področji iz realnega sveta vsebovanega v sliki.
- Glede na cilj ločimo dve vrsti segmentacije:
  - popolna segmentacija – rezultira v množici disjunktnih (tujih) regij, ki enolično ustrezajo objektom v vhodni sliki
  - delna segmentacija – dobljene regije ne ustrezajo direktno objektom
- Pri delni segmentaciji sliko razdelimo na ločene oz. posamezne regije, ki so homogene glede na izbrano lastnost (npr. sivina, barva, vsebina, tekstura):
  - delno segmentirano sliko še nadalje obdelamo, pri čemer vključimo znanje iz višjih nivojev (npr. znanje o naravi iskanih objektov, medsebojnih relacij med objekti, lastnosti ozadja...)
- Segmentacijske metode so večinoma vezane na problem (tj., so problemsko naravnane), ki ga rešujejo
  - problem definira način, po katerem se poudarjajo iskane lastnosti objektov, ter način, kako se uravnavajo in sklepajo kompromisi med želenimi lastnostmi

- ▷ Ločimo 3 skupine segmentacijskih metod, in sicer glede na dominantno lastnost, ki jo uporabljajo:
  1. segmentacija z globalnim znanjem o sliki oz. njenih delih (npr. pragovna operacija)
     - ⋆ znanje zbrano v raznih histrogramih (npr. sivin)
  2. segmentacija na osnovi robov
  3. segmentacija na osnovi regij
  - – slednji dve pomenita dualni problem: vsaka regija je namreč lahko predstavljena s sklenjeno konturo in obratno

Pragovna operacija

- ▷ Najpreprostejša in najhitrejša segmentacijska metoda.
- ▷ Vhod: originalna slika (npr. slika sivin I)
- ▷ Izhod: segmentirana slika S
- ▷ Več variant pragovne operacije:
  - ⋆ **enopragovne**
    1. s konstantnim pragom T

$$\mathsf{S}(i,j) = \begin{cases} 1; & \text{IF } \mathsf{I}(i,j) \geq T \\ 0, & \text{ELSE} \end{cases}$$

    2. z variabilnim pragom T

$$\mathsf{S}(i,j) = \begin{cases} 1; & \text{IF } \mathsf{I}(i,j) \geq T(i,j) \\ 0, & \text{ELSE} \end{cases}$$

    – enopragovne operacije, še predvsem z globalnim pragom, so primerne za segmentiranje preprostih scen (tj. scen, kjer se sivine objektov bistveno razlikujejo od sivin ozadja)
  - ⋆ vrednost $\mathsf{S}(i,j) \neq 0$ označuje piksle objekta(ov), vrednost $\mathsf{S}(i,j) = 0$ pa piksle ozadja

Pragovna operacija *(2)*

⋆ **večpragovne**

$$\mathsf{S}(i,j) = \begin{cases} 0; & \text{IF } \mathsf{I}(i,j) < T_1 \\ 1; & \text{IF } T_1 \leq \mathsf{I}(i,j) < T_2 \\ 2; & \text{IF } T_2 \leq \mathsf{I}(i,j) < T_3 \\ \vdots & \vdots \\ n-1; & \text{IF } T_{n-1} \leq \mathsf{I}(i,j) < T_n \\ n; & \text{ELSE} \end{cases}$$

A. Metode za določanje globalnih pragov

1. Eksperimentalno
2. Polovica med minimalno in maksimalno sivino v sliki: $T = \frac{min+max}{2}$

Za izračun pragov po ostalih metodah, moramo določiti nekaj pomožnih izrazov iz slike I:

⋆ histrogram sivin: $\mathbf{h} = [h_0, h_1, \ldots, h_Q]$

⋆ izraz A_k – tj. število pikslov, ki imajo sivino manjšo ali enako k

$$A_k = \sum_{i=0}^{k} h_i$$

Pragovna operacija *(3)*

- ⋆ izraz B_k – tj. obtežena vsota pikslov do sivine k (vrednost je proporcionalna povprečni svetlosti v sliki, če upoštevamo zgolj piksle do sivine k)

$$B_k = \sum_{i=0}^{k} i h_i$$

- ⋆ A_Q – število vseh pikslov v sliki, tj. $A_Q = MN$
- ⋆ B_Q – vsota vseh sivin v sliki, tj. $B_Q = \sum_i \sum_j \mathsf{I}(i,j)$

3. Globalni povprečni prag (MEAN)

$$T = \frac{B_Q}{A_Q}$$

4. Globalni prag določen z mediano (MEDIAN)
   - ▷ prag T izberemo tako, da velja

$$\frac{A_T}{A_Q} \approx 0,5$$

Pragovna operacija *(4)*

5. Globalni optimalni prag (OPTIMAL)
   - ▷ iterativni postopek
   - ▷ potrebujemo začetni približek za prag v koraku 0, tj. T_0
   - ▷ izračunamo dva pomožna izraza v k-tem koraku oz. za prag T_k

$$\mu_{T_k} = \frac{B_{T_k}}{A_{T_k}} \quad \text{in} \quad \gamma_{T_k} = \frac{B_Q - B_{T_k}}{A_Q - A_{T_k}}$$

   - ▷ μ_{T_k} je povprečna sivina trenutnega ozadja, γ_{T_k} je povprečje trenutnega objekta(ov)
   - ▷ nov prag v koraku $k+1$ določimo kot

$$T_{k+1} = \frac{\mu_{T_k} + \gamma_{T_k}}{2}$$

   - ▷ postopek zaključimo, kadar

$$| T_{k+1} - T_k | < \epsilon \, ,$$

   kjer je ϵ majhno realno število

Pragovna operacija *(5)*

6. Globalni prag določen z entropijo (KAPUR)

   ▷ prag določimo tako, da maksimiziramo informacijo med ozadjem in objekti – tj. poiščemo maksimum izraza:

$$\max_T \left(H_{ozadje}(T) + H_{objekti}(T) \right) ,$$

   kjer

$$H_{ozadje}(T) = -\sum_{i=0}^{T} \frac{p_i}{P_T} \log \frac{p_i}{P_T}$$

$$H_{objekti}(T) = -\sum_{i=T+1}^{Q} \frac{p_i}{1-P_T} \log \frac{p_i}{1-P_T}$$

$$\mathbf{p} = \frac{1}{MN}\mathbf{h} \quad \text{in} \quad P_T = \sum_{i=0}^{T} p_i$$

   ⋆ $\mathbf{p} = [p_0, p_1, \ldots, p_Q]$ – normaliziran histogram sivin

Pragovna operacija *(6)*

originalna slika

metoda MEAN ($T = 119$)

metoda MEDIAN ($T = 144$)

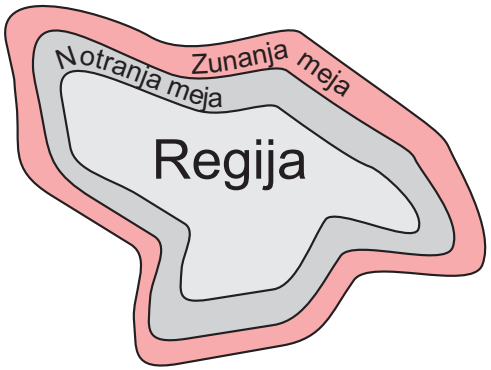
metoda OPTIMAL ($T = 89$)

Pragovna operacija na sliki robov

- V sliki robov imajo ”nepravi” robni piksli vrednosti zelo blizu 0 (vendar $\neq 0$!):
  - razlogi so: šum v podatkih, majhne nepravilnosti v osvetlitvi scene ipd.
- Prave robne piksle (oz. močne robove) lahko dobimo s pragovno operacijo:
  - uporabimo lahko vso znanje o pragovni operaciji!

A. Sledenje meji

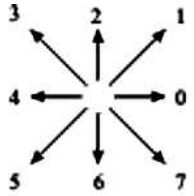
- Vsaka regija ima definirano mejo (*region border*).
- Ločimo dve vrsti mej:
  - notranja meja – dejanska meja regije in je del regije (tj. krivulja na skrajnem zunanjem delu regije)
  - zunanja meja – je meja ozadja in ni del regije (tj. najbližja in najkrajša krivulja, ki oklepa opazovano regijo in ni del regije)
- Algoritem sledenja meji (na naslednji prosojnici) deluje za regije večje od 1 piksla.
- Pri sledenju se lahko včasih vrnemo v isto točko.
- S tem algoritmom ni možno najti meje za luknje znotraj regij!



Pragovna operacija na sliki robov (2)

Algoritem sledenja notranji meji (za 8-sosedstvo)

1. Preiskuj segmentirano sliko od leve proti desni navzdol. Prvi najdeni piksel regije (z vrednostjo $\neq 0$), označi s $\mathbf{p_0}$. To je prvi piksel meje. Spremenljivka dir hrani smer pomika vzdolž meje (od prejšnjega do trenutnega piksla meje). Priredi: $dir = 7$.
2. Preišči 3x3 okolico trenutnega piksla v proti urni smeri. Začni pri pikslu, ki je v smeri:
   - ▷ $(dir + 7) \bmod 8$, IF dir je liho število
   - ▷ $(dir + 6) \bmod 8$, ELSE

   8-sosedstvo

   Prvi najdeni piksel, ki ima enako vrednost kot trenutni piksel, je novi element meje (označimo ga s $\mathbf{p_n}$). Ažuriraj spremenljivko dir.
3. IF $(\mathbf{p_n} = \mathbf{p_1})$ AND $(\mathbf{p_{n-1}} = \mathbf{p_0})$ THEN Konec, ELSE Korak 2.
4. Notranja meja je določena s piksli $\mathbf{p_0}, \mathbf{p_1}, \ldots, \mathbf{p_{n-2}}$.

- ▷ Problem za sledenje so področja, kjer je meja prekinjena (npr. zaradi šuma, prekrivanja objektov)
  - ⋆ rešitev: uporabimo hevristične pristope (tj. predvidimo potek meje na takšnih področjih)
- ▷ Sledenje v sivinskih slikah je možno, vendar predstavlja težji problem kot na binarnih slikah oz. že segmentiranih.

Ujemanje šablon

- Ujemanje rešuje naslednji problem: V sliki I želimo poiskati znan objekt (lahko vzorec ipd.), pri čemer je objekt lahko tudi rotiran.
- Iskan objekt predstavimo v obliki podslike oz. šablone H.
- Položaj iskanega objekta v sliki določimo kot tisti položaj v sliki, kjer izračunamo najboljše ujemanje med šablono in sliko, glede na nek kriterij ujemanja.

Algoritem:

1. `Izračunaj kriterij ujemanja` C `za vsako možno lokacijo in rotacijo iskanega objekta (šablone` H`) v sliki` I`.`
2. `Lokalni maksimum kriterija` C`, pri čemer je` $C \geq T$`, določa položaj iskanega objekta v sliki.`

Podrobnosti:

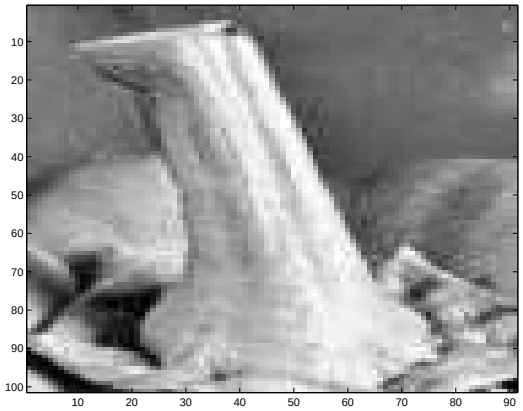
- Kriterij ujemanja C lahko definiramo kot korelacijo med iskano šablono in sliko:

$$C_1(i,j) = \left(\max_{(u,v)\in \mathsf{H}} |\, \mathsf{I}(i+u, j+v) - \mathsf{H}(u,v) \,| \right)^{-1}$$

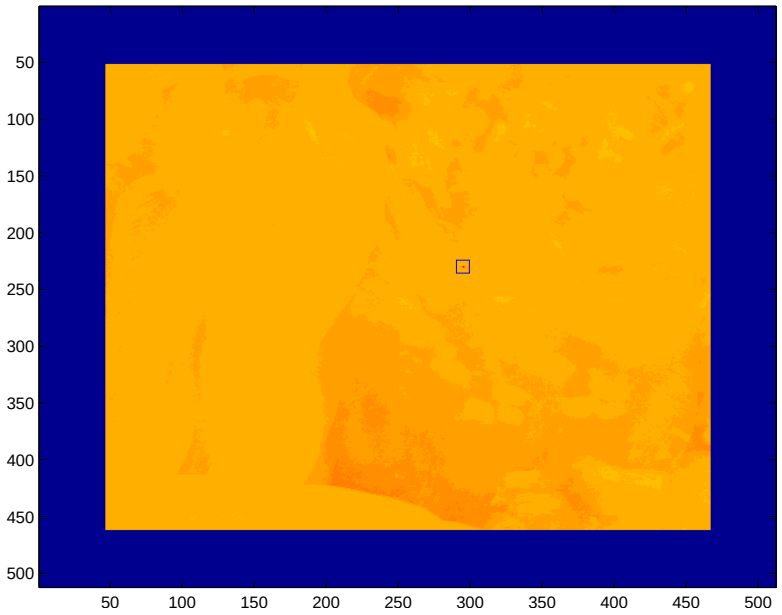
$$C_2(i,j) = \left(\sum_{(u,v)\in \mathsf{H}} |\, \mathsf{I}(i+u, j+v) - \mathsf{H}(u,v) \,|^p \right)^{-1} \qquad p = \{1, 2\}$$

Ujemanje šablon *(2)*

originalna slika



iskan objekt



Kriterij ujemanja C_1 –
pravilni in najdeni položaj (230,295)

5. Zaporedje slik in sledenje gibanja

Osnovni pojmi

Slike razlik

Statično ozadje

Optični pretok

Sledenje objektom

Osnovni pojmi

- Zaporedje digitalnih slik $\mathfrak{I}$ je zaporedje K-tih 2D-digitalnih slik:

$$\mathfrak{I} = \{\mathsf{I}_0, \mathsf{I}_1, \ldots, \mathsf{I}_{K-1}\}$$

- Ob tem še zahtevamo:
  - ⋆ če je bila slika I_k zajeta v času t_k, potem je morala biti slika I_{k+1} zajeta v času t_{k+1}, pri čemer $t_k < t_{k+1}$
- Časovni interval med sosednjimi slikami je običajno fiksen (npr. $\triangle t$). Zato velja:

$$t_{k+1} = t_k + \triangle t \quad \text{oz.} \quad t_{k+1} = t_0 + (k+1)\triangle t$$

- Na osnovi tega lahko rečemo, da je zaporedje slik časovno spremenljivo zaporedje (oz. funkcija)

$$\mathfrak{I} = \mathcal{I}(x, y, t) \quad - \text{za zvezne razmere}$$

- Glede na način zajemanja ločimo različne vrste zaporedij slik:
  1. Zaporedje slik vizualne scene (npr. posnete z video kamero)
     - ⋆ tipične naloge: zaznavanje gibanja, segmentiranje gibajočih se objektov, določanje parametrov gibanja
  2. Zaporedje medicinskih slik oz. prerezov skozi določene strukture
     - ⋆ dejansko ne gre za pravo gibanje – prerezi so res zajeti v različnih časovnih trenutkih, vendar pa je struktura statična
     - ⋆ tipične naloge: segmentiranje opazovane strukture, 3D-rekonstrukcija objekta(ov)

Slike razlik

- Najpreprostejši način za detektiranje sprememb v zaporedju slik je odštevanje slik – slika razlik
- Slika razlik $\triangle \mathsf{I}_k$ med dvema zaporednima slikama se izračuna kot

$$\triangle \mathsf{I}_k = \mathsf{I}_{k+1} - \mathsf{I}_k$$

- Na področjih gibanja bodo spremembe velike!
- Sliko razlik lahko segmentiramo s poljubno segmentacijsko metodo za statične 2D slike.
- Pragovna operacija na sliki razlik:

$$\mathsf{S}(\mathbf{p}) = \begin{cases} \text{gibajoči se}; & \text{IF } | \triangle \mathsf{I}_k(\mathbf{p}) | \geq T \\ 0; & \text{ELSE} \end{cases}$$

- Slike razlik primerne zgolj za obdelovanje zaporedij, zajetih s statično kamero.

Statično ozadje

- ▷ Metoda je alternativa sliki razlik.
- ▷ Motivacija: ozadje je najpogosteje najbolj stabilen del slike.
- ▷ V kontroliranem okolju lahko sliko ozadja (brez gibajočih se objektov) zajamemo.
- ▷ V realnem okolju je ozadje prekrito z gibajočimi se objekti:
  - ⋆ statično ozadje je možno regenerirati iz zaporedja slik.
- ▷ Če imamo na voljo sliko ozadja, I_{Ozadje}, potem lahko uporabimo idejo slik razlik:

$$\triangle \mathsf{I}_k = \mathsf{I}_k - \mathsf{I}_{Ozadje}$$

  - ⋆ velike vrednosti v $\triangle \mathsf{I}_k$ (gledano absolutno) določajo področja velikih sprememb (npr. gibajočih se objektov)

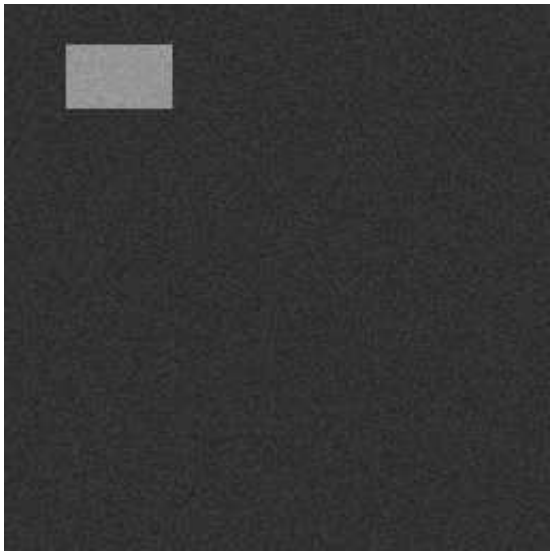
Postopki tvojenja statičnega ozadja

1. Povprečenje celotnega zaporedja slik
2. Na osnovi opazovanja robov skozi čas
   - ▷ izbran piksel opazujemo skozi celotno zaporedje
   - ▷ slike, kjer se je vrednost piksla močno spremenila, ne upoštevamo (področje robov)
   - ▷ za preostale piksle pa izračunamo povprečje

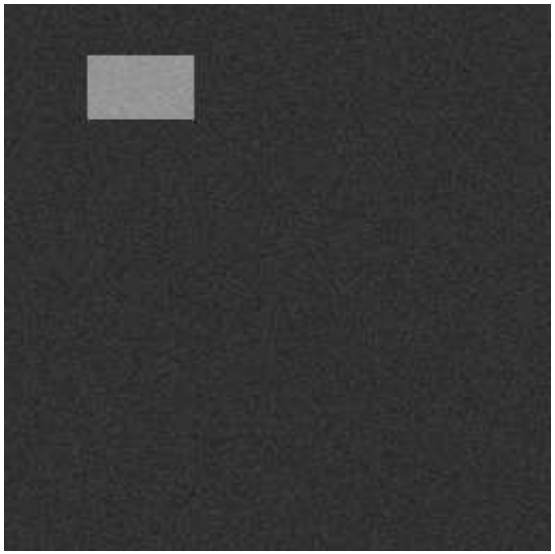
Statično ozadje *(2)*

3. Z drsečim oknom (skozi čas)
   - ▷ kadar imajo vsi istoležni piksli znotraj drsečega okna (drsečo okno – vnaprej predpisano število zaporednih slik) vrednost v preddefiniranem območju, potem prištejemo povprečje teh pikslov k skupni vsoti in povečamo število elementov.
   - ▷ drseče okno premaknemo za eno sliko in postopek ponovimo.
   - ▷ na koncu skupno vsoto delimo s številom vseh elementov ter dobimo končno vrednost za ta piksel.

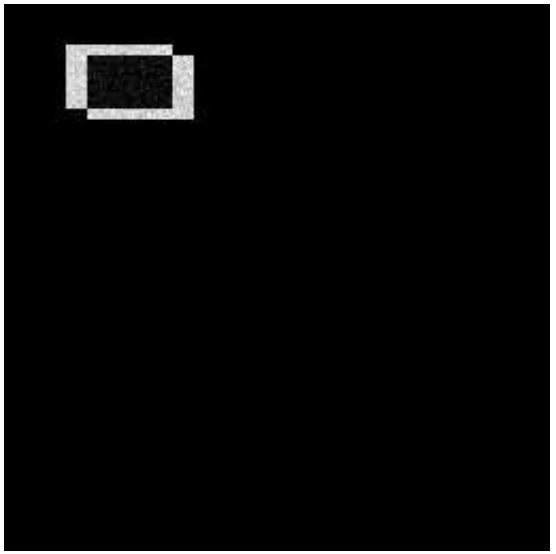
Zgled: Odštevanje slik



slika I_2



slika I_1



slika razlik $|\triangle I_2|$

Optični pretok

- Imejmo gibajoč se predmet v sceni (prostoru), ki jo snemamo npr. z video kamero.
- Gibanje objekta se v sliki kaže kot premik pikslov v smeri x in y.
- 2D vektorsko polje oz. matrika gibanja U definira, za koliko se je premaknil vsak piksel v sliki:

$$\mathrm{U}(\mathbf{p}) = [u_x, u_y]$$

- Polje gibanja ni poznano, poznamo le zaporedje slik $\mathcal{I}(x, y, t)$!
- Optični pretok je približek za pravo 2D vektorsko polje gibanja in se izračuna iz zaporedja $\mathcal{I}(x, y, t)$.
- Izračun optičnega pretoka temelji na naslednji predpostavki: svetlost gibajočega se predmeta ostane konstantna, kar zapišemo s formulo

$$\frac{d\mathcal{I}}{dt} = 0 \quad \text{oz.} \quad [\mathcal{I}_x, \mathcal{I}_y][u_x, u_y]^T + \mathcal{I}_t = 0$$

$\mathcal{I}_x$ in $\mathcal{I}_y$ – prostorska odvoda

$\mathcal{I}_t$ – odvod po času

Optični pretok *(2)*

Kako določimo vektor premika $\mathbf{u}$ za en piksel?

- Vzamemo Q pikslov iz okolice opazovanega piksla ter tvorimo matriko A in vektor $\mathbf{b}$:

$$\mathsf{A} = \begin{bmatrix} \mathcal{I}_x(\mathbf{p_1}) & \mathcal{I}_y(\mathbf{p_1}) \\ \mathcal{I}_x(\mathbf{p_2}) & \mathcal{I}_y(\mathbf{p_2}) \\ \vdots & \vdots \\ \mathcal{I}_x(\mathbf{p_Q}) & \mathcal{I}_y(\mathbf{p_Q}) \end{bmatrix} \quad \text{in} \quad \mathbf{b} = \begin{bmatrix} \mathcal{I}_t(\mathbf{p_1}) \\ \mathcal{I}_t(\mathbf{p_2}) \\ \vdots \\ \mathcal{I}_t(\mathbf{p_Q}) \end{bmatrix}$$

- Rešimo naslednji sistem:

$$\underbrace{\mathbf{u}}_{2\text{x}1} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} = \underbrace{-\mathsf{A}^{-g}}_{2\text{x}Q} \underbrace{\mathbf{b}}_{Q\text{x}1}, \tag{4}$$

kjer je

$$\mathsf{A}^{-g} = (A^T A)^{-1} A^T$$

Optični pretok *(3)*

Algoritem za računanje optičnega pretoka

VHOD: Zaporedje slik $\mathfrak{I}$. Maska H velikosti LxL pikslov, pri čemer $Q = L^2$ (tipično $L = 5$)

IZHOD: 2D vektorsko polje U

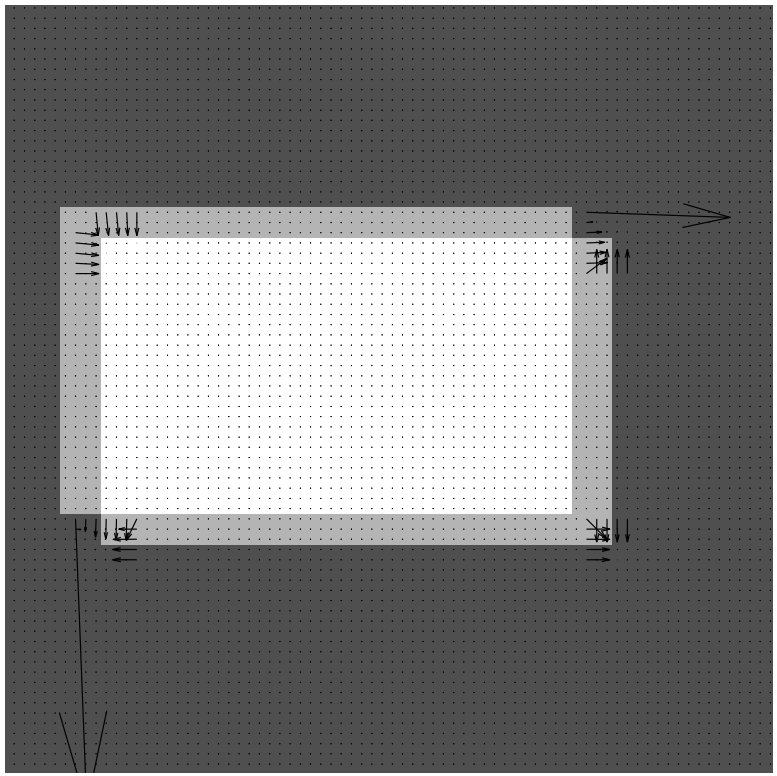
1. Filtriraj vsako sliko iz zaporedja z Gaussovim filtrom s standardnim odklonom σ_s vzdolž obeh prostorskih dimenzij (tipično $\sigma_s = 1,5$ piksla).
2. Filtriraj vsako sliko iz zaporedja z Gaussovim filtrom s standardnim odklonom σ_t vzdolž časovne dimenzije (tipično $\sigma_t = 1,5$ okvirja).
3. FOR vsako sliko zaporedja
   FOR vsak piksel v sliki
   Izračunaj vektor premika $\mathbf{u}$ po enačbi (4), pri čemer vzemi piksle $\mathbf{p_i}$ iz H-okolice trenutnega piksla.

▷ Optični pretok lahko uporabimo tudi za razvrščanje pikslov na gibajoče se in statične (npr. s pragovno operacijo).

Optični pretok *(4)*

Zgled:

- Ozadje: konstantna sivina 40
- Objekt:
  - velikost: 30x50 pikslov
  - konstantna sivina 140
  - položaj levega zgornjega oglišča $(20, 30)$
  - dejanski premik med slikama $\mathbf{u} = [3, 4]$
- Izračun optičnega pretoka:
  - filtriranje vzdolž prostorskih dimenzij: $\sigma_s = 0,4$ (filter 3x3 piksle)
  - filtiranje vzdolž časovne osi ni bilo izvedeno



Optični pretok za sliko 5 z vrisanim položajem objekta na sliki 4 in 5

Sledenje objektom

- Sledenje objektom (značilnicam) je problem iskanja ujemanja objektov (značilnic) iz slike v sliko skozi daljše zaporedje slik.
- Ogledali si bomo sledenje v slikovni ravnini (in ne v 3D-prostoru).
- Koncept sledenja:
  1. $k = 0$
  2. `Določi položaj objekta (značilnice) na začetni sliki;` $k = k + 1$
  3. `Predvidi položaj objekta (značilnice) v` k`-ti sliki, in sicer na osnovi razpoznanih položajev objekta (značilnice) v slikah od 0 pa do` $k - 1$`.`
  4. `Določi natančen položaj objekta (značilnice) v okolici predvidenega položaja.`
  5. `IF` $k < K - 1$ `THEN` $k = k + 1$`; Korak 3; ELSE Konec`

6. Geometrija več pogledov

Dva pogleda (splošno)

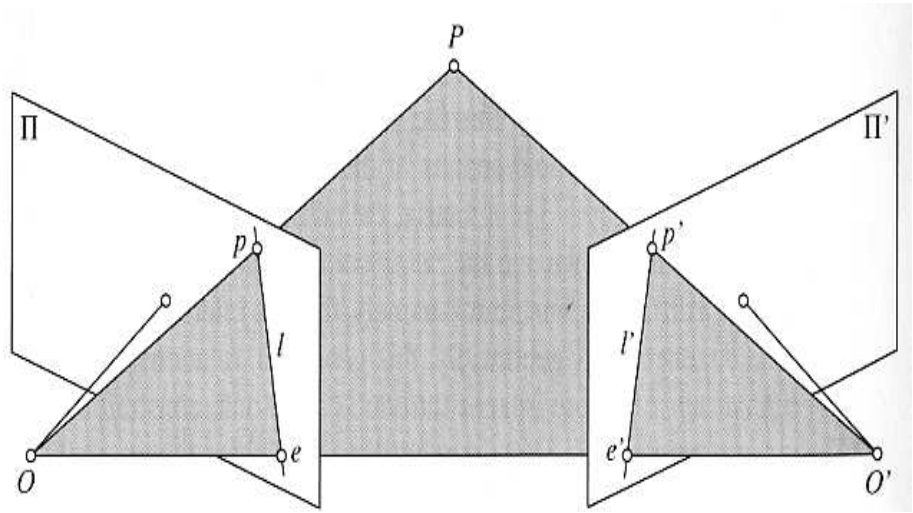
Trije pogledi

Določanje položaja točk v 3D

Stereo vid

Dva pogleda (splošno)

- Na osnovi ene slike ne moremo določiti globine (oddaljenosti) točke iz scene vzdolž projekcijskega žarka.
- Že iz dveh slik (iste scene, a posnete pod različnim kotom) lahko merimo globino (oddaljenost)
  - ⋆ uporabimo triangulacijo
- Dve kameri oz. pogleda.
- O in O' – optična centra (luknjici) prve oz. druge kamere
- P – točka v prostoru
- p in p' – sliki točke P na virtualni slikovni ravnini Π in Π'
- Virtualni ravnini se nahajata pred luknjico – uporabljeni zaradi lažjega risanja.



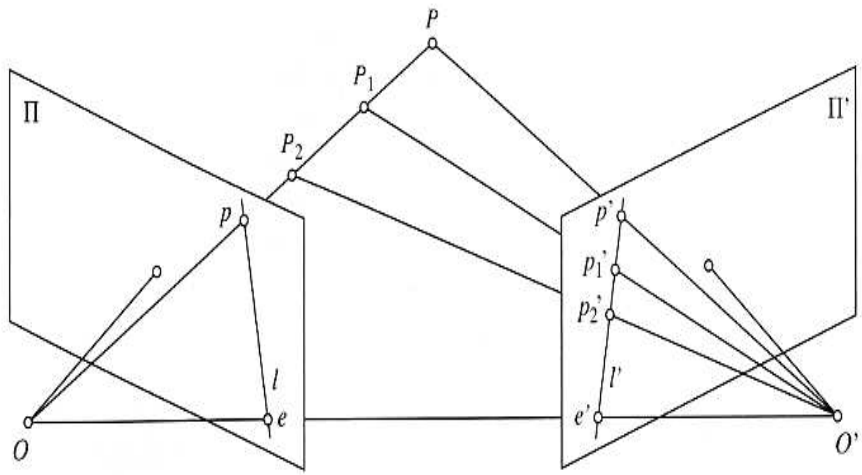
- Vse enačbe in razlage veljajo tudi za fizikalno sliko oz. mrežnico, ki se nahaja za luknjico!

Dva pogleda (splošno) *(2)*

- Točke O, O' in P definirajo ravnino, ki jo imenujemo **epipolarna ravnina**
  - ⋆ točki p in p' ležita na tej ravnini
- Točki e in e' se imenujeta **epipola** posamezne kamere
  - ⋆ epipol e dobimo na presečišču virtualne ravnine Π in osnovnice (daljice) OO'
- daljici $l = pe$ in $l' = e'p'$ se imenujeta **epipolarni premici** (daljici)

Epipolarna omejitev:

- Če sta točki p in p' sliki (tj. projekciji) iste točke P, potem mora p' ležati na epipolarni premici, prirejeni točki p.
  - ⋆ Ta omejitev igra ključno vlogo v stereo vidu ter pri analizi gibanja.
- Če poznamo položaj točke p, potem potencialne ujemajoče se točke lahko v drugi sliki ležijo le na epipolarni premici l'



Dva pogleda (splošno) *(3)*

A. Umerjene oz. kalibrirane razmere

- Predpostavimo, da so notranji parametri obeh kamer poznani.
- Velja:

$$\mathbf{p}^T \, \mathsf{E} \, \mathbf{p}' = 0, \quad \text{kjer} \quad \mathbf{p} = [u, v, 1] \text{ in } \mathbf{p}' = [u', v', 1]$$

- Matrika E – osnovna matrika (*essential matrix*), velikosti 3x3
- Produkt $\mathsf{E}\,\mathbf{p}'$ interpretiramo kot koordinatni vektor, ki predstavlja epipolarno premico povezano s točko p' v prvi sliki (tj. premica l)

$$\mathsf{E}\mathbf{p}' = [a, b, c]' \qquad \text{premica } l : au + bv + c = 0$$

  - ⋆ podobno velja za produkt $\mathsf{E}\mathbf{p}$, vendar za drugo sliko
- Matrika E je definirana kot

$$\mathsf{E} = [\mathbf{t}_\times]\mathsf{R}\,,$$

  kjer je $\mathbf{t}$ vektor OO', R je rotacijska matrika, ki poveže drugi pogled s prvim
  - ⋆ matrika $[\mathbf{a}_\times]$ je definirana kot

$$[\mathbf{a}_\times] = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

  - ⋆ poljuben vektor $\mathbf{w}'$ (predstavljen v drugem koordinatnem sistemu) je v prvem koordinatnem sistemu izražen kot $\mathsf{R}\mathbf{w}'$

Dva pogleda (splošno) *(4)*

B. Neumerjene oz. nekalibrirane razmere

- Notranji parametri obeh kamer niso poznani!
- Velja:

$$\mathbf{p}^T \, \mathsf{F} \, \mathbf{p}' = 0$$

- Matrika F – temeljna matrika (*fundamental matrix*), velikosti 3x3
- Matrika F je definirana kot

$$\mathsf{F} = \mathsf{K}^{-T} \mathsf{E} \mathsf{K}'^{-1}$$

  kjer sta matriki K in K′ matriki notranjih parametrov oz. kalibracijski matriki kamer
- Izraz F$\mathbf{p}'$ prav tako določa epipolarno premico l (izraz F$\mathbf{p}$ pa premico l').

1. Umerjanje oz. kalibracija

- Postopek določitve matrike F imenujemo umerjanje oz. kalibracija.
- Matriko F izpišemo kot

$$\mathsf{F} = \begin{bmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{bmatrix}$$

- Za določitev matrike F moramo določiti korespondenčne točke:
  - ⋆ tj. par ujemajočih se točk $(\mathbf{p_i}, \mathbf{p'_i})$, kjer $\mathbf{p_i}$ točka iz prve slike, $\mathbf{p'_i}$ pa pripadajoča točka iz druge slike

2. Algoritem osmih točk

- F_{33} postavimo na 1.
- Izbrati je treba osem korespondenčnih točk (parov!).
- Rešiti moramo naslednji nehomogeni sistem linearnih enačb:

$$\begin{bmatrix} u_1u'_1 & u_1v'_1 & u_1 & v_1u'_1 & v_1v'_1 & v_1 & u'_1 & v'_1 \\ u_2u'_2 & u_2v'_2 & u_2 & v_2u'_2 & v_2v'_2 & v_2 & u'_2 & v'_2 \\ u_3u'_3 & u_3v'_3 & u_3 & v_3u'_3 & v_3v'_3 & v_3 & u'_3 & v'_3 \\ u_4u'_4 & u_4v'_4 & u_4 & v_4u'_4 & v_4v'_4 & v_4 & u'_4 & v'_4 \\ u_5u'_5 & u_5v'_5 & u_5 & v_5u'_5 & v_5v'_5 & v_5 & u'_5 & v'_5 \\ u_6u'_6 & u_6v'_6 & u_6 & v_6u'_6 & v_6v'_6 & v_6 & u'_6 & v'_6 \\ u_7u'_7 & u_7v'_7 & u_7 & v_7u'_7 & v_7v'_7 & v_7 & u'_7 & v'_7 \\ u_8u'_8 & u_8v'_8 & u_8 & v_8u'_8 & v_8v'_8 & v_8 & u'_8 & v'_8 \end{bmatrix} \begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{bmatrix} = - \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- Če pa izberemo $n > 8$ korespondenčnih točk, potem lahko matriko F ocenimo z uporabo linearne metode najmanjših kvadratov, kjer minimiziramo izraz

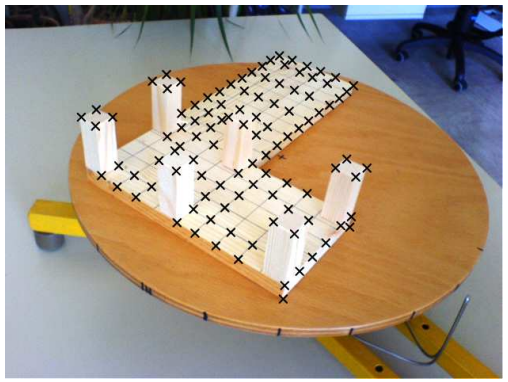
$$\sum_{i=1}^{n} \left(\mathbf{p_i}^T \mathsf{F} \mathbf{p_i}' \right)^2$$

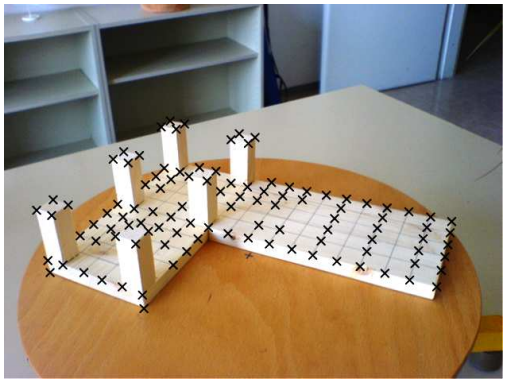
Dva pogleda (splošno) *(6)*

- Raziskave so pokazale, da dobimo boljše rezultate, če algoritem normaliziramo:
  1. podatke (točke) v obeh slikah premaknemo tako, da je center podatkov v koordinatnem izhodišču
  2. podatke nato še skaliramo tako, da je povprečna razdalja do koordinatnega izhodišča enaka $\sqrt{2}$ piksla
- Algoritem daje povprečne rezultate – boljši so robustni postopki kalibracije:
  - npr. metoda M-cenilk, metoda RANSAC, metoda LMedS (najmanjša mediana kvadratov)

3. Kako določiti korespondenčne točke za umerjanje kamer?

- Pare korespondenčnih točk lahko izberemo:
  1. ročno – ročno določimo položaje točk na slikah
  2. avtomatsko – v slikah iščemo določene karakteristike (npr. oglišča objektov)
- Paziti moramo, da so korespondenčne točke izbrane oz. razpršene po vsej sliki (tj., da ne ležijo preveč skupaj)!
  - sicer postane umerjanje nestabilno



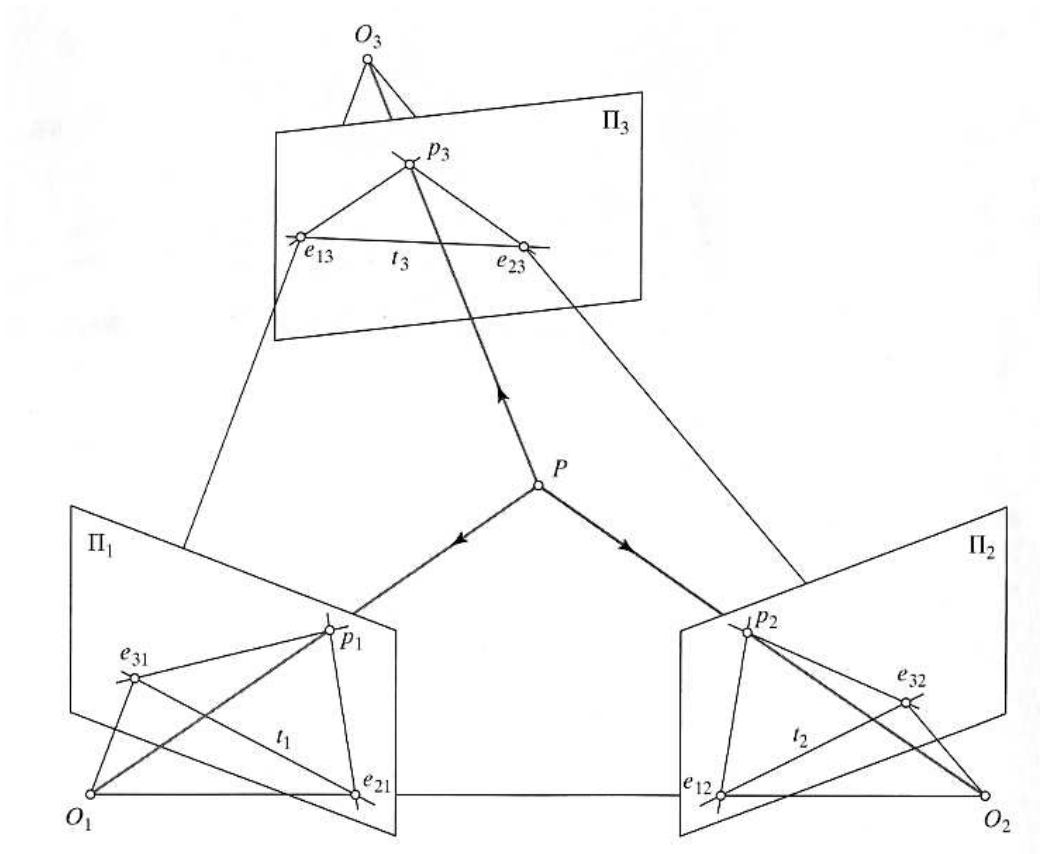


Trije pogledi

- Sceno opazujemo s tremi perspektivnimi kamerami.
- O_1, O_2 in O_3 – optični centri kamer
- Opazovana točka P; p_1, p_2 in p_3 – projekcije te točke na ustreznih kamerah
- Π_1, Π_2 in Π_3 – virtualne slikovne ravnine

- Vsak par kamer definira svojo epipolarno omejitev.
- Obravnavamo torej 3 neodvisne enačbe:

$$\begin{aligned} \mathbf{p_1}^T \mathsf{E}_{12}\, \mathbf{p_2} &= 0 \\ \mathbf{p_2}^T \mathsf{E}_{23}\, \mathbf{p_3} &= 0 \\ \mathbf{p_3}^T \mathsf{E}_{31}\, \mathbf{p_1} &= 0 \end{aligned}$$

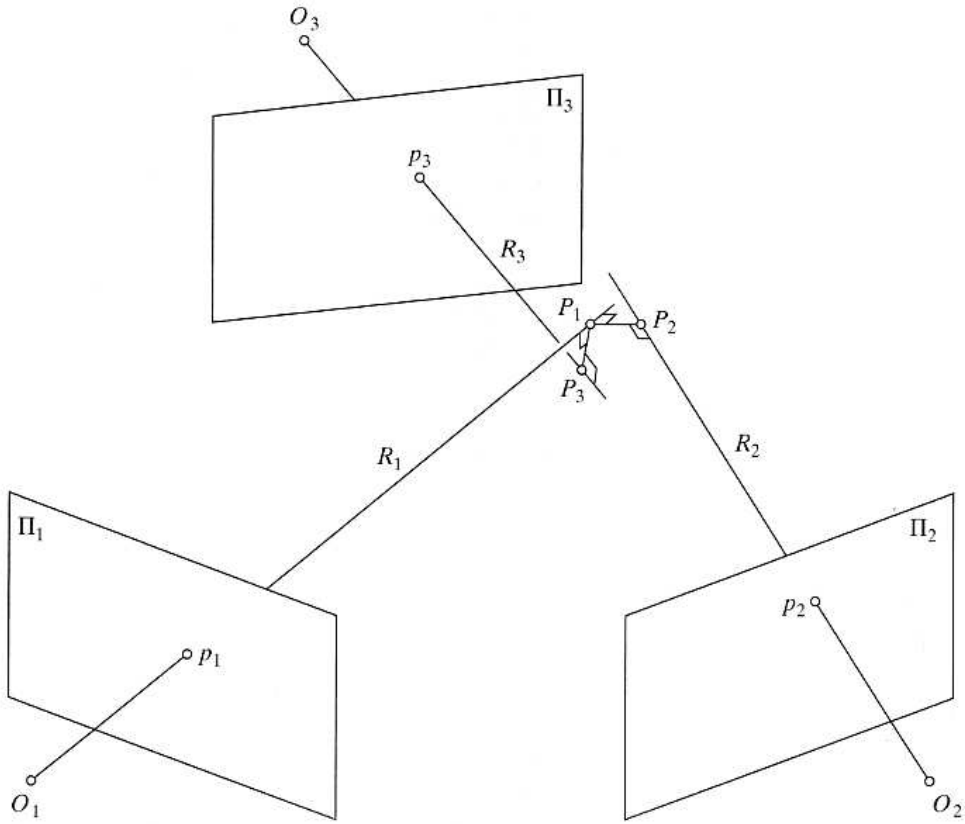


- E_{ij} – osnovna matrika prirejena slikama i in j

- Podane enačbe veljajo za umerjene zadeve, pri neumerjenih zadevah bi namesto osnovne matrike uporabili temeljno matriko (F_{ij}).

Določanje položaja točk v 3D

- V praksi se žarki R_1, R_2 in R_3, ki so prirejeni slikovnim točkam p_1, p_2 in p_3, ne sekajo!
- Položaj točke P v prostoru določimo: daljica PP_2 je pravokotna na žarka R_1 in R_2, daljica PP_3 pa na žarka R_1 in R_3.
- Enako situacijo imamo zgolj pri dveh pogledih!



A. Kako določiti položaj točke v praksi?

1. Postopek za 2 kameri

- Točki P_1 in P_2 sta približka za pravi položaj opazovane točke P.
- Poiščemo najkrajšo razdaljo med žarkoma R_1 in R_2:
  - ⋆ tj. daljica, ki je pravokotna na oba žarka
- Položaj točke P določimo na sredini te najkrajše razdalje!

Določanje položaja točk v 3D *(2)*

2. Postopek za 3 kamere

- Točke P_1, P_2 in P_3 so približki za pravi položaj opazovane točke P.
- Za vsak par kamer uporabimo postopek za 2 kameri
  - dobimo 3 približne položaje opazovane točke P v prostoru
  - te 3 točke določajo trikotnik
- Položaj točke P določimo kot težišče tega trikotnika!

Stereo vid

A. Rekonstrukcija

- Imejmo dve ujemajoči se slikovni točki p in p' opazovane točke P v prostoru.
- **Teoretično:** položaj točke P določimo kot presečišče žarka $R = Op$ in $R' = O'p'$.
- **Praksa:** žarka R in R' se nikoli ne sekata
  - vzrok: napake pri umerjanju in lokaliziranju značilnic na slikah
- **1. rešitev**
  - točko P postavimo na sredino daljice z najkrajšo razdaljo med žarkoma
- **2. rešitev**
  - matriki M in M′ sta projekcijski matriki
  - p in p' sta ujemajoči se slikovni točki
  - izraz $\mathbf{p} = \frac{1}{z}\mathrm{M}\mathbf{P}$ lahko zapišemo kot (isto seveda velja za drugi pogled!)

$$\mathbf{p} \times \mathrm{M}\mathbf{P} = 0$$

  - na osnovi tega dobimo naslednji sistem:

$$\begin{bmatrix} [\mathbf{p}_\times]\mathrm{M} \\ [\mathbf{p'}_\times]\mathrm{M'} \end{bmatrix} \mathbf{P} = \mathbf{0}_{4\times 1}$$

  - rešiti moramo torej predeterminiran sistem 4 neodvisnih linearnih enačb!

Stereo vid (2)

B. Primer paralelnih kamer

- Poseben primer stereo kamer (konfiguracije):
  - ⋆ fizikalni mrežnici (ravnini) sta horizontalno odmaknjeni in sta koplanarni
  - ⋆ kameri imata identično goriščno razdaljo f

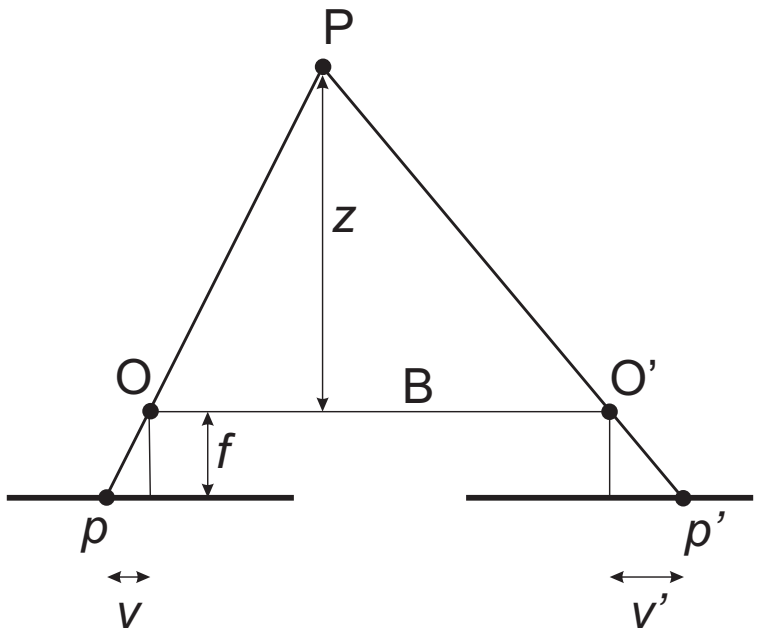
$$\mathbf{p} = [u, v] \qquad \mathbf{p}' = [u', v']$$

- Vrednost razlike med točkama (*disparity value*) d:

$$d = v' - v$$

- Pozor: u in u' sta enaka!
- Absolutna oddaljenost z točke P je odvisna od razlike d

$$z = \frac{fB}{d}$$



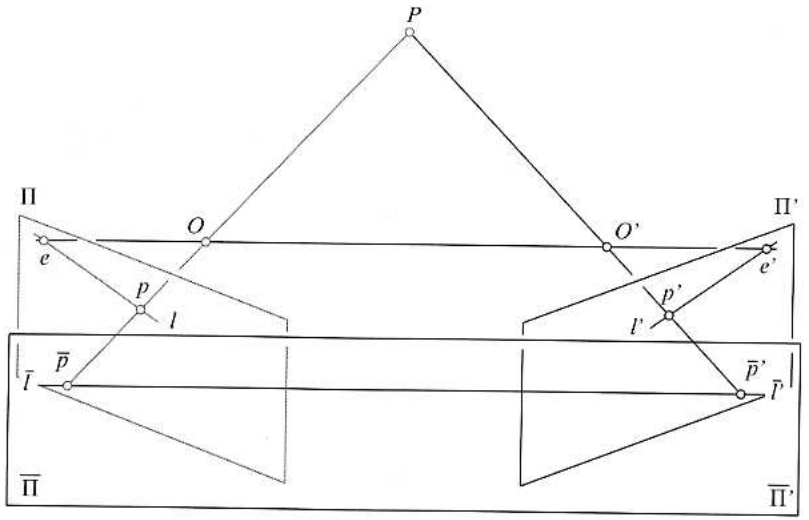
- Epipolarne premice sovpadajo s horizontalnimi linijami slike (tj. vrsticami) in so tudi vzporedne z osnovnico (tj. premica, ki povezuje optična centra), epipola sta v neskončnosti!

C. Rektifikacija oz. izravnavanje slik

- Rektifikacija je proces, ki originalni sliki obeh kamer (lahko poljubno postavljeni!) zamenja z ekvivalentnima slikama, ki imata skupno slikovno ravnino, le-ta pa je paralelna z osnovnico, ki povezuje optična centra.
- S tem postopkom pridemo na umeten način do situacije paralelnih kamer!
  - ⋆ Velja vse, kar smo povedali pri paralelnih kamerah (tudi enačbe)!
  - ⋆ Rekonstrukcija točke P v koordinatnem sistemu 1. kamere (po rektificiranju):

$$\mathbf{P} = z\mathbf{p} = -\frac{fB}{d}\mathbf{p}, \qquad \mathbf{p} = [u, v, 1]$$

- Rektifikacija je preslikava, ki ima 2 parametra:
  1. oddaljenost nove slikovne ravnine od osnovnice
  2. smer normale – izbira smeri normale te nove rektificirane ravnine na ravnini, ki je pravokotna na osnovnico
     - ⋆ običajno: izberemo ravnino, ki je vzporedna premici, kjer se sekata originalni sliki

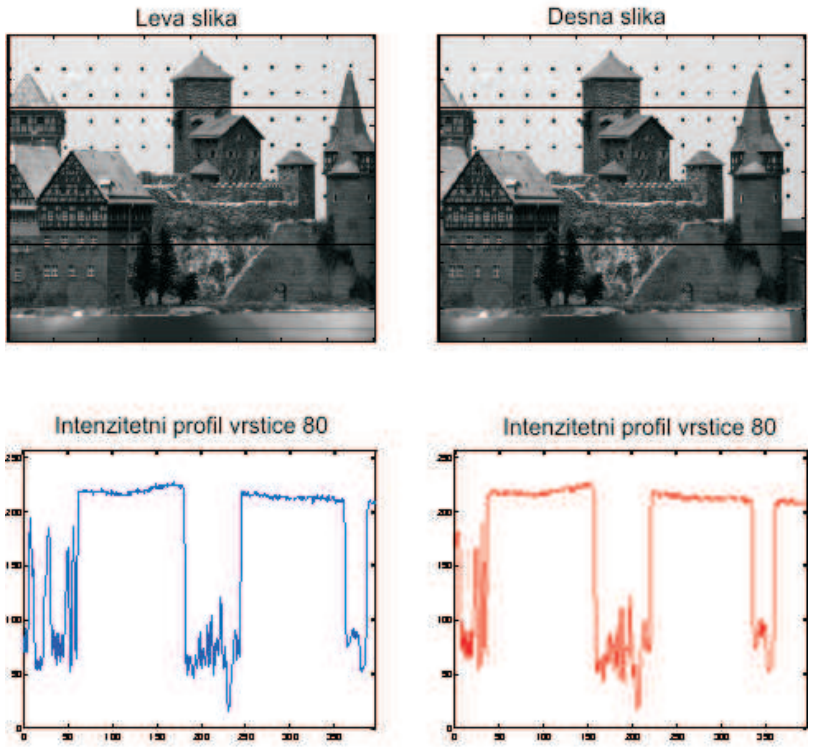


D. Ujemanje stereo slik (paralelne kamere)

- Gre za reševanje problema: kjer se nahaja določen del prve (tj. leve) slike na drugi (tj. desni) sliki.

Možni pristopi:

1. Iskanje na osnovi intenzitet sivin
   - primerjamo intenzitetne profile vrstic leve in desne slike ter poiščemo najboljšo razliko (premik) d
2. Iskanje ujemanja na osnovi značilnic
   - najprej v obeh slikah poiščemo značilnice, zatem pa iščemo ujemanje med temi značilnicami iz obeh slik
   - največkrat uporabljene značilnice: robovi, oglišča, linije (daljice), krivuljni segmenti, krogi/elipse, regije (poljubne)



Stereo vid *(5)*

Pri iskanju ujemanja upoštevamo razne omejitve:

- Podobnost (kompatibilnost)
  - podobne vrednosti sivin oz. značilnic v obeh slikah
- Enoličnost
  - en piksel oz. značilnica se ujema z enim pikslom oz. značilnico z druge slike
- Zveznost
  - vrednost razlike (*disparity*) se skozi sliko spreminja gladko
- Urejanje
  - če sta ujemajoča se para (m, m') in (n, n') potem, če je m levo od n, potem bo tudi m' levo od n' (in obratno)
- Epipolarna omejitev
  - ujemajoča se točka lahko leži le na pripadajoči epipolarni premici
  - ta omejitev je najbolj zanesljiva!

7. Uvod v razpoznavanje vzorcev

Definicije in osnovni pojmi

Model razvrščevalnika vzorcev

Zapisi vzorcev

Spoznavanje področja uporabe

Razvrščanje vzorcev

Preizkušanje razvrščevalnika vzorcev

Definicije in osnovni pojmi

- **Disciplina "razpoznavanje vzorcev":**
  1. Je znanstvena veda, ki se ukvarja z umetnimi zaznavnimi sistemi (stroji), ki lahko z gledanjem, s poslušanjem, s tipanjem in podobno s simboli opisujejo okolje, ki jih obkroža.
  2. Je proces kategoriziranja vzorca, izmerjenega ali opazovanega podatka kot člana enega izmed več razredov oz. kategorij. Je interdisciplinarna veda.
- **Okolje $\mathcal{O}$:**
  - ⋆ Je množica predmetov, pojavov in bitij (tj. objektov), v prostoru in času, ki jih razpoznavamo.
  - ⋆ Razpoznavalni sistem, ki bi se zavedal celotnega okolja, je danes še neizvedljiv.
- **Področje uporabe $\mathcal{PU}$:**
  - ⋆ Vsebuje samo tiste objekte in njihove medsebojne zveze in odnose, ki jih razpoznavamo, torej $\mathcal{PU} \subset \mathcal{O}$.
  - ⋆ **Primer:** razpoznavanje števk med 0 in 9 ter velikih in malih črk slovenske abecede; razpoznavanje registrskih tablic; kontrola kvalitete zamrznjenega sadja...

Definicije in osnovni pojmi *(2)*

- **Razred objektov:**
  - Je podmnožica tistih objektov iz danega področja uporabe $\mathcal{PU}$, na katere se nanaša oznaka (simbol, ime razreda) iz množice oznak razredov objektov.
  - Število razredov objektov (M), pri čemer $M \geq 2 \in \mathbb{N}$, je določeno z nalogo razpoznavalnega sistema.
  - Razredi objektov so tuje in neprazne množice. Njihova unija je enaka $\mathcal{PU}$.
  - **Primer:** $\mathcal{PU}$ je razpoznavanje določenih vrst sadja; razredi objektov so npr. jabolka, hruške, slive ter razred neznanih objektov
- **Vzorec:**
  - Je vsebina čutil, tj. odčitek merilnih naprav, ki daje organizmu (stroju) podatke o objektu ali o objektih in njihovih medsebojnih zvezah in odnosih.
  - Vzorce lahko popišemo z
    - 1) numeričnimi (digitalnimi) funkcijami, 2) množico vrednosti meritev, 3) množico vrednosti značilnic, 4) strukturirano množico oznak - imen podvzorcev ali 5) obrazci za zapis znanja.
    - Začetni opis vzorca (1 in 2) in izpeljani zapis vzorca (3 do 5).
  - Preprost in zapleten vzorec.
- **Razred vzorcev:**
  - Je slika razreda objektov. Sestavljajo ga vzorci, ki so slike objektov iz razreda objektov.
  - Veljajo vse trditve kot pri razredu objektov.
  - V posameznem razredu vzorcev so najbolj podobni si vzorci.

Definicije in osnovni pojmi *(3)*

- ▷ **Spoznavanje področja uporabe:**
  - ⋆ Je predpogoj za razpoznavanje objektov.
  - ⋆ Spoznavanje področja uporabe iz danih vzorcev je preslikava, ki preslika množico vzorcev objektov razpoznavanja v učno množico vzorcev $\mathcal{U}_M$.
- ▷ **Učna množica vzorcev $\mathcal{U}_M$:**
  - ⋆ Je končna množica vzorcev z danega področja uporabe $\mathcal{P}U$, iz katere se stroj (samodejno) nauči zvez med oznakami razredov in med objekti razpoznavanja.
  - ⋆ $\mathcal{U}_M$ je par

$$\mathcal{U}_M = (\mathcal{S}_N, \Omega)$$

$\mathcal{S}_N$ – množica vzorcev, ki opisuje področje uporabe (velikosti N)
Ω – množica oznak razredov objektov na danem področju uporabe.

  - ⋆ Vzorce iz $\mathcal{S}_N$ je v procesu spoznavanja področja uporabe označil učitelj.
  - ⋆ $\mathcal{U}_M$ mora zajemati vzorce iz vseh možnih razredov vzorcev.
- ▷ **Učenje:**
  - ⋆ Je proces, v katerem se vzpostavijo zveze med vzorci iz učne množice vzorcev $\mathcal{S}_N$ in med oznakami razredov vzorcev Ω.
  - ⋆ Pogosto učimo z nepopolno indukcijo. Pravila, lastnosti in relacije, ki veljajo med vzorci znotraj posameznega razreda ter med razredi vzorcev v $\mathcal{U}_M$, posplošimo na celotno $\mathcal{P}U$.
  - ⋆ Z učenjem izluščimo iz $\mathcal{U}_M$ pravila za posamezen razred vzorcev.

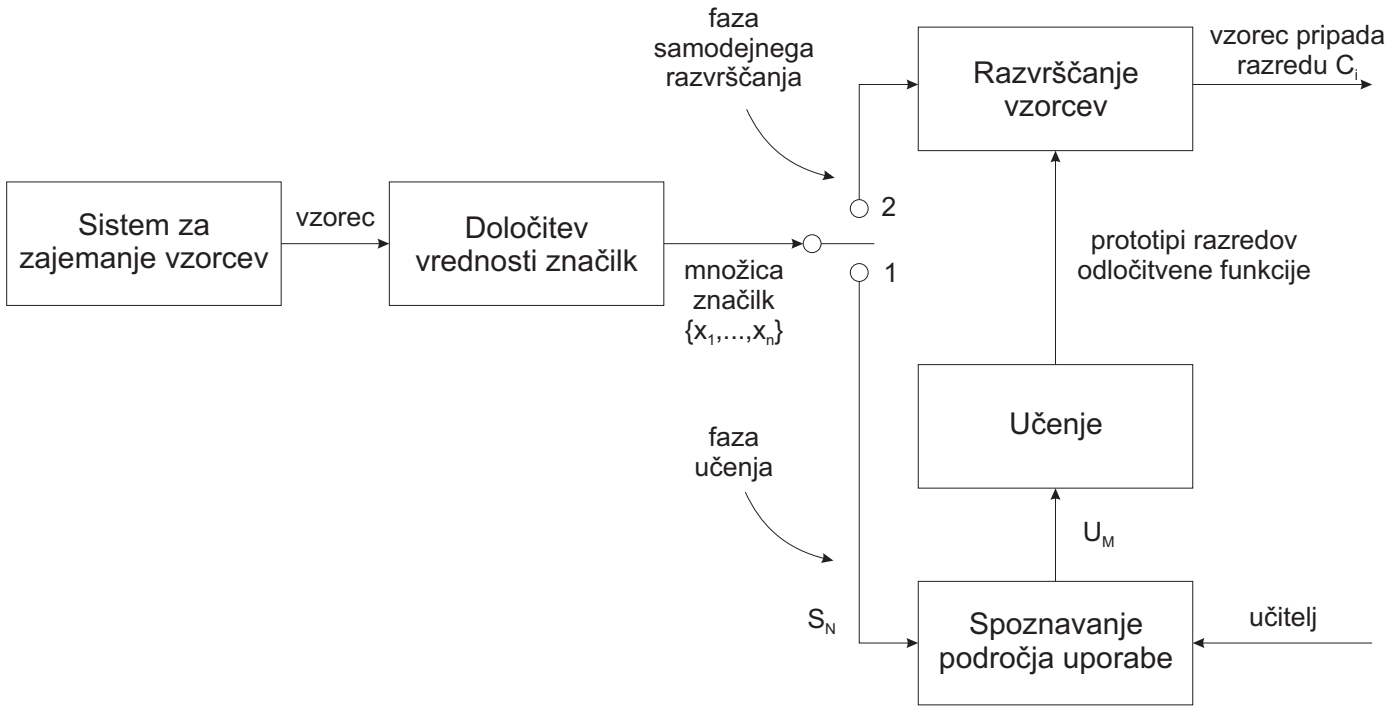
Definicije in osnovni pojmi *(4)*

- **Razvrščanje (klasifikacija) vzorcev:**
  - Je proces razporejanja vzorcev v razrede, ki so sestavljeni iz že prej razvrščenih, medseboj podobnih si vzorcev.
  - Razvrščamo s prileganjem, odločanjem, stavčno analizo ali z logičnim sklepanjem.
  - Postopek je odvisen od tipa zapisa vzorcev in temelji na podobnosti vzorcev, tj. merjenju razdalje med vzorci.
- **Razpoznavanje vzorcev:**
  - Predstavlja zadnjo fazo procesa zaznavanja, v kateri se določi istovetnost ali velika podobnost nove vsebine čutil z vsebino čutil, ki je že bila spoznana in zapomnjena.
  - V spominu iščemo tisti pojem, ki se nanaša na objekt razpoznavanja, želimo pa tudi zgraditi logično sliko dejstev o področju uporabe.
- **Primer:** razlika med razvrščanjem in razpoznavanjem vzorcev

Beseda v stavku	Pomen
Z	Skupaj s kom
Lučko	osebo ženskega spola
tečeva	hitiva ali se nama mudi
na kavo	na pijačo
v As.	v kavarno ali gostilno.

Model razvrščevalnika vzorcev

- Prikazan model za vzorce zapisane v obliki nestrukturirane množice vrednosti značilnic.



- Faza učenja (stikalo v položaju 1):
  - ⋆ Nadzorovano in nenadzorovano učenje
- Faza samodejnega razvrščanja (stikalo v položaju 2):
  - ⋆ Na samem začetku se izvede preizkus razpoznavalnika.

Zapisi vzorcev

A. Začetni zapis vzorcev

- Vzorec je zapisan kot množica meritev, ki so rezultat merjenja ene ali več količin (lastnosti), ki zaznamujejo objekte razpoznavanja:
  - vzorec je predstavljen kot eno- ali večdimenzionalna matrika meritev (označimo ga z $\mathbf{m}$)
  - **Primer:** začetni zapis vzorca za slivo

$$\mathbf{m} = [30\ g, 5\ cm, vijolična].$$

- Vzorci so podatki, zapisani z realnimi ali nenegativnimi števili, ter organizirani v eno- ali večdimenzionalno polje podatkov.
- Tipi podatkov:
  1. **Kvalitativni podatki**
     - nominalni podatki – jih ni moč urejati (npr. seznam imen barv)
     - ordinalni podatki – jih lahko urejamo (npr. majhen, srednji in velik).

     Aritmetične matematične operacije s to vrsto podatkov niso dovoljene!
  2. **Kvantitativni podatki**
     - absolutni podatki (npr. masa, merjena v kilogramih: 0 kg mase je hkrati tudi najmanjša možna masa)
     - relativni podatki (npr. temperatura, merjena v stopinjah Celzija)

     Aritmetične matematične operacije so dovoljene.

Zapisi vzorcev *(2)*

B. Zapis preprostih vzorcev z množico značilnic

- Pri zapisu vzorca v obliki množice vrednosti značilnic od meritev obdržimo ali iz njih izpeljemo le najpomembnejše lastnosti objekta glede na nalogo, ki jo rešujemo.
  - Bistvene lastnosti objektov so tiste, ki poudarjajo posebnosti posameznih razredov vzorcev.
  - Takšno lastnost objektov oz. vzorcev imenujemo značilnica.
- Pristopa za določanje značilnic iz preprostih vzorcev: 1) hevristični in 2) matematični pristop.

1. Hevristični pristop

- Določanje značilnic temelji na izkušnjah strokovnjakov oz. na posnemanju procesa razpoznavanja pri živih bitjih.
- Eksperti opredelijo, katere lastnosti objekta so značilnice.

Zapisi vzorcev *(3)*

2. Matematični pristop

- Pristop temelji na informacijsko zelo bogatem začetnem opisu objektov razpoznavanja
  - Za vsak nov objekt določimo čim več lastnosti.
- Nato z matematičnimi postopki izločimo odvečne podatke iz začetnega zapisa:
  - Temeljijo na optimizaciji ustrezne kriterijske funkcije napake.
- Ločimo optimalne in neoptimalne postopke:
  - Optimalni postopki poiščejo globalni minimum kriterijske funkcije, neoptimalni postopki pa zgolj lokalni minimum.

Optimalni postopki določanja značilnic

- Vzorec $\mathbf{m}$ – vzorec opisan z vrednostmi r spremenljivk oz. meritev:

$$\mathbf{m} = [m_1, m_2, \ldots, m_r]$$

- Vzorec $\mathbf{x}$ – vzorec opisan v obliki množice vrednosti značilnic

$$\mathbf{x} = [x_1, x_2, \ldots, x_n],$$

kjer $\mathbf{x} \in \mathbb{R}^n$ in n je število značilnic vzorca.

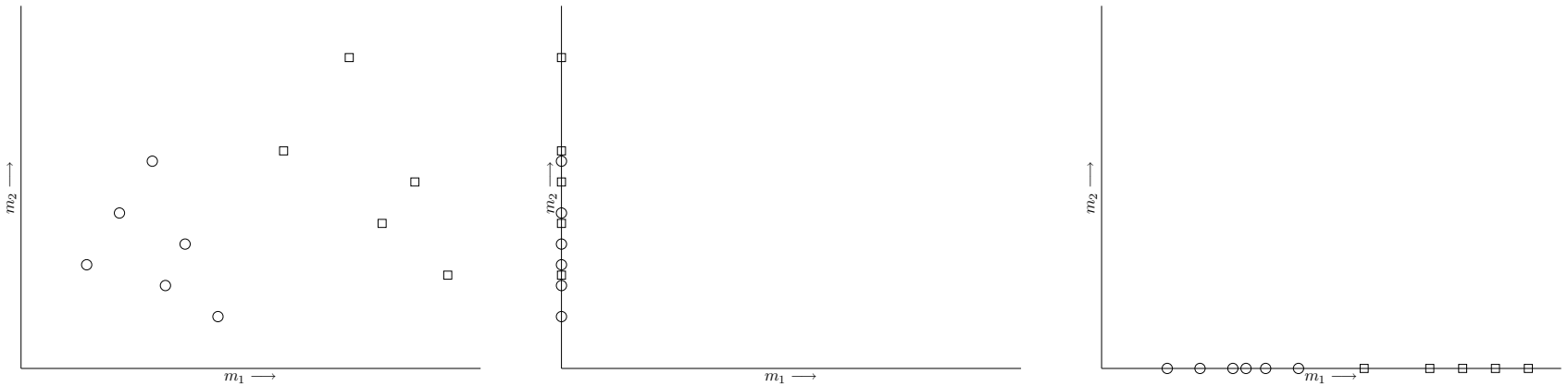
Zapisi vzorcev *(4)*

- ▷ **Cilj:** Poiskati le tisto informacijo iz množice meritev $\mathbf{m}$, ki je pomembna za razvrščanje vzorcev.
- ▷ Najboljša oz. optimalna množica značilnic je tista množica, ki v novem, manj razsežnem prostoru ne poveča verjetnosti napačnega razvrščanja vzorcev s teoretično najboljšim razvrščevalnikom (tj. Bayesov razvrščevalnik).
- ▷ Ločimo: a) postopke izbire značilnic ter b) postopke izpeljave značilnic.

a) Izbira značilnic

- ▷ Iz množice meritev izločimo spremenljivke preprostih vzorcev, ki ne prispevajo k natančnosti razpoznavanja oz. ne prispevajo k boljši ločljivosti razredov vzorcev v prostoru $\mathbb{R}^n$.
- ▷ Izbrane značilnice vzorcev ohranijo pomen in enoto izbranih meritev.

Zgled: dva razreda vzorcev, vsak vzorec opisan z dvema spremenljivkama

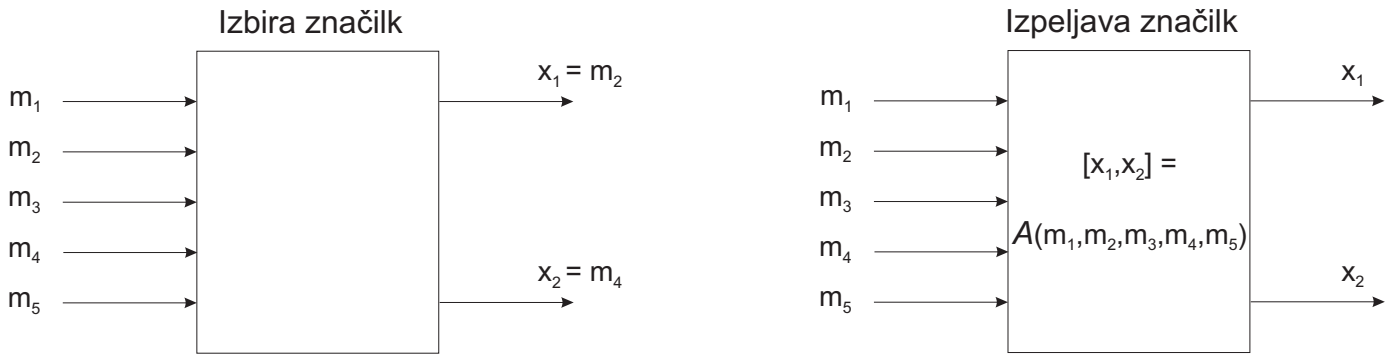


Zapisi vzorcev (5)

b) Izpeljava značilnic

- Gre za preslikavo r-dimenzionalnega prostora meritev v n-razsežni prostor značilnic, pri čemer je $n < r$.
- V novem prostoru se mora ohraniti (ali zgolj minimalno zmanjšati) stopnja ločljivosti razredov.
- Izpeljane značilnice ne ohranijo niti pomena niti enot meritev.
- Poznani postopki izpeljave značilnic: analiza glavnih komponent (PCA–principal component analysis) oz. dekompozicija na singularne vrednosti (SVD–singular value decomposition) oz. Karhunen-Loèvejeva transformacija, analiza neodvisnih komponent itd.

Zgled: Razlika med postopkoma izbire in izpeljave značilnic.



Spoznavanje področja uporabe

- Temelji na samodejnem razvrščanju (rojenju) vzorcev iz končne množice vzorcev $\mathcal{S}_{\mathcal{N}}$ ter na označevanju rojev vzorcev z oznakami razredov vzorcev.
- Je preslikava, ki preslika množico $\mathcal{S}_{\mathcal{N}}$ v učno množico $\mathcal{U}_{\mathcal{M}}$
  - ⋆ Preslikava ni avtomatična, saj označevanje vzorcev opravi učitelj.
- Opravimo ga v fazi učenja.
- Rezultat: množico "učnih" vzorcev avtomatsko razdeli v razrede oz. roje, učitelj pa priredi tem razredom oznake.
  - ⋆ To preslikavo oz. razbitje imenujemo rojenje.

A. Merjenje podobnosti med vzorci

- Rojenje temeljimo na merjenju podobnosti vzorcev (tj. razdalje med vzorci).
- **Evklidska razdalja:**

$$d(\mathbf{x_i}, \mathbf{x_j}) = \|\mathbf{x_i} - \mathbf{x_j}\|_2 = \sqrt{\sum_{k=1}^{n} (x_{i,k} - x_{j,k})^2}, \quad \mathbf{x_i}, \mathbf{x_j} \in \mathbb{R}^n, \tag{5}$$

- **Razdalja Manhattan ("City block"):**

$$d(\mathbf{x_i}, \mathbf{x_j}) = |\mathbf{x_i} - \mathbf{x_j}| = \sum_{k=1}^{n} |x_{i,k} - x_{j,k}|$$

Spoznavanje področja uporabe *(2)*

B. Predobdelava množice vzorcev

1. Ali kakšni značilnici vzorca ni prirejena (ji manjka) vrednost?
   - ▷ **Rešitev:** a) vzorec izločimo iz $\mathcal{S}_N$ ali b) manjkajočo vrednost nadomestimo s povprečno vrednostjo te značilnice, izračunane iz preostalih vzorcev.
2. Ali vrednost značilnice pri posameznem vzorcu zelo odstopa od izračunane povprečne vrednosti za to značilnico?
   - ▷ **Rešitev:** a) vzorec izločimo iz $\mathcal{S}_N$ ali b) uporabimo drugačno razdaljo med vzorci, ki ne poudarja razlike vrednosti značilnic.

Kdaj vrednost značilnice x_j "zelo" odstopa od povprečja?

- ▷ Izračunamo tri pomožne izraze za vsako značilnico j:
  - ⋆ Povprečna vrednost μ_j in varianca σ_j za j-to značilnico

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{i,j}, \quad \sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}(x_{i,j}-\mu_j)^2, \quad j = 1, 2, \ldots, n\,, \quad (6)$$

  - ⋆ Koeficient poševnosti oz. nesimetričnosti porazdelitve:

$$\Upsilon_j = \frac{\frac{1}{N}\sum_{i=1}^{N}(x_{i,j}-\mu_j)^3}{\sigma_j^3}. \quad (7)$$

Spoznavanje področja uporabe (3)

- ▷ Vrednost značilnice ”zelo” odstopa od povprečne vrednosti, če je izraz

$$\frac{|x_j - \mu_j|}{\sigma_j}$$

  - ⋆ ≥ 3, če gre za simetrične porazdelitve (tj. $|\Upsilon_j| < 0,5$)
  - ⋆ ≥ 5, če gre za asimetrične porazdelitve (tj. $|\Upsilon_j| > 0,5$).

3. Normiranje vrednosti značilnic vzorcev

- ▷ Postopek potreben, če želimo, da imajo vse značilnice v vzorcu enak doprinos pri računanju mere podobnosti.

a. Vrednosti značilnic simetrično porazdeljene okoli povprečne vrednosti μ_j (tj. $|\Upsilon_j| \approx 0$), potem normiranje z linearno transformacijo:

$$x_{i,j} = \frac{x^*_{i,j} - \mu_j}{\sigma_j}$$

  - ⋆ Skaliranje/linearizacija – če želimo značilnice na intervalu $[0, 1]$ ali $[-1, 1]$

$$x^{'}_{i,j} = s_{\min} + \frac{x_{i,j} - x_j^{\min}}{x_j^{\max} - x_j^{\min}}(1 - s_{\min})$$

$x_j^{\min}$ in $x_j^{\max}$ – minimalna ter maksimalna vrednost j-te značilnice

$s_{\min}$ – 0 za interval $[0, 1]$ oz. -1 za interval $[-1, 1]$

Spoznavanje področja uporabe *(4)*

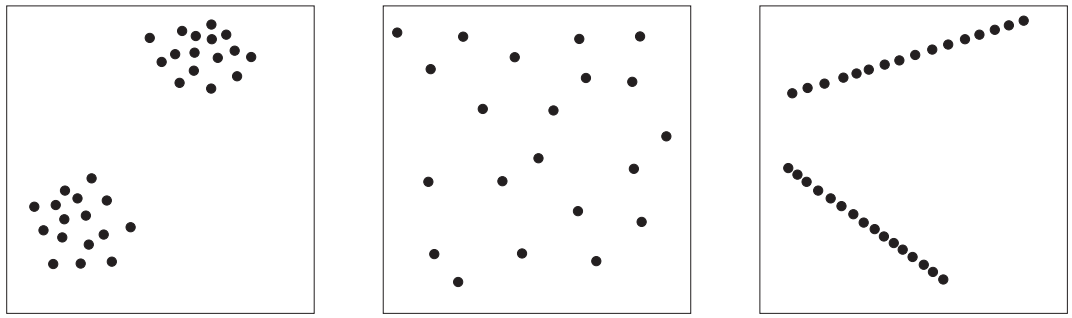
b. Vrednosti značilnic asimetrično porazdeljene okoli povprečne vrednosti μ_j (tj. $|\Upsilon_j| \neq 0$), potem normiranje z nelinearno transformacijo v dveh korakih:

$$z_{i,j} = \frac{x^*_{i,j} - \mu_j}{r\sigma_j}$$

$$x_{i,j} = \frac{1}{1 + e^{-z_{i,j}}}$$

- ⋆ Konstanto r lahko izberemo poljubno.
- ⋆ Po normiranju značilnice $x_{i,j}$ že na intervalu $[0, 1]$.

4. Ali se vzorci v prostoru značilnic porazdeljujejo naključno oz. po določenih zakonitostih (pravilih, funkcijah)?



dva roja naključno po zakonitosti

- ▷ Naključna porazdeljenost oz. porazdeljenost po določeni zakonitosti, kaže na neprimerno določitev značilnic.

Spoznavanje področja uporabe *(5)*

C. Postopek K povprečij

1. Inicializiramo K središč rojev, in sicer $\mathbf{c_1}(1), \mathbf{c_2}(1), \ldots, \mathbf{c_K}(1)$.
2. V k-ti iteraciji rojimo vzorce v K rojev, in sicer:

$$\mathbf{x} \in \mathcal{S}_i(k), \quad \text{če } \|\mathbf{x} - \mathbf{c_i}(k)\| < \|\mathbf{x} - \mathbf{c_j}(k)\| \quad \forall j = 1, \ldots, K \wedge j \neq i.$$

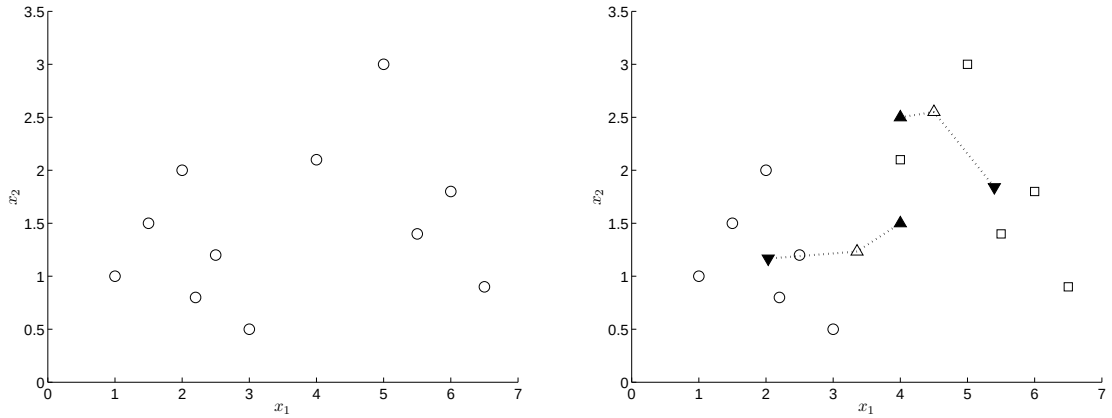
3. Izračunamo nova središča rojev v iteraciji $k+1$ kot:

$$\mathbf{c_j}(k+1) = \frac{1}{N_j} \sum_{\mathbf{x} \in \mathcal{S}_j(k)} \mathbf{x}, \quad j = 1, \ldots, K,$$

   kjer N_j določa število vzorcev v roju $\mathcal{S}_j(k)$.

4. Če $\mathbf{c_j}(k+1) = \mathbf{c_j}(k), \quad \forall j = 1, \ldots, K$, potem KONEC, sicer KORAK 2.

Zgled: 11 vzorcev in rezultat rojenja v 2 roja (▲ – začetni položaj središča)



Razvrščanje vzorcev

- ▷ Cilj: V fazi samodejnega razvrščanja dobimo na vhod razpoznavalnika neznan vzorcec $\mathbf{x}$, ki ga moramo razvrstiti v enega izmed razredov vzorcev C_i.
- ▷ Je proces razporejanja vzorcev v razrede, ki so sestavljeni iz že prej razvrščenih, medsebojno podobnih si vzorcev.
  - ⋆ Razvrščanje vzorcev temelji na merjenju podobnosti oz. merjenju razdalje med vzorci, in sicer znotraj razreda ter med razredi.
- ▷ Razvrščevalnik vzorcev vključuje znanje o posameznem razredu bodisi:
  1. z zabeleženjem in shranjevanjem vseh vzorcev posameznega razreda,
  2. z integriranjem celotnega znanja o razredu v t.i. tipičnega predstavnika razreda (šablona, pravzorec).
     - ⋆ Tipični predstavnik razreda običajno določimo kot povprečje vseh vzorcev razreda.

$$\mathbf{c_j} = \frac{1}{N_j} \sum_{k=1,\ldots,N_j} \mathbf{x_{jk}}$$

- ▷ Postopek razvrščanja je odvisen od načina predstavitve znanja.

Razvrščanje vzorcev *(2)*

A. Razvrščanje po pravilu "najbližji sosed"

- Razvrščamo na osnovi prileganja oz. merjenja podobnosti med neznanim vzorcem $\mathbf{x}$ in vzorci iz učne množice $\mathcal{U}_M$.
- Ideja: V učni množici $\mathcal{U}_M$ poiščemo vzorec, ki je najbolj podoben oz. najbližji glede na izbrano razdaljo vzorcu $\mathbf{x}$. Neznan vzorec $\mathbf{x}$ razvrstimo v razred, iz katerega prihaja njemu najbližji vzorec iz $\mathcal{U}_M$.

1. Znanje o posameznem razredu shranjeno v obliki vseh vzorcev:

$$\mathbf{x} \in \mathcal{C}_i, \quad \text{če} \min_{k=1,\ldots,N_i} d(\mathbf{x}, \mathbf{x_{ik}}) < \min_{l=1,\ldots,N_j} d(\mathbf{x}, \mathbf{x_{jl}}), \quad \forall j = 1, \ldots, M \wedge j \neq i, \tag{8}$$

N_i – število vzorcev v razredu $\mathcal{C}_i$.

2. Znanje o posameznem razredu shranjeno v obliki tipičnih predstavnikov razreda:

$$\mathbf{x} \in \mathcal{C}_i, \quad \text{če } d(\mathbf{x}, \mathbf{c_i}) < d(\mathbf{x}, \mathbf{c_j}), \quad \forall j = 1, \ldots, M \wedge j \neq i,$$

kjer je tipični predstavnik $\mathbf{c_j}$ največkrat določen kot povprečje vzorcev posameznega razreda.

- Izpeljanke osnovnega pravila:
  - "k najbližjih sosedov"
  - "(k, l) najbližjih sosedov"

Razvrščanje vzorcev *(3)*

B. Razvrščanje z odločitvenimi funkcijami

- ▷ Takšno razvrščanje razdeli prostor značilnic $\mathbb{R}^n$ v M neprikrivajočih se področij (M je enako številu razredov).
- ▷ Odločitvena funkcija je v splošnem funkcija n spremenljivk.
  - ⋆ Za vsak razred $\mathcal{C}_i$ uvedemo svojo odločitveno funkcijo φ_i, pri čemer velja:

$$\varphi_i(\mathbf{x}) > \varphi_j(\mathbf{x}), \quad \forall \mathbf{x} \in \mathcal{C}_i \quad \forall j \wedge j \neq i.$$

- ▷ Neznan vzorec $\mathbf{x}$ razvrstimo v tisti razred $\mathcal{C}_i$, za katerega vrne odločitvena funkcija φ_i največjo vrednost med vsemi odločitvenimi funkcijami.
- ▷ Mejo (t.i. ločilno mejo) med razredoma $\mathcal{C}_i$ in $\mathcal{C}_j$ zapišemo kot:

$$\varphi_{ij}(\mathbf{x}) = \varphi_i(\mathbf{x}) - \varphi_j(\mathbf{x}) = 0.$$

  - ⋆ Razvrščevalnik moramo naučiti $\binom{M}{2} = \frac{M(M-1)}{2}$ ločilnih mej!
- ▷ **Razvrščanje neznanega vzorca $\mathbf{x}$ v razred $\mathcal{C}_i$ lahko izvedemo tudi s pomočjo ločilnih mej:**

$$\mathbf{x} \in \mathcal{C}_i, \quad \text{če} \quad \varphi_{ij} > 0, \quad \forall j \wedge j \neq i.$$

- ▷ Odločitvene funkcije oz. ločilne meje največkrat zapišemo v obliki polinomov.

Linearne odločitvene funkcije

$$\varphi(\mathbf{x}) = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = w_0 + \mathbf{w}\mathbf{x}^\top$$

$\mathbf{w} = [w_1, w_2, \ldots, w_n]$ – vektor koeficientov (uteži), w_0 – prag.

- Ločilna meja φ_{ij}, ki ločuje razreda $\mathcal{C}_i$ in $\mathcal{C}_j$ v prostoru $\mathbb{R}^n$, je:

$$\varphi_{ij}(\mathbf{x}) = \varphi_i(\mathbf{x}) - \varphi_j(\mathbf{x}) = 0$$

oz. zapisano v kompaktni obliki

$$\varphi_{ij}(\mathbf{x}): \quad (\mathbf{w_i} - \mathbf{w_j})\mathbf{x}^\top = -w_{i0} + w_{j0}$$

- Ločilne meje so v prostoru značilnic $\mathbb{R}^n$ dejansko $(n-1)$-razsežne hiperravnine!
  - ⋆ $n = 1$ ločilna meja je točka
  - ⋆ $n = 2$ ločilna meja je premica
  - ⋆ $n = 3$ ločilna meja je ravnina.

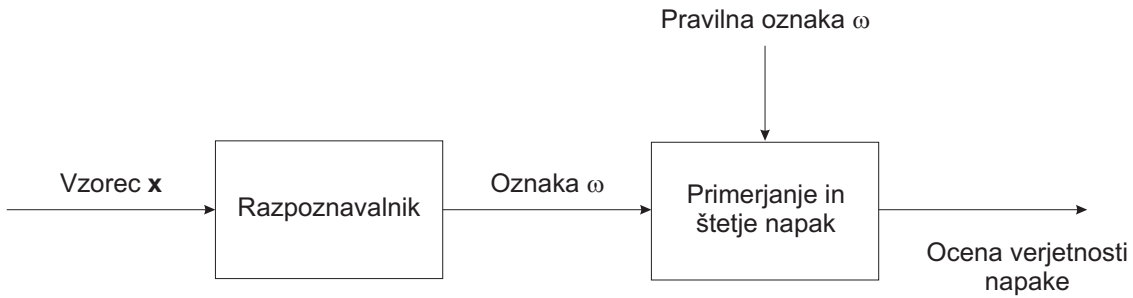
Zgled: Ločilna meja za dvodimenzionalni prostor značilnic

$$\varphi_{ij}(\mathbf{x}): \underbrace{(w_{i1} - w_{j1})}_{A} x_1 + \underbrace{(w_{i2} - w_{j2})}_{B} x_2 = \underbrace{-w_{i0} + w_{j0}}_{C}.$$

$$Ax_1 + Bx_2 = C \quad \Longrightarrow \quad x_2 = -\frac{A}{B}x_1 + \frac{C}{B}$$

Preizkušanje razvrščevalnika vzorcev

- Učinkovitost naučenega razvrščevalnika ugotavljamo na osnovi vzorcev iz t. i. preizkusne oz. testne množice vzorcev $\mathcal{T}_M$.
  - ⋆ Preizkusna množica ima sorodne lastnosti kot učna množica.
- Učinkovitost razvrščanja ocenjujemo na osnovi števila napačno razvrščenih vzorcev.



- Stopnja napačnega razvrščanja, $P(\mathbf{x} \in \mathcal{C}_i, \mathcal{C}_j)$, je verjetnost, da vzorec $\mathbf{x}$ iz razreda $\mathcal{C}_i$ napačno razvrstimo v razred $\mathcal{C}_j$.
  - ⋆ Je ne moremo natančno izračunati, ampak jo iz množice $\mathcal{T}_M$ zgolj ocenimo!

$$\hat{P}_e^i = \frac{n_i}{N_i^{\mathcal{T}}}$$

  $N_i^{\mathcal{T}}$ – število vseh vzorcev iz $\mathcal{C}_i$ v preizkusni množici $\mathcal{T}_M$, n_i – napačno razvrščeni vzorci iz $\mathcal{C}_i$ iz $\mathcal{T}_M$.

- Predikcijsko napako $\hat{P}_e$ za razvrščevalnik vzorcev pa določimo iz ocen verjetnosti napačnega razvrščanja za posamezen razred $\mathcal{C}_i$.

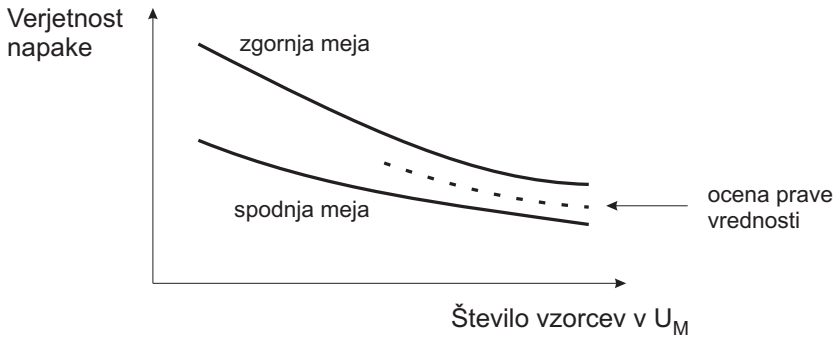
Preizkušanje razvrščevalnika vzorcev (2)

Metoda izpusti enega

- Postopek: Iz celotne množice vzorcev izpustimo en vzorec za preizkušanje, vse preostale vzorce pa uporabimo za učenje razvrščevalnika vzorcev.
- Predikcijsko napako lahko seveda zgolj ocenimo:
  - V praksi raje izračunamo spodnjo in zgornjo mejo predikcijske napake.
- **Spodnja meja predikcijske napake:**
  - Celotno učno množico najprej uporabimo za učenje, nato pa še za preizkus razvrščevalnika.
- **Zgornja meja predikcijske napake:**
  - Je povprečje N napak, kjer vsako napako izračunamo tako, da izpustimo en vzorec $\mathbf{x}_i$, ki ga uporabimo za preizkušanje, preostale pa za učenje:

$$\hat{P}_e = \frac{1}{N} \sum_i \hat{P}_e^{(\mathbf{x}_i)},$$

$\hat{P}_e^{(\mathbf{x}_i)} = 1$, če smo vzorec $\mathbf{x}_i$ napačno razvrstili z razvrščevalnikom (naučili ga brez vzorca $\mathbf{x}_i$), sicer pa 0.



8. Osnovni pojmi o nevronskih mrežah

Razlika med delovanjem računalnikov in možganov

Organski nevron

Model umetnega nevrona

Umetne nevronske mreže

Razdelitev nevronskih mrež

Razlika med delovanjem računalnikov in možganov

1. Postopek razpoznavanja
   - ▷ računalniki – se ne učijo, ampak so programirani za razpoznavanje določenih objektov
   - ▷ možgani – ljudje razpoznavamo na osnovi izkušenj (poizkušanje, analiziranje napak)
2. Čas, potreben za izvrševanje nalog razpoznavanja (razvrščanja) vzorcev
   - ▷ računalniki – ekstremno hitri (milijonkrat hitreje preklapljajo kot možgani), a določenih nalog ne morejo izvesti v realnem času
   - ▷ možgani – sestojijo iz množice nevronov, s katerimi večino razpoznavanj izvedemo hipoma
3. Zgradba in način izračunavanja
   - ▷ računalniki – a) informacijo obdelujejo zaporedno; b) sestojijo iz množice sofisticiranih komponent, a če le ena od njih odpove, je celoten sistem praktično neuporaben
   - ▷ možgani – a) informacijo obdelujejo paralelno; b) sestojijo iz množice identičnih gradnikov (tj. nevronov); kljub odpovedi mnogih nevronov, se obnašanje možganov bistveno ne spremeni

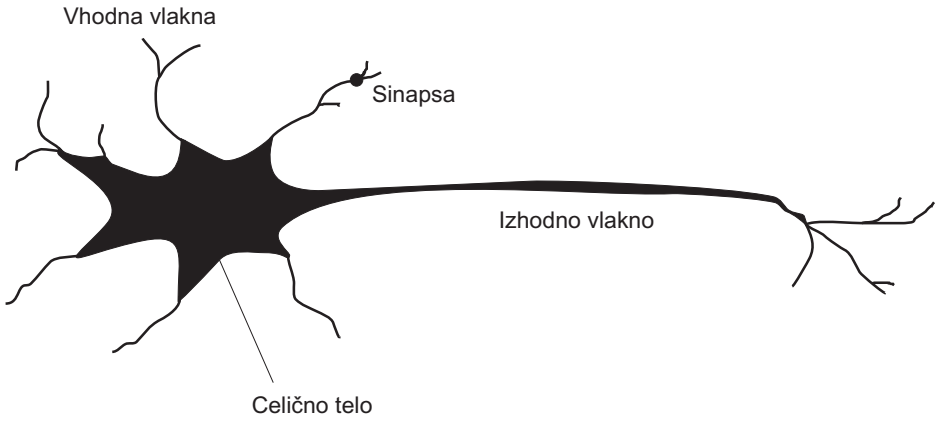
Druge pomembne prednosti možganov:

- ▷ so adaptivni sistemi; imajo sposobnost posploševanja; informacijo obdelujejo decentralizirano z redundancami; možgani so ”narejeni” iz proteinske oz. biotehnologije.

Organski nevron

Trije glavni deli nevrona:

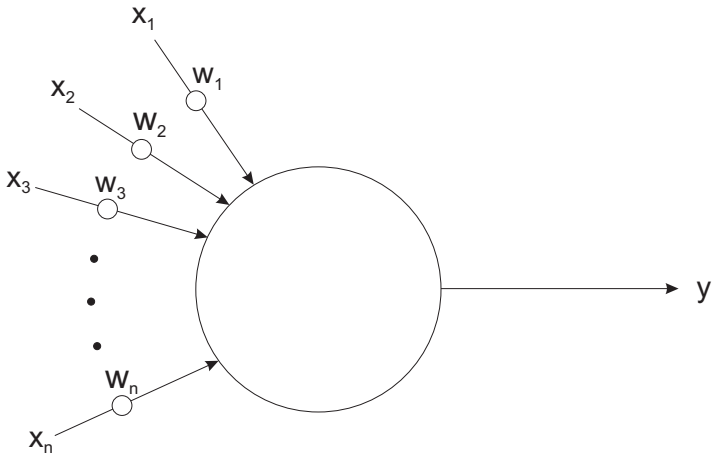
- ⋆ množica vstopajočih vlaken oz. dendritov
- ⋆ celično telo oz. soma
- ⋆ eno izstopajoče vlakno oz. akson.



- ▷ Človeški možgani vsebujejo približno 10^{11} nevronov.
- ▷ Teoretično lahko ima nevron tudi do 10.000 vhodnih povezav.
- ▷ Sinapsa – mesto, kjer se akson prejšnjega nevrona in dendrit trenutnega nevrona stikata.
- ▷ Nevroni generirajo električne impulze, ki se prenašajo vzdolž aksona do sinaps drugega nevrona:
  - ⋆ Od sinaps odvisno ali gre za vzbujanje (+ učinkovitost sinaps) ali zaviranje tega nevrona (–).
  - ⋆ Učinkovitost oz. obnašanje sinaps se med delovanjem nevrona lahko spreminja.
- ▷ Frekvenca električnih impulzov variira od nekaj impulzov na sekundo do 100 impulzov na sekundo.

Model umetnega nevrona

- ⋆ 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)



- ▷ Vhod x_i modelira prisotnost ter frekvenco vlaka električnih impulzov v vlaknu:
  - ⋆ binarne vrednosti – 1 - prisotnost vlaka električnih impulzov, 0 - odsotnost
  - ⋆ realne vrednosti (npr. med 0 in 1) – modeliranje frekvence električnih impulzov
- ▷ Obtežen vhod v celično telo – seštevek električnih impulzov iz vseh dendritov

$$s = \sum_i w_i x_i$$

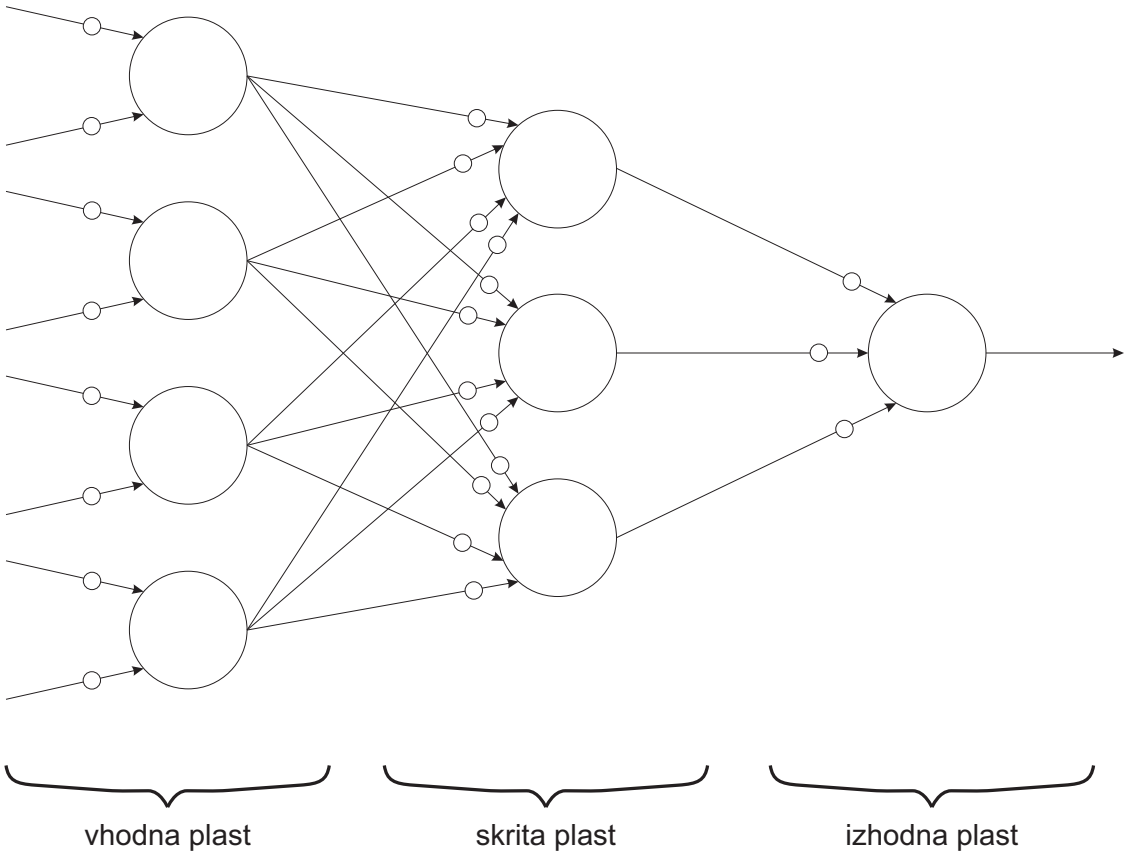
- ▷ Izhod y – monotono naraščajoča funkcija obteženega vhoda s

$$y = f(s)$$

  - ⋆ Funkcija f – prenosna oz. aktivacijska funkcija nevrona
  - ⋆ Najbolj znane: pragovna (McCulloch-Pittsov model), linearna ter sigmoidna funkcija.

Umetne nevronske mreže

- Nevronska mreža (NN) je množica medsebojno povezanih nevronov.
- **topologija nevronske mreže** – razmestitev oz. organiziranost nevronov v mreži



Razdelitev nevroskih mrež

1. Glede na tip podatkov na vhodnih in izhodnem vlaknu:
   - ▷ binarne nevronske mreže
   - ▷ zvezne nevronske mreže

2. Ali prejšnji izhodi nevronske mreže vplivajo na trenutni izhod:
   - ▷ dinamične nevronske mreže
     - ⋆ izhodi nevronov so rekurzivno povezani nazaj na vhod
   - ▷ statične nevronske mreže

Pleada različnih vrst nevronskih mrež:

- ▷ binarni in zvezni perceptroni, Hopfieldove nevronske mreže, nevronske mreže z radialnimi baznimi funkcijami, Elmanove nevronske mreže, samoorganizirajoče se nevronske mreže itd.

9. Zvezni večplastni perceptroni

Osnovni pojmi

Splošni zvezni večplastni perceptroni

Učno pravilo gradientnega sestopa

Enonevronski zvezni perceptroni

Dvoplastni zvezni perceptroni

Hitrost učenja in inicializacija sinaptičnih uteži

Osnovni pojmi

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

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

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

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

  - Predpostavimo: $\mathbf{t}(\mathbf{x_i}) = \mathrm{f}(\mathbf{x_i})$

- Z zveznim večplastnim perceptronom lahko realiziramo neskončno mnogo različnih funkcij $\mathrm{f}_\mathbf{w}$ (odvisne od vektorja uteži!).

- Učenje funkcijske relacije f z zveznim perceptronom je iterativni proces:
  - Med učenjem postaja vsaka dvojica $(\mathbf{x_i}, \mathrm{f}_\mathbf{w}(\mathbf{x_i}))$ bolj in bolj podobna dvojici $(\mathbf{x_i}, \mathbf{t}(\mathbf{x_i}))$.
  - Učenje zaključimo – ujemanje med dvojicami iz učne množice ter izhodi naučene nevronske mreže ”dovolj natančno” glede na izbrani kriterij (npr. srednja absolutna napaka).

Osnovni pojmi *(2)*

- Posplošitev oz. generalizacija
  - Po učenju vrača nevronska mreža za poljuben vhod $\mathbf{x_i}$ določen izhod $\mathrm{f}_{\mathbf{w}}(\mathbf{x_i})$.
  - Lastnost velja za vse vhode, četudi morebiti ne pripadajo učni množici $\mathcal{U}_M$ in jih zato nismo uporabili v procesu učenja.

- Pri delu z nevronskimi mrežami je ključnega pomena izbira ustrezne **topologije nevronske mreže**
  - Topologija označuje organiziranost in število nevronov v posameznih plasteh nevronske mreže.

- Možno je pokazati, da s tem, ko izberemo topologijo nevronske mreže, hkrati določimo tudi razred funkcij, ki jih s takšno nevronsko mrežo lahko realiziramo.
  - Pri določanju topologije moramo približno oceniti, v kateri razred funkcij spada neznana funkcija f, ki jo identificiramo!

Splošni zvezni večplastni perceptroni

Poljuben problem lahko rešimo že s pomočjo zveznega triplastnega perceptrona!

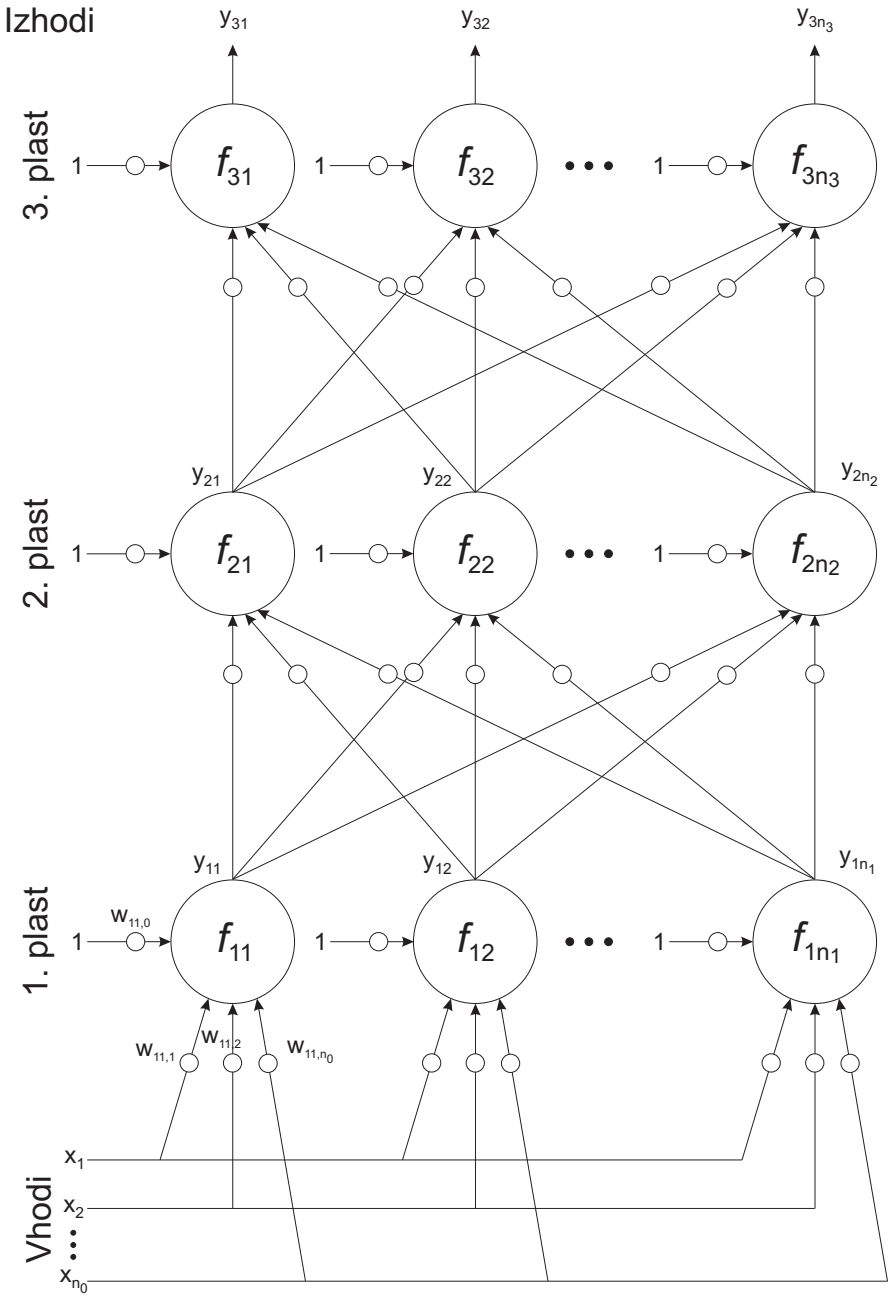
- Obtežen vhod s_{1k} za k-ti nevron v prvi plasti:

$$s_{1k} = \hat{\mathbf{w}}_{1k}\hat{\mathbf{x}} = w_{1k,0} + w_{1k,1}x_1 + w_{1k,2}x_2 + \ldots + w_{1k,n_0}x_{n_0}$$

- Izhod y_{1k} k-tega nevrona v prvi plasti:

$$y_{1k} = f_{1k}(s_{1k})$$

f_{1k} – prenosna funkcija za ta nevron.



Splošni zvezni večplastni perceptroni *(2)*

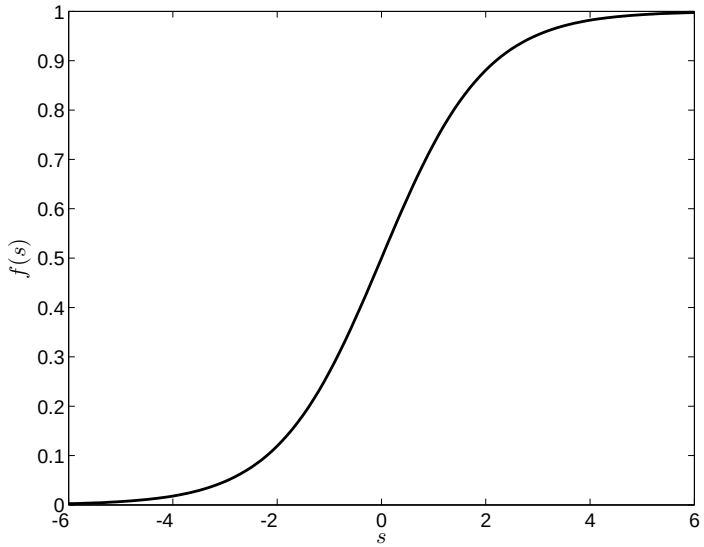
- Običajno ista prenosna funkcija f za vse nevrone v nevronski mreži.
- Najbolj pogosti prenosni funkciji sta linearna in sigmoidna prenosna funkcija.

A. Linearna prenosna funkcija

$$y = f(s) = k_1 s + k_0$$

B. Sigmoidna prenosna funkcija

$$y = f(s) = \frac{1}{1 + e^{-s}}$$



Učno pravilo gradientnega sestopa

- Problem učenja: adaptiranje sinaptičnih uteži w tako, da bodo po končanem učenju izhodi y za vsak vhod $\mathbf{x_i}$ približno enaki želenim vrednostim $\mathbf{t}(\mathbf{x_i})$.
- **Srednja kvadratna napaka (MSE)** – Kriterijska funkcija za merjenje uspešnosti prilagajanja

$$E = \frac{1}{2} \sum_{\mathbf{x_i} \in \mathcal{S}_N} \sum_{j=1}^{n_3} (t_j(\mathbf{x_i}) - y_j(\mathbf{x_i}))^2 \tag{9}$$

- Napaka E funkcija vseh sinaptičnih uteži v nevronski mreži.
- Napaka E je za določeno učno množico $\mathcal{U}_M$ ter za poljubno začetno inicializacijo sinaptičnih uteži v splošnem zelo velika.
- Med učenjem adaptacija sinaptičnih uteži tako, da se napaka E zmanjša do svojega globalnega (ali lokalnega) minimuma.

Učno pravilo gradientnega sestopa (vzvratno učenje)

Srednja kvadratna napaka večplastne nevronske mreže se zmanjša, če vektor sinaptičnih uteži $\mathbf{w}$ v koraku k spremenimo kot $\mathbf{w}(k) = \mathbf{w}(k-1) + \triangle\mathbf{w}$, kjer

$$\triangle\mathbf{w} = -\epsilon \nabla E(\mathbf{w}),$$

stopnja učenja ϵ je majhno pozitivno število, $\nabla E(\mathbf{w})$ pa je gradientni vektor, odvisen od vektorja uteži $\mathbf{w}$.

Učno pravilo gradientnega sestopa (2)

- Učno pravilo gradientnega sestopa ne zagotavlja, da bomo resnično dosegli globalni minimum srednje kvadratne napake E.
- Izpisano učno pravilo za izračun spremembe sinaptične uteži w_i je naslednje oblike:

$$\triangle w_i = -\epsilon \frac{\partial E}{\partial w_i} = \sum_{\mathbf{x_k} \in \mathcal{S}_N} \sum_{j=1}^{n_3} \epsilon \left(t_j(\mathbf{x_k}) - y_j(\mathbf{x_k}) \right) \frac{\partial y_j}{\partial w_i}$$

- Globalno oz. paketno učenje
  - Pri adaptiranju upoštevamo vse vhodne vzorce $\mathbf{x_k}$ (glej prvo vsoto v enačbi).
  - Globalni čas učenja – Število, kolikokrat smo spremenili uteži za vse vzorce iz učne množice.

- Lokalno oz. inkrementalno učenje
  - Pri adaptaciji v vsakem učnem koraku uporabimo le en vzorec $\mathbf{x_k}$ iz učne množice (prva vsota ni več potrebna!).

Enonevronski zvezni perceptroni

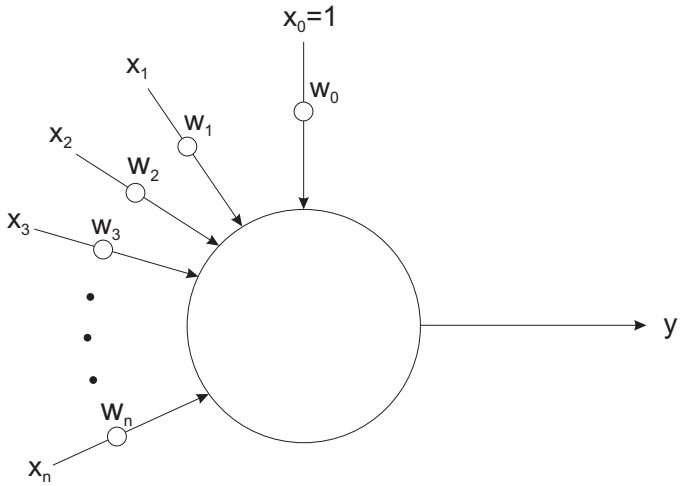
A. Učenje

- Izhod:

$$y = \mathrm{f}_{\mathbf{w}}(\mathbf{x_i}) = f(s(\mathbf{x_i}))$$

- Obteženi vhod:

$$s(\mathbf{x_i}) = \hat{\mathbf{x}}\hat{\mathbf{w}} = \sum_{j=0}^{n} w_j x_{i,j}$$



Formule za učno pravilo gradientnega sestopa se poenostavijo!

- Srednja kvadratna napaka – $E = \frac{1}{2}\sum_{\mathbf{x_i}\in\mathcal{S}_N}(t(\mathbf{x_i}) - y(\mathbf{x_i}))^2$
- Adaptacija posamezne sinaptične uteži

$$\triangle w_j = \epsilon \sum_{\mathbf{x_i}\in\mathcal{S}_N} (t(\mathbf{x_i}) - y(\mathbf{x_i}))\frac{df}{ds}x_{i,j}$$

- Adaptacija razširjenega vektorja uteži

$$\triangle \hat{\mathbf{w}} = \epsilon \sum_{\mathbf{x_i}\in\mathcal{S}_N} (t(\mathbf{x_i}) - y(\mathbf{x_i}))\frac{df}{ds}\hat{\mathbf{x}}_\mathbf{i}$$

Enonevronski zvezni perceptroni (2)

Zgled: Učimo enonevronski zvezni perceptron z dvema vhodoma po pravilu gradientnega sestopa: $\mathcal{S}_N = \{[1,1],[1,-1],[-1,1]\}$, pri čemer so ciljne vrednosti enake $t([1,1]) = 0{,}9$, $t([1,-1]) = 0{,}1$ in $t([-1,1]) = 0{,}1$. Začetni nabor sinaptičnih uteži je $\hat{\mathbf{w}} = [0; 0{,}5; 0{,}5]$. Prenosna funkcija je linearna: $f(s(\mathbf{x_i})) = s(\mathbf{x_i}) = w_0 + w_1 x_1 + w_2 x_2$.

- Izračunajmo trenutne izhode in napako E

$$(y[1,1]) = 1,\ y([1,-1]) = 0 \text{ in } y([-1,1]) = 0$$

$$E = \frac{1}{2}\left((0{,}9-1)^2 + (0{,}1-0)^2 + (0{,}1-0)^2\right) = 0{,}015$$

- Nevron ni naučen, zato potrebno učenje. Izračunajmo odvod prenosne funkcije:

$$\frac{df}{ds} = (s)' = 1$$

- Vse dobljeno vstavimo v enačbo za adaptacijo uteži.

$$\triangle\hat{\mathbf{w}} = \begin{bmatrix} \triangle w_0 \\ \triangle w_1 \\ \triangle w_2 \end{bmatrix} = \epsilon(-0{,}1)\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \epsilon(0{,}1)\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} + \epsilon(0{,}1)\begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0{,}1 \\ -0{,}1 \\ -0{,}1 \end{bmatrix}$$

- Nov razširjen vektor, pri čemer $\epsilon = 1$, je

$$\hat{\mathbf{w}}_{nov} = \hat{\mathbf{w}}_{star} + \triangle\hat{\mathbf{w}} = [0; 0{,}5; 0{,}5] + [0{,}1; -0{,}1; -0{,}1] = [0{,}1; 0{,}4; 0{,}4]$$

Enonevronski zvezni perceptroni *(3)*

B. Razvrščanje vzorcev v dva razreda (tj. razred $\mathcal{C}_A$ in $\mathcal{C}_B$)

- Objekti popisani kot vzorci v obliki množice značilnic: $\mathbf{x} = [x_1, x_2, \ldots, x_n]$.
- Pri razvrščanju izhodno vrednost $y = \mathrm{f}_{\mathbf{w}}(\mathbf{x})$ perceptrona uporabimo za določitev oznake razreda, kateremu pripada vzorec $\mathbf{x}$.
- Trije načini, kako uporabiti izhod $\mathrm{f}_{\mathbf{w}}(\mathbf{x})$ kot oznako za razred $\mathcal{C}_A$ in razred $\mathcal{C}_B$ (ob pogoju, da je prenosna funkcija sigmoidna!)
  1. razvrščanje s hiperravninsko ločilno mejo z ena in nič označevanjem,
  2. razvrščanje s hiperravninsko ločilno mejo z označevanjem z dvojnim pragom in
  3. razvrščanje s hiperravninsko ločilno mejo z označevanjem z enojnim pragom.
- V vseh načinih perceptron realizira odločitveno pravilo oz. ločilno mejo (tj. $(n-1)$-razsežna hiperravnina), ki razdeli prostor značilnic v podprostora $\mathcal{X}_A$ in $\mathcal{X}_B$

$$x_n = -\frac{1}{w_n}\left(w_0 + w_1 x_1 + \ldots + w_{n-1} x_{n-1}\right)$$

- Ločimo dve fazi:
  - ⋆ Faza učenja
    - Množica vzorcev $\mathcal{S}_N$ iz učne množice $\mathcal{U}_M$ je
    $$\mathcal{S}_N = \mathcal{S}_A \cup \mathcal{S}_B, \qquad \mathcal{S}_A \subset \mathcal{X}_A, \mathcal{S}_B \subset \mathcal{X}_B$$
  - ⋆ Faza razvrščanja

Enonevronski zvezni perceptroni *(4)*

a. Razvrščanje s hiperravninsko ločilno mejo z ena in nič označevanjem

1. Faza učenja
   - ▷ želena oz. ciljna vrednost $t(\mathbf{x}) = 1, \qquad \forall \mathbf{x} \in \mathcal{S}_A$
   - ▷ želena oz. ciljna vrednost $t(\mathbf{x}) = 0, \qquad \forall \mathbf{x} \in \mathcal{S}_B$
   - ▷ Učimo z učnim pravilom gradientnega sestopa (v določenih primerih tudi s pravilom okrepitve).
2. Faza razvrščanja
   - ▷ $\mathbf{x}$ razvrstimo v razred $\mathcal{C}_A$, če $y(\mathbf{x}) > 0{,}5$
   - ▷ $\mathbf{x}$ razvrstimo v razred $\mathcal{C}_B$, če $y(\mathbf{x}) \leq 0{,}5$
   - ▷ Verjetnosti pojavitve obeh razredov morata biti enaki (apriorna verjetnost), tj. $p(\mathcal{C}_A) = p(\mathcal{C}_B)$!

Praktična trditev:

- ▷ Veliko problemov razvrščanja v dva razreda, kjer je optimalna ločilna meja odprta, nelinearna in konveksna meja ter hkrati presek obeh razredov ni prevelik, lahko smiselno rešimo z enonevronskim zveznim perceptronom.
- ▷ Optimalno rešitev dobimo, kadar $|\mathbf{w}| = \left(\sum_i w_i^2\right)^{\left(\frac{1}{2}\right)} \longrightarrow \infty$.

Enonevronski zvezni perceptroni *(5)*

b. Razvrščanje s hiperravninsko ločilno mejo z označevanjem z dvojnim pragom

1. Faza razvrščanja
   - ▷ $\mathbf{x}$ razvrstimo v razred $\mathcal{C}_A$, če $y(\mathbf{x}) > 0{,}5$
   - ▷ $\mathbf{x}$ razvrstimo v razred $\mathcal{C}_B$, če $y(\mathbf{x}) \leq 0{,}5$
   - ▷ Verjetnosti pojavitve obeh razredov morata biti enaki (apriorna verjetnost), tj. $p(\mathcal{C}_A) = p(\mathcal{C}_B)$!
2. Faza učenja
   - ▷ želena oz. ciljna vrednost $t(\mathbf{x}) \geq a, \quad 0{,}5 < a < 1, \qquad \forall \mathbf{x} \in \mathcal{S}_A$
   - ▷ želena oz. ciljna vrednost $t(\mathbf{x}) \leq b, \quad 0 < b < 0{,}5, \qquad \forall \mathbf{x} \in \mathcal{S}_B$
   - ▷ Če je med učenjem izhod $y(\mathbf{x}) \geq a$ za vzorec $\mathbf{x} \in \mathcal{S}_A$, potem je napaka $E = 0$, sicer pa $E = (a - y(\mathbf{x}))^2$.
   - ▷ Če je med učenjem izhod $y(\mathbf{x}) \leq b$ za vzorec $\mathbf{x} \in \mathcal{S}_B$, potem je napaka $E = 0$, sicer pa $E = (b - y(\mathbf{x}))^2$.
   - ▷ Hitrost učnega procesa se zelo poveča, saj adaptacijo uteži izračunamo zgolj na osnovi napačno razvrščenih vzorcev:

$$\triangle \hat{\mathbf{w}} = \epsilon \left(\sum_{\mathbf{x} \in \neg \mathcal{S}_A} (a - y(\mathbf{x}))^2 \frac{df}{ds} \hat{\mathbf{x}} + \sum_{\mathbf{x} \in \neg \mathcal{S}_B} (b - y(\mathbf{x}))^2 \frac{df}{ds} \hat{\mathbf{x}} \right)$$

Enonevronski zvezni perceptroni *(6)*

c. Razvrščanje s hiperravninsko ločilno mejo z označevanjem z enojnim pragom

1. Faza učenja
   - ▷ želena oz. ciljna vrednost $t(\mathbf{x}) > 0{,}5, \qquad \forall \mathbf{x} \in \mathcal{S}_A$
   - ▷ želena oz. ciljna vrednost $t(\mathbf{x}) \leq 0{,}5, \qquad \forall \mathbf{x} \in \mathcal{S}_B$
2. Faza razvrščanja
   - ▷ $\mathbf{x}$ razvrstimo v razred $\mathcal{C}_A$, če $y(\mathbf{x}) > 0{,}5$
   - ▷ $\mathbf{x}$ razvrstimo v razred $\mathcal{C}_B$, če $y(\mathbf{x}) \leq 0{,}5$
   - ▷ Verjetnosti pojavitve obeh razredov morata biti enaki (apriorna verjetnost), tj. $p(\mathcal{C}_A) = p(\mathcal{C}_B)$!

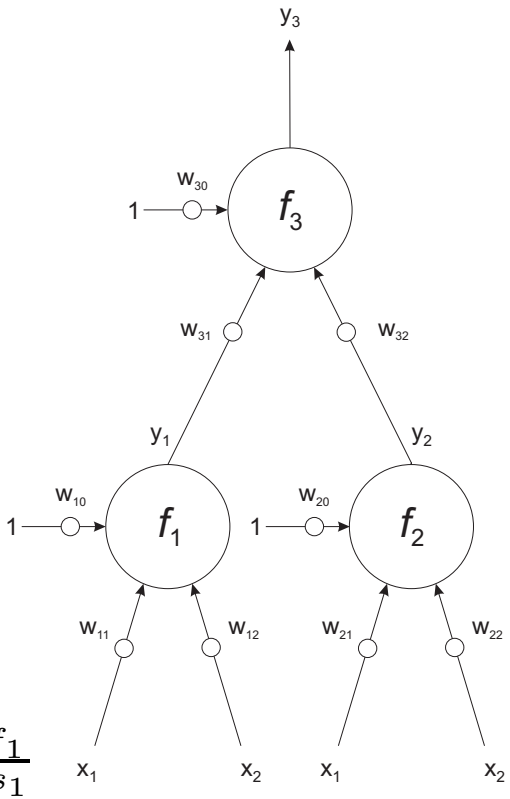
Razmislek:

- ▷ Opravka imamo z enojnim pragom, ki je enak v obeh fazah!
- ▷ $E = 0$, če je izhod perceptrona za vse vzorce enak 0,5
  - ⋆ Tedaj, kadar so vse sinaptične uteži enake 0.
  - ⋆ Uspešnost razvrščanja tedaj 50 % (vsi vzorci iz $\mathcal{S}_A$ so napačno razvrščeni!)
- ▷ Zato nujna previdnost. Smiselne rezultate dobimo, npr.:
  - ⋆ Če po vsakem koraku spreminjanja sinaptičnih uteži pomnožimo vsako utež tako, da absolutna vrednost razširjenega vektorja uteži $|\hat{\mathbf{w}}|$ ostane skozi proces učenja nespremenjena.

Dvoplastni zvezni perceptroni

A. Trinevronski dvoplastni zvezni perceptroni

$$y_3 = f_3(s_3), \quad s_3 = w_{30} + w_{31}y_1 + w_{32}y_2$$
$$y_1 = f_1(s_1), \quad s_1 = w_{10} + w_{11}x_1 + w_{12}x_2$$
$$y_2 = f_2(s_2), \quad s_2 = w_{20} + w_{21}x_1 + w_{22}x_2$$



▷ Učno pravilo gradientnega sestopa:

$$\triangle w_{30} = -\epsilon \frac{\partial E}{\partial w_{30}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3}$$
$$\triangle w_{31} = -\epsilon \frac{\partial E}{\partial w_{31}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} y_1(\mathbf{x})$$
$$\triangle w_{32} = -\epsilon \frac{\partial E}{\partial w_{32}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} y_2(\mathbf{x})$$
$$\triangle w_{10} = -\epsilon \frac{\partial E}{\partial w_{10}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{31} \frac{df_1}{ds_1}$$
$$\triangle w_{11} = -\epsilon \frac{\partial E}{\partial w_{11}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{31} \frac{df_1}{ds_1} x_1$$
$$\triangle w_{12} = -\epsilon \frac{\partial E}{\partial w_{12}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{31} \frac{df_1}{ds_1} x_2$$
$$\triangle w_{20} = -\epsilon \frac{\partial E}{\partial w_{20}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{32} \frac{df_2}{ds_2}$$
$$\triangle w_{21} = -\epsilon \frac{\partial E}{\partial w_{21}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{32} \frac{df_2}{ds_2} x_1$$
$$\triangle w_{22} = -\epsilon \frac{\partial E}{\partial w_{22}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{32} \frac{df_2}{ds_2} x_2$$

Dvoplastni zvezni perceptroni *(2)*

Zgled: S trinevronskim zveznim perceptronom razvrščajmo vzorce v dva razreda ($\mathcal{C}_A$ in $\mathcal{C}_B$). Razširjeni vektorji sinaptičnih uteži so: $\hat{\mathbf{w}}_\mathbf{1} = [0; 0{,}1; 0{,}1]$, $\hat{\mathbf{w}}_\mathbf{2} = [0; -0{,}05; -0{,}1]$ in $\hat{\mathbf{w}}_\mathbf{3} = [-0,2; -1; 1]$. Prenosne funkcije so sigmoidne.

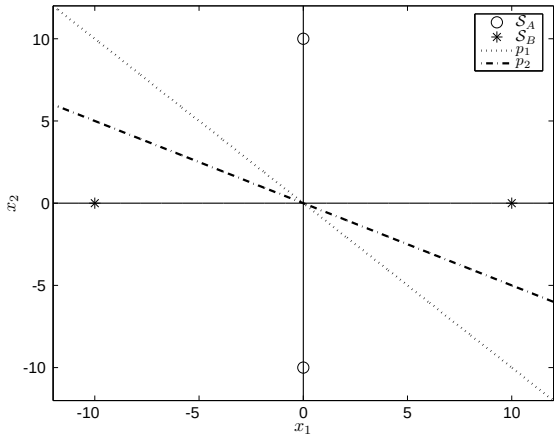
Množica vzorcev za razred $\mathcal{C}_A$ in $\mathcal{C}_B$:

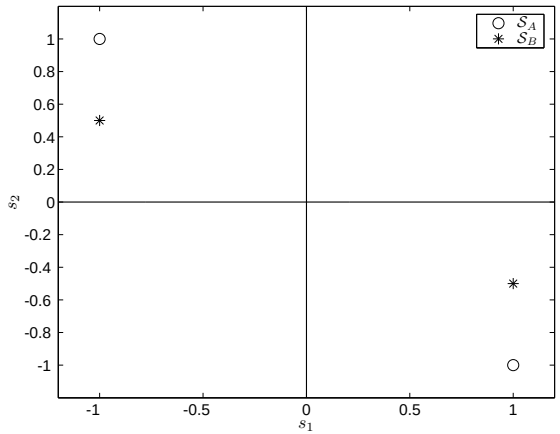
$\mathcal{S}_A = \{A_1 = [0, 10], A_2 = [0, -10]\}$ in $\mathcal{S}_B = \{B_1 = [10, 0], B_2 = [-10, 0]\}$

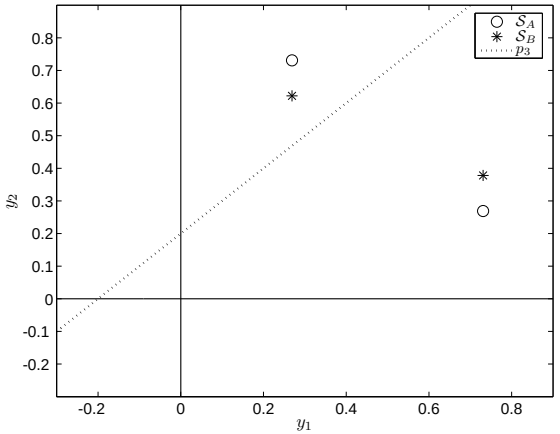
Ali je perceptron že naučen? Kako trenutno razvršča?

- ▷ Ločilna meja $p_1 : \; x_2 = -x_1$ in ločilna meja $p_2 : \; x_2 = -\frac{x_1}{2}$

$\mathbf{x}$	s_1	y_1	s_2	y_2	s_3	y_3
$A_1 = [0, 10]$	1	0,731	-1	0,269	-0,662	0,340
$A_2 = [0, -10]$	-1	0,269	1	0,731	0,262	0,565
$B_1 = [10, 0]$	1	0,731	-0,5	0,378	-0,554	0,365
$B_2 = [-10, 0]$	-1	0,269	0,5	0,622	0,154	0,538







Dvoplastni zvezni perceptroni *(3)*

Zgled (nadaljevanje): Perceptron še ni naučen! Uporabimo metodo z ena (za razred $\mathcal{C}_A$) in nič označevanjem.

Izračun pomožnih izrazov

$$\frac{df}{ds} = f'(s) = \left(\frac{1}{1+e^{-s}}\right)' = f(s)\,(1-f(s))$$

$\mathbf{x} = [x_1, x_2]$	$\frac{df_1}{ds_1}$	$\frac{df_2}{s_2}$	$\frac{df_3}{s_3}$	$t(\mathbf{x}) - y_3(\mathbf{x})$
$A_1 = [0, 10]$	0,197	0,197	0,224	$1 - 0,34 = 0,66$
$A_2 = [0, -10]$	0,197	0,197	0,226	0,435
$B_1 = [10, 0]$	0,197	0,235	0,232	-0,365
$B_2 = [-10, 0]$	0,197	0,235	0,249	-0,538

Primer izračuna za spremembo $\triangle w_{21}$:

$$\triangle w_{21} = -\epsilon \frac{\partial E}{\partial w_{21}} = \epsilon \sum_{\mathbf{x}\in\mathcal{S}_N} (t(\mathbf{x}) - y_3(\mathbf{x})) \frac{df_3}{ds_3} w_{32} \frac{df_2}{ds_2} x_1 =$$

$$= \epsilon \Big(0,66 \cdot 0,224 \cdot 1 \cdot 0,197 \cdot 0 + 0,435 \cdot 0,226 \cdot 1 \cdot 0,197 \cdot 0 +$$

$$(-0,365) \cdot 0,232 \cdot 1 \cdot 0,235 \cdot 10 + (-0,538) \cdot 0,249 \cdot 1 \cdot 0,235 \cdot (-10) \Big) = -0,5138\epsilon$$

$$w_{21_{\text{nov}}} = w_{21_{\text{star}}} + \triangle w_{21} = -0,05 - 0,5138 = -0,5638$$

Podobno določimo ostale uteži. Učenje ponavljamo do konvergence.

Dvoplastni zvezni perceptroni *(4)*

B. Razred funkcij, realiziranih z dvoplastnimi zveznimi perceptroni

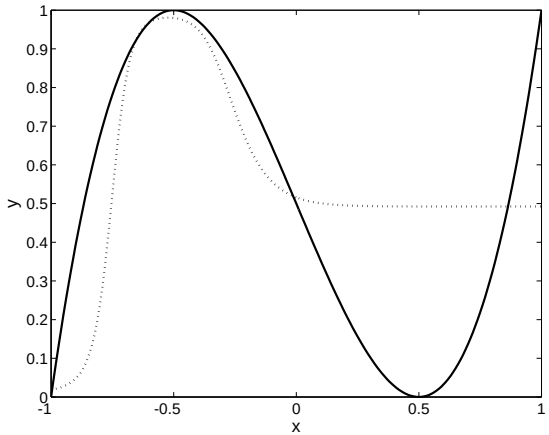
- ▷ Dvoplastne perceptrone lahko na osnovi učne množice $\mathcal{U}_M$, oblike $(\mathbf{x_i}, \mathbf{t}(\mathbf{x_i}))$, učimo takšne funkcije, ki bi potekala natančno skozi ali čim bolj blizu točk iz $\mathcal{U}_M$.
- ▷ Realizirana funkcija je odvisna od prenosne funkcije nevrona ter števila nevronov v prvi plasti takšne mreže.
- ▷ S premajhnim številom nevronov v prvi plasti ne moremo realizirati natančnega prileganja funkcije k podatkovnim točkam iz $\mathcal{U}_M$!
  - ⋆ Podatkovne točke so **preohlapno prilegane oz. podprilegane.**
  - ⋆ Perceptron se preslabo prilagodi k podatkovnim točkam.
- ▷ Pri prevelikem številu nevronov bo realizirana funkcija potekala natančno skozi podatkovne točke, hkrati pa bo močno oscilirala v intervalih med točkami.
  - ⋆ Podatkovne točke so **prestrogo prilegane oz. nadprilegane**.
  - ⋆ Perceptron se preveč prilagodi podatkovnim točkam.

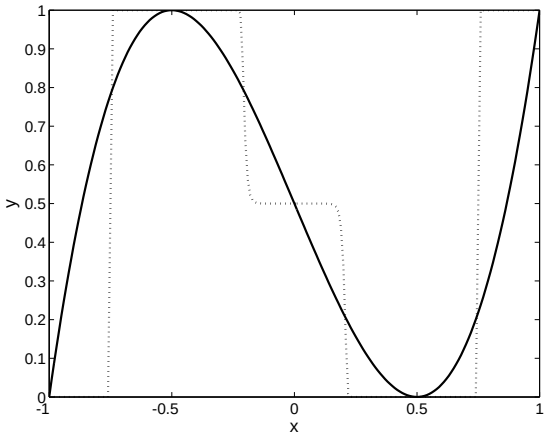
$y = 2x^3 - 1{,}5x + 0{,}5$

5 točk v $\mathcal{U}_M$

Podprileganje (levo) – 2 nevrona v 1. plasti

Nadprileganje (desno) – 5 nevronov v 1. plasti





Dvoplastni zvezni perceptroni *(5)*

- Nadprileganju se izognemo, če uporabljamo učne množice z veliko množico vzorcev.
- Podprileganju se izognemo, če kompleksne topologije nevronske mreže preveč ne poenostavimo.
- **Teorem:**

  Z dvoplastnim zveznim perceptronom s sigmoidno prenosno funkcijo za nevrone v prvi plasti in z enim linearnim nevronom v drugi plasti lahko aproksimiramo poljubno zvezno funkcijo $\mathrm{f} : \mathbb{R}^n \longmapsto \mathbb{R}$ v katerikoli domeni s poljubno natančnostjo. Prenosna funkcija nevrona v izhodni plasti je linearna funkcija, tj. $f_2(s) = s$.
- Koristneje je uporabiti triplastni zvezni perceptron, saj je število nevronov manjše kot pri dvoplastnem, še posebej če ima aproksimirana funkcija nezveznosti.
- **Teorem:**

  Vsako funkcijo, ki jo lahko aproksimiramo poljubno dobro z odsekoma linearno funkcijo, lahko realiziramo s triplastnim zveznim perceptronom, ki ima en linearen nevron v izhodni plasti (tj., izhodni nevron ima linearno prenosno funkcijo).

Hitrost učenja in inicializacija sinaptičnih uteži

A. Hitrost učenja

- Čas učenja oz. število učnih korakov je lahko zelo veliko za zvezne perceptrone, predvsem zaradi počasnega spreminjanja uteži:

$$\triangle \hat{\mathbf{w}} = -\epsilon \nabla E$$

- **Metoda momenta**
  - Hitrost učenja izboljšamo, če k izračunani spremembi sinaptičnih uteži $\triangle \hat{\mathbf{w}}(k)$ v koraku k prištejemo vektor, ki je proporcionalen izračunani spremembi uteži v koraku $\hat{\mathbf{w}}(k-1)$:

$$\triangle \hat{\mathbf{w}}_{nov}(k) = \triangle \hat{\mathbf{w}}(k) + \alpha \triangle \hat{\mathbf{w}}(k-1)$$

    momentni parameter α – skalar iz intervala $[0, 1]$.

B. Skaliranje vhodov in izhodov

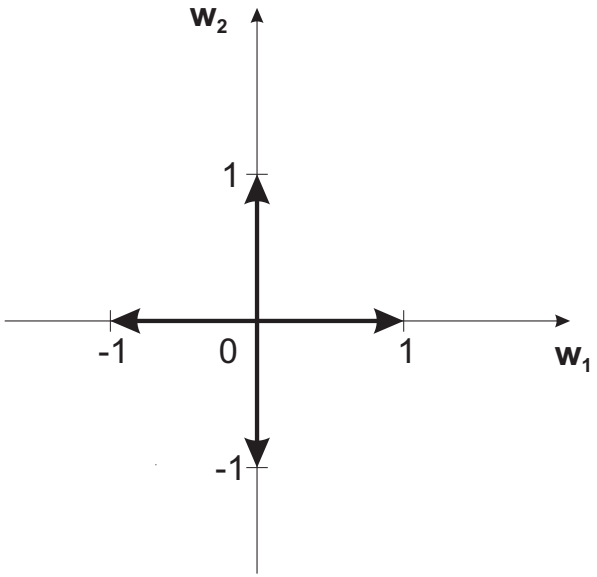
- **Vhodi:**
  - Vhodov x_i v nevronsko mrežo teoretično ni treba skalirati.
  - Velike vhodne vrednosti, pomnožene z velikimi utežmi, zelo upočasnijo proces učenja, saj je sprememba sinaptičnih uteži $\triangle \mathbf{w} \approx 0$.
- **Izhodi:**
  - Sigmoidna prenosna funkcija – izhod NN $\in [0, 1]$ (skaliranje izhodov!)
  - Linearna prenosna funkcija – skaliranje izhodov nepotrebno.

Hitrost učenja in inicializacija sinaptičnih uteži (2)

C. Inicializacija sinaptičnih uteži

- Začetne vrednosti sinaptičnih uteži teoretični izbiramo popolnoma naključno s poljubnega intervala realnih števil.
- Ker lahko pristanemo v lokalnem minimumu funkcije napake E, moramo učni proces ponoviti z različnimi inicializacijami sinaptičnih uteži.
- Ob vsaki inicializaciji izberemo vektor uteži v določeni plasti nevronske mreže tako, da vektor uteži enakomerno porazdelimo po prostoru sinaptičnih uteži (s tem določimo uteži w_1 do w_n.
- Kako pa določiti prag T oz. utež w_0 za nevron?
  - ⋆ Preprosto pravilo: Vhodi nevrona morajo prispevati k njegovemu izhodu.
  - ⋆ To tedaj, kadar ločilna meja $\hat{\mathbf{w}} \cdot \hat{\mathbf{x}} = 0$, ki jo definira ta nevron, poteka skozi težišče $\mathbf{c}$ vhodnih podatkov.

$$\mathbf{c} = \frac{1}{N} \sum_i \mathbf{x}_i$$



  - ⋆ Izračun uteži w_0 za 2D-primer

$$w_0 = -w_1 c_1 - w_2 c_2$$

10. Samoorganizirajoče se nevronske mreže

Topologija in obnašanje

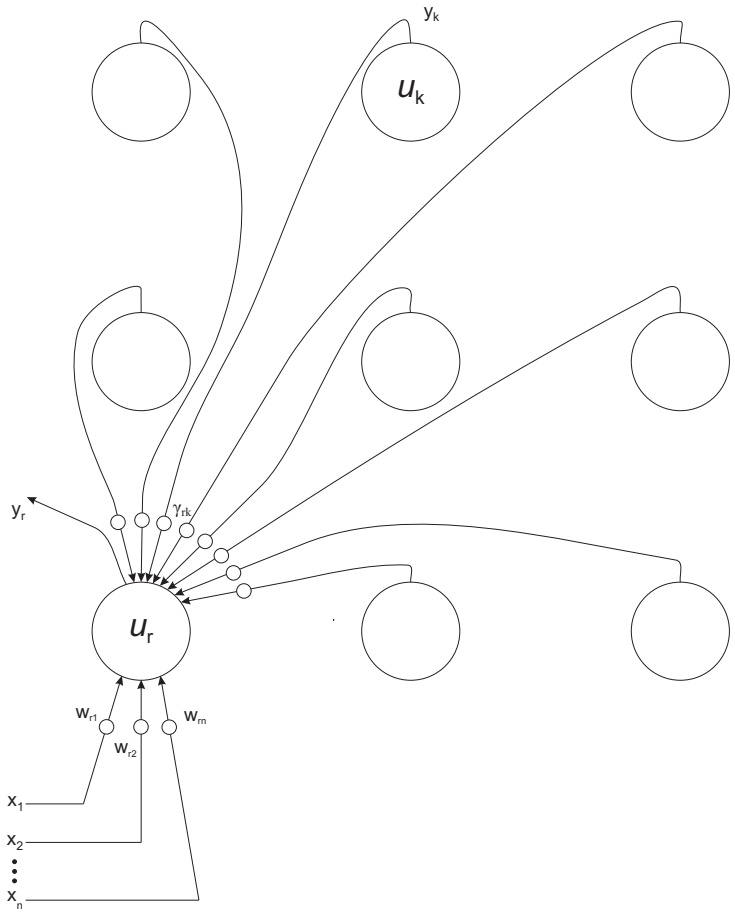
Ekvivalentni algoritem in spreminjanje sinaptičnih uteži

Ekvivalentni algoritem in postopek učenja

Napotki za uporabo ekvivalentnega algoritma

Topologija in obnašanje

- SOM iz K nevronov, ki so razporejeni v 1D, 2D ali ND rešetko.
- Vsak nevron ima eno izhodno linijo:
  - $y_r(t)$ – izhod nevrona u_r v času t
- $\mathbf{x}$ – zunanji oz. eksterni vhod (tj. opazovalni vektor)
- $w_{rj}(t)$ – sinaptična utež
- Notranji oz. interni vhod (stranske oz. bočne povezave):
  - Zmnožek izhoda določenega nevrona ter bočne sinaptične uteži γ
  - $K-1$ internih vhodnih linij
- Obtežen vhod v nevron u_r:



$$s_r(t) = \sum_j w_{rj}(t)x_j(t) + \sum_k \gamma_{rk}y_k(t)$$

- Izhod y_r nevrona u_r:

$$y_r(t+\triangle) = \lambda_r\{s_r(t)\}$$

funkcija λ_r – nelinearna monotona naraščajoča funkcija

- Gre za rekurzijo. Zato potrebna zakasnitev, da se izhodi nevronov stabilizirajo.

Topologija in obnašanje *(2)*

- Po vstavljanju $\mathbf{x}(t)$ v SOM imajo med učenjem zaradi bočnega vzbujanja in zaviranja določeni nevroni velike izhodne vrednosti, vsi preostali pa majhne.

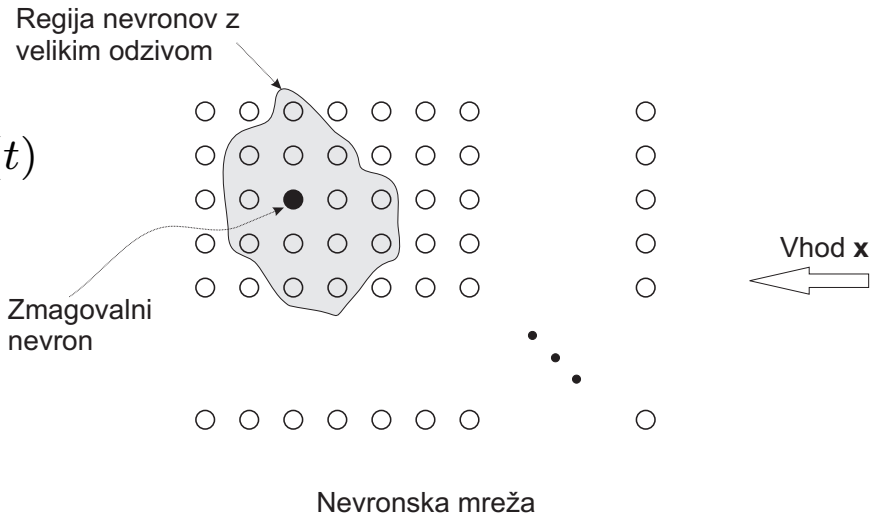
- Sinaptične uteži w spreminjamo po adaptacijskem pravilu Hebba:

$$\mathbf{w}_{r,nov}(t) = \mathbf{w}_{r,star}(t) + \epsilon(t) y_r(t) \mathbf{x}(t)$$

$\epsilon(t) \in [0, 1]$ in se s časom manjša

- Po vsakem koraku učenja uteži še normaliziramo:

$$\sum_j w_{rj} = 1$$



- Pravilo Hebba v glavnem spreminja uteži zmagovalnega nevrona ter uteži nevronov iz njegove okolice:
  - ⋆ Posledica: nevroni iz te soseščine postanejo bolj senzitivni na opazovalne vektorje, ki so podobni vhodu $\mathbf{x}(t)$, uporabljenemu v koraku adaptiranja.
- Sprejemno polje nevrona $\mathcal{R}_r$ – množica tistih opazovalnih vektorjev oz. vhodov $\mathbf{x}$, za katere je nevron u_r zmagovalec.
  - ⋆ Posledica: vektor uteži $\mathbf{w}_r$ nevrona u_r postane podoben opazovalnim vektorjem $\mathbf{x}$ v svojem sprejemnem polju $\mathcal{R}_r$.

Ekvivalentni algo. in spreminjanje sinaptičnih uteži

- Ekvivalentni algoritem SOM adaptira sinaptične uteži v dveh korakih:
  - 1) iskanje zmagovalnega nevrona in 2) dejansko ažuriranje uteži.

A. Iskanje zmagovalnega nevrona

- V ekvivalentnem algoritmu je zmagovalni nevron tisti, ki ima vektor sinaptičnih uteži najbolj podoben opazovalnemu vektorju $\mathbf{x}$ med vsemi nevroni v SOM:

$$d(\mathbf{w}_s(t), \mathbf{x}(t)) = \min_r d(\mathbf{w}_r(t), \mathbf{x}(t))$$

  d – mera razdalje za vhodni prostor

- Majhna razlika glede na vpeljano definicijo o zmagovalnem nevronu.

B. Dejansko ažuriranje uteži

- Vsak vektor sinaptičnih uteži $\mathbf{w}_r$ v nevronski mreži spremenimo po pravilu Hebba:

$$\mathbf{w}_r(t+1) = \mathbf{w}_r(t) + g(r, s, t) \cdot (\mathbf{x}(t) - \mathbf{w}_r(t))$$

  t – čas oz. učni korak, $g(r, s, t)$ – skalarna funkcija, ki vrača vrednosti iz $[0, 1]$.

- Funkcija $g(r, s, t)$ določa okolico zmagovalnega nevrona, ki jo bomo ažurirali, ter jakost ažuriranja sinaptičnih uteži.
  - Na začetku učenja sta okolica in jakost ažuriranja veliki, ob zaključku pa majhni.

Ekvivalentni algo. in spreminjanje sinaptičnih uteži (2)

- Funkcijo $g(r, s, t)$ razbijemo v dva dela:

$$g(r, s, t) = \epsilon(t) \cdot h(r, s, t)$$

$\epsilon(t)$ – stopnja učenja, funkcija $h(r, s, t)$ – skalarna funkcija, ki določa velikost okolice adaptiranja.

- Funkcijo h zapišemo v obliki eksponentno padajoče funkcije:

$$h(r, s, t) = e^{\frac{-d_L^2(u_r, u_s)}{\sigma^2(t)}}$$

$\sigma(t) = \frac{\sigma_0}{\sqrt{1+t}}$, z izrazom $d_L(u_r, u_s)$ pa merimo razdaljo med nevronoma u_r in u_s v rešetki SOM.

- Z ekvivalentnim algoritmom lahko sicer določimo odziv oz. izhod nevrona, vendar nevronov izhod nima nikakršnega pomena v tem algoritmu

Ekvivalentni algoritem in postopek učenja

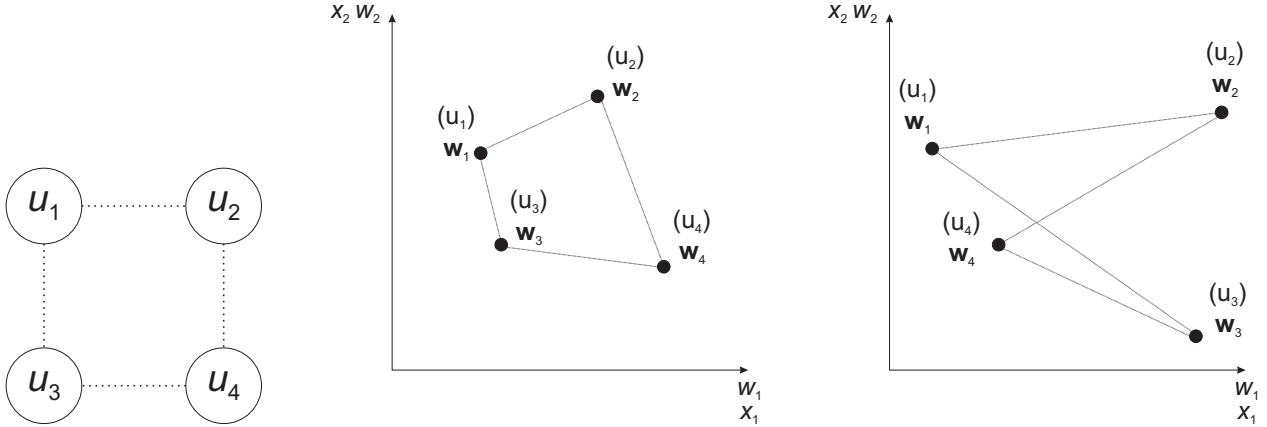
- Pri učenju z ekvivalentnim algoritmom se izoblikujeta dve fazi:
  - faza urejanja sinaptičnih uteži ter kvantizacijska faza.

A. Faza urejanja sinaptičnih uteži (začetna faza)

- Se pojavi na samem začetku procesa učenja.
- SOM postane urejena – sosednji nevroni imajo podobne sinaptične uteži.
- Pravilno urejena nevronska mreža – sosednji nevroni imajo tudi sosednja sprejemna polja.
  - Spomnimo: sprejemno polje nevrona u_s oz. njegove sinaptične uteži $\mathbf{w}_s$ tvorijo vsi tisti opazovalni vektorji $\mathbf{x}$, za katere je u_s zmagovalni nevron:

$$\mathcal{R}_{\mathbf{w}_s} = \left\{ \mathbf{x} \mid d(\mathbf{x}, \mathbf{w}_s) = \min_r d(\mathbf{x}, \mathbf{w}_r) \right\}$$

Zgled: Urejena in neurejena SOM iz 2×2 nevronov; 2D opazovalni vektorji.



Ekvivalentni algoritem in postopek učenja (2)

B. Kvantizacijska faza (končna faza)

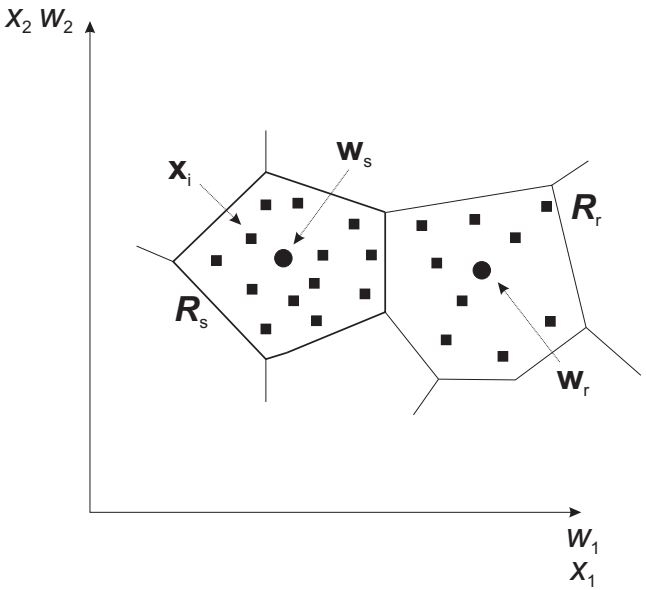
- Prostor opazovalnih vektorjev $\mathcal{X}$ se kvantizira (razdeli)
  - oz. vhodni prostor N n-dimenzionalnih vhodov se nadomesti z manjšo, reprezentativno množico K n-dimenzionalnih sinaptičnih uteži NN.
  - Vhodni prostor se razdeli na K disjunktnih podprostorov $\mathcal{X}_i$:

$$\mathcal{X} = \bigcup_i \mathcal{X}_i \quad \wedge \quad \mathcal{X}_i \cap \mathcal{X}_j = \emptyset \quad \forall i \neq j$$

- Podprostor $\mathcal{X}_i$ je dejansko sprejemno polje nevrona u_i:
  - Vektor sinaptičnih uteži $\mathbf{w}_i$ je tipični predstavnik opazovalnih vektorjev iz sprejemnega polja nevrona u_i.

Zgled: Primer kvantizacije 2D prostora vhodov s SOM.

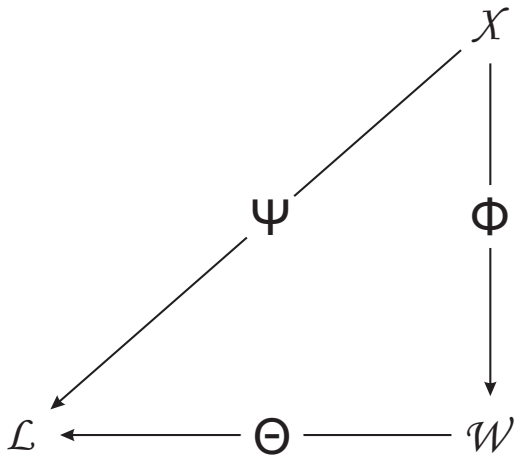
- Sprejemni polji $\mathcal{R}_s$ in $\mathcal{R}_r$.
- Sinaptična utež je tipičen predstavnik vhodnih vektorjev iz svojih sprejemnih polj.
- Meja med sprejemnima poljema – premica, ki jo tvorijo točke, ki so enako oddaljene od vektorjev $\mathbf{w}_s$ in $\mathbf{w}_r$.



Napotki za uporabo ekvivalentnega algoritma

A. Topologija nevronske mreže

- Trije prostori pri delu z ekvivalentnim algoritmom:
  1. prostor vhodnih oz. opazovalnih vektorjev $\mathcal{X}$
  2. prostor vektorjev sinaptičnih uteži $\mathcal{W}$
  3. nevronska rešetka $\mathcal{L}$
- Tri preslikave pri delu z ekvivalentnim algoritmom:
  1. kvantizacijska preslikava Φ
  2. projekcijska preslikava Θ
  3. preslikava značilnic $\Psi = \Phi \cdot \Theta$

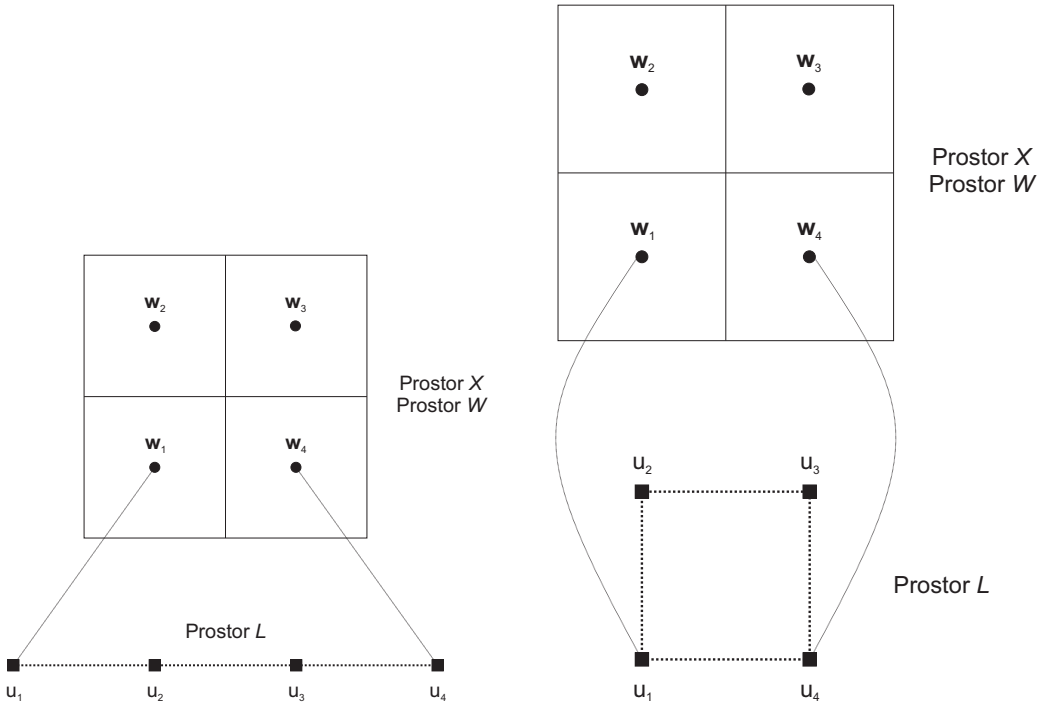


Definicija:

- Če sta sprejemni polji $\mathcal{R}_i$ in $\mathcal{R}_j$ v prostoru vhodov $\mathcal{X}$ sosednji, potem je nevron u_i z vektorjem sinaptičnih uteži $\mathbf{w}_i$ sosed nevronu u_j (utež $\mathbf{w}_j$). Če ta lastnost velja za vse pare sprejemnih polj, potem preslikava značilnic Ψ ohranja topologijo.
- Ohranjanje topologije je komplement lastnosti pravilnega urejanja.
- Popolno ohranjanje topologije pogosto ni možno, ker je dimenzija vhodnega prostora $\mathcal{X}$ običajno večja od dimenzije nevronske rešetke $\mathcal{L}$
- Večja razlika med dimenzijama, večje število defektov:
  - ⋆ Defekt – situacija, kjer sosednji nevroni v SOM nimajo sosednjih sprejemnih polj.

Napotki za uporabo ekvivalentnega algoritma (2)

Zgled: Primer neohranjanja in ohranjanja topologije SOM. Vhodi so dvodimenzionalni.



Nevronska rešetka 1×4 Nevronska rešetka 2×2

- ▷ Levo:
  - ⋆ Ni ohranjanja topologije, saj imata nevrona u_1 in u_4, z vektorjema sinaptičnih uteži $\mathbf{w}_1$ in $\mathbf{w}_4$, imata sosednji sprejemni polji, čeprav nevrona nista soseda v nevronski rešetki.
- ▷ Desno:
  - ⋆ Popolno ohranjanje topologije, saj imajo vsi sosednji nevroni dejansko tudi sosednja sprejemna polja.

Napotki za uporabo ekvivalentnega algoritma (3)

B. Proces učenja

- ▷ SOM hrani znanje v sinaptičnih utežeh.
- ▷ Učenje SOM se razlikuje od učenja perceptronov:
  1. Množico učnih vzorcev $\mathcal{S}_N$ razdelimo na M podmnožic $\mathcal{S}_i$, pri čemer podmnožico $\mathcal{S}_i$ tvorijo vsi vzorci iz razreda $\mathcal{C}_i$.
  2. Za vsako podmnožico $\mathcal{S}_i$ uvedemo svojo SOM, ki jo učimo z vzorci oz. opazovalnimi vektorji iz množice $\mathcal{S}_i$.

Praktični napotki za uspešno učenje SOM

- ▷ Med učenjem vsak vektor $\mathbf{x}$ iz učne množice za to mrežo, kjer $\mathbf{x} \in \mathcal{X}$, vstavimo večkrat v ekvivalentni algoritem (≥ 10).
- ▷ Po določenem času učenja T_E se sinaptične uteži v SOM več ne spreminjajo.
  - ⋆ Npr. Če $\epsilon(t) = \frac{1}{\sqrt{1+t}}$, potem je po 10^4 korakih učenja sprememba vektorja sinaptičnih uteži le še okrog 1% glede na trenutno vrednost vektorja uteži.
  - ⋆ Ni potrebe, da bi izvajali učenje z več kot 10^5 koraki učenja.
  - ⋆ Če želimo daljši čas učenja, moramo modificirati funkcijo za stopnjo učenja.

Napotki za uporabo ekvivalentnega algoritma (4)

- Vsak opazovalni vektor moramo večkrat uporabiti v procesu učenja, zato lahko množico z učnimi primerki S_{nova} tvorimo kot:

$$S_{nova} = \left\{ \mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^i, \ldots \mathbf{x}^\Gamma \right\}, \quad \mathbf{x}^i \in \mathcal{S}_N$$

  Iz originalne množice vzorcev $\mathcal{S}_N$ naključno vzamemo Γ-krat vhodni oz. opazovalni vektor.

- Kako pa v fazi učenja uporabljamo opazovalne vektorje iz množice $\mathcal{S}_{nova}$?

  1. Število elementov množice $\mathcal{S}_{nova} > T_E$
     - ⋆ V procesu učenja jemljemo opazovalne vektorje kar zaporedoma iz množice $\mathcal{S}_{nova}$.

  2. Število elementov množice $\mathcal{S}_{nova} < T_E$
     - ⋆ Med učenjem jemljemo vzorce iz množice $\mathcal{S}_{nova}$ zaporedoma do zadnjega elementa.
     - ⋆ Nato vse elemente v množici $\mathcal{S}_{nova}$ naključno premešamo ter nadaljujemo s procesom učenja, in sicer tako, da ponovno zaporedno jemljemo učne primerke iz preurejene množice $\mathcal{S}_{nova}$.

Napotki za uporabo ekvivalentnega algoritma (5)

C. Merjenje učinkovitosti ekvivalentnega algoritma

1. Energija kvantizacijskega šuma – mera, kako kvalitetno se je izvedla kvantizacijska faza ekvivalentnega algoritma.

$$\mathrm{E}_{\mathrm{kvant\_šum}} = \frac{1}{N} \sum_{\mathbf{w}_s} \sum_{\mathbf{x}_i \in \mathcal{R}_s} ||\mathbf{x}_i - \mathbf{w}_s||^2$$

2. Ocena energije topologije – praktična mera za ocenjevanje, kako dobro se ohranja topologija nevronske mreže.
   - ▷ Izračun: Za vsak par nevronov, ki imajo sosednji sprejemni polji, izračunamo razdaljo med tema nevronoma v nevronski rešetki. Na koncu vsoto vseh teh razdalj še delimo s številom vseh parov nevronov s sosednjimi sprejemnimi polji.
   - ▷ Pri popolnem ohranjanju topologije je ta ocena energije topologije enaka 1.
3. Množica preprostih mer učinkovitosti:
   - ▷ Število geometrijskih sosedov – število parov nevronov v nevronski rešetki, ki so oddaljeni za 1 oz. razdalja $d_L = 1$.
   - ▷ Število efektivnih bližnjih sosedov – število parov sosednjih sprejemnih polj, ki pripadajo nevronom z razdaljo $d_L = 1$.
   - ▷ Število efektivnih sosedov – število parov sosednjih sprejemnih polj.
   - ▷ Število efektivnih oddaljenih sosedov – število parov sosednjih sprejemnih polj, ki pripadajo nevronom z razdaljo $d_L > 1$.