Elenka Otazky, Hej

Júlia, Hugo, Oliver

May 2023

1 Part 1

1.1 Describe the Viola-Jones method for face detection. Which 3 basic ideas does it use?

- Integrálne obrazy - rýchle vyhodnotenie príznakov
  - integrálne obrazy sú obrazy, ktoré pre každý bod sčítajú intenzitu v ľavej hornej časti obrazu od daného bodu (čiže je to suma intenzít do ľava a hore od daného bodu)

$$IO(x,y) = \sum_{i=1}^{x}\sum_{j=1}^{y} I(i,j)$$

  - Tento prístup je výhodný v tom, že keď chcem určiť akú hodnotu má suma intenzít v takomto obdĺžniku, tak mi stačí pozrieť sa do integrálneho obrazu a skombinovať dané 4 hodnoty. To znamená, že keď si raz predvypočítam integrálny obraz pre nejaký obrázok, tak ľubovoľnú oblasť viem spočítať tromi aritmetickými operáciami:

$$IO(A) - IO(B) - IO(C) + IO(D)$$

    ⇒ čiže nemusím sčítavať všetky intenzity v tejto oblasti, stačí mi zistiť hodnoty z integrálneho borazu a skombinovať ich.
  - To používajú autori na to aby spočítali práve tie príznaky, ktoré sú rozdiely medzi intenzitami v bielych a čiernych oblastiach.
  - príznak je slabý klasifikátor tváre. Slabý klasifikátor je myslený ako klasifikátor, ktorý má aspoň o trochu väčšiu úspešnosť než je náhodná klasifikácia, to znamená, že stačí ak má náhodný klasifikátor úspešnosť 51% a keď ich skombinujem dohromady, tak mi dajú silný klasifikátor, ktorý má vysokú úspešnosť.
  - príznak je teda tvár alebo netvár ak slabý klasifikátor je 1 alebo -1:

$$h_j(x) = \begin{cases} 1, & p_j f_j(x) < p_j \theta_j \\ -1, & inak \end{cases}$$

  - pre každý klasifikátor musím určiť prah Θ_j a paritu p_j a za základe tohto viem zhodnotiť či sa jedná alebo nejedná o tvár.

- Boosting - výber príznakov/klasifikátorov
  - vraví akým spôsobom vyberám slabšie klasifikátory do silnejšieho klasifikátoru
  - mám množinu slabých klasifikátorov, ktoré som si určila. Silný klasifikátor je potom kombinácia týchto slabých klasifikátorov, kde každému z nich dám nejakú váhu, s ktorou ho beriem do úvahy.
  - Napríklad pre klasifikáciu modrých a červených kruhov si môžem zvoliť jednoduchý lineárny klasifikátor. Následne sa pozriem, ktoré objekty boli klasifikované nesprávne a tým zvýšim váhu, čím natrénujem klasifikátor, ktorý dané objekty klasifikuje lepšie.
  - Následne môžem lineárne skombinovať tieto slabé klasifikátory tak, aby bol výsledný klasifikátor úspešný.
  - Jeden z boostovacích algoritmov je napríklad AdaBoost algoritmus.
- Kaskády - rýchle odmietnutie ne-tvárových vzoriek rýchle vyhodnotenie či niečo môže/nemôže byť tvár a následne spresnovať detekciu
  - Kaskádovitá klasifikácia využíva kaskádu klasifikátorov (každá úroveň vylúči nejaké oblasti, ktoré nie sú tváre, čiže postupne sa zväčšuje kvalita klasifikácie):
    * na začiatku by sa mal použiť klasifikátor, ktorý je veľmi dobrý vo vylučovaní oblastí, ktoré nie sú tváre. Keď mi tento klasifikátor vylúči nejaké oblasti, tak tie už ďalej nepostupujú. Pri tomto klasifikátore nepotrebujem aby si bol nutne istý, že to čo pošle ďalej je tvár, ale že to čo vylúči určite tvár nebola.
    * Tréning kaskády = klasifikátory sú postupne trénované na falošne pozitívnych vzrokách predchádzajúcich klasifikátorov. Pri trénovaní kaskády musím nastaviť rôzne parametre (koľko stupňov kaskády chcem natrénovať, koľko príznakov chcem mať v každom stupni, prah v každom klasifikátore)

Fázy algoritmu:

1. Trénovacia časť
   - Pri tréningu sa hľadajú príznaky, ktoré dokážu správne identifikovať či daná oblasť fixnej veľkosti je alebo nie je tvár. Príznaky sú jednoduché obdĺžnikové. Výber príznakov je robený AdaBoost algoritmom.
2. Klasifikácia
   - Pri klasifikácii už len vezmeme okno pevnej veľkosti a škálujem obraz, a pomocou nájdených príznakov vyhodnotím či je to tvár alebo nie je. Takto s oknom prechádzam celý obraz.

1.2 Describe Itti's model of visual attention

Modely vizuálnej pozornosti fungujú tak, že sa preskúmajú výsledky eye trackingu ľudí z iných štúdií a na tom sa natrénuje model, ktorý dokáže vyrobiť mapu pozornosti. Ittiho model berie do úvahy aj biologické znalosti o tom ako funguje oko. Preto sa vstupný obraz rozdelí na 3 paralelné kanály: farba, intenzita, orientácia a v tých sa porovnávajú zmeny. Významnosť oblasti sa v čase zmenšuje čo modeluje nasýtenie pozornosti. Ak sa človek pozerá na nejaký obrázok tak napríklad tváre majú veľkú dôležitosť ale potom ako ich preskúma (pozornosť sa nasýti) sa pozrie aj na ostatné časti obrázku.

Biologické okienko Máme gangliové bunky, čo sú približne kruhové receptívne polia. Tie sa delia na dva typy:

1. ON-center OFF-surround: prítomnosť svetla v strede receptívneho poľa spôsobuje vzruch a v prstencovom okolí spôsobuje anti-vzruch (inhibíciu).
2. OFF-center ON-surround: presne opačne ako v predošlom prípade

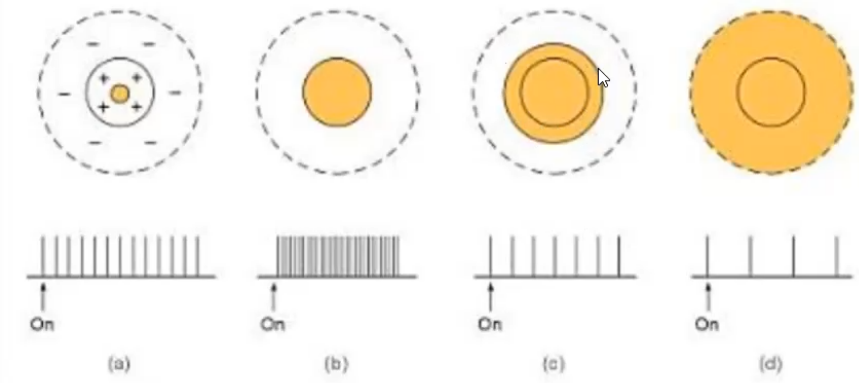


Figure 2.19 Response of an excitatory-center-inhibitory-surround receptive field as stimulus size is increased. Shading indicates the area stimulated with light. The response to the stimulus is indicated below each receptive field. The largest response occurs when the entire excitatory area is illuminated, as in (b). Increasing stimulus size further causes a decrease in firing due to center-surround antagonism. *(Adapted from Hubel and Wiesel, 1961.)*

Okrem toho, že sa dajú bunky rozdeliť podľa toho aká časť na nich reaguje sa delia ešte ďalej:

1. **M bunky** vedia sledovať rýchle zmeny v podnetoch nesú najmä informáciu o hrubých črtách predmetov a pohybe. Teda vedia dobre rozoznávať intenzitu. Sú väčšsie ako P bunky.
2. **P bunky** rozoznávajú farby. Existujú v pároch, ktoré reagujú na červenú a zelenú alebo modrú a žltú

V Ittiho model sa snaží replikovať správanie oka preto sa obrázok delí na nasledovné kanály:

1. $R = r - (g + b)/2$ aplikuje sa Gaussovká pyramída $R(\sigma)$
2. $G = g - (r + b)/2$ aplikuje sa Gaussovká pyramída $G(\sigma)$
3. $B = b - (r + g)/2$ aplikuje sa Gaussovká pyramída $B(\sigma)$
4. $Y = (r + g)/2 - |r - g|/2 - b$ aplikuje sa Gaussovká pyramída $Y(\sigma)$
5. $I = (r + g + b)/3$ intenzita na ktorú sa aplikuje Gaussovká pyr amída $I(\sigma)$
6. $O(\sigma, \theta)$ kanál reagujúci na orientáciu používa Gaborove pyramídy kde sa používa Gaussián prenásobený sinusoidou (nemám šajn prečo...nevysvetľovalo sa to)

Otočenie θ je z množiny $\{0, 45, 90, 132\}$ a škál filtra se berie 9 $\sigma \in [0...8]$. Po tom ako sa kanály rozdelia sa model pozrie ako jednotlivé vlastnosti vyzerajú v centre a ako v okolí, kde centrum-okolie sa berie ako rozdiel medzi vrstvami Gaussových pyramíd. Centrum sa berie ako $c \in \{2, 3, 4\}$ a okolie $s = c + d, d \in \{3, 4\}$. Centrum a okolie sa spája tak, že sa jemenšia škála interpoluje na veľkosť hrubšej (neviem aký prístup sa používa na interpoláciu nebolo spomenuté) a následne sa mapy bodovo odčítajú. Táto operácia sa bude značiť $\ominus$.

1. Mapa intenzít: $I(c, s) = |I(c) \ominus I(s)|$
2. Mapy farieb: $RG(c, s) = |(R(c) - G(c)) \ominus (G(s) - R(s)|, BY(c, s) = |(B(c) - Y(c)) \ominus (Y(s) - B(s)|$
3. Mapy orientácií: $O(c, s, \theta) = |O(c, \theta) \ominus O(s, \theta)|$

Z toho vyjde celkovo 42 rôznych máp. Teraz máme problém, že každá z tých máp bude veľmi pravdepodobne inak naškákovaná a keď ich chceme spojiť bude ich treba najskôr znormalizovať. Postup normálizácie značenej operátorom $\mathcal{N}$:

1. Namapujem všetky hodnoty do intervalu $< 0, M >$ kde M je hodnota globálneho maxima
2. Potom spočítam m čo bude priemer lokálnych maxím (tam neberiem M)
3. Nakoniec danú mapu vynásobím $(M - m)^2$

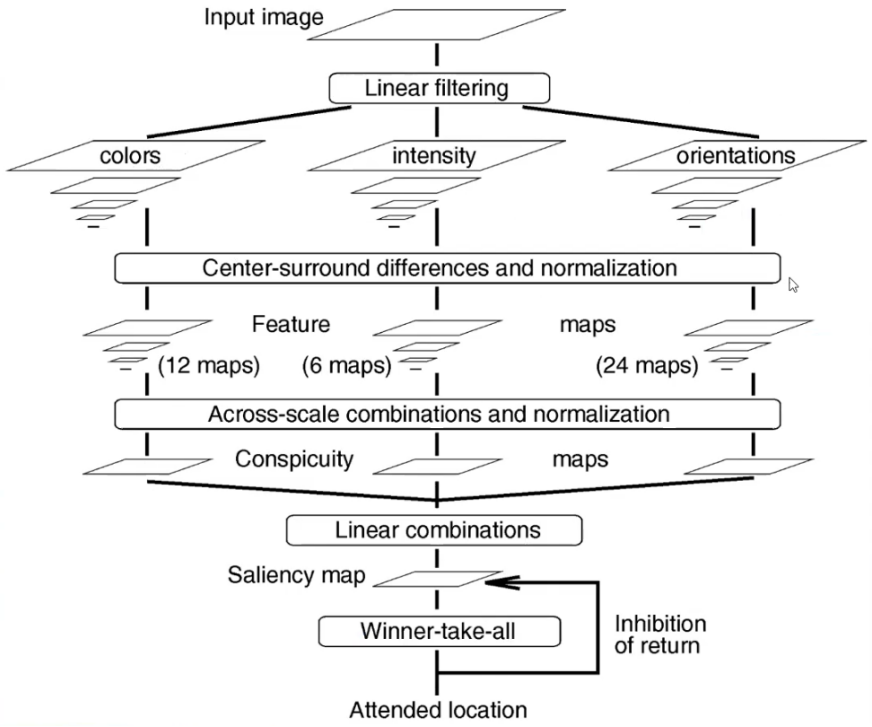
Následne sa mapy v každom kanále sčítajú, pričom ich treba opäť najskôr interpolovať a potom bodovo sčítať.

$$\overline{I} = \oplus_{c=2}^{4} \oplus_{s=c+3}^{c=4} \mathcal{N}(I(c,s))$$

$$\overline{C} = \oplus_{c=2}^{4} \oplus_{s=c+3}^{c=4} [\mathcal{N}(RG(c,s)) + \mathcal{N}(BY(c,s))]$$

$$\overline{O} = \sum_{\theta \in \{0,45,90,135\}} \mathcal{N}[\oplus_{c=2}^{4} \oplus_{s=c+3}^{c=4} \mathcal{N}(O(c,s,\theta))]$$

Posledný krok je už len lineárna kombinácia troch máp kde každá má rovnakú váhu. Nižšie je znázornený diagram modelu.



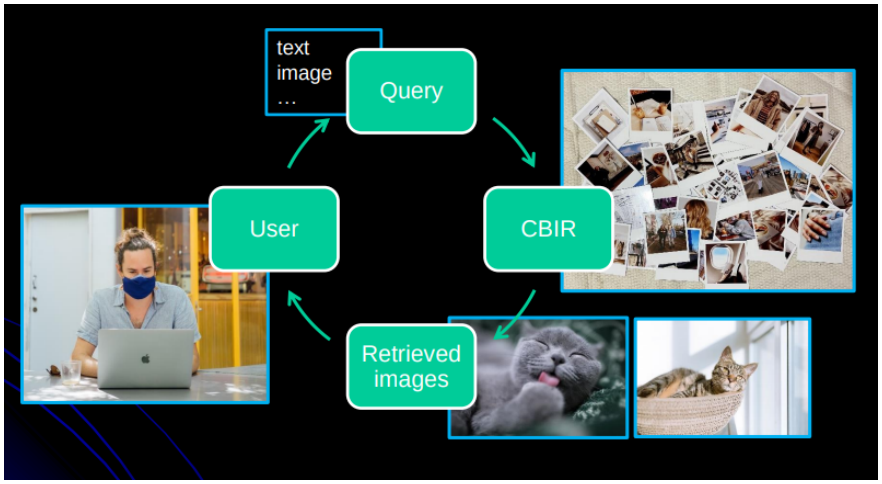
Aby bol model schopný simulovať ako sa človek pozerá na obrázok tak používa vyššie spomenuté nasýtenie pozornosti. Po nejakom časovom úseku sa najvýznamenjšia oblasť potlačí aby sa pozornosť mohla presunúť inde.

1.3 What is the navigational approach to selecting images from a database (CBIR)?

CBIR = content based image retrieval: hľadanie v databáze na základe obsahu obrázku

Na toto vyhľadávanie sa môžeme pozerať buď ako na jednokrokový alebo na viackrokový proces, kde máme užívateľa, ktorý chce vyhľadať nejaký obrázok, čiže môže zadať buď text, alebo zadá obrázok, ktorému sa má hľadaný obrázok podobať.

Tento vstup sa premení na nejaký dotaz do databázy, systém prehľadá databázu, na základe dotazu porovná, ktoré obrázky vyhovujú a vráti nejaké obrázky. ⇒ Tu by jednokrokový prístup končil, ale ak umožníme používateľovi reagovať na ponúknuté obrázky (povie, ktoré z nich sú v poriadku a ktoré z nich dotazu nezodpovedajú). Tejto väzbe sa hovorí relevance feedback ⇒ cyklus pokračuje upravením dotazu a novým vyhľadaním. Cyklus sa opakuje kým užívateľ nie je spokojný.



Historicky bolo prvým nástrojom zadanie textu, na základe ktorého sa vyhľadával obrázok. Nedostatky tohto prístupu: ako dobre vie uživateľ špecifikovať čo konkrétne hľadá (napr. flowers in a pond, water lilies, numphaea caerulea). Obrázok uložený v databáze nemusí byť asociovaný s týmito kľúčovými slovami. ⇒ Užívateľ nemá ako vedieť akými slovami je daný obrázok v databáze anotovaný.

Pri anotácii databázy a obrázkov narážame na limitácie:

1. Vyhľadávanie možné len v niektorých jazykoch: Väčšina databáz nie je anotovaná niektorými menej používanými jazykmi ⇒ väčšina databáz sa však dá dobre prehľadávať pomocou angličtiny

2. Ľudská subjektivita a veľký objem databáz: v prípade, že má anotáciu robiť človek, nie je dosiahnuteľné aby zvládol anotovať dnešné množstvá obrázkov, na to by bolo treba veľmi veľa ľudí. Navyše, ľudská anotácia je veľmi subjektívna ⇒ každý anotátor môže obrázok nazvať inak.

3. Nejasnosť reprezentácie abstraktných termínov

Miesto voľného anotovania sa používa anotácia s nejakým obmedzeným slovníkom, ktorý vychádza z ontológií - postupného zjemňovania daného výrazu (Society, civilization - industries - machines - machine driven by wind - windmill)

Ďalšie možnosti vyhľadávania obrázkov:

1. vyhľadávanie na základe textu asociovaného s obrázkami ako je to napríklad pri webstránkach, kde obrázok môže byť nájdený na základe slov obsiahnutých spolu s ním na stránke. Ďalším príkladom je možnosť anotácie fotiek na disku.

Ďalší prístup: Vieme na obrázku analyzovať objekty, segmentovať ich a vybrať, ktorá časť obrázku znázorňuje čo. Na základe týchto objektov vieme určiť aj ich vzájomný súvis, postavenie, atď. Na základe takejto analýzy obrázku vieme určiť nejaké kľúčové slová. Ak máme anotované obrázky, môžeme začať hľadať obrázky, ktoré sú podobné dotazu zadanému od užívateľa.

Podobnosť obrázkov je určená podľa:

1. farby

- systém môže užívateľovi dovoliť zadať hlavnú farbu obrázku, farby s daným percentuálnym zastúpením, vyberať si ich môže z farebnej palety

2. textúry
   - chceme aby obrazy s rovnakou textúrou, ktoré ale nemusia byť rovnako farebné boli zobrazené ako podobné (používajú sa vektory popisujúce textúru)
   - $d_{pick\ and\ click}(D,Q) = min_{i \in D} \|T(i) - T(Q)\|^2$
3. tvaru
4. vzájomnej sémantickej súvislosti v obraze
5. zvolenej metriky na skúmanie podobnosti

Podobnosť je matematicky daná podobnosťou medzi extrahovanými príznakmi:

$$S_Q, D = s(F_Q, F_D) = g(d(F_Q, F_D))$$

S - miera podobnosti F - príznaky extrahované z daných 2 obrázkov d - metrika g - váha (monotonická nerastúca funkcia)

1.4 How does object search using local features work? Describe the steps involved. Describe the selected detector and descriptor.

Na prednáške sme si hovorili najskôr, že môžeme detekovať obrázky pomocou šablóny. To je strašne moc shitty a zle to funguje. Ak je objekt čiastočne prekrytý tak dostaneme o dosť menšiu odozvu ako by sme chceli. Takisto ak objekt v obrázku prejde nejakou transformáciou ako napríklad škálovanie, skosenie tak to tiež nefunguje. Lokálne príznaky sú o niečo lepšie. Všeobecný postup pri detekcii objektov je takýto:

1. Detekcia: nájdenie zaujímavých bodov teda bod v ktorého okolí sa niečo deje
2. Deskripcia: vytvorenie vektora, ktorý popisuje okolie zaujímavého bodu, toto budeme chcieť aplikovať na všetky významné body ktoré sme v obrázku našli
3. Párovanie: hľadanie korešpondencie medzi príznakmi

1.4.1 Detekcia

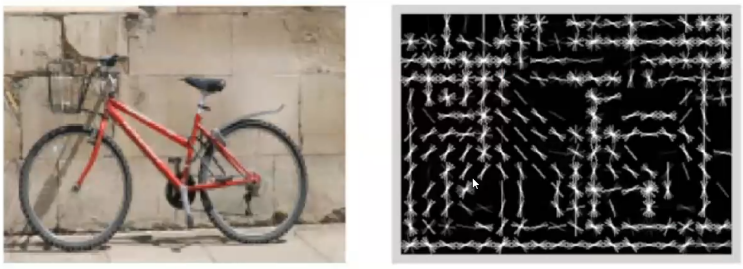
V tejto úlohe môžeme postupovať naivne a vybrať: všetky body, náhodné body alebo rovnomerne rozmiestnené body. To môže fungovať v prípade ak mám nejaké homogénne obrázky ale všeobecne to nie je moc dobre aplikovateľné. Lepší prístup je použiť detektory: Harris, SUSAN, SIFT, SURF, FAST. Na to aby boli lokálne príznaky na niečo užitočné musím vedieť identifikovať rovnaké oblasti aj po aplikácii transformácii. Pri transformáciách uvažujeme geometrické (mení sa tvar objektu, otočenie škálovanie) alebo fotometrické (mení sa intenzita a podobne).

1.4.2 Deskripcia

Môžem zobrať intenzity ktoré sú v okolí zaujímavých bodov ale to nie je moc dobrý prístup, pretože zmena intenzity robí neporiadok. Deskriptory ktoré môžeme použiť: HOG, SIFT, SURF, BRIEF.

1. Histogram of Oriented Gradients: pre každý bod spočíta gradient a potom pre bunky 8x8 vytvorí histogram gradientov. Tie následne normalizuje po blokoch 16x16 a z toho bude výsledný príznak. Je najčastejšie používaný.
2. SIFT používa trochu upravey HOG deskriptor v ktorom aplikuje na okolie bodu gausovské váhovanie

3. SURF počíta Haar wavelet
4. BRIEF je binárny deskriptor, ktorý porovnáva intenzitu dvojíc pixelov v okolí zaujímavých bodov.

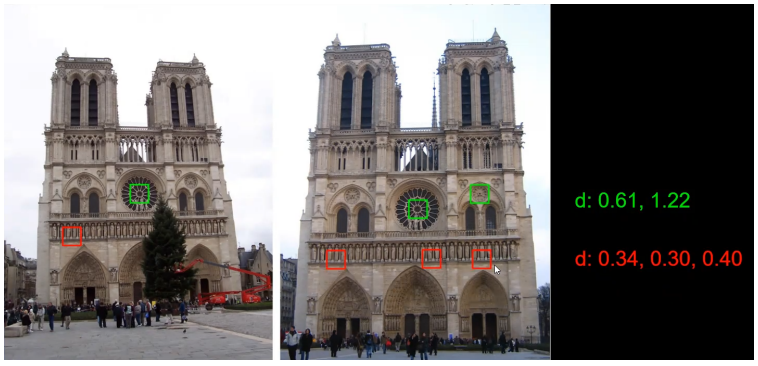


1.4.3 Párovanie

Máme rovnaké body v nejakých 2 rôznych obrazoch, môžeme si predstaviť, že jeden bod prešiel nejakou transformáciou. V obidvoch nájdeme významné body a chceme sa pozrieť či popisujú rovnaké oblasti scény. Pozrieme sa na deskriptory a následne zistíme či sa tieto deskriptory rovnajú, to spravíme pre každú dvojicu. Na porovnanie môžeme použiť rôzne metriky:

1. Suma absolútnych rozdielov: $\sum_i |p_i - q_i|$
2. Suma štvorcov rozdielov: $\sum_i (p_i - q_i)^2$
3. Kosínusová metrika: $\cos p, q = \frac{pq}{|p||d|}$ ak dva vektory zvierajú malý uhol môžeme z toho usúdiť, že sa podobajú. Je to podobnostná metrika na [0, 1] a chceme aby bola hodnota čo najväčšia

Body môžeme párovať naivne tak, že sprárujeme dva najbližšie body ak vzdialenosť je menšia ako nami stanovený prah. Tam je problém, že môžem mať v obrázku oblasti ktoré sa na seba veľmi podobajú, to viem určiť tak že sa pozriem na prvý najbližší bod a druhý najbližší bod. Aj je pre druhý najbližší veľmi podobná vzdialenosť tak to nie je dobré. Takéto body mi v analýze obrázku moc nepomôžu preto ich treba vylúčiť. Porovnávam vzťah $\frac{|A-B|}{|A-C|}$ kde B je najbližší a C je druhý najbližší.



1.5 Describe the Lucas-Kanade method and its iterative refinement to compute the optical flow.

1.5.1 Opticky tok z otazky 2.12

Pohyb oblastí s rovnakou intenzitou v obraze. Nemusí nutne zodpovedať reálnemu 3D pohybuje ale je to to najlepšie čo vieme vytiahnuť z obrázkov. Predpoklady sú, že intenzita sledovaného bodu sa nemení. Bod sa nepohne príliš ďaleko respektíve máme vhodné fps. a bod sa bude sa bude pohybovať rovnako ako jeho

okolie. Za týchto predpokladov je v malom časovom okne obrazová funkcia priemetu bodu konštantná. Teda derivácia podľa času bude nula:

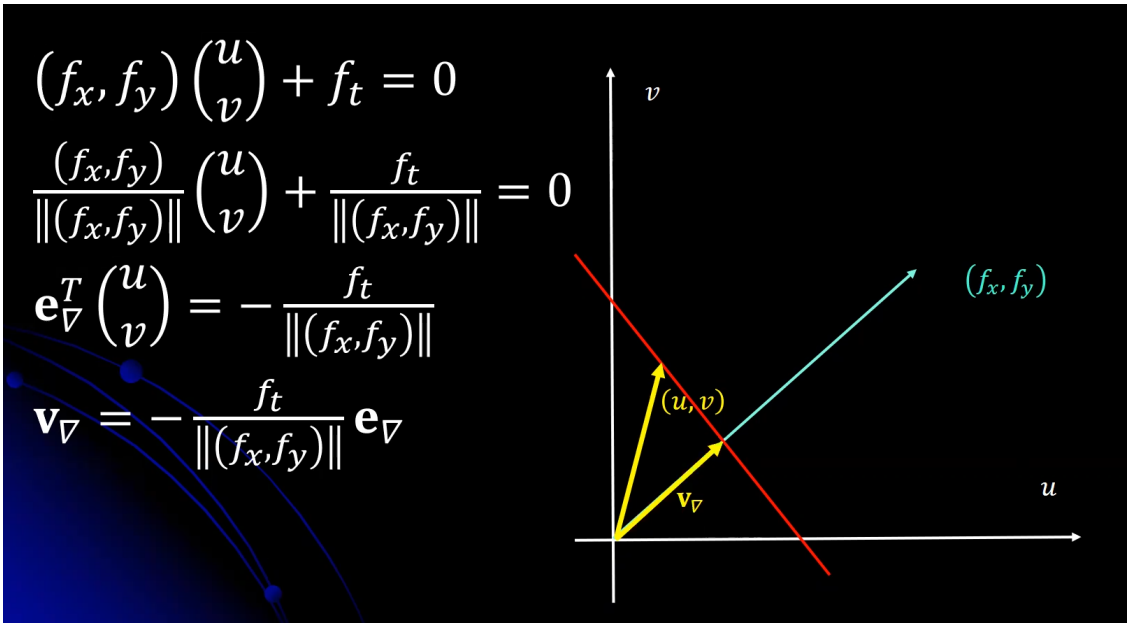
$$\frac{\partial f(x(t), y(t), t)}{\partial t} = 0$$

$$\frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial t} = 0$$

1. Gradient: ∇f
2. Optický tok: $\begin{pmatrix} u \\ v \end{pmatrix} = \left(\frac{dx}{dt}\ \frac{dy}{dt}\right)^T$
3. Časová derivácia: $\frac{\partial f}{\partial t}$

Z toho máme jednu rovnicu s 2 neznámami ktoré chceme vypočítať. Presné riešenie nebudem vedieť nájsť ale viem, že riešenie bude ležať na priamke.

$$\nabla f^T \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\partial f}{\partial t} = 0$$



Zložku v kolmom smere na gradient nepoznáme čo je vlastne problém clony. Ak by som vedel určiť ako sa pohne jeden konkrétny bod tak budem vedieť vypočítať smer skutočného pohybu. Na tom už je založený prístup ktorý sa používa v metóde Lucas-Kanade.

1.5.2 Rozšíření Lucas-Kanade

Sleduje kromě jednoho bodu i jeho okolí (např. 3x3) a potom máme rovnici pro každý bod z okolí, nejenom pro bod, jehož pohyb se snažíme zjistit. Tímpádem máme soustavu 9 rovnic na zjištění 2 neznámých.

$$\begin{pmatrix} f_x(P_1) & f_y(P_1) \\ f_x(P_2) & f_y(P_2) \\ \dots & \dots \\ f_x(P_9) & f_y(P_9) \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} -f_t(P_1) \\ -f_t(P_2) \\ \dots \\ -f_t(P_9) \end{pmatrix}$$

Matici fx a fy si označíme jako A, $\begin{pmatrix} u \\ v \end{pmatrix}$ je hledaný optický tok, který označíme jako d, a matici -ft si označíme jako b. Potom $Ad = b$. Metodou nejmenších čtverců dostaneme $A^T Ad = A^T b$, což jde rozepsat jako:

$$\begin{pmatrix} \sum f_x f_x & \sum f_x f_y \\ \sum f_x f_y & \sum f_y f_y \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = - \begin{pmatrix} \sum f_x f_t \\ \sum f_y f_t \end{pmatrix}$$

Tato rovnice je vyřešitelná pokud vlastní čísla červeně zvýrazněné matice jsou kladná, dostatečně velká a přibližně stejně velká. (V tom případě je matice invertovatelná). Také je potřeba aby v obraze nebyl velký šum.

Stejnou podmínku má Harris corner detector na detekci rohů, tedy Lucas-Kanade nám říká, že pokud dokážeme najít v okolí bodu roh, dokážeme vypočítat jaký pohyb vykonal daný bod (aneb že rohy se dobře sledují).

1.5.3 Iterativní upřesňování

Nutné pokud je porušen jeden z předpokladů

- Předpoklad konstantní intenzity
- Předpoklad společného pohybu pixelů v okně
- Předpoklad malého pohybu (jen o jeden pixel)

Algoritmus iterativního upřesňování

1. Vypočítej optický tok pomocí Lucas-Kanade
2. Warpujeme f(x, y, t) (= aplikujeme na obrázek v čase transformace) podle odhadnutého toku
3. porovnáme s příštím snímkem, opakujeme dokud chyba mezi časovými kroky není akceptovatelná. Dokud f(x, y, t) cca= f(x, y, t+1)

Pomůže po warpování obraz vyhladit, a vzároveň pro urychlení můžeme používat derivace obrazu před warpováním.

Při pohybu větším než o jeden pixel řešíme pomocí samplování pomocí gaussovské pyramidy. Pohyb najdu na nejnižší úrovni, warpuju, transformuju a na vyšší úrovni je díky warpování už pohyb menší než původně.

Všechny rovnice výše počítají s pohybem po přímce, pro komplikovanější pohyby by rovnice vypadaly jinak.

1.6 Describe the Kalman filter.(SNAD HOTOVO, ALE CONFUSING TO JE POŘÁD

Kalmanův model slouží k přepovídání pohybu už nějakou dobu sledovaného objektu, abychom ho nemuseli znovu a znovu hledat v celém obrázku, pričom predpokladám, že pohyb bude lineárny.

Teda snažím sa naučiť parametre a, b pre $y = ax + b$, pomocou sekvenčnej aktualizácie parametrov. To znamená, že predikcie modelu porovnávam so zlatými dátami a aktualizujem parametre až kým nebudem spokojný s výsledkom. Celý model je založený na cyklu predikce budoucího stavu a porovnávání predikce s actually příštím snímkem.

Kalmanov filter si uchováva informácie o stave objektu x_k v čase k. Medzi ne patrí napríklad pozícia rýchlosť alebo zrýchlenie.

Zlaté dáta alebo niečo čo viem zmerať z videa označím z_k. Okrem toho treba zahrnúť ešte procesný šum w_k a merací šum v_k. Predpokladá sa, že oba šumy sú z normálneho rozdelenia so strednou hodnotou v 0, a svojí vlastní kovarianční maticí. Objekt ešte môže ovplyvňovať nejaká externá vlastnosť u_k ktorá vstupuje do systému. Napríklad pri voľnom páde gravitačné zrýchlenie. Ten to parameter je optional.

Stav v čase k je daný pomocí:

$$x_k = Ax_{k-1} + B_k u_k + w_k$$

Kde A je přechodová matice mezi stavem $k-1$ a stavem k, $B_k u_k$ je konstantní vstup co je optional, a w_k je šum.

Změřená pozice v čase k je daná pomocí:

$$z_k = Hx_k + v_k$$

Kde H je matice měření, což znamená že z vektoru x_k, který má veškeré informace o např. rychlosti a pozici si vytáhne pouze informace o pozici, protože rychlost nemůže změřit. v_k je šum.

Já se k x_k nedostanu, můžu se dostat jenom k z_k, a x_k musím aproximovat. Funguje to teda podobne ako skryté Markovské modely.

Predikcie modelu:

- Predikcia stavu: $\overline{x_k^-} = A\overline{x_{k-1}}(+Bu_k)$
  - značení s mínusem znamená před korekcí
- Kovariancia chyby predikcie stavu: $P_k^- = AP_{k-1}A^T + Q$
  - Při výpočtu zmizí ten konstantní optional šum
  - Výpočet kovariance chyby predikce stavu v čase k můžu vypočítat z kovariance chyby predikce stavu v čase $k-1$
- $P_{k+1^-} = \mathbf{E}[(x_{k+1} - \overline{x_{k+1^-}})(x_{k+1} - \overline{x_{k+1^-}})^T]$
  - Za předpokladu že bych znala x_{k+1}

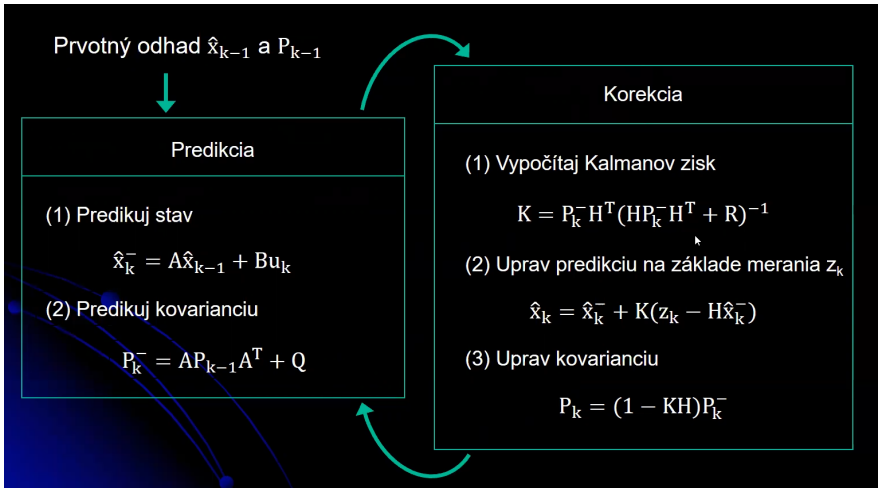
Po tom ako model spraví nejakú predikciu pohybu objektu sa ešte vykonáva korekcia parametrov:

- Korekcia na základe merania:
  - $z_k = Hx_k + v_k$

– $\overline{x_k} = \overline{x_k^-} + K(z_k - H\overline{x_k})$

- Korekcia kovariancie chyby: $P_k = (I - KH)\overline{P_k}$ kde K je Kalmanov zisk.

Kalmanův zisk určuje jakou váhu bude mít predikovaná vs změřená hodnota - které se bude přikládat větší důležitost. Jde vypočítat ze složení kovariančních matic chyby. Potom zvolíme K aby minimalizovalo stopu matice P_k (Matice korekce kovariance chyby)



2 Part 2

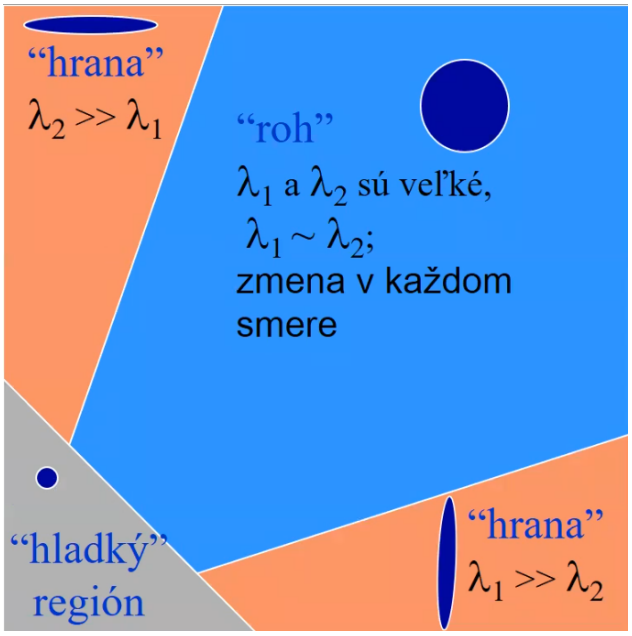
2.1 Describe the Harris corner detector. Against which transformations is it, and is it not invariant? Why?

Morovcov detetor počíta hodnotu sebepodobnosti v 8 smeroch posunutia(vektor u):

$$E(u) = \sum_i w(p_i)[I(p_i + u) - I(p_i)]^2$$

Rohovitosť bodu je daná najmenšou hodnotou E. Harrisov detektor prináša zlepšenie Moravcovho detektora s tým, že sa vie pozrieť do ľubovoľného smeru pomocou Taylorového rozvoja. Potom môžeme hodnotu sebepodobnosti vypočítať ako uAu^T. Postup detektoru:

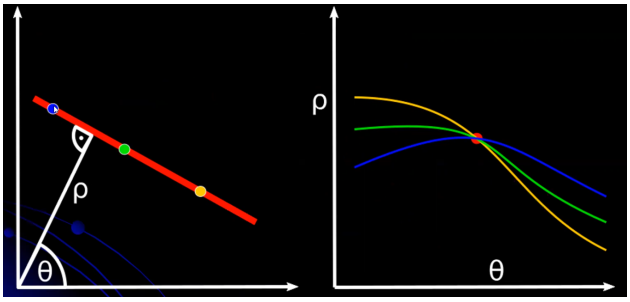
1. Spočítam prve derivacie z I v x aj y smere
2. Aplikujem Gaussov filter na maticu vytvorenu z derivacii I (beriem gausov filter ako váhovú funkciu w) takze dostanem $A = \sum_i w(p_i) \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$
3. Viem detekovať rozdiel v oboch smeroch potrebujem určiť kedy som našiel roh (viď obrázok). Pre každý bod obrazu spočítam funkciu odozvy $R = det(a) - \alpha trace(A)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$. Okrem toho sa používajú aj iné odozvy napríklad najmenšie vlastné číslo.
4. R treba odprahovať a následne vybrať lokálne maximá aby som potlačil nemaximálne body



Tento detektor je invariantný voči zmene intenzity obrázku pretože počítam deriváciu a pričítanie konštanty zachováva výsledok. Tak detektor neovplyvní otočenie obrázku pretože používame na výpočet vlstné čísla a to nejako funguje idk.

2.2 Describe the Hough transform for lines of arbitrary direction.

Používa sa na detekciu objektov, ktoré sa dajú jednoducho analyticky vyjadriť, väčšinou krivky ale priamky. Výhodou je, že funguje dobre aj pri aj ak je v obrázku veľa šumu, pretože viem jednoducho dopočítať priebeh funkcie. Používajú sa polárne súradnice $\rho = x\cos\theta + y\sin\theta$ kvôli tomu aby som vedel zapísať aj zvislú priamku. Každú priamku viem reprezentovať dvojicou (ρ, θ) v parametrickom priestore. Ak zafixujem body x, y a mením ρ, θ dostanem krivku. Ak vyberiem body na nejakej priamke tak ich krivky s parametrickom priestore sa budú pretínať a bod v ktorom sa pretínajú udáva parametrické vyjadrenie (ρ, θ) hľadanej priamky. Keď sa mi podarí takéto body identifikovať odprahujem ich podľa toho koľko kriviek sa tam pretína, tým identifikujem najviac vyrazné priamky v obrázku.



2.3 Describe the SIFT (detector and descriptor).

Najznámejším detektorom, ktorý využíva škálový priestor je detektor SIFT, ktorý hľadá zaujímavé body v DoG (Difference of Gaussians) priestore.

Vezme obrázok a rozostrí si ho rôznymi Gausiánmi. Následne obrázok preškáluje(zmenší), a opäť ho rozostrí Gausiánmi. Toto robí pre rôzne veľkosti obrázku - v podstate vyrába Gaussovskú pyramídu a

každú úroveň pyramídy ešte rozostrí rôznymi Gausiánmi aby potom odčítaním Gausiánom vedel detekovať zaujímavé body.

Jedna úroveň Gaussovskej pyramídy sa v SIFTe nazýva oktáva.

Obraz rozostrený gaussiánom = J

SIFT pracuje tak, že sa pozrie na 26 okolie v priestore a hľadá extrém v odozve. To znamená, že v rozostrení sa pozrie o úroveň pod a nad a nájde hľadaný bod, čím určí pozíciu maxima/minima a určí v akej škále bola nájdená, čiže v akej oktáve sme ju našli.

V prípade, že už máme škálu a pozíciu určenú, tak vieme určiť aj orientáciu = smer, v ktorom sa intenzita najviac mení:

$$\Theta(x,y) = \arctan \frac{J(x,y+1) - J(x,y-1)}{J(x+1,y) - J(x-1,y)}$$

Takto vzniknú body, ktoré sú kandidátmi na zaujímavé body. Tie však ešte musíme prečistiť.

1. odstránime málo kontrastné body - prahujeme hodnotu D v nájdených extrémoch
2. odstránime body pozdĺž hrán
   - Hesseho matica $H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$
   - $det(H) = \lambda_1 \lambda_2$
   - $trace(H) = \lambda_1 + \lambda_2$
   - $\frac{trace(H)^2}{det(H)} = \frac{(\lambda_1+\lambda_2)^2}{\lambda_1\lambda_2} = \frac{(r+1)^2}{r}$

Ako deskriptor použijeme HOG prístup na okno 16x16 okolo každého pixelu, ktoré potom skombinujeme na okno 4x4 s HOGmi a spravime vahovany priemer gaussom cez tieto histogramy a mame HOG pre pixel v strede.

2.4 Describe the SURF (detector and descriptor).

SURF aproximuje LoG priestor aproximáciou Laplaciánu. Filtruje obrázok rôznymi veľkosťami filtrov.

Filter: aproximácia druhých parciálnych derivácií Gausiánu

Hesseho matica $H^{SURF}_{approx} = \begin{bmatrix} \hat{L}_{xx} & \hat{L}_{xy} \\ \hat{L}_{xy} & \hat{L}_{yy} \end{bmatrix}$

Integrálny obraz pre nejaký bod je suma intenzít bodov vľavo hore of daného bodu:

$$IO(x,y) = \sum_{i=1}^{x} \sum_{j=1}^{y} I(i,j)$$

Suma intenzít v obdĺžnikovom okne (A,B,C,D) pôvodného obrazu:

$$IO(A) - IO(B) - IO(C) + IO(D)$$

SURF využíva vlastnosť vyššie spomenutých integrálnych obrazov vďaka čomu je veľmi efektívny.

Deskriptor pouziva Haar wavelet. (?)

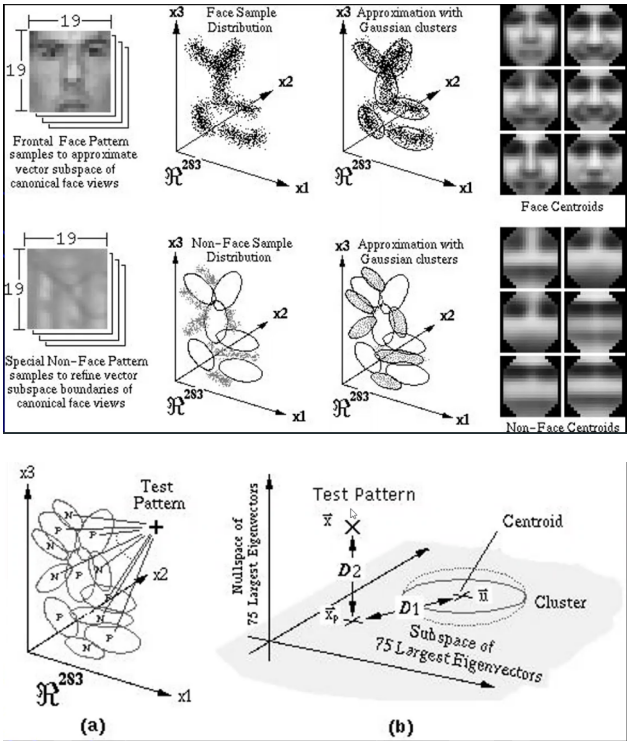
2.5 Describe the procedure of face detection using Gaussians. How can we use boosting in it?

Množina obrázkov musí mať rovnaký rozmer, v nej budú obrázky tvárí a hocičoho iného. Potom sa zvykne obrázok orezať tak aby sa nebrali do úvahy rohy. Intenzity daného obrázku sa zoradia za seba a to beriem

ako vektor príznakov. V tomto priestore sa snaží robiť n zhlukov pre tváre a n zhlukov pre zvyšné obrázky. Štandardne sa k tomu robí augmentácia ale aj korekcia osvetlenia a ekvalizácia histogramu. Osvetlenie sa odstráni tak, že identifikujem gradient a odčítam ho od pôvodného obrazu. Ekvalizácia len zvýši kontrast medzi jednotlivými črtami tváre. Na klasifikáciu obrázku potrebujem spočítať nasledovné vzdialenosti:

1. **Vzdialenosť v pod-priestore D1** teda vzdialenosť vzorky od stredov zhlukov v redukovanom priestore. Zvyčajne sa na to použije ešte PCA.
2. **Vzdialenosť k pod-priestoru D2**, ktorá počíta vzdialenosť pôvodného bodu a budu po transformácii PCA.

Potom sa tieto vzdialenosti dajú do neurónovej siete zamáva sa rukami a povieme si aký máme super model. Boosting v tomto kontexte funguje podobne ako v rozhodovacích stromoch. Dám do siete tváre, pozriem sa na to ktoré sú nesprávne klasifikované a tie pridám do trénovacej množiny.



2.6 Describe the methods of tracking eye movements and analyzing the data obtained.

Pri analýze oka nás zaujímajú najviac:

1. Fixácie: obdobie relatívnej stability, počas ktorého vnímame objekty
2. Kmity/Sakády: krátke pohyby 40ms - 120ms

Metódy sledovania oka sú:

1. **Elektronické** Používa sa nejako v medicíne nie je to presné
2. **Mechanické** Presné ale nasty. Človek má kontaktné šošvky v ktorých je cievka potom sa dá niekam do magnetického poľa a tak sa dá dobre odmerať ako sa mu hýbe oko.

3. **Videometódy**

   (a) Jednobodové: sleduje sa buď zrenička alebo dúhovka. Software analyzuje video a sleduje vybraný objekt

   (b) Dvojbodové zariadenie vysiela IR svetlo as sleduje k tomu aj odraz na rohovke. Poloha odrazu voči zreničke tak určuje smer pohľadu a vieme zistiť kam sa človek pozeral. Najskôr to treba kalibrovať.

   Analyzovanie záznamu:

   (a) I-VT: vypočíta rýchlosti medzi jednotlivými zaznamenanými bodmi. Potom sa určí prah a ak je rýchlosť ¡ prah tak sa označí bod ako fixácia oka. Ak je viacero za sebou idúcich fixácii tak ich treba zoskupiť do jednej.

   (b) I-DT: berie do úvahy, že oko sa aj v rámci fixácie môže trochu hýbať. Je tu na to nejaký odmávaný algoritmus, ktorý funguje podobne ako predošlý akurát má dva zvolené prahy.

2.7 Describe the possibilities of visualizing eyetracking data and methods of comparing them.

Zaznamenám pozície fixácii a spravím na to heatmap. Fixáciu môžem znázorniť ako "gaussovské kopčeky" ktoré potom sčítam. Z toho je vidno kam sa človek najviac pozeral, to sa dá použiť dobre na príklad v marketingu. Ak budeme nad heatmapou uvažovať ako nad rozdelením pravdepodobnosti, môžeme dve heatmapy medzi sebou porovnať pomocou KL divergenciu (zo straky je to relatívna entropia): $\sum_{x \in X} f(x) \log \frac{f(x)}{g(x)}$. Okrem toho sa dá sledovať aj trajektória pohľadu fixáciami. Na bežný eyetraking treba hardware a hlavne veľa ľudí. Dnes už sa častejšie používajú výpočtové modely ktoré odhadujú kam sa ľudia budú pozerať. Model sa natrénuje na výsledkoch iných štúdii s reálnymi ľudmi a potom snaží niečo rozumné predikovať. Známy model je Ittyho model.

2.8 Describe color image quantization. Types of palettes, their creation, quantization error

Euklidovská vzdialenosť v farebnom priestore nezodpovedá vizuálnej vzdialenosti, čo má za následok, že sú množiny farieb ktoré, sú okom prakticky nerozlíšiteľné. Kvantizácia má za následok kódovať podobné farby rovnako za účelom kompresie obrázku. Postup kvantizácie pri farebnom obrázku, je nasledovný. Najskôr treba rozdeliť 3D priestor to požadovaného počtu skupín a potom naň treba namapovať pôvodné farby obrázku.

Typy paliet

1. Pevná paleta: pravidelné rozdelenie RGB kocky.
2. Adaptívna paleta: každému obrázku je pridaná jedinečná paleta. Horšie na porovnávanie 2 rôznych obrázkov

Algoritmus median cut funguje na myšlienke, že reprezentatívne farby by mali zastupovať približne rovnaký počet pôvodných farieb. Výpočet:

1. Nájdem najmenší obal ktorý obsahuje všetky farby
2. Zoradím farby podľa najdlhšej osi (osi sú RGB v 3D priestore)
3. Rozdelím obal v bode mediánu
4. Opakujem až pokým nemám požadovaný počet množín

Kvalitu aproximácie viem zmerať chybou kvantovania, ktorá sa počíta ako: $\frac{1}{n}\sum_i d^2(x_i, Q(x_i)$ kde Q je kvanizačná funkcia a ako vzdialenosť sa zväčša používa euklidsovká. Ak chcem mať čo najlepšiu aproximáciu mal by som spĺňať podmienky optimality. Reprezentatívna farba je tá ktorá je k pôvodnej farbe najbližšia a reprezentatívna farba je centroidom farieb, kltoré repreznertuje.

2.9 Describe the Moravec and Harris corner detectors.

Roh je taká časť obrázka, že ak by sme cez ňu preložili nejaké malé okienko, tak posun toho okienka v ľubovoľnom smere zmení obsah okienka. Hľadáme body s nízkou sebepodobnosťou.

Moravcov detektor roho funguje tak, že posunie okienko do každého z 8 smerov a spočíta $E_{WSSD}(u) = \sum_i w(p_i)(I(p_i+u)-I(p_i))^2$, kde u je vektor posunu a $w(p)$ je váhová funkcia, môžeme použiť napr. funkciu, ktorá je 1 ak je bod v okienku a inak 0, alebo Gaussovu funkciu, ktorá dáva bodom na okraji okienka nižšie váhy. Rohovitosť je daná ako najmenšia hodnota E medzi oknami určenými skúmaným bodom a jeho 8 susedmi. (WSSD = weighted sum of squares of distances). Na záver ešte musíme nájsť vhodný prah pre E. Nevýhodou tohoto detektoru je, že nie je invariantný voči rotáciam obrazu, keďže skúma len 8 smerov.

Vylepšenie je Harrisov detektor, ktorý využíva Taylorov rozvoj. $E_{WSSD}(u) \approx \sum_i w(p_i)(\nabla I(p_i)^T u)^2 = u^T A u$, pričom A vyopčítam ako vážený priemer pomocou gaussovej funkcie (gaussov filter).

$$A = \sum_i w(p_i) * \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix}$$

Pomer vlastných čísel matice A nám určuje či sa tam nachádza hrana alebo roh. Ak sú obidve vlastné čísla malé, tak je tam hladký región. Použijeme response funkciu (napr. $R = det(a) - \alpha trace(A)^2$). Potom R odprahujeme a nájdeme lokálne maximá ktoré určujú rohy.

2.10 Describe object search using template matching

Porovnávanie so vzorom sa používa na lokalizáciu známych vzorov v obraze. Najlepšia zhoda je založená na kritériu optimality, ktoré je vo veľkej časti prípadov záleží na objekte, ktorý chceme rozpoznať. Potom párovanie prebieha tak, že určujem kritériu zhody pre všetky možné polohy a otočenia vzoru v obraze. Nie je to dobré, neviem načo sa to učíme...nezmyselne dlho to trvá. Ako miera podobnosti sa používa zväčša: korelácia, korelácia s nulovým priemerom, suma štvorcov vzdialeností, normalizovaná korelácia. Samotná korelácia sa nedá dobre prahovať, zvyšné podmienky fungujú o niečo lepšie.

1. Korelácia: rýchla ale nevhodná: $h(m,n)\sum_{k,l} g(k,l)f(m+k,n+l)$
2. Korelácia s nulovým priemerom: stále rýchla ale detekuje aj náhodné výskyty
3. Suma vzdialeností: menej rýchla a citlivá na zmenu intenzity
4. Normalizovaná korelácia: pomalá je invariantá ku kontrastu a intenzite

2.11 Describe the possibilities of visualizing eyetracking data and comparing them. Describe a selected method for finding salient regions in an image

Na dátach z eye trackingu môžeme vizualizovať:

1. Fixačné body: Zobrazujú miesta, na ktoré sa subjekt pozerá dlhšiu dobu. Tieto body sa zvyčajne zobrazujú ako malé kruhy, pričom veľkosť kruhu môže reprezentovať dĺžku fixácie.
2. Trasy pohľadu: Tieto vizualizácie ukazujú sled pohľadu subjektu v čase. Môžu byť zobrazené ako krivky alebo lomené čiary, ktoré spájajú jednotlivé fixácie.

3. : Heatmapy: zobrazujú oblasti, na ktoré sa subjekty pozerajú najviac. Využívajú farbu a intenzitu, pričom teplejšie farby (napríklad červená) reprezentujú oblasti s väčšou koncentráciou pohľadov a chladnejšie farby (napríklad modrá) reprezentujú menej pozorované oblasti.

Významné salietné oblasti budú tie na ktoré sa sústredilo najviac ľudí. Porovnať vizualizácie môžeme tak, že ich jednoducho prekryjeme (to funguje napr. s heatmapami) ale ešte sa dá použiť KL-Divergencia.

2.12 Describe the calculation of optical flow.

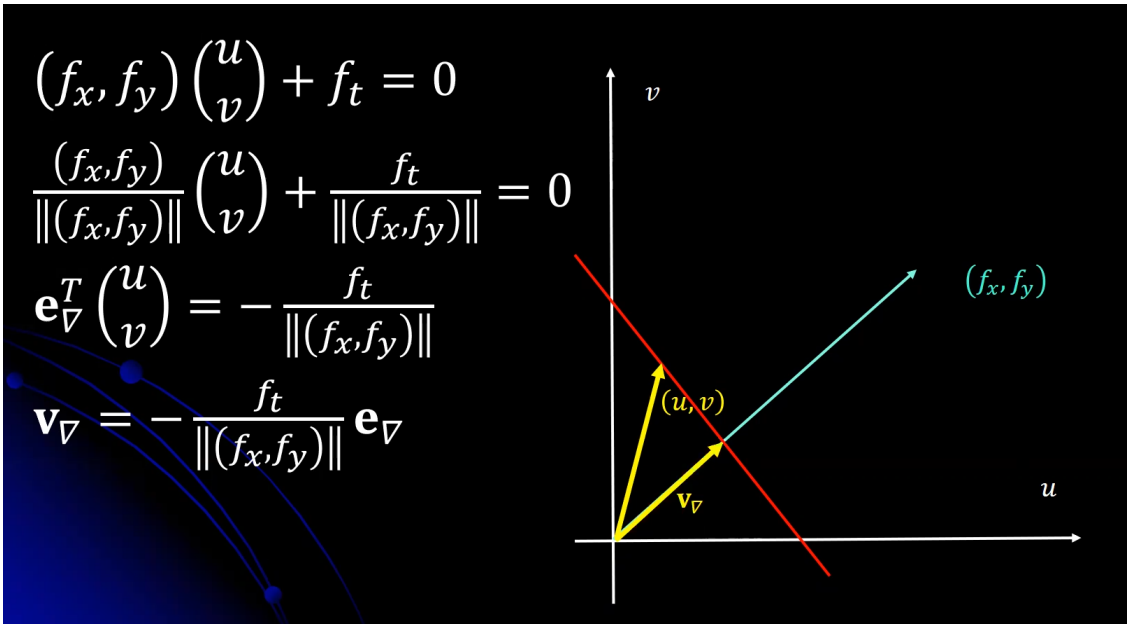
Pohyb oblastí s rovnakou intenzitou v obraze. Nemusí nutne zodpovedať reálnemu 3D pohybuje ale je to to najlepšie čo vieme vytiahnuť z obrázkov. Predpoklady sú, že intenzita sledovaného bodu sa nemení. Bod sa nepohne príliš ďaleko respektíve máme vhodné fps. a bod sa bude sa bude pohybovať rovnako ako jeho okolie. Za týchto predpokladov je v malom časovom okne obrazová funkcia priemetu bodu konštantná. Teda derivácia podľa času bude nula:

$$\frac{\partial f(x(t), y(t), t)}{\partial t} = 0$$
$$\frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial t} = 0$$

1. Gradient: ∇f
2. Optický tok: $\begin{pmatrix} u \\ v \end{pmatrix} = \left(\frac{dx}{dt}\frac{dy}{dt}\right)^T$
3. Časová derivácia: $\frac{\partial f}{\partial t}$

Z toho máme jednu rovnicu s 2 neznámami ktoré chceme vypočítať. Presné riešenie nebudem vedieť nájsť ale viem, že riešenie bude ležať na priamke.

$$\nabla f^T \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\partial f}{\partial t} = 0$$



Zložku v kolmom smere na gradient nepoznáme čo je vlastne problém clony. Ak by som vedel určiť ako sa pohne jeden konkrétny bod tak budem vedieť vypočítať smer skutočného pohybu. Na tom už je založený prístup ktorý sa používa v metóde Lucas-Kanade.

→ využívá se pro clustrování z balíčku n hodnot ve k.

2.13 Describe the K-means method. What is it used for?

K-means klastrovanie je nehierarchická metóda, ktorá vytvára k klastrov. Optimalita klastrov je daná minimálnym intra-cluster rozptylom:

$$W = \sum_{k=1}^{K} \sum_{x_i \in C_k} \sum_{x_j \in C_k} \|x_i - x_j\|^2 = \sum_{k=1}^{K} 2N_k \sum_{x_i \in C_k} \|x_i - m_k\|^2 = \sum_{k=1}^{K} WSS_k$$

kde m_k je centroid klastru k, N_k je množstvo bodov v klastri k.

2.13.1 K-Means Algoritmus:

Inicializácia:
Náhodne inicialnizujeme do priestoru K bodov, ktoré budú reprezentovať objekty, ktoré sú klastrované. Tieto body reprezentujú počiatočné centroidy skupín.

Pokračovanie algoritmu:
Priradenie objektu do skupiny s najbližším centroidom:

$$C(x) = \underset{k}{\operatorname{argmin}} \|x - m_k\|^2$$

Po priradení všetkých objektov sa prepočítajú pozície K centroidov:

$$m_k = \frac{\sum_{x:C(x)=k} x}{N_k}, k = 1, ...K$$

Tento krok opakujeme pokým algoritmus nebude zastavený. Zastaví sa, pokiaľ je MSE menej než nejaký threshold, alebo ak sa poloha niektorého z centroidov nezmení počas niekoľkých po sebe nasledujúcich krokov.

2.13.2 K-Means Clustering:

K-Means clustering bude vždy konvergovať. Nie vždy síce nájde globálne optimum, avšak vždy sa mu podarí nájsť lokálne optimum.

Táto metóda je pomerne senzitívna na outliery a šum v dátach, a taktiež na počiatočnú inicializáciu centroidov. Keďže počiatočná pozícia centroidov je náhodná, vravíme o K-means, že je nedeterministickým algoritmom a teda klastre spočítané v každej iterácii nemusia byť konzistentné.

2.13.3 Voľba K:

Pri výpočte si musíme dávať pozor na to, aký počet klastrov zvolíme. Existuje neikoľko metód určujúcich aké K je optimálne:

1. Elbow point metóda
   Spočítame celkové WSS(= within-cluster sum of squares) pre rôzne K. Na mieste kde v grafe vznikne elbow sa nachádza optimálne riešenie.
2. Gap value metóda
3. Silhouette value metóda

2.14 What is the difference between sampling and quantizing an image function? Describe the principle of both. Describe the methods of quantization of color images. How is the quality of quantization evaluated?

Obě metody dělají ze spojitých přechodů barev diskrétní a snižují počet barev, které je nutné využít na vykreslení obrázku.

Při vzorkování (sampling) se vždy udělá integrál čtvercové oblasti a vezme se průměrná barva jako barva, kterou bude tato oblast obarvená po nasamplování obrázku. Výsledný obrázek vypadá jako minecraft.

Při kvantizaci je snaha o diskretizaci continuous barevného gradientu, a namapování rozsahu barev na jednu reprezentativní barvu tohoto rozsahu. Výsledné obrázky mají ostrůvky stejných barev v místech, kde původní barva byla podobná.

Methods of quantization

- Obrazově nezávislé metody
  - Color space partitioning - s přednastavenýma barvama do kterých řadím
  - Thresholding - podle předem určených tresholdů
- Obrazově závislé metody
  - Color clustering - např. median cut
  - Image segmentation – areas with the same characteristics
  - Thresholdin - s adaptivními thresholds (např. pomocí thresholdování histogramu podle peaks)

Kvalita kvantizace se hodnotí pomocí Quantization error, což je vzdálenost původní barvy od nové barvy (centroidu, barvy, na kterou původní barva byla namapována). Většinou se používá Euklidovská vzdálenost v RGB, přestože tam vizuelní a euklidovská vzdálenost neodpovídají, mělo by to být v CIE.

Kvalita je MSE obrázku. (Průměr umocněných vzdáleností každého pixelu od jejich centroidu)

Vykreslení tohoto erroru dává Error map = jak moc špatně to je.

2.15 What are Gaussian and Laplace pyramid images? Give an example of their use in computer vision.

- Gaussovská pyramída
  - Princíp: vezmem obrázok, rozostrím ho a zmenším rozlíšenie. Toto spravím niekoľkokrát za sebou a mám Gaussovskú pyramídu obrazov.

- Gaussovská pyramída môže následne slúžiť na priestorové prehľadávanie - tzn, že ak sa pozrieme na naivný prístup kde mám objekt a hľadám rovnako veľký objekt v danom obraze, čo malo veľkú zložitosť. Pri pyramídovom prístupe možno komplexnosť znížiť. Čo môžme spraviť je, že si vyrobíme Gaussovské pyramídy aj zo šablóny aj z obrazu a začneme tou najviac degradovanou verziou. Nájdeme oblasť, ktorá je len približne tam, kde by daný objekt mal byť. Potom vo vyššej pyramíde neprehľadávame celý obraz, ale len tú časť, ktorú sme identifikovali na nižšej úrovni.
- Keď máme Gaussovskú pyramídu, vezmeme obrázok z najnižším rozlíšením, prejdeme celý obrázok a identifikujeme nejakú oblasť, kde by cca mal byť daný objekt, lenže nás zaujíme presné umiestnenie daného objektu, tak prejdeme do vyššej úrovne a už prehľadávame len identifikovanú oblasť kde postupne spresňujeme umiestnenie objektu.
- V každej úrovni Gaussovskej pyramídy máme takmer rovnakú informáciu, pretože Gaussov filter je nízko priepustný filter, ktorý odrezáva vysoké frekvencie a prepúšťa len tie nízke, tzn že v každej úrovni Gaussovskej pyramídy zajnižšie frekvencie v obraze zostávajú. Toto je redundantná informácia, ktorú nepotrebujeme zachovávať na každej úrovni a hlavne z Gaussovskej pyramídy sa strácajú nejaké informácie v podobe vysokých frekvencií. Keby som si nejakým spôsobom zachovala práve tieto informácie, bolo by to fajn, pretože by som neprišla o žiadnu informáciu. $\Rightarrow$ nepotrebujeme v jednotlivých úrovniach ukladať celý obraz, ale stačila by nám chyba, ktorú sme spravili, tým, že sme odstránili tie vysoké frekvencie.

- Laplaceovská pyramída ukladá vysoké frekvencie. Laplaceovská pyramída je v podstate doplnkom Gaussovskej pyramídy, čiže keď mám obrázok a vyrábam z neho Gaussovskú pyramídu, tak keď k rozostrenej Gaussovskej pyramíde priložím tie detaily (vysoké frekvencie), tak dostávam pôvodný obrázok. Keď sa na to pozrieme v oblasti frekvencií, tak v Gaussovskej pyramíde v tom najväčšom rozlíšení ukladám všetky frekvencie a postupne zužujem tie frekvencie, ktoré sú ukladané a v Laplaceovskej pyramíde ukladám len rozdiel medzi nasledovnou hodnotou a pôvodným obrázkom.
- Čiže z Laplaceovskej pyramídy viem pekne rekonštruovať pôvodný obraz, ktorý mám uložený v tej Gaussovskej pyramíde. V Laplaceovskej ho uložený nemám, ale viem ho rekonštruovať.
- Najnižšie rozlíšenie v Laplaceovskej pyramíde zodpovedá rovnakému objektu v Gaussovskej pyramíde. Vzťah medzi Gaussovskou a Laplaceovskou pyramídou sa dá vyjadriť nasledovne (i-čko hovorí o úrovni pyramídy a expand je zväčšenie oblasti v Gaussovskej pyramíde):

$$L_i = G_i - expand(G_{i+1})$$

$$G_i = L_i + expand(G_{i+1})$$

- V princípe sa mi v každom kroku ku môjmu rozostrenému obrázku pridávajú detaily v podobe vysokých frekvencií.
- Funkcia expand má sebe zodpovedajúcu funkciu reduce, ktorá sa používa na výrobu Gaussovskej pyramídy a vyzerajú nasledovne: Reduce:

$$G_l(i,j) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) G_{l-1}(2i+m, 2j+n)$$

Expand:

$$G_l(i,j) = 4 * \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m,n) G_{l+1}(\frac{i-m}{2}, \frac{j-n}{2})$$

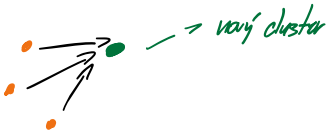
2.16 Describe the mean-shift method. What is it used for?

Mean-shift algoritmus je podobný algoritmu K-means, až na to, že nepotrebuje parameter K. Mean-shift zhlukuje body obrazu (pixely) pomocou konvergencie do lokálnych maxím hustoty.

Algoritmus nebol pôvodne zostrojený pre computer vision, avšak je použiteľný na segmentáciu farebných obrazov práve na zistení kde sú body v priestore husté. ⇒ Identifikujeme body, ktoré majú podobné farby tým, že nájdeme zhluky v priestore farieb.

Pracuje iteratívne:

- V každej iterácii pre určitú pozíciu počíta gradient hustoty bodov v blízkom okolí a pohybuje sa b smere gradientu, až kým nedosiahne lokálne maximum.
- Tento prístup je iniciovaný v každom bode
- Výsledkom sú súradnice lokálneho maxima pre každý jeden bod



- Následne všetky body, ktoré dokonvergovali do rovnakého maxima s určutou toleranciou zhlukujeme do jednej rovnakej oblasti.

V jednorozmernom prípade sa mean-shift dá predstaviť ako hľadanie módov v histograme:

- Inicializujeme ho v nejakom bode a určíme si okno W. Od tohto okna vlastne závisí výsledok.
- V okne W spočítame ťažisko bodov, ktoré doňho spadajú: $\bar{x} = \frac{1}{|W|} = \sum_{x \in W} x$
- Okno posunieme do ťažiska
- Opakujeme kým neskonverguje

Mean-shift segmentácia:

1. Z každej vzorky spustíme mean-shift a zapamätáme si lokálne maximum, do ktorého algoritmus dokoncergoval
2. Zhlukujeme vzorky, ktoré dokonvergovaly do rovnakého maxima (s určitou toleranciou), to znamená do oblasti, z ktorej vzorky konvergujú do jedného bodu (basin of attraction)

Treba ošetriť niekoľko situácií:

- Sedlové body: odstránime ich nejakou malou zmenou množiny dát, ktoré máme k dispozícii
- Odstránime viacnásobné maximá

Alternatívy/optimalizácie mean-shiftu:

- k-means + mean-shift

2.17 Describe the 4 basic morphological operations $\oplus, \ominus, \circ, \bullet$.

Značenie:
Mirroring: $\hat{A} = \{-a | a \in A\}$
Translation: $A_z = \{a + z | a \in A\}$
E je štruktúrny prvok

Pre binárne obrazy:

E je štrukturální prvek

- **Dilatácia** $\oplus$: expanzívna operácia - zväčšuje množinu, zaplní diery určitej veľkosti a tvaru, môže spojiť disjunktné komponenty.

$$A \oplus E = \bigcup_{e \in E} A_e = \bigcup_{e \in E} \{a + e | a \in A\} = \{a + e | a \in A, e \in E\} = \{x | x \in \hat{E}_x \cap A \neq \emptyset\}$$

  Vlastnosti: komutativita, asociativita, distributivita $(A \oplus (E \cup F) = (A \oplus E) \cap (A \oplus F)$

- **Erózia** $\ominus$: opak dilatácie - zmenšuje množinu, môže rozdeliť množinu.

$$A \ominus E = \bigcap_{e \in E} \{a - e | a \in A\} = \{x | E_x \subseteq A\}$$

  Vlastnosti: nekomutativita, $(A \ominus E) \ominus F = A \ominus (E \oplus F)$, $(A \ominus E) \oplus E \subseteq A \subseteq (A \oplus E) \ominus E$, dualita erozie a dilatácie: $(A \ominus E)^c = A^c \oplus \hat{E}$

- **Otvorenie** $A \circ E = (A \ominus E) \oplus E$ - posúvame E po vnútornej hranici A. Vyhladí okraje, odstráni tenké spojenia a malé výbežky (závisí na E), zachováva približnú veľkosť množiny.

  Vlastnosti: $(A \circ E) \circ E = A \circ E \subseteq A$

- **Uzavretie** $A \bullet E = (A \oplus E) \ominus E$ - posúvame E po vonkajšej hranici A. Platí $A \subseteq A \bullet E = (A \bullet E) \bullet E$, dualita s otvorením $(A \bullet E)^c = A^c \circ \hat{E}$. Vyhladí okraje, spojí oblasti, ktoré sú blízko a malé diery. Zachováva približnú veľkosť množiny.

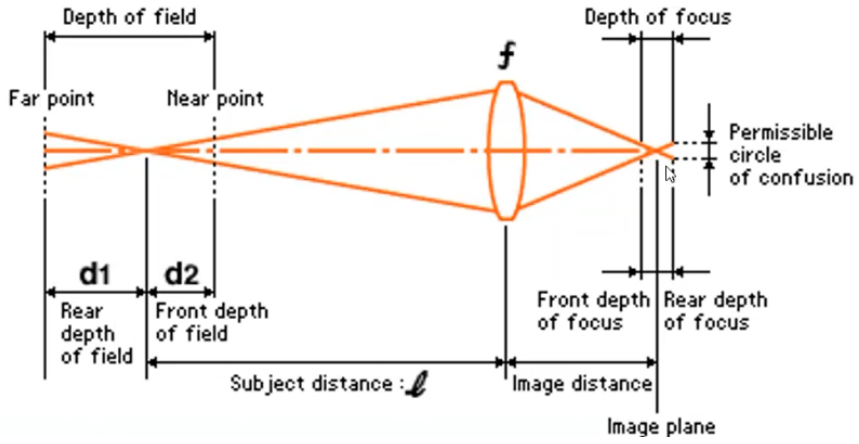
Pri vhodných štrukturálnych prvkoch na dajú tieto operácie použiť na detekciu tvarov v obrázku, granulometriu (hľadanie kruhov určitej veľkosti)

Morfologické operácie na greyscale obrázkoch: $f : (W \times H) \to [0, 1]$ - intenzita pixelu

- **Dilatácia** $(f \oplus h)(x, y) = \max_{(r,s) \in H}\{f(x - r, y - s) + h(r, s)\}$. Zvyšuje intenzitu pixelov (jas)
- **Erózia** $(f \ominus h)(x, y) = \min_{(r,s) \in H}\{f(x + r, y + s) - h(r, s)\}$. znižuje intenzitu pixelov (jas)
- **otvorenie, zatvorenie** - analogicky

Použitie: filtrácia tmavého alebo svetlého šumu.

2.18 Explain the concepts of depth of field, circle of confusion, and focal length. Illustrate with a figure. What affects the depth of field?



Ak sa nejaký bod premietne cez tenkú šošovku na obrazovú rovinu a jeho lúče sa nestretnú v jednom bode, tak obraz bude stále vyzerať ako ostrý, ak sa všetky lúče zobrazia to tzv. *circle of confusion (krúžok rozptylu)*.

Depth of field = hĺbka ostrosti. Veľkosť intervalu, v ktorom môže byť objekt, aby sa jeho obraz javil ostro. Závisí na vzdialenosti objektu a, veľkosti circle of confusion c, ohniskovej vzdialenosti f a clony (aperture) N (pomer ohniskovej vzdialenosti a preiemeru šošovky).

$$DOF = \frac{2a^2Nc}{f^2}$$

Ohnisková vzdialenosť - vlastnosť šošovky, ktorá udáva, v akej vzdialenosti sa pretínajú lúče, ktoré do šošovky prídu paralelne (spojky).

Rovnica tenkej šošovky:

$$1/a + 1/a' = 1/f$$

, kde f je ohnisková vzdialenosť, a je vzdialenosť obrazovej roviny od šošovky a a' je vzdialenosť objektu. Hrúbka šošovky je zanedbateľná.

2.19 What defects can lenses have? What is the cause of each error?

- chromatická chyba. — = chromatická aberace
  - Pozdĺžna - ak lúče idú paralelne. Rôzne vlnové dĺžky svetla sa zaostrujú v rôznych bodoch. Fialová najbližšie, červená najďalej.
  - Priečna - vlnové dĺžky sa zaostrujú v správnej vzdialenosti od šošovky, ale na iných miestach.
  - Často kombinácia oboch. Sú spôsobené tým, že rôzne vlnové dĺžke svetla majú rôzne indexy lomu.
- sférická chyba
  nesústreďuje všetky lúče do jedného bodu. Je to spôsobené tým že šošovka nie je fyzicky dokonale vyrobená. Rieši sa to optickými sústavami - viacero šošoviek za sebou.
- vingetting - tmavé okraje obrazu
  - prirodzená - žiarenie objektu na šošovku ! = žiarenie premietnuté na obrazovú rovinu. je spôsobená natočením objektu voči šošovke (priestorový uhol). Nedá sa uplne riešiť.
  - oprická - lúče zablokované okrajmi šošoviek v optickej sústave - preto sú okraje obrazku potom tmavšie
  - mechanická - niekedy chceme pricloniť a zabrániť lúčom z iných častí (napr. slnko)
- radiálne skreslenie - dá sa riešiť transformáciou výsledného obrázku
  - vankúšové (polštářové)
  - súdkové (fish eye)

2.20 Describe segmentation using thresholding. Describe Otsu's thresholding in detail.

Segmentácia obrazu - rozdelenie obrazu na oblasti, v ktorých chceme identifikovať niečo, čo poznáme z reálneho sveta (časti, ktoré majú vysokú koreláciu s vecami z reálnehgo sveta).

Segmentácia založená na prahovaní:
Je najjednoduchšou z techník - len porovnáme hodnotu voči nejakému prahu a na základe toho sa rozhodneme. Je výpočtovo nenáročná a rýchla.

Prahovanie je transformácia, ktorá nám rozdelí vstupnú množinu na 2 oblasti, ktoré sú buď vyššie alebo nižšie ako prah. Zobrazuje teda vstupný obraz $f(i, j)$ na výstupný obraz $g(i, j)$ tak, že:

- $g(i,j) = 1$, ak $f(i,j) \geq T$ (objekt)
- $g(i,j) = 0$, ak $f(i,j) < T$ (pozadie)

Globálne a lokálne prahovanie:
Ak na celý obraz aplikujeme 1 prah T, ide o globálne prahovanie
Ak v každom okne obrazu zvolíme iný prah T_W ide o lokálne prahovanie
Ak zvolíme prah $T(x,y)$, ktorý závisí iba od súradníc (x,y), hovoríme o adaptívnom (dynamickom) prahovaní

Prahovanie dáva dobré výsledky v prípade, že histogram obrazu je bimodálny (vieme v ňom určiť 2 výrazné kopčeky, ktoré potom prahom oddelíme).

V prípade, že nemáme bimodálny prah, môžeme použiť adaptívne prahovanie s tým, že sa snažíme deliť obraz kým tie jednotlivé okná, na ktoré to rozdelíme nemajú bimodálny histogram. Vtedy už môžeme skončiť s delením a určiť prah pre dané okno na základe histogramu.

Otsu prahovanie:
Používané na automatické prahovanie obrázku.
Otsu metóda je technika založená na rozptyle, ktorá slúži na hľadanie prahovej hodnoty, pri ktorej je váhovaný rozptyl medzi pixelmi popredia a pozadia najmenší. Kľúčovou myšlienkou je prejsť všetky možné hodnoty prahu a merať rozptyl pixelov pozadia a popredia. Potom nájdeme prahovú hodnotu, pri ktorej je rozptyl najmenší.

Rozptyl obrázku:

$$\sigma_i^2 = \sigma_b^2(t) + \sigma_w^2(t)$$

Intra-class rozptyl - minimalizujeme:

$$\sigma_w^2(t) = P_0(t)\sigma_0^2(t) + P_1(t)\sigma_1^2(t)$$

Inter-class rozptyl - maximalizujeme:

$$\sigma_b^2(t) = P_0(t)P_1(t)(\mu_0(t) - \mu_1(t))^2$$

2.21 Describe region-based segmentation - growing, splitting, their combination

Segmentácia obrazu - rozdelenie obrazu na oblasti, v ktorých chceme identifikovať niečo, čo poznáme z reálneho sveta (časti, ktoré majú vysokú koreláciu s vecami z reálnehgo sveta).

Segmentácia založená na oblastiach:
Nehľadám hranicu, ale priamo identifikujem body oblasti, ktoré patria k sebe.
Na základe vybranej vlastnosti (väčšinou jas alebo farba) sa snažíme zoskupiť body, ktoré majú túto vlastnosť podobnú. Používané metódy sú: zhlukovanie, narastanie (spájanie) oblastí

- zhlukovanie
  - hierarchické - vytvárame určité úrovne
    - Aglomeratívne (Bottom-up) - začneme na jednotlivých pixeloch a postupne ideme zdola nahor
    - Divizívne (Top down) - začneme na celej množine, ktorú postupne delíme
    - kombinované - kombinujeme tieto 2 prístupy
    - Narastanie oblastí
      - Začína pri učení malých jadier, ktoré sú homogénne a potom sa pridávajú pixely (alebo spájajú oblasti), aby homogénne oblasti boli maximálne
      - výsledok záleží na poradí spájania

        - originálny obraz s jadrovým bodom
        - počiatočné štádium rastu
        - Medzištádium
        - Finálna oblasť
        - ako kritérium spájania sa používajú 2 heuristiky:
        - Dve susedné oblasti sa spoja ak významná časť ich spoločnej hranice pozostáva zo slabých hrán (v pomere k obvodu menšej oblasti)
        - Dve susedné oblasti sa spoja ak významná časť ich spoločnej hranice pozostáva zo slabých hrán (dĺžke spoločnej hranice)
    * Delenie oblastí
        - Opačné ku spájaniu oblastí
        - Začína pri podsegmentovanom obraze, ktorý nespĺňa podmienku homogenity oblastí
        - Potom sa existujúce oblasti postupne delia, aby spĺňali základné podmienky segmentácie vrátane homogenity
    * Delenie a spájanie oblastí
        - spája výhody oboch prístupov
        - používa hierarchickú dátovú štruktúru obrazu
            1. Definuj počiatočnú segmentáciu, kritérium homogenity a dátovú štruktúru
            2. ak niektorá oblasť R nie je homogénna, rozdeľ ju na 4 podoblasti. Ak možno spojiť 4 podoblasti na rovnakej úrovni hierarchie do homogénnej oblasti, spoj ich.
            3. Ak a dve susedné oblasti R_i a R_j (na rozličných úrovniach hierarchie) dajú spojiť do homogénnej oblasti, spoj ich.
- nehierarchické - robíme rozdelenie bodov do daných zhlukov (K-means)

2.22 Describe the RANSAC method.

RANSAC - RANdom SAmple Consensus

Iteratívna medóda na určenie parametrov modelu z množiny nameraných bodov, ktorá obsahuje odľahlé dáta (outliery)

Využíva sa:

- Párovanie lokálnych príznakov
- Spájanie panorám
- 3D rekonštrukcia
- Rozpoznávanie obrazcov
- Augmented reality

Princíp fungovania:

- Náhodný výber minimálnej podmnožiny bodov na to, aby sme určili parametre modelov.
- Určenie podporovateľov modelu - tzn. bodov v určenej vzdialenosti od modelu
- Celý tento postup pre N vzoriek
- Vždy si pamätáme koľko z tých bodov nám tento model podporilo. Model, ktorý mal najviac podporovateľov vyhráva.

Voľba parametrov:

- Veľkosť vzorky s: typicky minimálny počet potrebný na výpočet parametrov modelu
- Prah vzdialenosti t: tak, aby nejaké percento bodov podporovalo víťaznú vzorku
- Počet opakovaní N: Chceme robiť toľko opakovaní, aby s pravdepodobnosťou p aspoň 1 vzorka neobsahovala odľahlé dáta

Aplikácie: spájanie panorám, hľadanie objektov v obrázku (nemusia byť celé viditeľné)

2.23 Describe the HOG descriptor.

HOG = histogram of oriented gradients
Môže sa požiť na popis celého obrázku, alebo len vybraných častí.
HOG sa pozerá na každý bod obrázku, v každom bode spočítame gradient a v určitých častiach obrázku si spravíme histogram orientácií týchto gradientov.

Algoritmus:

1. Gradient v každom bode.
2. Obrázok rozdelíme na 8x8 bodov. V tejto bunke si spravíme histogram gradientov.
3. Normalizácia 16x16 bloku. Normalizované časti spojíme do jedného deskriptora a vypočítame.

HOG prístup je používaný napríklad SIFT detektorom, ktorý detekovaný bod otočí tak, aby mal orientáciu zarovno s osou a v okolí tohto bodu vypočíta histogram orientovaných gradientov.
Vypočíta ho tak, že aplikuje na gradienty Gaussovské váhovanie, čiže body, ktoré sú ďaleko od zaujímavého bodu prispievajú menej ako body, ktoré sú priamo okolo.
Okolie 16x16 si rozdelí na body 4x4 a ďalej pracuje klasicky ako HOG.

2.24 Describe the BoVW method.

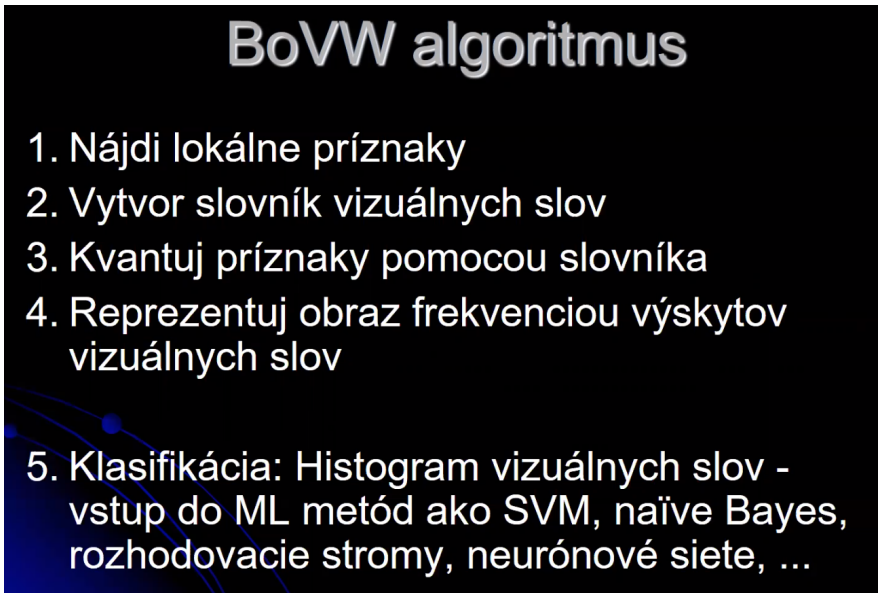
BoVW - Bag of Visual Words

Hľadania lokálnych príznakov a následnú klasifikáciu obrazov na základe toho aké lokálne príznaky sa tam nájdu.

V tomto prípade nemusím mať šablónu a nemusím hľadať presný objekt. Môžem hľadať objekty, ktoré sú medzi sebou podobné.
Ak objekt obsahuje nejaké špeciálne vizuálne slová, tak ho zaradíme do danej kategórie.
Vizuálne slovo - malá, vizuálne rozlíšiteľná časť obrazu.
V každom obraze nájdeme nejaké ľahko odlíšiteľné oblasti a nejakým spôsobom ich budeme klasifikovať.
Algoritmus:

- Máme databázu obrazov, z ktorej nájdeme zaujímavé lokálne príznaky, ktoré nám budú tvoriť vizuálny slovník (bude obsahovať reprezentantov vizuálnych slov, ktoré sme našli - títo reprezentanti sú vybraní zo zhluku podobných príznakov). Na nájdenie lokálnych príznakov použijeme detektory a na popísanie deskriptory.
- Kvantovanie príznakov pomocou slovníka ⇒ pomocou histogramu vieme reprezentovať koľko vizuálnych slov bolo reprezentovaných pomocou ktorého slova zo slovníka
- Reprezentujeme obraz frekvenciou výskytov vizuálnych slov

- Klasifikácia: Po analýze ľubovoľného obrazu, ktorý nám príde, vieme zistiť či obsahuje slová vizuálne podobné ktorej skupine príznakov ⇒ podľa tohto vieme obraz zaradiť do niektorej z kategórií.



2.25 How can the PCA method be used for face recognition? Describe the principle(eigenfaces).

Mám množinu tvárí a chcem zistiť či nejaký nový obázok je tvár človeka z môjho datasetu. Každý obrázok z datasetu premením do jedného vektoru tak na to môžem použiť PCA čím nájdem pod-priestor efektívne modelujúci tváre. Funguje to tak, že zoberiem n d-rozmerných vektorov a chcem nájsť nové bázové vektory také že v danom smere bude variancia dát čo najväčšia, tak aby boli bázové vektory ortonormálne. Báza budú vlastné vektory kovariančnej matice (netreba ukazovať prečo). Pri rozpoznaní hľadám prvok z množiny ktorý je najbližší k môjmu obrázku. Treba nastaviť prah od kedy vieme prehlásiť, že daná tvár sa v našom datasete nenachádza. Pri použití PCA sa z trénovacej množiny tvárí extrahujú hlavné komponenty, ktoré majú najväčší podiel na variabilite tvárí v tejto množine. Výsledkom sú eigenfaces, ktoré predstavujú bázy v priestore tvárí. Každý eigenface je reprezentovaný ako vektor a predstavuje určitý "archetyp" tváre.

2.26 Describe the Lucas-Kanade method for calculating optical flow.

Pohyb oblastí s rovnakou intenzitou v obraze. Nemusí nutne zodpovedať reálnemu 3D pohybuje ale je to to najlepšie čo vieme vytiahnuť z obrázkov. Predpoklady sú, že intenzita sledovaného bodu sa nemení. Bod sa nepohne príliš ďaleko respektíve máme vhodné fps. a bod sa bude sa bude pohybovať rovnako ako jeho okolie. Za týchto predpokladov je v malom časovom okne obrazová funkcia priemetu bodu konštantná. Teda derivácia podľa času bude nula:

$$\frac{\partial f(x(t), y(t), t)}{\partial t} = 0$$

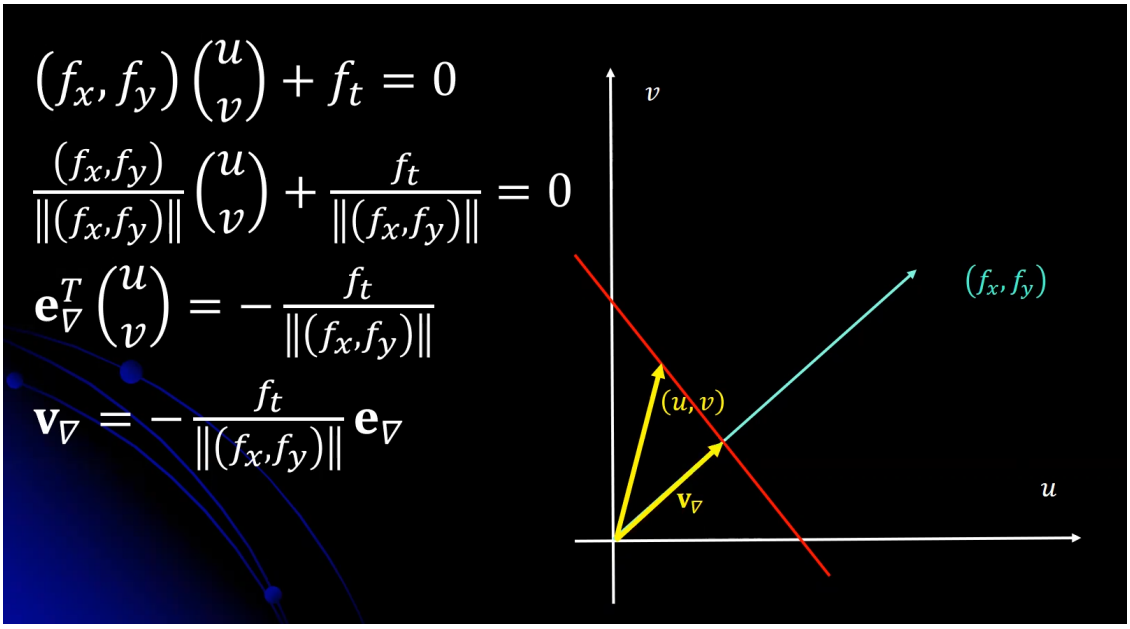
$$\frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial t} = 0$$

1. Gradient: ∇f
2. Optický tok: $\begin{pmatrix} u \\ v \end{pmatrix} = \left(\frac{dx}{dt}\ \frac{dy}{dt}\right)^T$

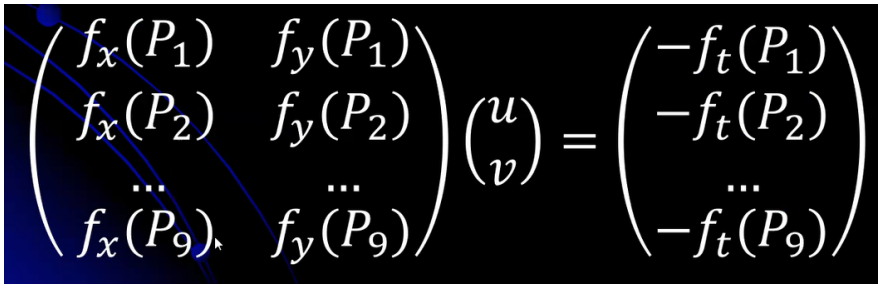
3. Časová derivácia: $\frac{\partial f}{\partial t}$

Z toho máme jednu rovnicu s 2 neznámami ktoré chceme vypočítať. Presné riešenie nebudem vedieť nájsť ale viem, že riešenie bude ležať na priamke.

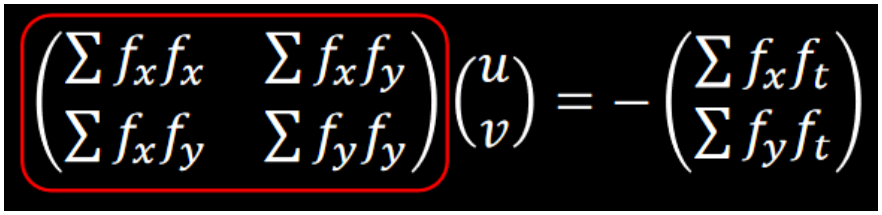
$$\nabla f^T \begin{pmatrix} u \\ v \end{pmatrix} + \frac{\partial f}{\partial t} = 0$$



Lucas-Kanade sleduje kromě jednoho bodu i jeho okolí (např. 3x3) a potom máme rovnici pro každý bod z okolí, nejenom pro bod, jehož pohyb se snažíme zjistit. Tímpádem máme soustavu 9 rovnic na zjištění 2 neznámých.



Matici fx a fy si označíme jako A, $\begin{pmatrix} u \\ v \end{pmatrix}$ je hledaný optický tok, který označíme jako d, a matici -ft si označíme jako b. Potom $Ad = b$. Metodou nejmenších čtverců dostaneme $A^T Ad = A^T b$, což jde rozepsat jako:



Tato rovnice je vyřešitelná pokud vlastní čísla červeně zvýrazněné matice jsou kladná, dostatečně velká a přibližně stejně velká. (V tom případě je matice invertovatelná). Také je potřeba aby v obraze nebyl velký šum.

Stejnou podmínku má Harris corner detector na detekci rohů, tedy Lucas-Kanade nám říká, že pokud dokážeme najít v okolí bodu roh, dokážeme vypočítat jaký pohyb vykonal daný bod (aneb že rohy se dobře sledují).

3 Part 3

3.1 Gaussian pyramid of images and its use in computer vision.

Gaussovská pyramída je hierarchická reprezentácia obrázka, ktorá sa vytvára postupným rozmazávaním a zmenšovaním rozlíšenia pôvodného obrázka. Slúži na priestorové prehľadávanie a znižovanie komplexity pri hľadaní objektov v obraze, keďže umožňuje identifikovať oblasť záujmu v hrubších úrovniach pyramídy a následne presnejšie lokalizovať objekt vo vyšších úrovniach.

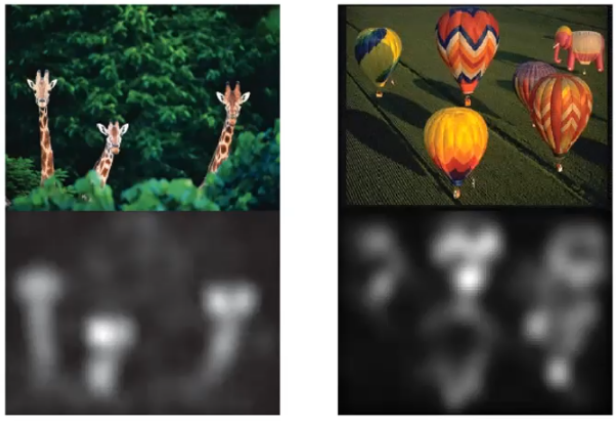
3.2 Laplace pyramid of images and its use in computer vision.

Laplaceova pyramída je doplnkom Gaussovskej pyramídy a ukladá vysokofrekvenčné detaily obrázka. Pomocou Laplaceovej pyramídy je možné rekonštruovať pôvodný obraz z Gaussovskej pyramídy. Každá úroveň Laplaceovej pyramídy obsahuje rozdiel medzi príslušnou úrovňou Gaussovskej pyramídy a expanziou nasledujúcej úrovne.

3.3 Describe using spectral residuals to find salient regions in an image

Robí fourierovu transformáciou na obrázku. Z nejakého dôvodu platí, že logaritmus frekvenčného spektra sa správa podobne pre ľubovoľé obrázky. Teda môžem zobrať veľa obrázkov spriemerovať ich spektrum a potom ho odčítať od obrázku v ktorom chcem identifikovať zaujímavé časti. Tým dostanem frekvencie ktoré sa líšia od priemeru.

jde o kumulativní průměr nad stejným obrázkem.



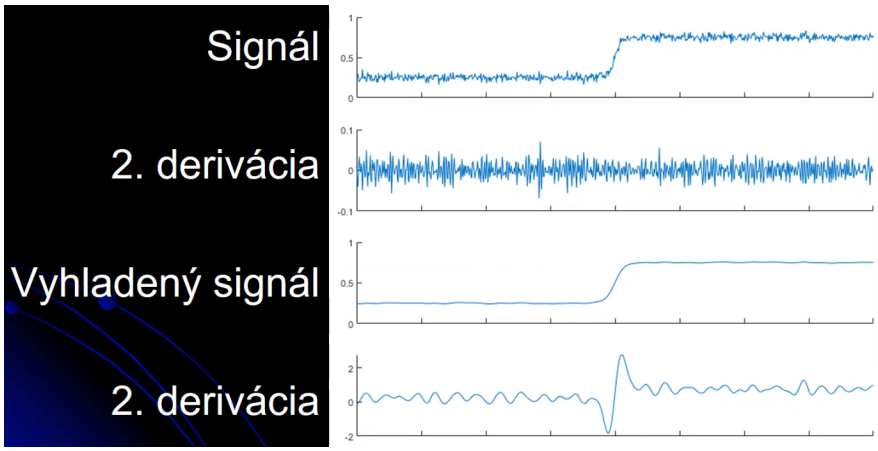
3.4 Describe the method of using significant frequencies in finding salient regions in an image (2 frequency-tuned methods).

Pozerám sa na to ako sa jednotlivé pixely odlišujú od priemernej farby v obraze. Teda vypočítam priemernú farbu potom obrázok rozostrím pomocou gaussovského rozmazania a počítam pre každý pixel vzdialenosť k priemernej farbe. Okrem toho môžem ešte robiť multiscale frequency-tunded saliency, kde sa nepozerám

na globálny priemer ale vyberám si farbu z nejakého blízkeho okolia pixelu. Väčšinou sa zoberie ako okolie, polovica, štvrtina a osmina rozmerov obrázka. Tie potom sčítam a dostanem novú mapu významnosti, ktorá vie zistiť ako významné sú objekty na rôznej škále.

3.5 What idea do we use to detect edges in noisy images? What is LoG?

Na detekciu hrán funguje dobre najskôr signál vyhladiť a potom naň aplikovať 2. deriváciu (viď obrázok). Môžem spraviť konvolúciu pomocou gausiánu $\frac{1}{2\pi\sigma^2}\exp -\frac{x^2+y^2}{2\sigma^2}$ a následne obrázok zderivovať. Konvolúcia sa však správa slučne preto stačí keď zderivujem gausián a potom ho použije ako konvolučný filter. A to je presne LoG: laplacián gausiánu $\nabla^2 G = \frac{\partial^2 G}{\partial x^2} + \frac{\partial^2 G}{\partial y^2}$



3.6 Describe Moravec's corner detector

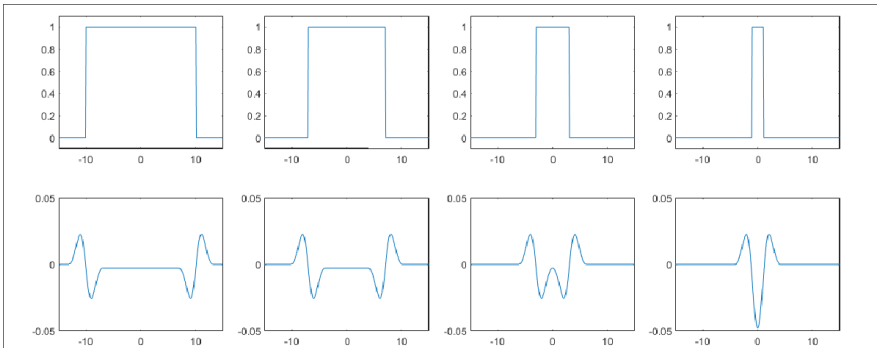
Idea je, že sa pozriem na to ako sa správa okno v okolí nejakého bodu. V Moravcovom detektore je rohovitosť bodu daná ako najmenšiu hodnotou z $E(u) = \sum_i w(p_i)(I(p_i+u)-I(p_i))^2$. Výsledky potom treba odprahovať pričom výsledky sú zavilé na intenzite bodov. To nie je moc super, pretože nevhodne zvolený prah nedokáže dané body nájsť.

3.7 Describe the SUSAN corner detector

Zoberiem kruhové okolie kde majú body podobnú intenzitu tomu hovoríme USAN. Keď chcem vypočítam USAN hodnotu pre nejaký bod, tak sa pozriem koľko iných bodov na tomto okolí sú mu podobné. Čím menšiu podobnosť mám tým je pravdepodobnejšie, že bod sa nachádza na rohu. Zaujíma ma SmallestUSAN, teda detektor rohov bude brať lokálne minimá USAN hodnôt.

3.8 How do we detect blobs using the Laplacian of the Gaussian?

Použíjem LoG ako konvolučný filter na obrázok. Ten zvýrazní škvrny nejakej veľkosti. Pri to je dôležité aby sme používali normalizovaný LoG aby fungoval správne. Treba ešte vybrať správny parameter σ na to aby som vedel detektovať škvrny správnej veľkosti.



3.9 Describe the FAST detector and the BRIEF descriptor.

FAST detektor sa pozerá koľko bodov sa nepodobá na stred. Je veľmi podobný detektoru SUSAN, ktorý ale sledoval všetky body v kruhovom okolí, zatiaľ co detektor FAST pozerá len body na kružnici. Porovnáva tedda hodnoty intenzity bodu a jeho susedov na Bresenhamovej kružnici s polomerom $r = 3$.

BRIEF deskriptor: binárny deskriptor (vracia hodnoty 0/1), ktorý vezme okolie body a pre dvojice bodov v okolí zistí či intenzita v jednom bode je väčšia ako intenzita v druhom bode. Existuje aj orientovaný BRIEF, ktorý okrem iného využíva aj rotáciu okolia.

3.10 What is a Bayer color filter array? How do we use it?

Bayerov filter sa používa vo fotoaparátoch na zachytávanie farebného obrazu. Mám rôzne senzory ktoré zachytávajú: krátku, strednú alebo dlhú vlnovú dĺžku. Bayerov filter je mini-matica (väčšinov 2x2), ktorá určuje ako sú senzory usporiadané tak aby vedeli čo najlepšie reprezentovať obraz. Následné potrebujem určiť 3 kanály pre každý pixel a skombinovať informácie z kanálov. To sa robí metódou demosaikovania, ktorá len spriemeruje susedné hodnoty. Ako v tej úlohe čo bolo treba programoveď veď

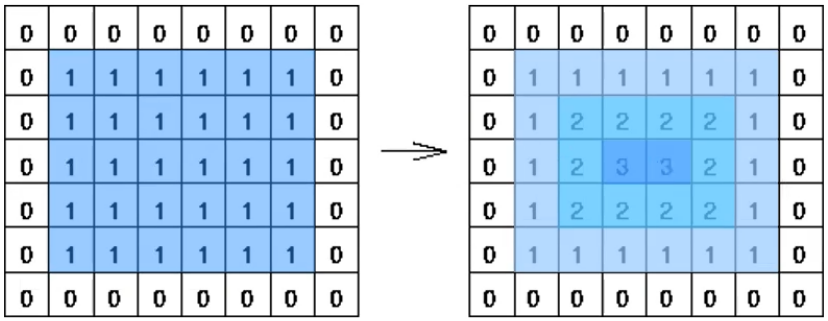
3.11 What can we determine from a CIE chromaticity diagram?

CIE chromatický diagram je grafická reprezentácia farebného priestoru, ktorá umožňuje interpretáciu vzťahov medzi farbami. Z tohto diagramu je možné určiť niekoľko dôležitých vlastností a informácií:

1. Farebná nasýtenosť: Vzdialenosť od stredu diagramu ku krivke znázorňujúcej farby reprezentuje nasýtenosť farby. Vzdialenosť od stredu (tzv. sýtosť) predstavuje, do akej miery je farba čistá alebo zmiešaná so svetlom inej vlnovej dĺžky.
2. Dajú sa odtiaľ dobre vyčítať farby štandardných osvetlovacích telies pri zahriatí na určité teploty
3. Viem určiť dominantnú vlnovú dĺžku nejakej farby tak že spojím bod v referenčnou farbou (vščsinou bielovu a potiahnem priamku až na okraj diagramu)
4. Gamut je oblasť farieb ktoré zariadenie vyprodukovať sa dá vyznačiť ako nejaká plocha na diagrame

3.12 Describe the distance transform. What is it used for, and how is it related to morphology?

Distance transform je operátor aplikovaný na binárne obrazy ktorý počíta vzdialenosť od množiny. Výsledkom operátora sú úrovne šedej ktoré hovoria ako veľmi vzdialený je je bod od najbližšieho okraja. V súvislosti s morfológiou je distance transform spojený so zväčšovacími a zmenšovacími operáciami (dilaterácia a erozia). Pri dilatácií robím DT pozadia a pri erózii DT objektu.

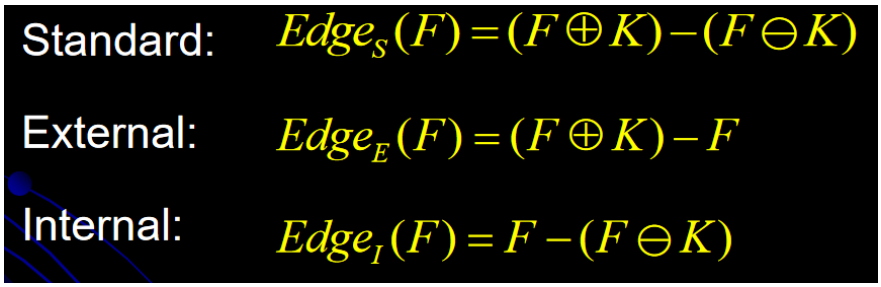


3.13 What are the White top hat and Black top hat transforms used for?

Keď chceme použiť thresholding na obrázky s neuniformným pozdaím, použijeme predtým top hat transformáciu, ktorá zmenší rozdiely medzi intenzitami pozadia v rôznych častiach obrázku. Tiež sa používa na zvýšenie kontrastu v obrázkoch. Podľa farby objektov, resp. pozadia si vyberieme medzi white alebo black top-hat transformáciou. $WTT = A - (A \circ E)$, $BTT = (A \bullet E) - A$, E je štrukturálny prvok.

3.14 How do you find edges in binary and gradients in grayscale images using morphological operations?

Hledání hran v obrázku, kde F je obrázek a K je strukturální prvek:



Hledání gradientu v grayscale obrázcích má ty samé rovnice.

3.15 How do we combine two images to make the transition between them invisible?

Vytvoríme Laplaceove pyramídy z obidvoch obrázkov L_a, L_b, Gaussovskú pyramídu masky G_m (ak je nejaká) a skombinujeme ich úrvovne ako priemer vážený príslušnou úrovňou pyramídy masky a tým vytvoríme L_c, ktorá je Laplaceovská pyramída zmiešaného obrazu. Rekonštrukciou úrovní L_c dostávame skombinovaný obrázok.

3.16 What is the difference between a saliency map and a task map?

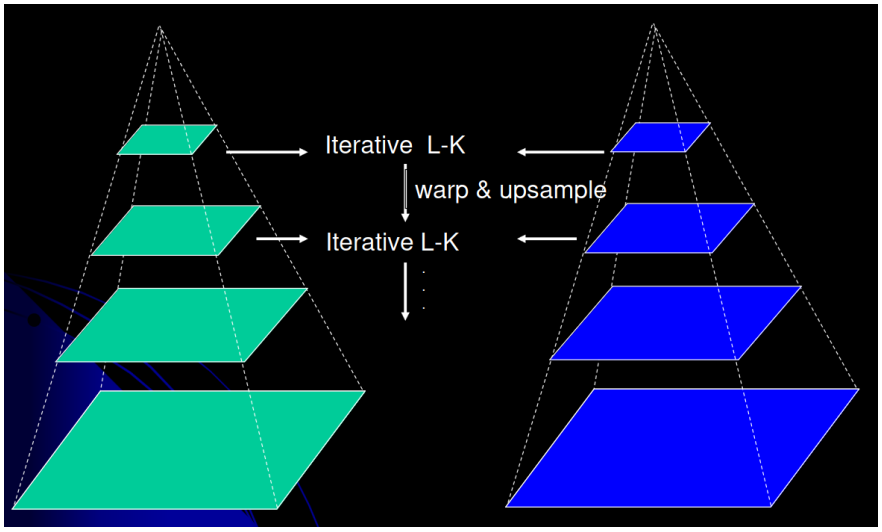
Saliency map je heatmap obrazu, ktorý sa používa na identifikáciu a vizualizáciu oblastí v obraze, ktoré sú vizuálne významné alebo zaujímavé pre ľudské vnímanie. Tieto oblasti majú tendenciu vystupovať z množiny a zaujímať našu pozornosť. Saliency mapy sa často vytvárajú pomocou algoritmov analýzy obrazu a vnímani. Task map je niečo podobné, v zásade je to tiež heatmapa, ktorá vizualizuje dôležité oblastí v obraze v súvislosti s konkrétnou úlohou. Preto závisia od konkrétneho kontextu a aplikácie. Napríklad, ak je úlohou detekcia tváre, task mapa môže zvýrazniť oblasti, kde je pravdepodobné, že sa nachádzajú tváre.

3.17 What is a heatmap, and how do we get them from eyetracker data?

Heatmapa je vizualizácia, ktorá zobrazuje distribúciu fixácií alebo pozornosti na obrazovom priestore. Je to efektívny spôsob vizualizácie správania subjektov získaného pomocou eyetrackingu. Z dát treba identifikovať fixácie a sakády na čo si nastavíme nejaké vhodné prahy. To je super teraz spravím niečo ako gausián okolo každej fixácie to spočítam pre všetkých ľudí ktorých som meral a na konci mi to už stačí len odprahovať a vizualizovať.

3.18 How would you handle calculating optical flow for large motion?

V prípade ak je skok medzi snímkami príliš veľký môže sa stať, že vypočítaný optický tok nedá správne výsledky. Riešením tohoto problému bude pyramídový prístup, kde podsamplovaním sa strácajú najmenšie pohyby a zmenšujú sa tie veľké. Problém vyriešim tak, že budem aplikovať Lucas-Kanade prístup na Gausovsku pyramídu. Optický tok z vyššej vrstvy sa prenáša na nižšiu vrstvu. To znamená, že výsledok výpočtu optického toku z jednej vrstvy sa použije ako počiatočná hodnota pre výpočet optického toku na nasledujúcej vrstve. Optický tok z vyššej vrstvy sa upravuje a zjemňuje podľa zmenšeného rozlíšenia na nižšej vrstve. Keď je výpočet optického toku dokončený pre všetky vrstvy, výsledky sa interpolujú a upravia tak, aby sa získal celkový optický tok pre pôvodné rozlíšenie obrazu.



3.19 What is a difference image and a cumulative difference image?

Rozdielový obraz je obraz vytvorený z absolútneho rozdielu medzi zodpovedajúcimi pixelmi dvoch obrázkov. Zvýrazňuje oblasti, kde sa hodnoty pixelov výrazne líšia, často sa používa na detekciu zmien alebo pohybu medzi dvoma snímkami.
Rozdiel medzi zodpovedajúcimi bodmi:

$$d(x,y) = \begin{cases} 0 & |f(x,y,t) - f(x,y,t+1)| \leq \epsilon \\ 1 & \text{inak} \end{cases}$$

0 = nedošlo k výraznej zmene jasovej úrovne medzi zodpovedajúcimi miestami týchto 2 po sebe idúcich obrazov

Kumulatívny rozdielový obraz je nahromadením rozdielových obrázkov v sekvencii snímok. Predstavuje kumulatívnu zmenu alebo pohyb, ku ktorému došlo v priebehu času. Sčítaním alebo spriemerovaním rozdielových obrazov poskytuje kumulatívny rozdielový obraz komplexnú reprezentáciu celkových zmien alebo pohybov pozorovaných v celej sekvencii, čo umožňuje analýzu dlhodobých trendov alebo vzorov.

3.20 How do we find the background of a scene to determine motion?

Asi najjednoduchší pristup je rozdielový obraz popísaný vyššie. Je to jednoduchá a rýchla metóda na detekciu pohybu, avšak môže byť citlivá na zmeny osvetlenia a šum. Preto sa často kombinuje s ďalšími technikami alebo sa upravuje, aby sa zlepšila jej presnosť a robustnosť.

Keby som chcel niečo presnejšie použijem dynamický kumulatívny rozdielový. To znamená, že referenčný obraz sa bude v čase meniť. Referenčný obraz, môžem spočítať ako:

$$b(x, y, t) = \phi(f(x, y, t - n)_{n=1}^{k})$$

Kde sa najčastejšie za ϕ používa priemer alebo medián. Treba však myslieť na to, že táto metóda je náročná na pamäť.