Multimedijski sistemi – izpitna vprašanja

1. Razložite bistveno razliko med barvnim prostorom RGB in CIE Lab. Čemu je primarno namenjen RGB in čemu Lab prostor?

Bistvena razlika: CIE LAB barve transformira tako, da bolje odslikavajo človeško zaznavanje barve. Če sta človeku dve barve podobne želimo da sta si enako blizu tudi v barven prostoru.

RGB: aditivni model, uporablja se v monitorjih (CRT) in projektorjih
CIE Lab: uporaba za pretvorbo iz RGB v CMYK, ker lahko s CIE Lab predstavimo vse barvne odtenke

2. Podana je 4-bitna sivinska slika – za podano sliko določite histogram in rezultat operacije *razteg histograma*.

6	**5**	**7**	**3**	**5**	**3**	$f_{ac}(a)=a_{min}+(a-a_{low})*\frac{a_{max}-a_{min}}{a_{high}-a_{low}}$	13	11	15	6	11	6
2	**1**	**3**	**0**	**2**	**0**		4	2	6	0	4	0
5	**3**	**5**	**1**	**7**	**2**	$a_{low} = 0$, $a_{high} = 7$	11	6	11	2	15	4
1	**3**	**3**	**2**	**1**	**4**	$a_{min} = 0$, $a_{max} = 15$	2	6	6	4	2	9
3	**2**	**1**	**4**	**3**	**2**	h(i)|4 5 6 9 2 5 1 4	6	4	2	9	6	4
7	**5**	**7**	**3**	**0**	**0**	i |0 1 2 3 4 5 6 7	15	11	15	6	0	0

3. V obliki psevdokode zapišite postopek za zlivanje dveh slik na podlagi maske z uporabo Laplaceove piramide. // Opiši proces zlivanja slik z uporabo slikovnih piramid.

1. Sestavi Laplacovi piramidi LA in LB za sliki A in B.
2. Sestavi Gaussovo piramido GR za izbrano regijo R v sliki.
3. Sestavi novo Laplacovo piramido LS tako, da vsak nivo piramide posebej izračunaš kot uteženo kombinacijo istega nivoja v piramidah LA in LB, kjer za utež uporabiš uteži piramide GR:
   LS(i, j) = GR(i, j)*LA(i, j) + (1 – GR(i, j))*LB(i, j)
4. Kolapsiraj piramido LS, rezultat pa je sestavljena slika.

Sliko predstavimo kot zaporedje slik, kjer vsaka slika vsebuje nižje frekvence. Sliko zgladimo z Gaussovim filtrom in jo zmanjšujemo za polovico; postopek ponavljamo; z odštevanjem zaporednih nivojev dobimo frekvenčne pasove → Laplacova piramida (LS - vsota vseh nivojev te piramide je originalna slika).

4. Naštejte strategije reševanja problema tipkarskih napak pri tekstovnem poizvedovanju.

1. predlagaj popravke za besede, ki jih ni v slovarju
2. Ponudi najbolj podobno (črkovna ali fonetična podobnost)
3. Če je več enako podobnih popravljenih besed, ponudi tisto, ki jo uporabniki najpogosteje uporabljajo

5. Naštejte in na kratko opišite osnovne korake pri kompresiranju slik po standardu JPEG.

1. Transformacija signala (DCT) → sliko razdeli v 8x8 bloke, vsak blok pa v koeficiente DCT
2. Kvantizacija → kvantizira vrednosti koeficientov DCT
3. Mapiranje koeficientov v simbole → npr. kodiramo razliko med DC koeficienti zaporednih blokov
4. Kodiranje s Huffmanovim postopkom → simbolom fiksne dolžine predpišemo kodne besede različnih dolžin (krajše besede bolj pogostim simbolom)

6. Imamo dva sistema za priklic (A in B) na podlagi primera Q, ki ju želimo ovrednotiti. Sistema za podani primer Q podata oceno podobnosti za vsak vzorec v bazi X_i. Poleg tega imamo na voljo tudi *zlati standard* G_Q, t.j., za vsak vzorec v bazi X_i vemo ali dejansko pripada istemu razredu kot podani primer Q. Recimo, da smo za 10 vzorcev izračunali ocene podobnosti A_Q in B_Q:

A_Q = [0.5 0.3 0.6 0.22 0.4 0.51 0.2 0.33 0.23 0.7]
B_Q = [0.04 0.1 0.68 0.22 0.4 0.11 0.8 0.53 0.5 0.08]
G_Q = [1 0 1 1 1 0 0 0 1 1]

Na podlagi podatkov narišite ROC krivulji za sistema A in B ter določite, kateri sistem je na podlagi analize boljši (odgovor tudi utemeljite s koncepti ROC analize).

sum(G_Q==1) = 6 sum(G_Q==0) = 4
A' = [0.7 0.6 0.51 0.5 0.4 0.33 0.3 0.23 0.22 0.2]
$G_{A'}$ = [1 1 0 1 1 0 0 1 1 0]
B' = [0.8 0.68 0.53 0.5 0.4 0.22 0.11 0.1 0.08 0.04]
$G_{B'}$ = [0 1 0 1 1 1 0 0 1 1]

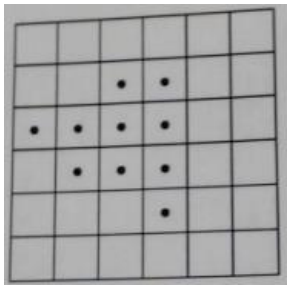
Boljši je sistem A, saj je krivulja najbližje točki (0, 1) (to je mera analize uspešnosti pri ROC).

7. Inženir je dobil nalogo izdelati sistem za poizvedovanje po dokumentih. Problem je omejen tako, da je slovar terminov poznan vnaprej, nabor dokumentov pa je omejen in se ne spreminja. Sistem mora omogočati poizvedovanje z Boolovimi izrazi. Razložite, kakšno podatkovno strukturo mora inženir implementirati za učinkovito poizvedovanje in kakšne so prednosti te strukture pred naivnimi rešitvami, kot je incidenčna matrika.

Rešitev je izgradnja ***obrnjenega indeksa*** → za vsak termin tvorimo seznam (indeksov) dokumentov, ki ta termin vsebujejo, zgradimo ga vnaprej:
1. zgradimo seznam parov (termin, dokumentID)
2. seznam parov uredimo po abecedi
3. če se nek temin v seznamu parov pojavi večkrat, preštejemo kolikokrat se pojavi, nato pa za ta termin sestavimo seznam dokumentov v katerih se pojavi (dolžinaSeznama=štPojavitev).

Pri poizvedovanju z boolovimi izrazi torej pogledamo preseke seznamov dokumentov vseh terminov, ki nastopajo v seznamu (pri urejenih seznamih je časovna zahtevnost linearna).
Prednost obrnjenega indeksa je hitrejše poizvedovanje (linearna kompleksnost), slabost incidenčne matrike pa je omejitev računalniškega spomina saj le-ta za vsak(!) posamezen termin vsebuje info kater dokument vsebuje ali ne (matrika je redka, vsebuje veliko ničel).

8. Za regijo, ki jo opisuje spodnja binarna slika izračunajte absolutno verižno kodo, Freemanovo diferenčno verižno kodo ter absolutno Freemanovo diferenčno verižno kodo.



AVK = 066634301
FDVK = 600517517 (odštevanje parov, nato mod8(št. smeri))
AFDVK = 005175176 (shift v levo dokler ne dobimo najmanjše številke)

	(6-0)	*(6-6)*	*(6-6)*	*(3-6)*	*(4-3)*	*(3-4)*	*(0-3)*	*(1-0)*	*(0-1)*
mod8	*6*	*0*	*0*	*-3*	*1*	*-1*	*-3*	*1*	*-1*
	6	*0*	*0*	*5*	*1*	*7*	*5*	*1*	*7* *(fdvk)*

9. Naštejte in opišite korake slepe povratnozančne poizvedbe.

1. s *standardno* metodo poišči najbolj relevantne dokumente
2. predpostavi, da je *K* najvišje rangiranih dokumentov relevantnih *(zato slepa!)*
3. izvedi *povratnozančno* relevanco (npr. Rocchio)

10. Naštejte tri glavne tipe slik, ki jih srečamo v video toku stisnjenim s standardom MPEG-1. Vsak tip tudi na kratko opišite.

Slike **I** (intra frames): približno 12 slik se pojavi med zaporednimi slikami tipa **I**; omogočajo naključen dostop (so polne slike; lahko jih *v celoti* dekodiramo *neodvisno* od ostalih slik)

Slike **P** (predictive frames): kodirane s predikcijo iz predhodne slike **I** ali referenčne **P**; za vsak makroblok v sliki izračunamo premik glede na prejšnjo sliko (=vektor gibanja → samo iz kanala Y)

Slike **B** (bi-directional frames): kodirane s predikcijo prejšnje/prihodnje slike **I**/**P** preko povprečenja *kompenziranih* slik (so v parih; za dekodiranje mora biti posamezna B slika med *I & B* oz. med *P & B*, pri čimer so *I* oz. *P* slike že dekodirane)

11. Inženir bi rad načrtal aplikacijo za zmanjševanje širine slike. Vendar bi rad zagotovil, da se ob zmanjšanju širine ne bi zožali tudi objekti v slikah. Razložite, kako bi ta problem naslovili in jasno opišite vse korake postopka.

Rezanje šiva: za vsak piksel v sliki izračunamo pomembnost (energijsko vrednost) ter iz slike odstranimo nepomembne piksle (piksli z manj energije).
Šiv: povezana pot pikslov od zgornje do spodnje vrstice (ali od levega do desnega roba slike)

1. izračunamo energijsko mapo za sliko
2. poiščemo *zvezno* pot pikslov od zgornjega do spodnjega roba (piksli na poti morajo imeti *minimalno* energijo). V vsaki naslednji vrstici izberemo piksel, ki je levo, desno ali tik pod pikslom prejšnje vrstice.
   M(i, j) = E(i, j) + min(M(i-1, j-1), M(i-1, j), M(i-1, j+1))
4. iz slike izrežemo zvezno pot (šiv)

12. Imamo dvorazredni razvrščevalnik slik C, ki za zbirko slik vrne odzive, ki so podani v spodnji tabeli, skupaj z zlatim standardom. Za razvrščevalnik narišite ROC krivuljo ter izračunajte F mero za optimalno pragovno vrednost.

$\xi_{score}^{(C)}$ **= [0.5 0.3 0.6 0.22 0.4 0.51 0.2 0.33]** →→ [0.6 0.51 0.5 0.4 0.33 0.3 0.22 0.2]
ξ_{id} **= [1 0 1 1 1 0 0 0]** → → [1 0 1 1 0 0 1 0] (P=4, N=4)

$FP_R = FP/N$ → $FP = FP_R*N = 1/4*4 = 1$ Re=TP/P=3/4
$TP_R = TP/P$ → $TP = TP_R*P = 3/4*4 = 3$ Pr=TP/(TP+FP)=3/4
F = 2PrRe/(Pr+Re) = 3/4

13. Naštejte tri tipe opisov slike in za vsak tip navedite po en opisnik.

Barva (histogram), tekstura (sopojavitvena matrika), oblika (Freemanova koda).

14. Imel si slike 1, 2, 3, 4, 5 in si jih moral povezat z ustreznim a, b, c, d, e odzivom Fourirjevega spektra. Pojasni svojo odločitev?

15. Podane imamo tri dvo-bitne slike, I_1, I_2 in I_3. Z uporabo energije sopojavitvene matrike za jedro d = (x, y) = (1, 1) določite, katera izmed slik I_2 in I_3 je bolj podobna sliki I_1. Energija sopojavitvene matrike je definirana kot $\xi_{ener}(A) = \Sigma_{i,j} C^2(i, j)$.

$$I_1 = \begin{matrix} 0 & 1 & 3 & 0 & 0 \\ 0 & 2 & 3 & 2 & 2 \\ 0 & 1 & 2 & 1 & 1 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \end{matrix} \quad I_2 = \begin{matrix} 0 & 1 & 3 & 0 & 0 \\ 0 & 2 & 3 & 3 & 3 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 2 & 0 & 0 \\ 1 & 1 & 2 & 0 & 0 \end{matrix} \quad I_3 = \begin{matrix} 0 & 1 & 3 & 0 & 0 \\ 1 & 2 & 3 & 2 & 2 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \end{matrix}$$

$$C_1 = \begin{matrix} 1 & 3 & 2 & 0 \\ 1 & 0 & 2 & 1 \\ 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{matrix} *(1/16) \quad C_2 = \begin{matrix} 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 1 \end{matrix} *(1/16)$$

$$C_3 = \begin{matrix} 1 & 2 & 2 & 0 \\ 3 & 3 & 0 & 1 \\ 0 & 2 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{matrix} *(1/16) \quad C_1^2 = \begin{matrix} 8 & 5 & 10 & 3 \\ 5 & 6 & 5 & 0 \\ 5 & 7 & 7 & 1 \\ 3 & 1 & 3 & 1 \end{matrix} *(1/256) \quad C_2^2 = \begin{matrix} 8 & 8 & 6 & 5 \\ 7 & 5 & 3 & 3 \\ 4 & 4 & 2 & 2 \\ 2 & 4 & 4 & 3 \end{matrix} *(1/256) \quad C_3^2 = \begin{matrix} 7 & 12 & 2 & 2 \\ 12 & 16 & 7 & 3 \\ 6 & 6 & 0 & 2 \\ 3 & 5 & 0 & 1 \end{matrix} *(1/256)$$

$E(C_1^2) = 70/256 = 0.2734375$
$E(C_2^2) = 70/256 = 0.2734375$
$E(C_3^2) = 84/256 = 0.328125$

Izmed I_2 in I_3 je sliki I_1 bolj podobna slika $\mathbf{I_2}$.

16. Podani so trije dokumenti, ki jih vnesemo v sistem za iskanje po dokumentih. Sistem za iskanje upošteva samo besede, ki so daljše od štirih znakov. Za ta nabor dokumentov slovar ter obrnjen indeks, v katerega nato indeksirate dokumente. Na podlagi dokumentov določite tudi seznam besed, ki jih opisani sistem opredeli kot stop besede.

Dokument 1	**Dokument 2**	**Dokument 3**
A question is an expression used to make a request for information.	**Information is a sequence of symbols that can be interpreted as a message.**	**A message is a vessel which provides information.**

[termin, docID]	[termin, docID – a..z]	[termin, stPojavitev, seznamDocID]
question, 1	expression, 1	***[obrnjen indeks]***
expression, 1	information, 1	
request, 1	information, 2	expression, 1, [1]
information, 1	information, 3	information, 3, [1, 2, 3]
information, 2	interpret, 2	interpret, 1, [2]
sequence, 2	message, 2	message, 2, [2, 3]
symbol, 2 →→	message, 3 →→	provide, 1, [3]
interpret, 2	provide, 3	question, 1, [1]
message, 2	question, 1	request, 1, [1]
message, 3	request, 1	sequence, 1, [2]
vessel, 3	sequence, 2	symbol, 1, [2]
provide, 3	symbol, 2	vessel, 1, [3]
information, 3	vessel, 3	

Stop besede: [a, is, an, to, for, of, that, can, be, as, which] (nimajo velike informacijske vrednosti)
Slovar:[expression, information, interpret, message, provide, question, request, sequence, symbol, vessel]

17. Uporabnik je želel narediti aplikacijo ki zmanjša sliko na polovico velikosti. To je naredil tako, da je odstranil vsako drugo vrstico in vsak drugi stolpec. Je to v redu? Kako bi izboljšal aplikacijo?

Ta pristop ni primeren, saj ne ohranja pomembnih struktur v sliki/videu; ker je človeški vid bolj občutljiv na robove bi brisali tudi robove, zati bo objekti na sliki delovali popačeni.
To aplikacijo bi izboljšal tako, da bi sliko manjšal z *rezanjem šivov* → raje bi odstranjeval vsebino v gladkih predelih slike oz. piksle v sliki, ki so nepomembni (imajo manj energije). Šiv je *povezana* pot energijsko minimiziranih pikslov od zgornje do spodnje vrstice → z iteriranjem rezanja šivov bi torej po zmanjšanju slike ohranil pomembne objekte na sliki in zavrgel nepomembne.

18. Opišite glavne korake obogatene resničnosti z oznako od zajema posamezne slike, do ustreznega izrisa dodane vizualne informacije na sliko. Razložite, zakaj sploh potrebujemo oznako.

Bistvo: Iz slike deformiranega ploščatega markerja/oznake je možno oceniti orientacijo kamere.
1. Detekcija markerja (z iskanjem povezanih regij) → pravilno označimo oglišča/značilne točke (*določimo možno oznako (rektifikacija oznake), vzamemo model markerja in ju primerjamo z NCC*) → s tem marker vpnemo v svetovni koordinatni sistem, tako pa lahko preko homografije (ocenimo jo s pomočjo referenčne & dejanske slike) izračunamo *R & t* kamere
2. Orientacija kamere -> izračun projekcijske matrike ($P = K[R|t]$) → ugotovimo orientacijo in položaj kamere v svetovnem koordinatnem sistemu
3. Projekcija 3D modela v sliko → ker smo ugotovili kje v svetovnih koordinatah leži kamera, lahko v sliko enostavno projekciramo 3D model ($x_c = P*x_w$)

19. Razložite postopek Huffmanovega kodiranja. Na kakšni predpostavki to kodiranje temelji? Recimo, da imate zgrajeno Huffmanovo tabelo kodnih besed. Kako mora neka slika kršiti to predpostavko, da bo kodiranje s to tabelo neučinkovito?

Huffmanovo kodiranje: simbolom fiksne dolžine predpiše kodne besede različnih dolžin. Gre za statistično metodo → predpostavka: krajše kodne besede pripišemo bolj pogostim simbolom

Postopek: zapišeš znake po padajočih verjetnostih pojavitve (štPojavitevZnaka/štVsehZnakov), potem pa združuješ po 2 simbola in seštevaš verjetnosti in na vsakem koraku spet urediš po padajočih verjetnostih dokler ne prideš do verjetnosti 1, potem pa v obratni smeri poljubno na vsako od 2 vej s katerima si prej združeval napišeš 0 ali 1; ko prideš do začetka maš torej za vsak znak sestavljeno kodno zamenjavo.

Kršitev: Slika je neprimerna, če so barve/vrednosti pikslov enakomerno razpršene, to pomeni da nobena ne prevladuje, in ni nekega učinkovitega kodiranja s katerim bi zmanjšal število znakov v kodni zamenjavi ker so vse kodne zamenjave enako dolge (po možnosti celo daljše od osnovnega znaka, kar je precej neučinkovito).

V mislih moramo imeti, da ima Huffman omejitev vsaj 1bit/simbol oz. cela števila.

20. Kaj je podvzorčenje v videu? (nekaj takega)

Ker ljudje bolje opazijo spremembe v intenziteti kot spremembe v barvi, lahko barvnost podvzorčimo.
Poseben zapis določa, koliko pikslov (glede na izvorne 4 piksle) dejansko obdržimo → J:a:b.
V videu pri MPEG-1 podvzorčimo le ***Slike I***.

21. Imel si eno sliko narisano ter podano transformacijsko matriko in si moral povedat, kako bo izgledala transformirana slika in kakšen bi bil hiter in učinkovit postopek za to transformacijo?

22. Recimo, da imamo video sekvence kodirano s standardom MPEG-1, tipi slik v zaporedju pa so: $I_1, B_2, B_3, P_4, B_5, B_6, P_7, B_8, B_9, P_{10}, B_{11}, B_{12}, I_{13}, B_{14}, B_{15}, P_{16}, B_{17}, B_{18}, P_{19}, B_{20}, B_{21}, P_{22}, B_{23}, B_{24}, I_{25}$ … Če želimo prikazati sliko z zaporedno številko 18 – katere vse slike mora dekoder prej nujno dekodirati? Kaj pomeni skalabilnost kodeka?

Trenutna slika B je kodirana s prejšnjo in naslednjo sliko tipa I/P preko povprečenja kompenziranih slik.
Za prikaz B_{18} rabimo torej prej dekodirati: **I_{13}, P_{16}, P_{19}, (B_{17}), B_{18}**

Skalabilnost kodeka:
- Hkratna podpora različnih resolucij in bitnih hitrosti
- Bitna hitrost je odvisna od resursov, ki so v nekem trenutku na voljo. Če se možna bitna hitrost prenosa v komunikacijskem kanalu *naenkrat zniža*, je pomembno, da je kodiranje izvedeno tako, da ni treba ponovno kodirati za spremenjeno hitrost prenosa.

23. Prednosti digitalnega videa pred analognim?

- Shranimo ga v digitalnih napravah
- Zapis omogoča direktno manipulacijo (odstranjevanje šuma, rezanje, itd.) in uporabo v multimedijskih aplikacijah
- Omogoča direktni dostop do različnih delov videa
- Presnemavanje ne poslabša kvalitete
- Lažje dekodiranje in boljša toleranca šuma

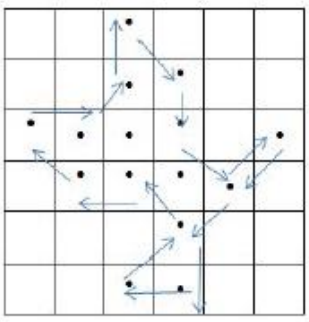
24. 2 ROC krivulji. Izbrati morate klasifikator, ki bo najprimernejši za aplikacijo. Naročnik pravi, da si lahko njegova aplikacija privošči, da na 100 obravnavanih pozitivnih primerov vsaj 90 primerov opredeli kot true pozitivne. Na podlagi ROC analize izberite bolj primeren klasifikator in utemeljite svoj odgovor. Ocenite število napačno klasificiranih primerov v izbranem scenariju.

TPr = TP / P = true_pozitivni / vsi_pozitivni = 90/100 = 0.9

Na grafih odčitaš *FPr* pri *TPr=0.9* in vidiš da je ta pri prvem grafu mislim da 0.4 pri drugem pa 0.8 kar pomeni, da je prvi klasifikator boljši saj nam poleg TP klasificira manj FP *(prav tako je delovna točka bližje točki (0,1) → ta razdalja je pravzaprav mera analize uspešnosti pri ROC*).

$d = Sqrt((x_2-x_1)^2+(y_2-y_1)^2)$

25. Freemanova koda



(absolutna) Freemanova verižna koda: 767155641343012
(Freemanova) Diferenčna verižna koda: 712401652175115 (spodaj)
(absolutna Freemanova) Normalizirana diferenčna verižna koda: 016521751157124 (shift v levo do najmanjše številke)

(6-7)	(7-6)	(1-7)	(5-1)	(5-5)	(6-5)	(4-6)	(1-4)	(3-1)	(4-3)	(3-4)	(0-3)	(1-0)	(2-1)	(7-2)
-1	1	-6	4	0	1	-2	-3	2	1	-1	-3	1	1	5
mod8: *7*	*1*	*2*	*4*	*0*	*1*	*6*	*5*	*2*	*1*	*7*	*5*	*1*	*1*	*5*

26. Pri poizvedovanju po videoposnetkih (Video Google) gre za preslikavo konceptov razvitih za tekstovno poizvedovanje za primer videa. Določite preslikavo spodnjih terminov tekstovnega poizvedovanja v termine poizvedovanja po videoposnetkih.

Gre za preslikavo standardnih postopkov za tekstovno poizvedovanje na domeno slik & videa.

[tekst |→| vizualna informacija]

beseda → opisnik (vizualna beseda)

koren → centroid (gručenje podobnih vizualnih besed)

dokument → okvir/slika

korpus → video posnetek

27. Opišite postopek stiskanja slike tipa I po standardu MPEG-1.

Slike I omogočajo naključni dostop, saj jih lahko dekodiramo neodvisno od ostalih slik.

1. Barvno podvzorčenje (Cr&Cb kot pri JPEG)
2. Sliko razdeli v makrobloke (= 6 blokov 8x8)
3. Vsak makroblok kodiraj podobno kot pri JPEG; razlika je v kvantizacijskih tabelah.

28. Korpus *N = 1000* dokumentov, poizvedba *q = "Multimedijski sistemi"*. Termin '*multimedija*' se pojavi v *20* dokumentih in v dokumentu d_1 *10*-krat. Termin '*sistem*' se pojavi v *100* dokumentih in v d_1 *20*-krat. S postopkom *tf-idf* izračunajte utež d_1 ob poizvedbi *q*.

$tf\text{-}idf_{t,d} = tf_{t,d} * idf_t = tf_{t,d} * \log_{10}(N/df_t)$ // utež termina

$Score(q, d) = sum(tf_{t,d} - idf_{t,d})$ // izračun uteži dokumenta *d* ob poizvedbi *q* == vsota uteži vseh terminov

$Score(q, d) = 10 * \log_{10}(1000/20) + 20 * \log_{10}(1000/100) =$ ***36.9897***

29. JPEG Na kratko opišite bistvene dele korakov kompresije barvne slike s postopkom JPEG. V katerem koraku in kako pride do izgube v kompresiji? Kaj je to kvantizacija?

1. predprocesiranje → RGB sliki ločimo intenzitete in barvnost (prostor Y Cr Cb):
   - *Y* pustimo kot je, *Cr & Cb* pa navadno podvzorčimo s faktorjem 2 (saj je človeško oko manj občutljivo na spremembe barve kot na spremembe svetlosti)
2. transformacija/stiskanje → kot pri običajni JPEG kompresiji (le da za vsak kanal posebej):
   - različna kvantizacijska tabela za Y kot za Cr & Cb (Y=16x16, Cr&Cb=8x8)
   - Apliciramo 2D DCT na vsak blok velikosti 8x8 (kanala Cr & Cb) → projekcija bloka na 8x8 baznih funkcij
   - Tabelo koeficientov DCT delimo s kvantizacijsko tabelo, zaokrožimo rezultat in ponovno zmnožimo
3. mapiranje koeficientov v simbole (kodiranje razlike med DC koeficienti zaporednih blokov)
4. kodiranje simbolov → ponavadi s Huffmanom

Izguba: nastopi v koraku stiskanja s kvantizacijo DCT koeficientov.
Kvantizacija: postopek, kjer koeficiente DCT delimo s kvantizacijsko tabelo, zaokrožimo rezultat in ponovno zmnožimo.

30. Razlika med progresivnim in prepletenim zapisom slike v videu?

Prepleten zapis: siko razdelimo na dve sliki (ena s sodimi, druga z lihimi vrsticami)
- slabost: hitri premiki → striženje
- prednost: manjše utripanje brez povečanja hitrosti prikaza celih slik (v analognih prikazovalnikih)

Progresiven zapis: vsako sliko zapišemo v celoti

31. V aplikaciji obogatene resničnosti smo detektirali marker in izračunali homografijo med markerjem in našim modelom markerja. Razložite, kako s pomočjo te homografije izračunamo parametre kamere, ki so potrebni za reprojekcijo umetno ustvarjenega objekta v sliko.

Ko imamo izračunano homografijo, lahko iz nje dobimo rotacijsko matriko *R* in translacijsko matriko *t*. Z izračunanima *R & t* lahko izračunamo projekcijsko matriko *P* po enačbi:

$$P = K[R \mid t],$$

kjer je K kalibracijska matrika kamere.S *P* matriko lahko projekciramo posamezni objekt/točko iz svetovnih koordinat v koordinate kamere:

$$X_c = P * X_w$$

Po tem na koncu še X_c normaliziramo, tako da za zadnji element dobimo 1 (homogene koordinate).

Tu rabimo izpeljavo R in T iz H:
- Oceno R in t inicializiraj preko homografije H
- Dokler se spreminjata R in t:
  1. Z nelinearno optimizacijo minimiziraj E(R,t) po parametru R → dobimo oceno ~R
  2. Z najmanjšimi kvadrati minimiziraj E(~R,t) po parametru t

$H = K[r_1, r_2, t] = [h_1, h_2, h_3]$ $B = [r_1, r_2, t] = [b_1, b_2, b_3]$ $P = K[R|t]$
$X_c = P*X_w$ $\lambda H = K\sim B$ $\sim B = \lambda K^{-1}H$
$B = \lambda\sim B*(-1)^{|\sim B|<0}$ → *objekt mora biti pred kamero*
$r_1 = \lambda b_1$ $r_2 = \lambda b_2$ $r_3 = r_1 X r_2$
$t = \lambda b_3$ $\lambda = ((||K^{-1}h_1||+||K^{-1}h_2||)/2)^{-1}$

32. MPEG-1: kaj je makroblok? Kaj je rezina?

Makroblok je sestavljen iz 16x16 blokov slike za kanal Y ter 8x8 blokov slike za kanala Cr & Cb.
Rezina je zaporedje makroblokov od leve proti desni in od zgoraj navzdol.

33. Katere barvne modele poznaš, za katere naprave se uporabljata?

Aditivni model (RGB): izhodiščna barva je *črna*, nato se *dodajajo* primarne barve → uporablja se za CRT monitorje, projektorje

Substraktivni model (CMY = 1-RGB): izhodiščna barva je *bela*, nato se *dodajajo pigmenti*, s katerimi *odvzemamo (absorbiramo)* barve → uporablja se za tiskalnike, fotoaparate (film), (voščenke)

34. Kako zgradimo obrnjen indeks? Zakaj uporabljamo pozicijski indeks?

Obrnjen indeks vedno zgradimo vnaprej, zberemo dokumente za indeksiranje, tokeniziramo tekst, normaliziramo besede, sestavimo obrnjen indeks → vsak dokument ima lasten *ID*, nato zgradimo tabelo parov *(termin, id)*, ta seznam uredimo po abecedi, termine napišemo le 1x (poleg zapišemo še frekvenco ponavljanja), nato pa za vsak termin zapišemo seznam *ID*-jev dokumentov, kjer se termin pojavi.
Pozicijski indeks uporabimo pri iskanju fraz (2 ali več terminov/besed je povezanih). To dosežemo tako, da za vsak termin v slovarju shranimo pozicijo, kjer se pojavlja. Pozicijski indeks poveča velikost seznama indeksa v primerjavi z običajnim (zgoraj omenjenim) obrnjenim indeksom (poveča se zahteva po shranjevanju ter čas iskanja).

35. Kako bi skrčil sliko po širini, ne da bi se objekti popačili?

Rezanje šivov: za vsak piksel izračunamo njegovo pomembnost (energijo) → odstranimo nepomembne piksle (tiste z manj energije). Naredimo potrebno število šivov po dolžini (posamezna pot šiva sledi pikslom z najmanj energije *gor->dol* oz. *levo->desno*).
S tem bi se izognil brisanju robov objektov (na katere je človeški vid bolj občutljiv), saj bi raje odstranjeval vsebino v gladkih predelih slike oz. piksle, ki so v sliki nepomembni.

36. Kako poteka poizvedovanje s povratnimi zankami?

Uporabnik posreduje povratno informacijo o *relevanci prvotno* vrnjenih dokumentov → **povratnozančna relevanca**. Primer le-te je algoritem Rocchio, ki poizvedbo in dokumente predstavi v vektorskem prostoru (imamo poizvedbo ter nekaj relevantnih & nerelevantnih primerov) → želimo si umetne poizvedbe (novo iskanje), ki je *max* podobna relevantnim && *min* podobna nerelevantnim rezultatom.
Ločimo 2 tipa povratnih zank:

1. Slepa: predpostavi, da je *K* najvišje rangiranih dokumentov relevantnih
2. Navadna: uporabnik klika na dokumente, katere je dobil in to potem pove, kateri so najvišji relevantni dokumenti

37. Lokalni binarni vzorci

Opiše globalno teksturo z lokalnimi opisniki → za vsak piksel izračunamo 8-bitno številko (piksel ima 8 sosedov), pri čemer vsak bit nastavimo odvisno od tega ali je večji (b_i==1) ali manjši (b_i==0) od gledanega piksla. Nato vsa 8-bitna števila vseh pikslov pretvorimo v desetiška, kjer ta števila v histogramu predstavljajo strukturo teksture.
[piksli si od b_0 do b_7 sledijo od levega zgoraj v smeri urinega kazalca okrog gledanega piksla]

38: Razlika med MPEG-1 in MPEG-4

MPEG-1: cilja na nizke kompleksnosti dekoderjev; predvajanje nazaj in hitro predvajanje; normalno predvajanje z naključnim dostopom; slika je razdeljena na makrobloke in rezine
MPEG-4: omogoča višje resolucije (HDTV, BluRay); poudarja interakcijo (tudi na nivoju posameznih objektov); boljša učinkovitost kodiranja z več hitrostmi prenosa; robustno deluje v okoljih s pogostimi napakami; vsak objekt se prenaša v svojem toku

Stisljivost je mnogo večja pri MPEG-4 kot pri MPEG-1.

39. Preko wireless povezave priklopimo monitor na Siol box, na ekranu se v videu oz. sliki pojavljajo zelene črte. Kaj je to?

Pri prenosu prihaja do izgube rezin.

40. Prednosti obrnjenega indeksa

Rabi manj računalniškega spomina kot navadno indeksiranje (ni redek kot incidenčna matrika), vedno ga zgradimo vnaprej, pri urejenih seznamih ponuja linearno kompleksnost.
[Navadno indeksiranje je hitrejše od iskanja brez indeksiranja.]

41. MPEG-1 - kako poteka stiskanje?

Potek stiskanja: imamo tri barvne komponente Y, Cr in Cb. (2:1:1). Y pustimo kot je, Cr in Cb pa podvzorčimo s faktorjem 2. Stiskanje poteka kot pri JPEG: barvne slike (vsak kanal ima tudi svojo kvantizacijsko tabelo in ga posebej stiskamo). V osnovi torej naredimo DCT in kvantizacijo koeficientov DCT (delimo z kvantizacijsko tabelo, ki visoke frekvence bolj grobo kvantizira, zaokrožimo in spet zmnožimo), nato pride na vrsto pretvarjanje v simbole (za DCT koeficiente imamo 1x AD (povprečna sivina) in 63x AC komponent, pri čemer AC komponente še z RLE kodiramo zaradi pojavljanja zaporedij ničel) in nato še Huffman (oboje na koncu je brezizgubno stiskanje).

42. Ravnanje histograma

1. narišemo histogram slike I
2. izračunamo kumulativni histogram h_c (vrednosti se seštevajo od leve proti desni)
3. normaliziramo h_c z $\max(h_c) \rightarrow h_{nc}$
4. pomnožimo h_{nc} z največjo izhodno intenziteto (npr. 255) $\rightarrow h_{mnc}$
5. h_{mnc} uporabimo kot vpogledno tabelo za intenzitete ostalih pikslov (slikovnih elementov) → približno uniformen histogram.

43. SIFT (ali kako opisati značilne regije?)

Najprej moramo vsako detektirano regijo *normalizirati* v referenčno obliko; eliptično regijo rotiramo in jo nato skaliramo v krog → tako transformirano regijo opišemo z opisnikom (npr. SIFT).
SIFT (Scale Invariant Feature Transform) je opisnik, ki opisuje značilne regije. Regije razdelimo na 4x4 podpodročij (16 celic), izračunamo gradiente na vskem pikslu in jih zgladimo preko nekaj sosedov, nato v vsaki celici izračunamo histogram orientacij gradientov (ti imajo 8 smeri), iz vseh 4x4 podpodročij potem naredimo histogram 128 dimenzij → ta histogram je potem naš SIFT.
SIFT je 16 skupaj nanizanih histogramov orientacij gradientov.

44. Izgradnja slovarja, obrnjenega indeksa in stop besed.

Izgradnja slovarja: v slovar napišeš vse termine (samo 1x), ki se pojavljajo v vseh dokumentih in sicer že osnovne izraze (se pravi samo korene && brez stop besed)
Obrnjen indeks: podobno, termine nanizaš po abecedi, dodaš stPojavitev v različnih dokumentih, nato pa pripišeš še seznam *ID*-jev dokumentov kjer se ti termini pojavljajo.
Stop besede: to so besede, ki se pojavljajo zelo pogosto v vseh dokumentih in kot take *nimajo* velike informacijske vrednosti za priklic.

45. ROC: dva primera; 100 pozitivnih, vsaj 80 TP - kater klasifikator boljši, koliko je narobe klasificiranih negativnih (se pravi koliko je FPr?)?

TPr =80/100 =0.8 → označil si na y-osi kje je to, pogledal kaj je tam na x-osi (je blo 0.3 in na drugem 0.6) → prvi primer je boljši ker ima manjši FPr pri 0.8
FPr = 0.3

Boljši je tisti, kjer je delovna točka (0.3, 0.8) bližje točki (0,1) ALI pa kjer ima ROC krivulja večjo ploščino. Lahko tudi izračunaš Fmero → večja je, boljše je (max = 1).

46. Podvzorčenje, kako, zakaj?

Ljudje bolj opazimo spremembe v intenziteti kot pa spremembe v barvi, zato lahko podvzorčimo barvnost. Za to obstaja poseben zapis, ki določa koliko pikslov (glede na izvorne štiri piksle) dejansko obdržimo → oblika ***J:a:b*** *(J=horizontalna referenca vzorčenja; a=št. kromatskih vzorcev v 1. vrstici J pikslov; b=št. dodatnih kromatskih vzorcev v 2. vrstici J pikslov)*.

4:4:4 → obdržimo vse, ni podvzorčenja
4:2:2 → horitontalno podvzorčenje Cr in Cb s faktorjem 2
4:1:1 → horizontalno podvzorčenje s faktorjem 4
4:2:0 → horizontalno in vertikalno podvzorčenje s faktorjem 2

47. 2 ROC krivulji. Izbrati morate klasifikator, ki bo najprimernejši za aplikacijo. Naročnik pravi, da si lahko njegova aplikacija privošči, da na 100 obravnavanih negativnih primerov največ 10 primerov opredeli kot pozitivne. Na podlagi ROC analize izberite bolj primeren klasifikator in utemeljite svoj odgovor. Ocenite število pravilno klasificiranih primerov v izbranem scenariju.

48. Stabilizacija videa: Objektno-centrična stabilizacija - kaj to je, kakšen je postopek/algoritem?

Namen: Skozi čas želimo *spreminjati* položaj *okvirja slike*, tako da izločimo močne *tresljaje* glede na objekt zanimanja.
Objektno-centrična stabilizacija (lokalna): ročno izberemo *objekt* (poiščemo ga lahko tudi z NCC glede na podan model), preračunamo okvirje videa, tako da se položaj objekta glede na okvirje ne spreminja bistveno → potek: objekt ročno označimo, nato objekt poiščemo v naslednji sliki z NCC, določimo trajektorijo (tresenje odpravimo z Gaussovim filtriranjem trajektorije), preračunamo centre slik in nato izrežemo nove slike (njihov center je na sledenem objektu).

49. Globalna stabilizacija videa.

Globalna stabilizacija s poravnavo: ocenimo globalno strukturo in preračunamo frame-e, tako da se položaj globalne strukture ne spreminja bistveno → algoritem mora sam oceniti globalno strukturo.
- potek: določimo značilne točke, določimo korespondenco teh točk skozi pare zaporednih slik, zaporedje slik poravnamo glede na te točke (stabiliziramo video) in nato opravimo filtriranje zaporednih transformacij (izboljšamo stabilizacijo).

Globalna stabilizacija z optičnim tokom: za vsak piksel izračunamo njegov najbolj verjetni premik v naslednjo sliko → uporaben tudi za brisanje artefaktov in teksta v videu ALI vrisovanje manjkajočih delov slike (lukas-kanade).

50. K-ta povprečja (za gruče oz. ko želimo ločiti objekt/ozadje)

Izberemo število razredov (*K*), naključno izberemo centre *K*-tih razredov, za vsak center določimo gruče (točke, ki so jim najbližje), nato na novo izračunamo centre razredov (nov center se izračuna kot povprečje vseh točk, ki smo jih pripisali temu centru/razredu). Postopek ponavljamo, dokler se spremeni vsaj 1 center vseh *K*-razredov (drugače končamo → imamo končne gruče).
Postopek je preprost in konvergira k minimumu razdalje članov razreda, slabost pa je, da je odvisen od začetne izbire centrov in predvideva sferične razrede (občutljiv je na izven ležeče točke).

51. Transformacijski pristop (podobnosti oblik)

Podobnosti med dvemi oblikami merimo kot *stopnjo nelinearne transformacije*, potrebne za preslikavo ene oblike v drugo. Eno sliko transformiramo v drugo s *premik+skaliranje*. Podobnost je nelinearna *transformacijska* distanca. Slabost tega pristopa je, da ne podpira indeksiranja.

52. Kaj je skalabilnost kodeka in kako se to odraža v MPEG-2?

Skalabilnost kodeka:
- Hkratna podpora različnih resolucij in bitnih hitrosti
- Bitna hitrost je odvisna od resursov, ki so v nekem trenutku na voljo. Če se možna bitna hitrost prenosa v komunikacijskem kanalu naenkrat zniža, je pomembno, če je kodiranje izvedeno tako, da ni treba ponovno kodirati za spremenjeno hitrost prenosa.

MPEG-2 skalabilnost kodeka: posnetek ločimo v 2 nivoja:
- osnovni nivo (ON): vsebuje dovolj informacije za grobo rekonstrukcijo slike
- izboljševalni nivoji (IN): vsebujejo inkremente informacije za izboljševanje osnovnega nivoja (lahko jih je več)

-> MPEG-2 ločeno prenaša osnovni nivo slike (visoka prioriteta) in izboljševalni nivo slike (nizka prioriteta)

~ prostorska skalabilnost (velikost slike), frekvenčna skalabilnost (DCT koeficienti), SNR skalabilnost (signal-šum), časovna skalabilnost

53. Postopek BTC kodiranja in dekodiranja, slabosti BTC.

BTC (Block Truncation Coding): preprost postopek izgubnega stiskanja delov slik → sliko razdeli v bloke, vsak blok aproksimira z 2 intenzitetama (*srednja vrednost & standardni odklon* → po kompresiji enaki!).

Postopek stiskanja BTC (kompresija):
1. sliko razdelimo v mrežo blokov velikosti *nxn*
2. vsak blok upragujemo s srednjo vrednostjo sivinskih nivojev znotraj bloka
3. shranimo upragovano sliko (masko) (bitna globina=1bit → kjer je pod povprečjem je 0, nad pa 1)
4. shranimo povprečno vrednost (8bit) in stanadardno deviacijo (8bit) vsakega bloka

Postopek raztezanja BTC (dekompresija):
Za vsak blok iz srednje vrednosti in variance izračunamo 2 sivini (a, b → eno za 'črne' & eno za 'bele' piksle v maski) → varianca bloka mora biti enaka originalni varianci bloka pred kompresijo.

Slabosti: opazne popačitve zaradi izgubne kompresije → vidijo se robovi med bloki; artefakte opazimo na delih z nizkim kontrastom, kjer se sivine prelivajo med temno in svetlo.

54. Utež termina v dokumentu (term frequency, inverse document frequency)

Uporablja se za rangiranje dokumentov → nekateri termini slabo diskriminirajo med dokumenti, zato pogostejšim terminom zmanjšamo uteži.

tf = kolikokrat se termin pojavi v dokumentu

df_t = v koliko dokumentih v korpusu se pojavi termin t

$idf_t = \log_{10}(N/df_t)$ = inverzna pogostost v dokumentih (N=št. vseh dokumentov v korpusu; odf je nizek za pogost termin, za redek termin pa visok)

$tf\text{-}idf_{t,d} = tf_{t,d} * idf_t = tf_{t,d} * \log_{10}(N/df_t)$ ← utež termina

55. Kako opisujemo teksture?

56. Kako lahko pride do tega, da Huffmanov kod ni optimalno kodiran če imaš vnaprej definirano kodno tabelo?

Tako imamo statičen Huffmanov kod, ki ne računa optimalnih kodov glede na statistično analizo pojavitev simbolov ampak kodira na podlagi vnaprej določene kodne tabele. Huffmanov kod statistično računa frekvence pojavitev možnih vrednosti in glede na pogostost simbolov le-tem (pogostejšim) pripiše krajše kodne zamenjave, s čimer optimalno kodira signal/sliko.

57. Podobnosti slik (energija)

Podobnost glede na barvo: ugotavljamo s histogrami → Hellingerjeva razdalja, Hi kvadrat, presek histogramov, … Problem histogramov je, da barvni histogrami ne upoštevajo položaja, so pa izredno občutljivi na spremembe osvetlitve
Podobnost glede na teksturo: ugotavljamo z sopojavitveno matriko (gledamo, kolikokrat se piksli iz ene določene vrednosti pojavijo skupaj na relaciji d=(dy,dx) v sliki) → iz te matrike lahko izračunamo veliko značilnic (*energija, entropija, kontrast, homogenost, korelacija*)
Podobnost glede na obliko: ugotavljamo sFreemanovo kodo → diskretizirano obliko zapišemo samo s številko; sprehodimo se po *obrisu* regije in vsak premik zakodiramo s smerjo v kateri smo se premaknili

58. Kakšen postopek za detekcijo rezov v videoposnetku bi uporabili, če veste, da se v posnetku spreminja svetlost slik in to neodvisno od prisotnosti reza? Skicirajte osnovne korake postopka in razložite, zakaj bi uporabili ravno ta postopek.

Opazovanje robov: gradimo histograme robov && primerjamo po pikslih koliko robov se je pojavilo in koliko jih je izginilo med zaporednimi slikami.
Imamo dva praga razlike $T_s < T_b$ → če presežemo zgorni prag T_b, je to rez; če presežemo spodnji prag T_s pa je to lahko potencialno začetek prehoda. Slika iz tega koraka se primerja z vsemi naslednjimi, beležimo kumulativno razliko, ki preseže prag T_s in ko kumulativna razlika preseže zgornji prag T_b razglasimo konec postopnega prehoda.

Adaptivna nastavitev praga: gledamo v oknu *t+M/2* in *t-M/2* in največjo razliko definiramo kot rez.

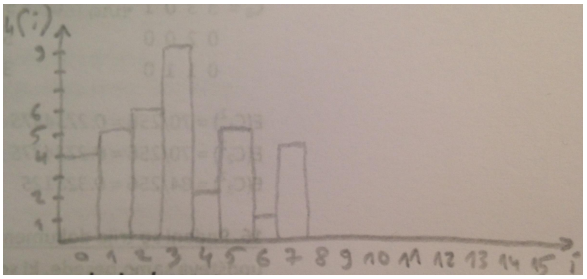
***59. Transformacijska matrika [0 1 0 ; 1/3 0 0 ; 0 0 1] elipsa in pravokotnik...kako se spremenijo, postopek.**

x' = [0 1 0]*|x|	dobimo	x' =y
y' = [1/3 0 0]*|y|		y' =1/3x tale shit potem
1' = [0 0 1]*|1|		1 =1

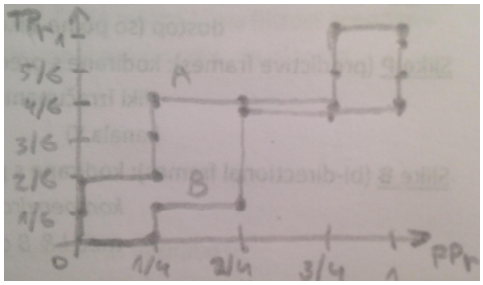
Osi x in y se zamenjata, torej se pravokotnik kar naenkrat postavi pokonci (ker smo zamenjali osi). Ker je y=1/3x, to pomeni, da naredimo zožanje tistega kar je bilo prej na osi x. Torej *rotacija+zožanje*.
Poenostavil bi to tako, da bi zaznal oglišča likov in njihove parametre in transformiral le njih (namesto preslikavanja vsakega piksla posebej).

Mankajoče slike:

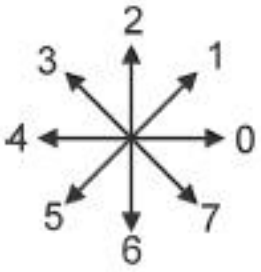
Naloga 2:



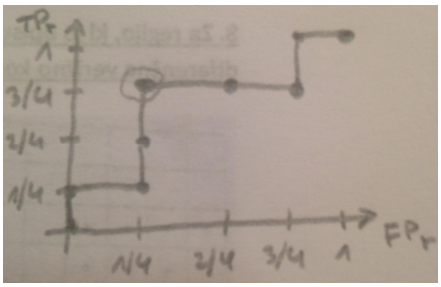
Naloga 6:



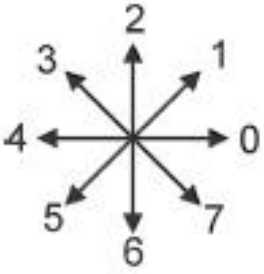
Naloga 8:



Naloga12:



Naloga 25:



Naloga 36: