**3. Detekce hran, Cannyho a Sobelův detektor, prostor měřítek. Matematická morfologie, topologie diskrétního obrazu, diskrétní metriky.**

**Motivace**

Výsledky neurofyziologického a psychologického výzkumu ukazují, že pro zrakové vnímání vyšších organismu jsou

důležitá místa v obraze, kde se náhle mění hodnota jasu (hrany). Tato místa nesou **více** **informace** než jiná místa.

|  |  |
| --- | --- |
| Tato místa chceme  • bud zvýraznit - zvýraznění vysokých kmitočtů operací ostření  • nebo detekovat - detekce hranových bodů  Detekce má časté použití v počítačovém vidění: rozpoznání obsahu obrazu, 3D rekonstrukce scény, problém korespondence, sledování aj. |  |

**Původ hran**

Hrany vznikají díky nespojitostem v normále k povrchu, hloubce, odrazivosti povrchu (barve), odleskům nebo nespojitostech v osvětlení (stínům).

**Poznámka**:

|  |  |
| --- | --- |
| Jinak k hranám je dobrý si uvědomit co je třeba takový **gradient**. Gradient je vektor který má směr, který odpovídá největší změně (třeba jasu). Velikost vektoru odpovídá velikosti změny.  Ty šipky na obrázku jsou gradienty.  Tam kde takový gradient má vysokou hodnotu, tak je hrana. Gradient se dá určit **derivacemi** podle jednotlivých dimenzí.  Gradient spojité funkce n proměnných je vektor parciálních derivací.    Pro **n = 1** (jednorozměrný signál) je roven obyčejné derivaci.  Pro **n = 2** má velikost (nezávisí na natočení souřadné  soustavy) a směr (daný jediným úhlem )    V [souřadnicovém](http://cs.wikipedia.org/wiki/Sou%C5%99adnice) vyjádření je v daném místě gradientem [vektor](http://cs.wikipedia.org/wiki/Vektor), jehož složky tvoří jednotlivé [parciální derivace](http://cs.wikipedia.org/wiki/Parci%C3%A1ln%C3%AD_derivace) funkce vyjadřující dané [skalární pole](http://cs.wikipedia.org/wiki/Skal%C3%A1rn%C3%AD_pole). Pro trojrozměrné pole je gradient:    nebo ve 2D jen:  x, y, z jsou osy ve 3D, dW/dx je derivace podle x. | Lepší 3D reprezentace, protože tam jsou vidět lépe ty **změny** a je vidět, že gradient je **tečnou** k té funkci (k tomu kopci)  nebo má směr tečny, podobně jako derivace pro nějakou funkci..  Gradient na 3D povrchu - červená značí největší růst, modra pomalejší, na vrcholu je růst i gradient **nulový**. |

**Hrana, Hranový bod**

**Hrana** (angl. edge) je dána vlastnostmi obrazového elementu a jeho okolí;

popisuje rychlost změny a směr největšího růstu obrazové funkce f(x, y);

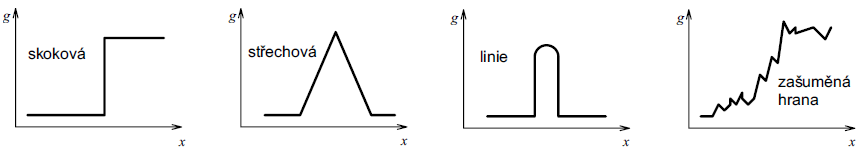
je vhodnou diskrétní aproximací gradientu f(x, y), je tedy vektorem o dvou složkách.

**Hranový** **bod** (angl. edgel = edge element, jako pixel = picture element)

je bod s velkým modulem gradientu;

některé body v obraze jsou tedy hranové a jiné ne.

**TYPICKÉ JASOVÉ PROFILY V OKOLÍ HRANOVÝCH BODU**

****

Příklady jednotlivých profilů

|  |  |  |
| --- | --- | --- |
|  |  | Obrázek s šumem. |

První tři profily zleva, tj. skoková hrana, střechová hrana, tenká linie, jsou idealizované.

Poslední profil odpovídá zašuměné hraně, kterou lze najít v reálném obrázku.

**3 KATEGORIE HRANOVÝCH DETEKTORU**

Detektory založené na

1. hledání maxim prvních derivací (Roberts, Prewittová, Sobel apod., Canny); (hledá se gradient a zjišťuje se, jestli je dostatečně velký)

2. hledání průchodů druhých derivací nulou (Marr-Hildreth); (druhá derivace zjišťuje, kde ten gradient nejméně mění svůj směr a takové místo se označí jako hrana)

3. lokální aproximaci obrazové funkce parametrickým modelem, např. polynomem dvou proměnných (Haralick). (a ten třetí případ, prostě se snažíme sestrojit např. polynom tak, aby odpovídal hranám)

Příklad **Laplace Operator**:

Extrém 1. derivace odpovídá průchodu 2. derivace nulou.

|  |  |
| --- | --- |
|  |  |

Na obrázku vlevo je jas, od černé přechází v bílou. Uprostřed je první derivace, tedy místo s největší změnou má maximum. Mohl bych teď vzít nějaký threshold a říct, že pokud to maximum převyšuje threshold, tak je to hranový pixel anebo můžu udělat druhou derivaci (vpravo) a tam, kde to prochází **nulou (angl. zero-crossings)** bude hrana (druhá derivace = zrychlení).

Když se koukneš na první obrázek, tak jas se zvyšuje se zrychlením a to zrychlení klesá postupně až je nulové a pak naopak se jas zvyšuje zpomaleně až dosáhne bílé.

No takže ten jas pořád se zvyšuje víc a víc a potom až to překročí střed tak se zvyšuje pořád míň a míň až se zvětšovaní jasu zastaví, takže někde je hranice, kdy to zrychlení je nulové.

**DISKRÉTNÍ APROXIMACE DERIVACE**

Dva přístupy aproximace derivací diskrétní funkce, vzniklé vzorkováním spojité funkce (oba vedou k podobným algoritmům):

* rekonstruuj spojitou funkci a spočítej její derivaci;
* aproximuj derivace konečnými diferencemi.

Poznámka: Aproximovat můžeš různými způsoby od toho jsou různé masky konvoluce.

Máš, že v jednom pixelu je 0.3 jas ve druhé 0.8 a co je mezi tím? jak spočítáš derivaci? derivace je definována pro spojité funkce přece, tak ten prostor mezi pixely něčím vyplníš a na tom spočítáš derivaci.

Jednoduchá derivace je odečíst dva sousední pixely. 0.8 - 0.3 = 0.5, takže bude tam nějaká hrana. Když bych měl dva pixely 0.8 a 0.8 vedle sebe, tak rozdíl je 0, takže tam hrana není, ale je to nesymetrické. Tak se dělá symetrická aproximace derivace, kdy odečítám ne sousední pixely ale ob jeden pixel, odečtu f(x+1) - f(x-1) a tuto hodnotu uložím do f'(x). A to je vlastne ta maska Prewittové [-1 0 1]. Tedy -1\*f(x-1) + 0\*f(x) + 1\*f(x+1). Problém u nesymetrické: f'(x)= f(x)- f(x-1). Prostě preferuješ odečítat pixely nalevo a ty napravo neuvažuješ. Taky by to fungovalo i takto:

f'(x)= f(x) - f(x+1). Ale odečítáš od současné pozice tu sousední na některou jednu stranu. Zatímco u symetrické současnou neuvažuješ a odečítáš od sebe levého souseda a pravého.

**CITLIVOST DERIVACE NA ŠUM, PŘED DERIVOVÁNÍM NUTNO VYHLADIT**

Derivace je hodně citlivá na šum, protože derivace = **tečna** k té původní funkci, když původní funkce pořád mění svůj směr, tak derivace pořád skáče, takže se to nejprve rozmazává gaussianem.

Můžeme obě operace rozmazání a derivaci dát do jednoho operátoru (do jedné konvoluční masky), taková maska je např. Sobel.

**VZTAH HRANY A HRANICE**

|  |  |
| --- | --- |
| Nalezené hranové body v obraze lokálními operátory se  někdy používají pro hledání hranic objektu.  Za předpokladu, že objektu odpovídá oblast homogenního jasu, jsou body hranice právě pixely s vysokou hodnotou gradientu.  **Směr hrany je kolmý na směr gradientu.** |  |

**DERIVACE A KONVOLUCNÍ MASKY 3 × 3**

Roberts, jen 2 × 2, Prewittová, Sobel, Robinson, Kirsch, Laplacián (aproximuje 2. všesměrovou derivaci)

**ROBERTSŮV OPERÁTOR V OKOLÍ 2 × 2**

|  |  |
| --- | --- |
| Robertsův odpovídá derivaci podle šikmých směrů.  Je citlivý na šum, protože odečítá blízké pixely (okolí použité pro aproximaci je malé).  Velikost gradientu (délka vektoru) se počítá podle: | Konvoluční masky |

****

**OPERÁTOR PREWITTOVÉ**

|  |  |  |
| --- | --- | --- |
|  |  | **,...** |

Příklad:

|  |  |  |
| --- | --- | --- |
| originál 256 × 256 | prahové hrany v severním směru | východní prahové hrany |

**SOBELŮV OPERÁTOR**

Více preferuje středové body a méně body v okolí, takže je to kombinace gaussianu a derivace, proto na rozdíl od Prewittové není tolik vidět šum.

Sobelův operátor se často používá pro detekci vodorovných a svislých hran, na což postačí masky h1, h3.

|  |  |  |
| --- | --- | --- |
|  |  | ,... |

**ROBINSONŮV OPERÁTOR**

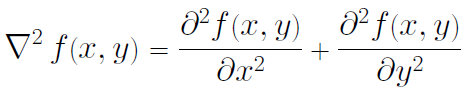
|  |  |  |
| --- | --- | --- |
|  |  | .. |

**KIRSCHŮV OPERÁTOR**

|  |  |  |
| --- | --- | --- |
|  |  | .. |

**LAPLACIÁN OBRAZOVÉ FUNKCE**

Laplacián je operátor druhých derivací, předtím ty operátory jsi počítal v obou směrech x a y, laplacián počítáš pouze jednou a už ti nedává směr, ale pouze velikost gradientu.

, **** je skalár, oproti gradientu přicházíme tedy o směr hrany.

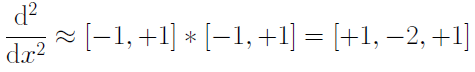
Podobně jako velikost gradientu , také  je invariantní vůči natočení souřadné soustavy.

Pro monotónně rostoucí jasovou funkci **f(x, y)** v příslušném okolí je Laplacián **nulový** tam, kde je velikost gradientu

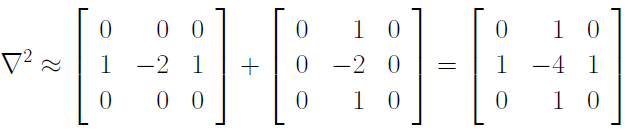
 maximální -> průchody **nulou** (angl. zero-crossings).

**DISKRÉTNÍ APROXIMACE LAPLACIÁNU**

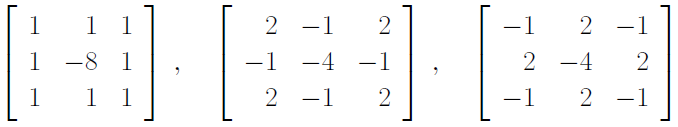
Diskrétní druhá derivace je složením (konvolucí) prvních derivací:



Diskrétní Laplacián je součtem druhých parciálních derivací:



Alternativní používané tvary (8-okolí, zvýraznění středu):



**OSTŘENÍ LAPLACIÁNEM**

Někdy nechceme detekovat hranové body, ale pouze zvýraznit hrany (ostření). Laplacián je vhodný, neboť zdůrazňuje vysoké frekvence (srov. DoG).

**Srovnání:**

|  |  |
| --- | --- |
| Roberts operator | Prewitt edge detector |
| Sobel edge detector | Robinson compass operator |
| Kirsch compass operator | Laplacian |

**Poznámka:** jakože tenčí čáry? Protože na obrázku nahoře uprostřed u Laplace příklad, když jsem si zvolil nějakej **threshold**, tak se mi klidně mohlo stát, že nad tím thresholdem bude nejen ten pixel s maximální derivací, ale i blízké pixely kolem něj jsou větší než threshold, ale u laplaciánu (obrázek nahoře vpravo) to projde nulou právě v jednom pixelu.

**CANNYho HRANOVÝ DETEKTOR**

V jistém smyslu završení období hledání **nejlepšího** hranového detektoru. Používán pro většinu aplikací.

**Algoritmus**:

1. Najdi přibližně směry gradientu.

2. Pro každý pixel najdi 1D derivaci ve směru gradientu pomocí optimální masky spojující vyhlazení a derivaci.

3. Najdi lokální maxima těchto derivací.

4. Hranové body získej prahováním s hysterezí.

5. Proved syntézu hran získaných pro různě velká vyhlazení (málokdy se používá).

Takže nejprve rozmažeme gaussianem, pak vypočteme gradienty.

|  |  |  |
| --- | --- | --- |
| magnitudy gradientů (velikosti šipek)  čím výraznější přechod tím větší gradient | směry gradientů (stejná barva, stejný směr) černá je třeba 0°, šedá je třeba 180° | z toho prvního obrázku s magnitudami vybrali přes threshold jen dostatečné silné čary a zjistili jejich směr a ten odlišili barevně |
| pak obarví ty hrany jen podle 4 směrů | posčítají všechny magnitudy v tom dominantním směru | pak se dělá **hystereze**..přes threshold Th (vyšší) se najdou takové body, které mají dostatečně velkou odezvu a pak jakoby jedeš po těch čarách (viz. obr) dokud platí, že odezva je větší než druhý threshold Tl  Takže ty body, kde je odezva < Tlow (Tl) jsou automaticky vyřazeny.  Body, které mají odezvu > Thigh (Th) jsou automaticky hranové body.  A teď co s těmi ostatními v intervalu mezi Tlow a Thigh?  Pokud ty hranové body sousedí a dotýkají se Thigh tak budou taky hrany.  Pokud některé hranové body tvoří křivku, ve které není přítomen žádný hranový bod vyšší jak Thigh, tak se považuje za šum a zahodí se. |

**Zhrnuto**:

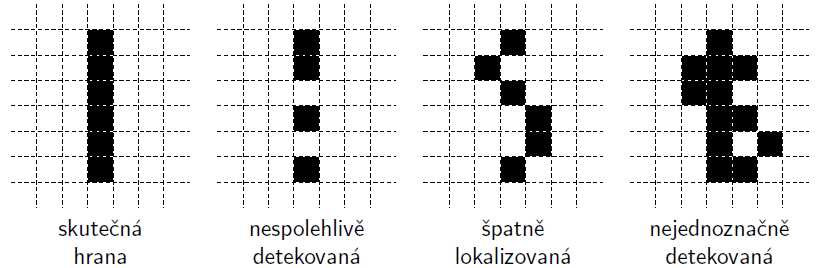
|  |  |
| --- | --- |
| 1) rozmažeme a vypočteme gradient (**velikost** - magnitudu a **směr** gradientu)  2) v každém bodě (pixelu) vezmeme okénko a vypočteme **histogram** orientací do 4 směrů (nahoru-dolů, zleva-doprava, šikmo 1, šikmo 2) a vybereme z toho histogramu dominantní směr a známe součet všech těch magnitud v daném směru - to je ta odezva se kterou dále pracujeme  [pro každý bod okénko a v něm jeden histogram se 4 směry, do každého směru (do každé přihrádky histogramu) nasčitáváš magnitudy]  3) provádí se **hystereze** - najdeme body s odezvou > Thigh a ty považujeme za hranové body odstraníme body s odezvou < Tlow a potom iterativně procházíme od hranových bodů a hledáme, které body z těch zbylých k nim můžeme připojit.. Ve výsledku připojíme ty body, které se nějak dotýkají hranových a ty, které se nedotýkají hranových odstraníme. | Výsledek: |

**OPTIMÁLNÍ LINEÁRNÍ OPERÁTOR PRO DETEKCI HRAN**

Předpokládán zjednodušený model: ideální schodová hrana; aditivní gaussovský a v každém pixelu stejný a nezávislý šum.

Požadavky, které chceme maximalizovat :

* spolehlivá detekce (nalezeno co nejvíce existujících hran);
* dobrá lokalizace (malá chyba detekované pozice hrany);
* jednoznacná odezva (nalezeno co nejméne neexistujících hran).



**NALEZENÍ MAXIM PRVNÍ DERIVACE V 1D**

Místo prahování se hledá lokální maximum, protože tomu prahování může odpovídat více bodů, ale maximum je jen jedno a může se hledat maximum i subpexilově proložením paraboly

|  |  |
| --- | --- |
| Prahování nevhodné (ledaže maxima jsou velmi ostrá) | Správné je hledat lokální maxima derivace (non-maxima suppression) |

Umožňuje subpixelovou (tj. lepší než celočíselnou) přesnost nalezení maxima: např. proložíme parabolu a analyticky spočítáme maximum.

**NALEZENÍ HRANOVÝCH BODŮ PRAHOVÁNÍM S HYSTEREZÍ**

|  |  |
| --- | --- |
| To je ta hystereze u canny edge detektoru.  Chceme potlačit krátké (tj. typicky nevýznamné) řetězy hranových  bodů, ale přitom zabránit fragmentaci dlouhých řetězů.  Tučné jsou ty, které mají odezvu vyšší než Thigh, tenké jsou ty, které mají větší než Tlow. Ty čáry uvnitř se odstraní protože nesousedí s žádným hranovým bodem, který je větší než Thigh, ta tenká čára na okraji se stane taky hranou, protože sousedí s body, které mají odezvu větší než Thigh. |  |

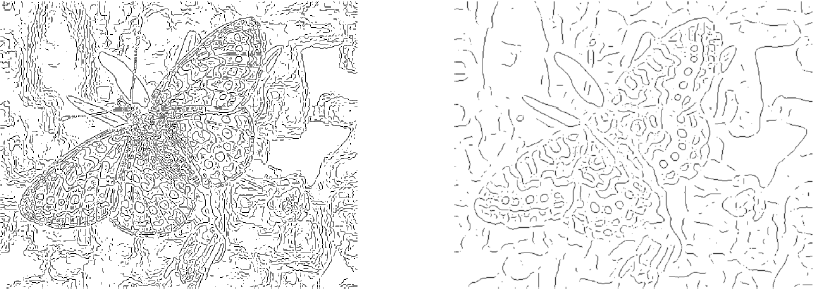
**PROBLÉM VOLBY MĚRÍTKA VYHLAZENÍ**

Jaké zvolit  Gaussiánu při počítání derivací? Čím větší , tím ...

* lepší potlačení šumu,
* více slabých hran zanikne,
* menší přesnost lokalizace hran.

Problém není omezen na (Cannyho) detekci hran. Je to obecný problém při detekci primitiv (lokálních vlastností)

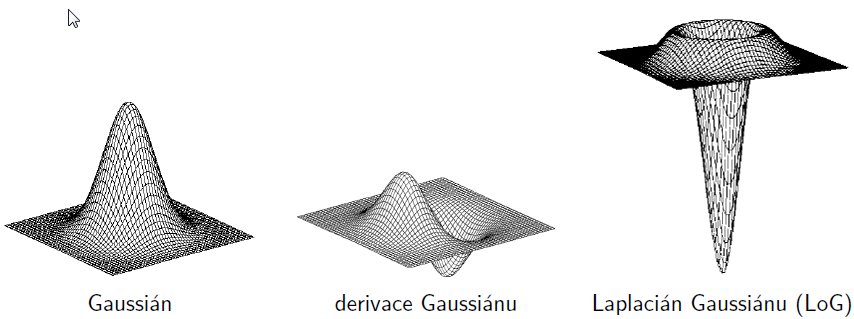
v obrazech (např. významných bodů). Často nás nezajímají detaily, i když nevznikly díky šumu. Jako bychom se chtěli podívat na obraz z ‘větší dálky’ a detekovat jen významnější hrany (či jiná primitiva).



Když budou dvě hrany blízko sebe tak s větším sigma (větším rozmazáním) se ztratí detaily a uvidím jen jednu hranu.

Laplacián je ještě citlivější na šum než gradient -> opět kombinujeme s vyhlazením Gaussiánem. Oba operátory lze spojit do jediného, označovaného jako **LoG** (Laplacian of Gaussian).

**2D OPERÁTORY, S NIMIŽ JSME SE SETKALI**



**DoG JAKO APROXIMACE LoG**

Aproximovat lze pomocí diference (rozdílu) dvou obrazů, které vznikly konvolucí s Gaussiánem o různém sigma.

**JAK POČÍTAT PRŮCHODY NULOU?**

Při implementaci detektoru založeného na hledání průchodů nulou je třeba se vyhnout naivnímu řešení pomocí

prahování LoG obrazu v intervalu hodnot blízkých k nule. Výsledkem by byly hodně nespojité hrany.

Lepší je použít opravdový detektor průchodů nulou, např. v masce 2 × 2 s reprezentativním bodem třeba v levém

horním rohu. Hrana se zde indikuje tehdy, pokud se uvnitř okna opravdu mění znaménko.

**Nevýhoda**: Příliš vyhlazeny ostré tvary. Například ostré rohy se ztrácejí. Snaží se spojovat hrany do uzavřených křivek. “Talíř špaget”.

**Matematická morfologie**

**MORFOLOGIE**

V **biologii**: studium velikosti, tvaru a vnitrní struktury zvířat, rostlin a mikroorganismu a hledání souvislostí mezi jejich

vnitřními částmi.

V **digitálním** **zpracování** **obrazu**: matematický nástroj pro předzpracování i segmentaci obrazu.

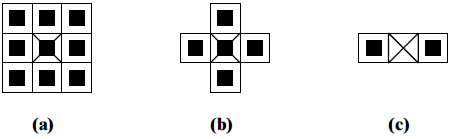
**BODOVÁ MNOŽINA**

|  |  |
| --- | --- |
| Obrázky lze modelovat pomocí *bodových množin* libovolné dimenze (např. *N*-rozměrný euklidovský prostor).  2D euklidovský prostor a systém jeho podmnožin je přirozeným definičním oborem pro popis rovinných útvarů.  Digitální protějšek euklidovského prostoru.  Binární matematická morfologie – množina dvojic celých čísel ().  Šedotónová matematická morfologie – trojice (). |  |

**MORFOLOGICKÁ TRANSFORMACE **

Je dána relací mezi ***obrazem***(bodovou množinou *X*) a typicky menší bodovou množinou ***strukturním elementem*** *B*.

Strukturní element *B* je vztažen k “lokálnímu” počátku.

****

**Aplikace morfologické transfomace**  na obraz *X* odpovídá systematickému posunu *B* po obraze. Výsledek transformace v každé poloze odpovídá relaci.

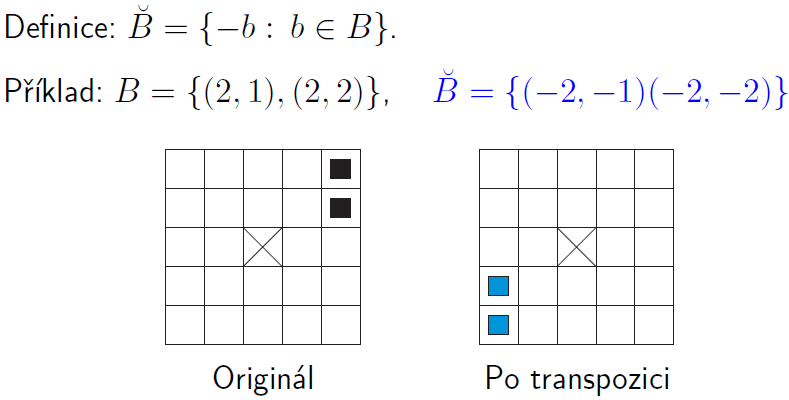
**TRANSLACE MNOŽINY O RADIUSVEKTOR**

***Translace Xh***bodové množiny *X* o radiusvektor *h*



**SYMETRICKÁ BODOVÁ MNOŽINA**

Vůči reprezentativnímu bodu *O*. Někdy se také říká transponovaná bodová množina.



**BINÁRNÍ MATEMATICKÁ MORFOLOGIE**

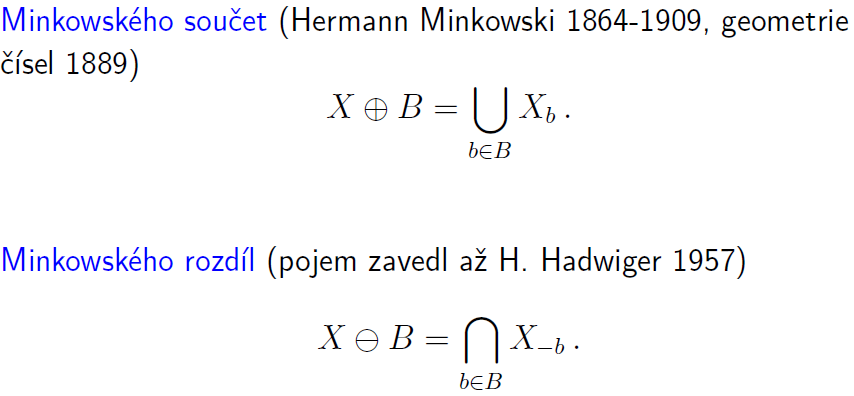
Pracuje s binárními obrázky. Definiční obor . Obor hodnot *{* 0*,* 1*}*.

2 základní operace: **dilatace** a **eroze**. Nejsou invertovatelné.

2 používané formalismy pro součet a rozdíl

* Ve školské aritmetice obvyklé sčítání a odečítání.
* Minkowského součet, rozdíl
* Rozdílnost obou přístupů hraje roli u eroze.

**MINKOWSKÉHO SOUČET, ROZDÍL**

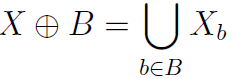


**BINÁRNÍ DILATACE ****

Sčítá dvě bodové množiny.



Dilataci můžeme vyjádřit jako sjednocení posunutých bodových množin

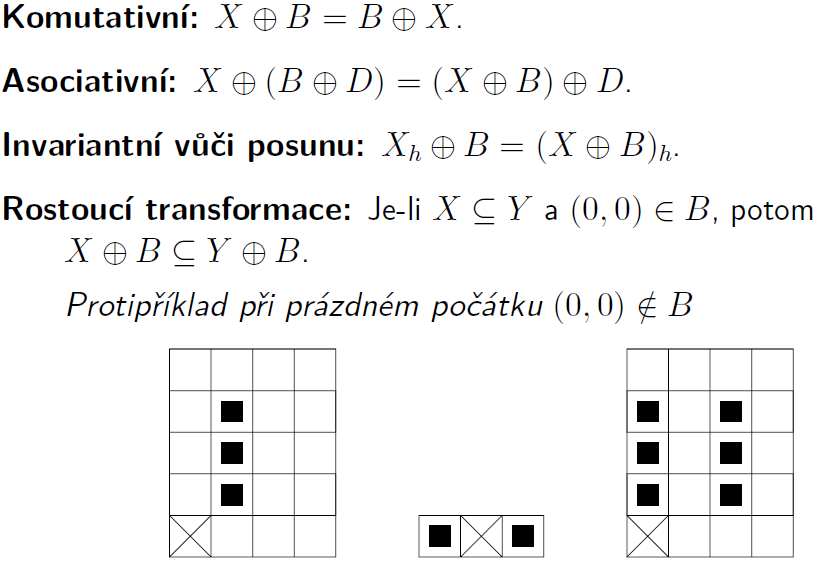
 pro každý puntík v B sjednotím

|  |  |
| --- | --- |
| B jsou 2 puntíky, tak vezmu první ten má souřadnice (0,0)  posunu X o (0,0) takže nechám tam kde je a uložím do nové matice, pak vezmu druhý puntík v B (1,0), posunu X o (1,0) a doplním do té výsledné matice a konec.  Takže sjednocuju přes translace pro všechny puntíky v B. |  |

**BINÁRNÍ DILATACE ISOTROPICKÝM STRUKTURNÍM ELEMENTEM 3*×* 3**

|  |  |
| --- | --- |
| Dilatace se používá k zaplnění malých děr a úzkých zálivů  v objektech. Zvětší původní velikost objektu. Má-li být velikost zachována, potom se použije dilatace s erozí. |  |

**VLASTNOSTI DILATACE**



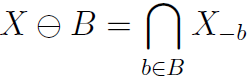
**BINÁRNÍ EROZE **

Skládá dvě množiny pomocí Minkovského rozdílu. Jde o duální morfologickou transformaci k dilataci.



Pro každý bod obrazu *p* se ověřuje, zda pro všechna možná *p* + *b* leží výsledek v *X*. Pokud ano, je výsledek 1, jinak 0.

Erozi můžeme vyjádřit jako průnik všech posunů obrazu *X* o vektory 

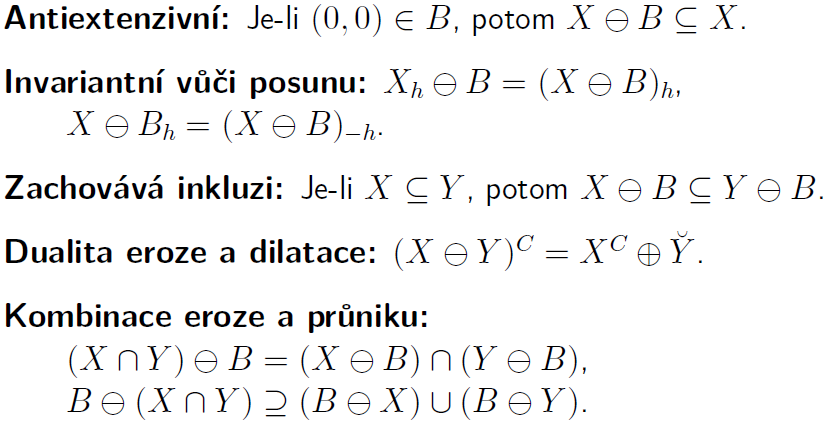


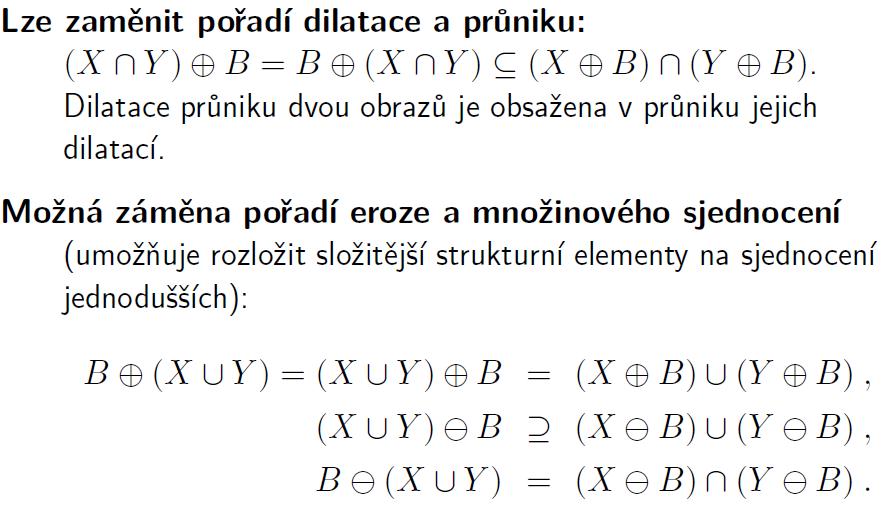
|  |  |
| --- | --- |
| vezmu první puntík z B (0,0)  X posunu o -(0,0), takže nechám tam kde je a překreslím (tužkou, abych mohl pak některé puntíky vygumovat) do nové matice  pak vezmu druhý puntík z B (1,0)  X posunu o -(1,0) = (-1,0) a zase vkreslím do nové matice, místa, kde se to **překrylo** **nechám**, tam kde se to **nepřekrylo** **vygumuju** puntíky |  |

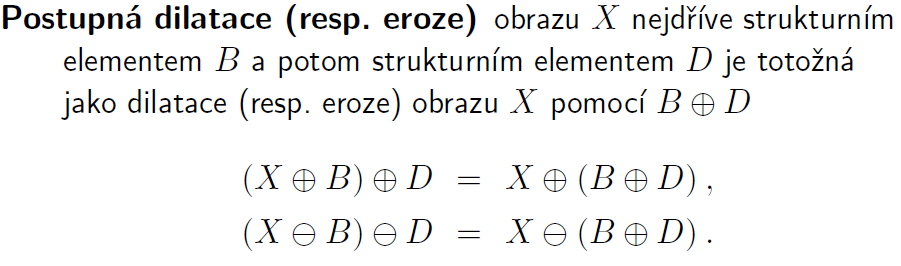
**BINÁRNÍ EROZE ISOTROPICKÝM STRUKTURNÍM ELEMENTEM 3*×* 3**

|  |  |
| --- | --- |
| Objekty menší než strukturní element vymizí (např. čáry tlouštky 1).  Eroze se používá ke zjednodušení struktury (rozložení objektu na jednodušší části). |  |

**VLASTNOSTI EROZE**







**BINÁRNÍ OTEVŘENÍ **

Eroze následovaná dilatací. 

Pokud se obraz *X* nezmění po otevření strukturním elementem *B*, říkáme, že je otevřený vzhledem k *B*.

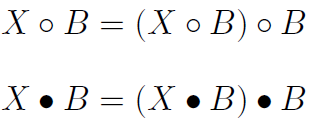
**BINÁRNÍ UZAVŘENÍ **

Dilatace následovaná erozí. 

Pokud se obraz *X* nezmění po uzavření strukturním elementem *B*, říkáme, že je uzavřený vzhledem k *B*.

Otevření a uzavření jsou **duální** morfologické transformace 

**Idempotence** – důležitá vlastnost v matematice. Zde: po jednom otevření, resp. uzavření, je množina již otevřena, resp. uzavřena. Další použití těchto transformací již nic nezmění.

****

**HOMOTOPICKÉ TRANSFORMACE**

Opírají se o souvislost v obraze. Homotopické transformace nemění homotopický strom.

**Příklad**: dvěma různým obrázkům odpovídá stejný homotopický strom.

|  |  |
| --- | --- |
|  |  |

**KOSTRA (skelet) POMOCÍ MAXIMÁLNÍCH KRUHŮ**

Podlouhlé objekty má smysl reprezentovat kostrou (viz animace člověka v počítačové grafice zachycující kinematiku).

Blum v roce 1964 navrhl “Medial axis transformation” (analogie, vypalování trávy).

Formální definice kostry se opírá o pojem maximálního **kruhu** (**koule** ve 3D).

|  |  |
| --- | --- |
| **Maximální kruh** *B* vepsaný do množiny *X* se dotýká hranice ve dvou a více bodech.  **Kostra** je sjednocením středů maximálních kruhů. |  |

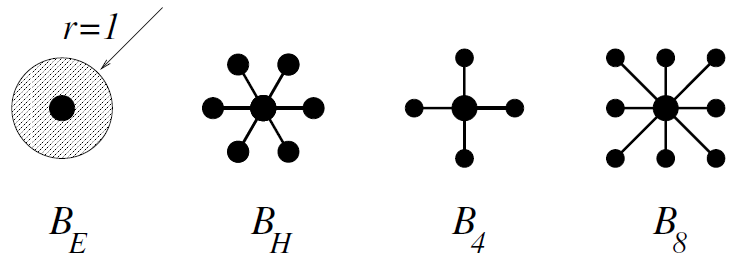
Příklady koster a ukázka problému se šumem (vpravo).

|  |  |  |
| --- | --- | --- |
|  |  |  |

**DISKRETNÍ KRUHY O POLOMĚRU 1**

V diskrétním rastru mohou kruhy vypadat různě, a to díky několika možným způsobům zavedení vzdálenosti.

Příklady:



**TŘÍDĚNÍ ALGORITMU BINÁRNÍ SKELETONIZACE OBLASTÍ**

**Vpisování kruhů** podle definice se prakticky nepoužívá. Přílišná výpočetní složitost. Porušuje se souvislost. Skelet tloušťky *>* 1.

**Sekvenční ztenčování**. Oblast se eroduje vhodným strukturním elementem, který zaručí, aby nebyla porušena souvislost. Obvykle homotopické ztenčování, s využitím strukturních elementů z Golayovy abecedy.

**Přes vzdálenostní transformaci**. Rychlý výpočet. Nejčastěji používané.

**ZTENČOVÁNÍ A ZTLUŠŤOVÁNÍ**

Nechť *X* je obraz a *B* = (*B*1*,B*2) je složený strukturní element zavedený v transformaci tref či min.

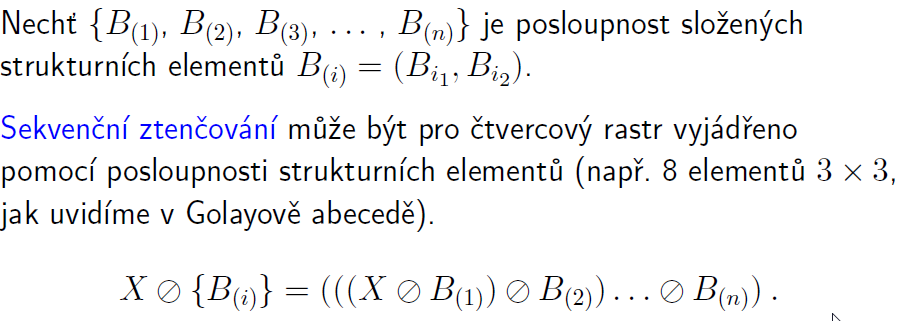
 

Část **ztenčované** oblasti určená strukturním elementem *B* množinově odečítá od objektu samého.

Oblast se sjednocuje s částí pozadí danou strukturním elementem *B* u **ztlušťování**.

Ztenčování a ztlušťování jsou **duální** transformace.

**SEKVENČNÍ ZTENČOVÁNÍ A ZTLUŠŤOVÁNÍ**

****

**UŽITEČNÉ SEKVENCE Z GOLAYOVY ABECEDY**

Existuje několik posloupností strukturních elementů , které jsou z praktického pohledu užitečné.

Ukažme jen dvě z nich z **Golayovy abecedy** (1969) pro oktagonální rastr. Strukturní elementy rozměru 3 *×* 3 uvedeme ve dvou základních polohách, ostatní si domyslete pootočením.

Stručný zápis složeného strukturního elementu: **1** ověřuje příslušnost k objektu, **0** ověřuje příslušnost k pozadí a konečně hodnota ***\**** znamená, že prvek nehraje roli.

Ztenčování a ztlušťování prvky Golayovy abecedy je **idempotentní**.

**ZTENČOVÁNÍ ELEMENTEM L, HOMOTOPICKÁ NÁHRADA SKELETU TL. 1**

|  |  |
| --- | --- |
|  |  |



**OŘEZÁNÍ VOLNÝCH KONCŮ ELEMENTEM E**

|  |  |
| --- | --- |
|  |  |

Pokud by se ztenčování nechalo běžet až do dosažení **idempotence**, zůstaly by v obraze pouze uzavřené linie.

**MOTIVACE PRO REKURZIVNÍ MORFOLOGII Vzdálenostní transformace**

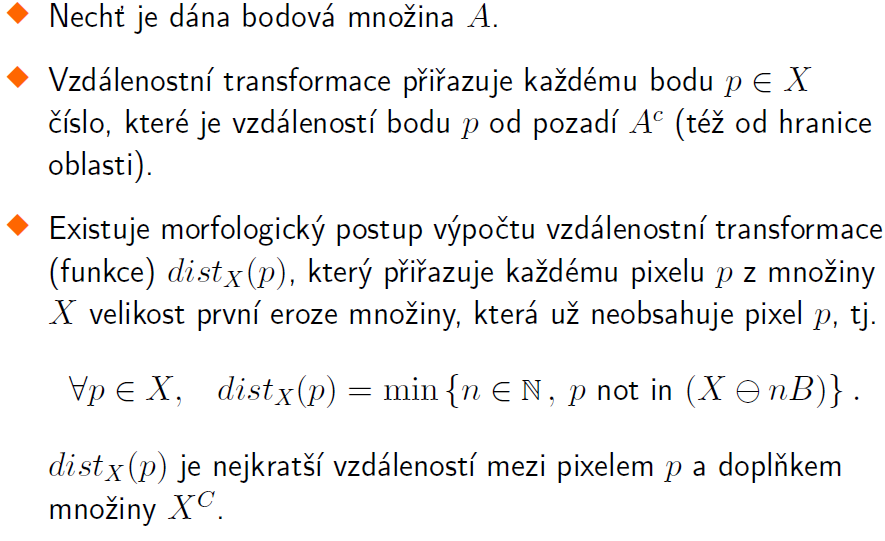
Dosud nezáleželo na pořadí, v jakém byly použity morfologické transformace v různých místech obrazu. Mohly být použity v náhodném pořadí, po řádcích, paralelně.

Speciálnější případ, kdy je vhodně předepsáno pořadí operací v obrazu, přináší zrychlení výpočtu. Výsledek operace totiž bude záviset nejen na vstupním obraze a transformaci, ale na předchozích výsledcích.

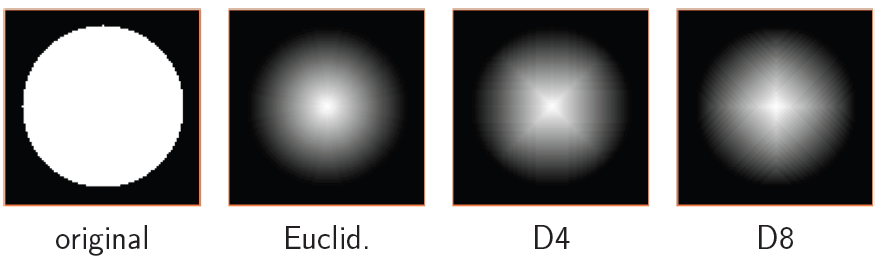
Tím se může při výpočtu akumulovat potřebná globální informace, a tak lze algoritmy výpočetně zjednodušit.

**Vzdálenostní transformace** a algoritmus jejího výpočtu je jedním důležitým příkladem tohoto přístupu.

**VZDÁLENOSTNÍ TRANSFORMACE**



**VZDÁLENOSTNÍ TX A METRIKY**



**Litaratura**:

http://suraj.lums.edu.pk/~cs436a02/CannyImplementation.htm