Segmentace obrazu – Houghova transformace a plošné segmentace

Houghova transformace

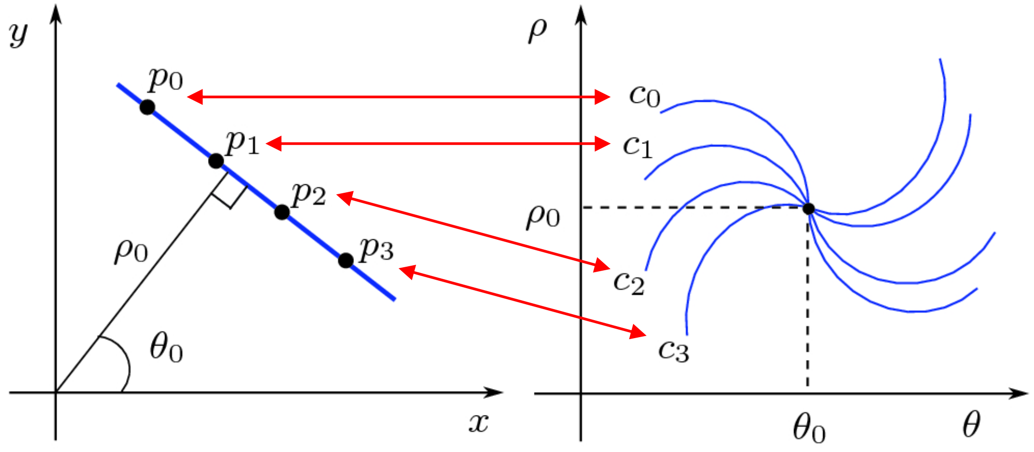
- HT slouží pro detekci tvaru (hranice, kontury) objektu v obrázku.
- Je zapotřebí znát parametrický tvar rovnice, která popisuje konturu objektu nebo jeho části.
- Pomocí HT se nejčastěji detekují křivky jako jsou přímky, kružnice, elipsy apod. – (**klasická**) HT.
- Pokud neexistuje analytický popis tvaru objektu, používá se **zobecněná** HT.

Houghova transformace

- Klasická HT je výpočetně méně náročná, než zobecněná HT.
- Klasická HT je, přes svoje omezení, co se týče požadavku na analytický parametrický popis kontury objektu, plně dostačující pro většinu aplikací.
- Slovo klasická se nepoužívá a mluví se jen o HT.
- Výhodou HT je, že je relativně odolná vůči porušení (chybějícím částem) kontury objektu a vůči šumu.

Houghova transformace

- Princip HT
  - Zvolíme vhodný tvar, kterým by šla popsat kontura objektu nebo její část – např. přímka, kružnice apod.
  - Vyjádříme tento zvolený tvar, kterým budeme popisovat konturu objektu, pomocí parametrické rovnice.
  - Projdeme každý bod kontury objektu a najdeme všechny parametry rovnice tak, aby tvar popsaný touto rovnicí a s těmito parametry procházel daným bodem.
  - V prostoru parametrů, tzv. akumulátoru, kde jednotlivé osy reprezentují příslušné parametry, na místě určeném konkrétními hodnotami parametrů daného bodu zvýšíme hodnotu (např. o 1), přičemž na začátku obsahoval akumulátor samé 0.



Houghova transformace

- Princip HT
  - Poté, co jsme zpracovali všechny body kontury objektu a s jejich pomocí sestavili akumulátor (můžeme ho vizualizovat jako jasový obrázek) najdeme lokální maxima v tomto akumulátoru.
  - Body, které sledují předepsaný tvar (např. pro zvolený tvar přímky, body ležící v obrázku na přímce) se budou v prostoru parametrů akumulovat do stejného místa a tím budou jasnější (budou vytvářet maxima).
  - Každému maximu v akumulátoru odpovídají konkrétní hodnoty parametrů rovnice tvaru kontury (nebo její části) a po jejich dosazení do rovnice tvaru získáme analytický popis tvaru (části) kontury.

Houghova transformace

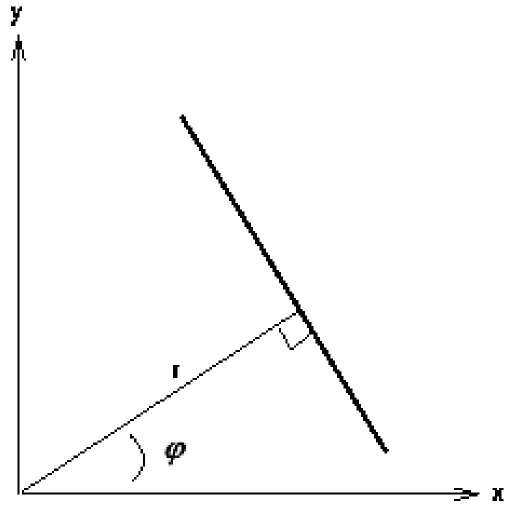
- HT pro **přímkovou** konturu objektu
  - Předpokládejme, že máme obrázek, který obsahuje „hranaté“ objekty, tj. jejich kontura je složena z přímkových úseků.
  - Vhodný analytickým popisem kontury bude tedy přímka. Rovnici přímky musíme vyjádřit v parametrické podobě. Tvar $y = k \cdot x + q$ (s parametry k, q) není vhodný, protože směrnice k jde k nekonečnu pro svislé přímky).

Houghova transformace

- Vhodnou parametrickou rovnicí přímky je tvar

$$x \cdot \cos\varphi + y \cdot \sin\varphi = r$$

- kde r je délka normály k přímce procházejí daným bodem a φ je úhel této normály od osy x.
- Pro každý bod (x, y), který leží na přímce jsou parametry r a φ konstantní.



Houghova transformace

- Postupně procházíme všechny body (x_i, y_i) kontury objektu a hledáme parametry r a φ v uvedené rovnici.
- Protože hledáme **dva** parametry, ale máme jen **jednu** rovnici není úloha jednoznačná a dostáváme nekonečně mnoho řešení. Každému bodu v obrázku tak odpovídá celá křivka.
- Bodům, které leží na přímce, budou odpovídat různé křivky, které se ale protínají v jednom bodě (lokální **maxima** v akumulátoru). Tím získáme potřebnou informaci, aby řešení bylo **jednoznačné** (získali jsme právě jednu rovnici).

Houghova transformace

- V praxi probíhá hledání vhodných parametrů následovně:
  - Zvolíme jemnost dělení úhlu φ (velikost kroku) a postupně procházíme všechny úhly od 0 do 360° s daným krokem (velikost kroku bude ovlivňovat přesnost proložení kontury hledanou přímkou – pozice, úhel) a
  - pro každý zvolený úhel dopočteme podle uvedené rovnice délku normály r.
  - Pro každý bod tak získáme soubor úhlů φ a jim odpovídajících délek r – dvojice (φ, r).
  - V parametrickém prostoru (akumulátoru) zvýšíme hodnotu (např. o 1) na místě daném příslušnou dvojicí (φ, r).
  - Každému bodu kontury objektu bude odpovídat v parametrickém prostoru (akumulátoru) jedna křivka (má sinusový charakter).
  - Výsledný akumulátor můžeme vizualizovat jako jasový obrázek.
  - Nalezením (pozic) maxim v akumulátoru najdeme příslušnou dvojici parametrů (φ, r), které po dosazení do výše uvedeného vztahu jednoznačně určí analytický popis kontury (její části).

  - ***Pozn.**: Kontura je v metodě HT popsána přímkou, přičemž hledáme popis kontury pomocí úseček. Proto je třeba následně aplikovat další algoritmy, které omezí přímku na hledanou úsečku.*

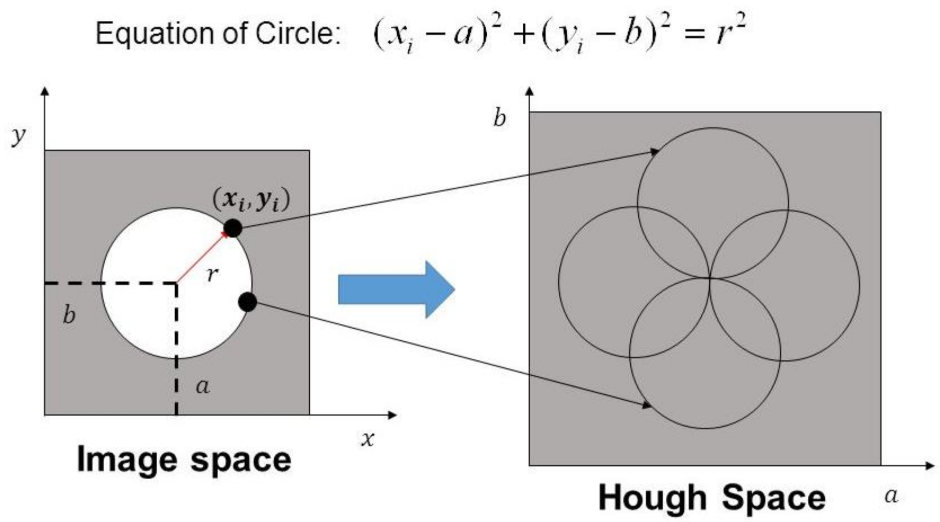
Houghova transformace

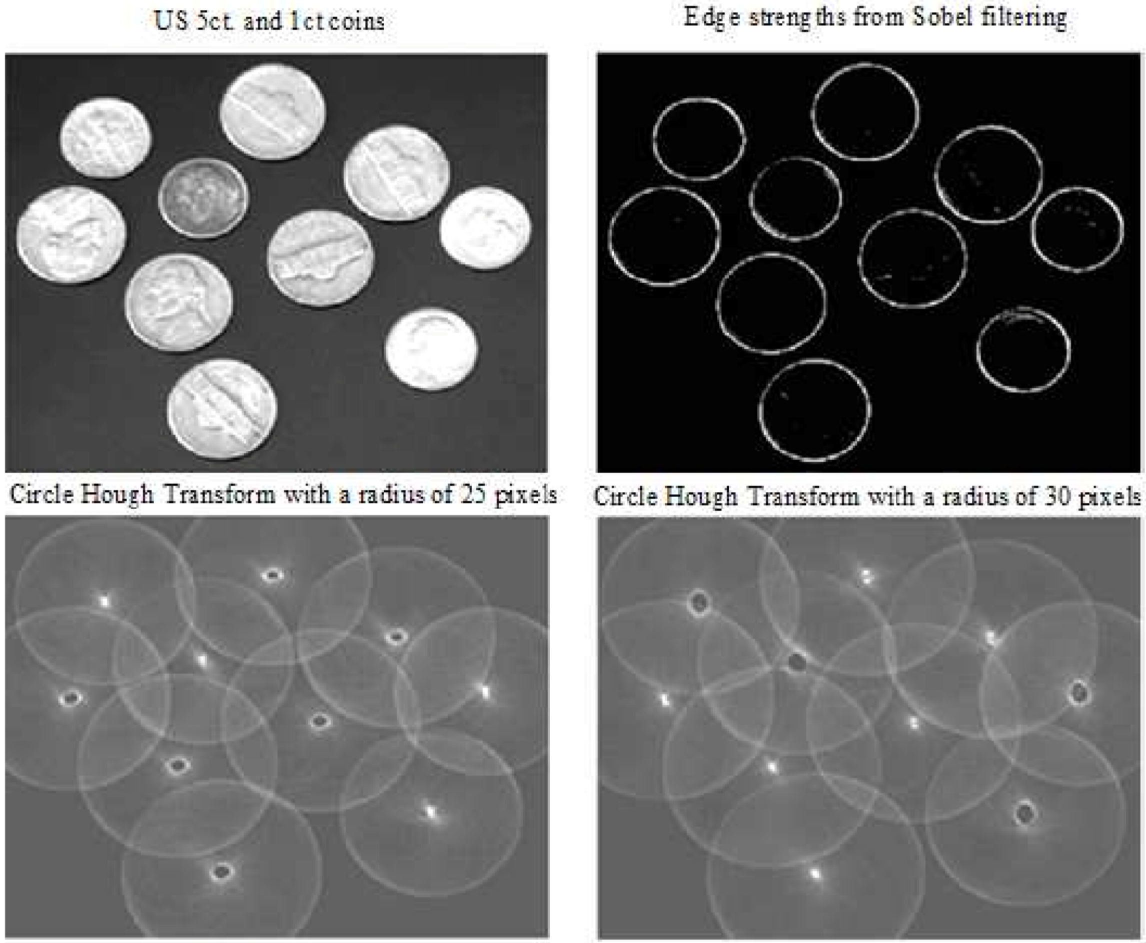
- Pokud má kontura objektu jiný tvar, je zapotřebí použít jinou analytickou parametrickou rovnici, která její tvar lépe vystihuje.
- Např. pokud bychom měli v obrázku kruhové objekty, pak příslušná parametrická rovnice by měla tvar

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

- kde a, b jsou souřadnice středu kružnice a r je poloměr kružnice.
- V této rovnici jsou už **tři** parametry a tedy parametrický prostor (akumulátor) je třídimenzionální.
- Stoupá také doba potřebná pro výpočet.
- Klasická HT se hodí pro objekty, které lze snadno popsat jednoduchými parametrickými rovnicemi s málo parametry.
- Pro složitější kontury objektů se používá zobecněná HT.

Houghova transformace

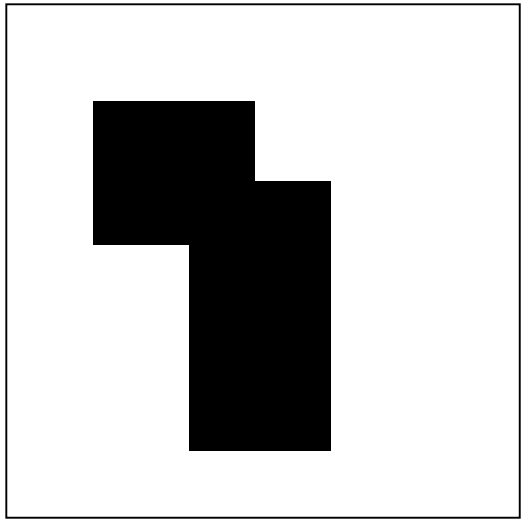


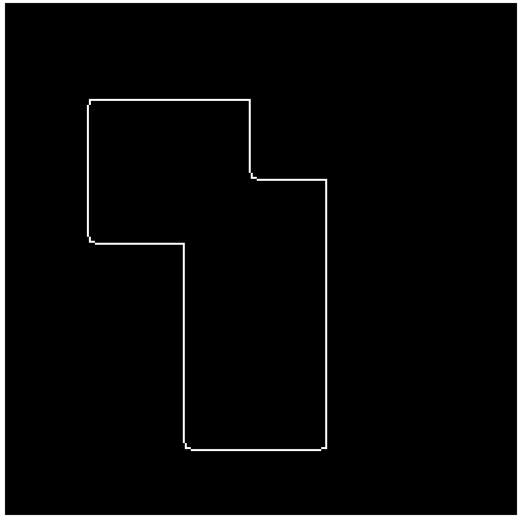


Pokud je poloměr kružnice konstantní,
pak stačí jen 2D akumulátor (a,b)

Houghova transformace

- Příklad:
  - Mějme následující obrázek a najděme analytický popis jeho kontury.





Po aplikaci Cannyho detektoru dostaneme konturu objektu

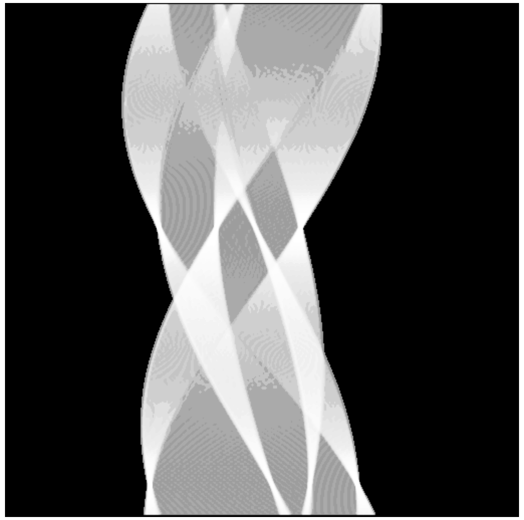
Houghova transformace

- Po zpracování všech bodů kontury (nalezení souborů dvojic parametrů (φ, r) dostaneme výsledný akumulátor (zobrazený jako jasová bitmapa)

ϕ $\uparrow$

$\rightarrow r$

ϕ $\uparrow$

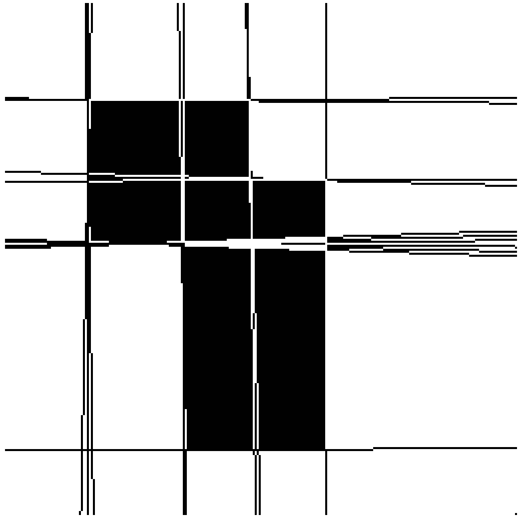


$\rightarrow r$

Pouze jasové zvýraznění – ekvalizace histogramu

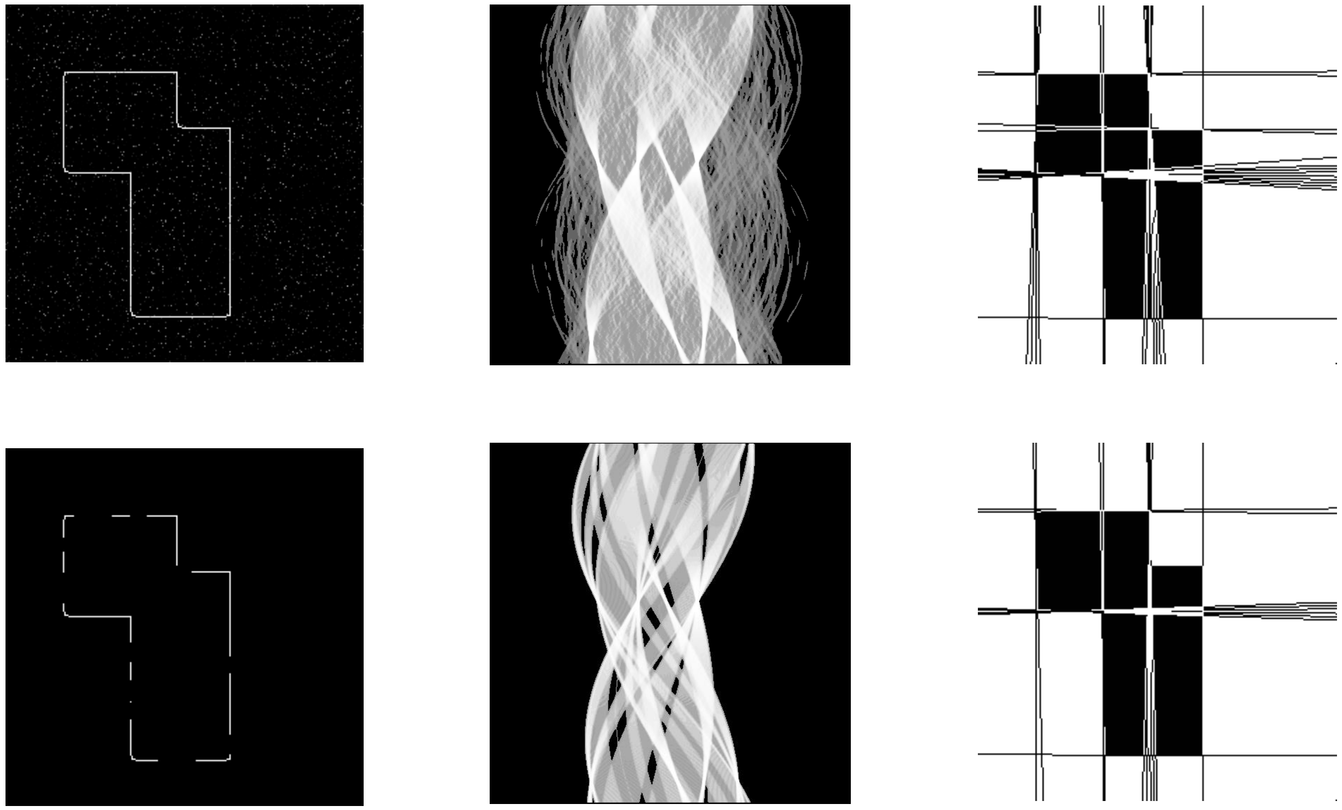
Houghova transformace

- Najdeme maxima (světlá místa) v akumulátoru. Každému maximu odpovídá jedna dvojice (φ, r), tedy jedna přímka.
- Promítnutí nalezených přímek do původního obrázku:



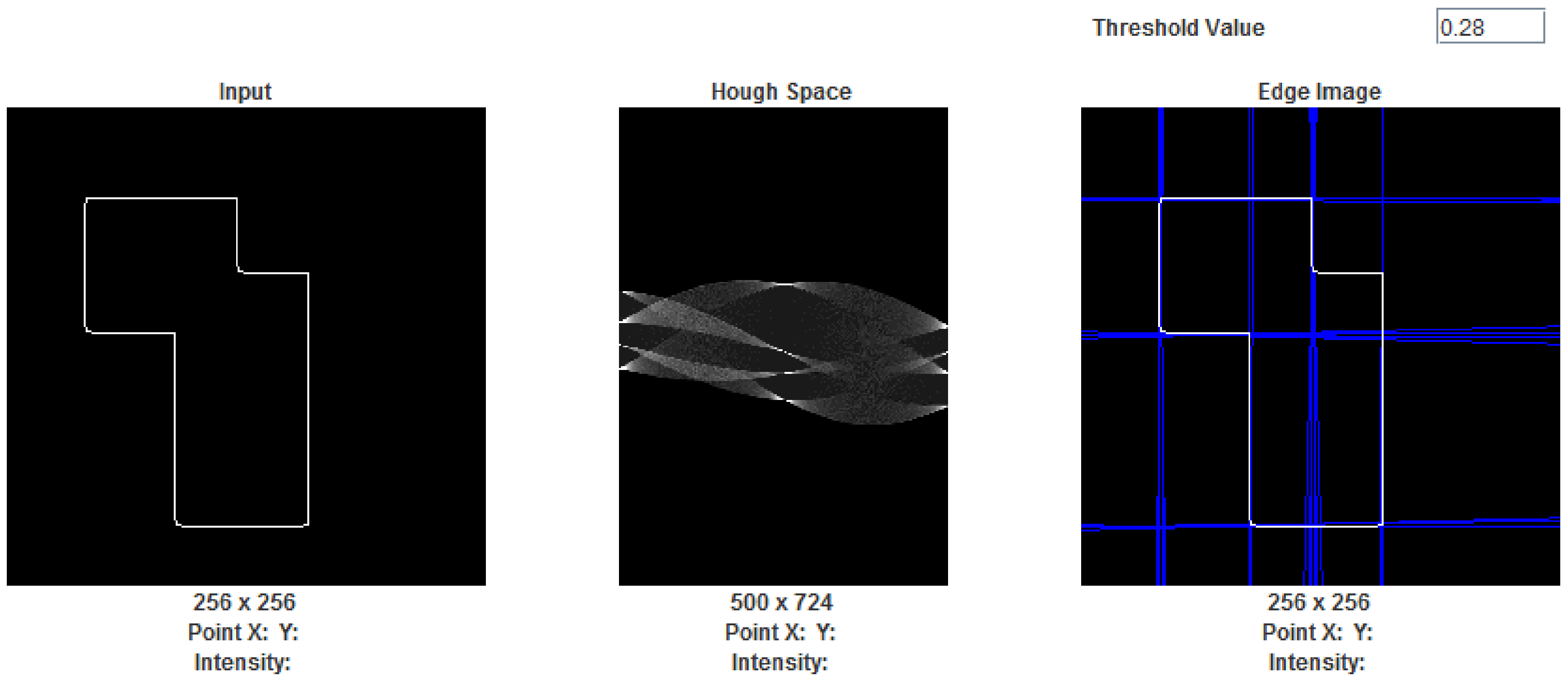
Houghova transformace

- HT je relativně odolná proti šumu i proti chybějícím částem kontury:



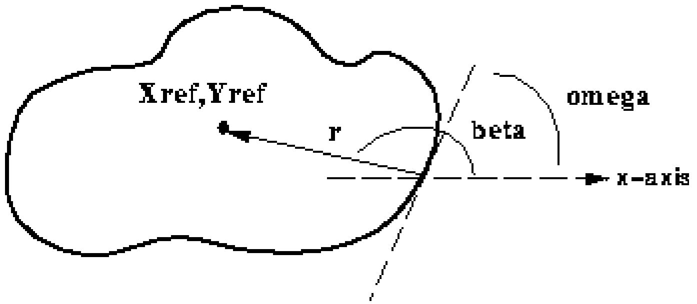
Houghova transformace

- Ukázka z Java apletu:



Zobecněná Houghova transformace

- Pokud nemáme analytický popis kontury, popíšeme konturu tabulkou bodů kontury – seznam dvojic (x_i, y_i)
- Zvolíme libovolný referenční bod a všechny body kontury vyjádříme pomocí vzdálenosti r od referenčního bodu a úhlem β.



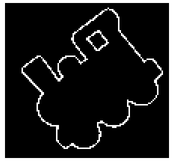
- Houghova transformace je pak definována parametrickými rovnicemi

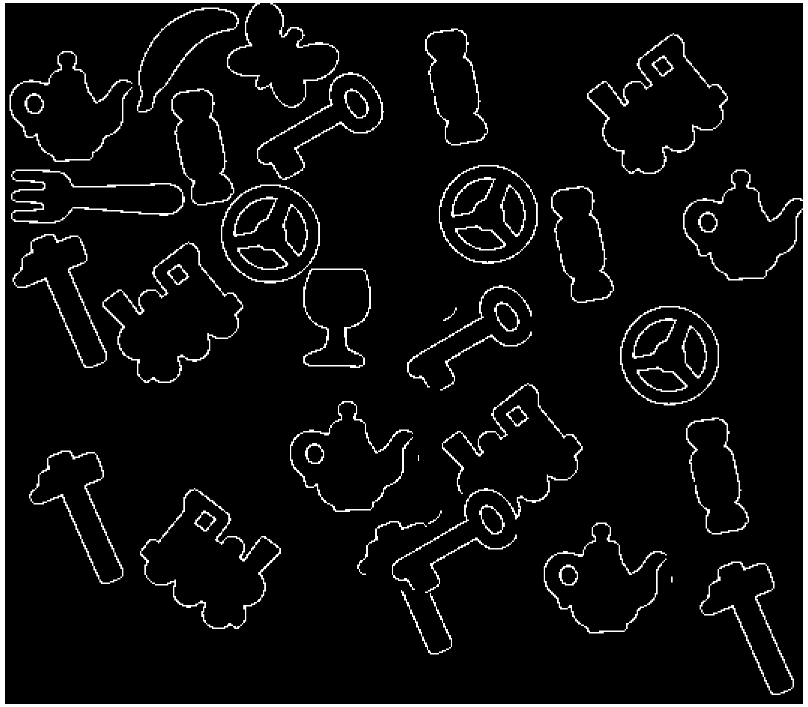
$$x_{ref} = x + r\cos(\beta)$$

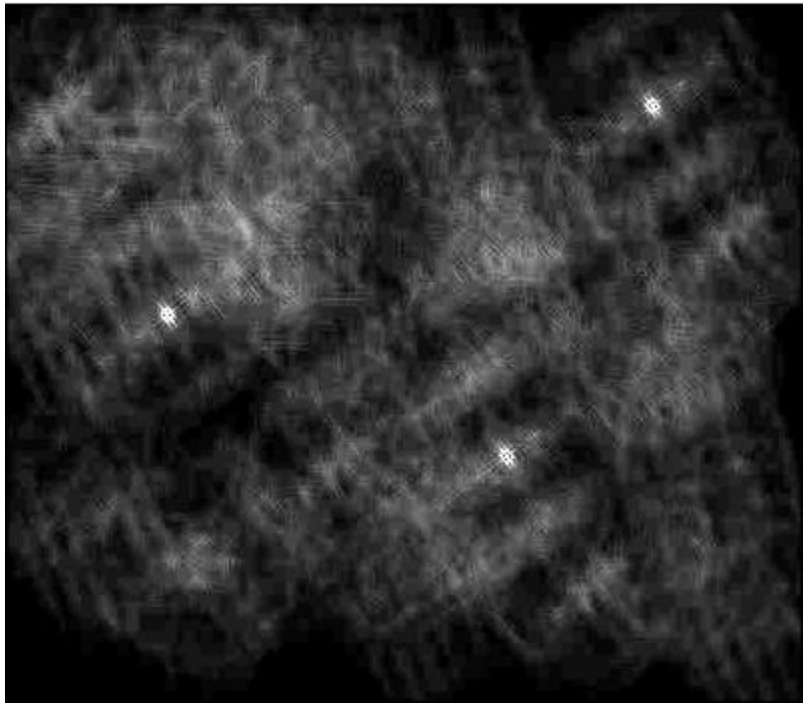
$$y_{ref} = y + r\sin(\beta)$$

Zobecněná Houghova transformace

- Příklad:







Referenční snímek

Analyzovaný snímek

Akumulátor

Segmentace obrazu

- Segmentace obrazu na základě **oblastí (ploch)**
  - Rozčlenění obrazu na (nepřekrývající se) oblasti, které úzce souvisí s objekty nebo částmi reálného světa
  - Segmentace probíhá na základě stejnorodosti (homogenity) nějaké vlastnosti, např. velikosti jasu, barvy, tvaru, textury apod.

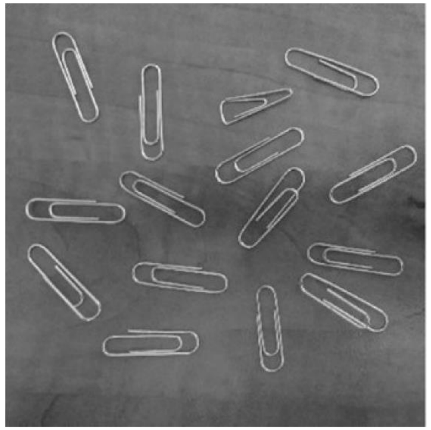
  - Prahovací metody (jednoduchý/dvojitý práh)
  - Otsuova metoda, adaptivní prahování
  - Split & merge algoritmus
  - Vzdálenostní transformace, Watershed transformace
  - Mean-shift segmentace
  - …

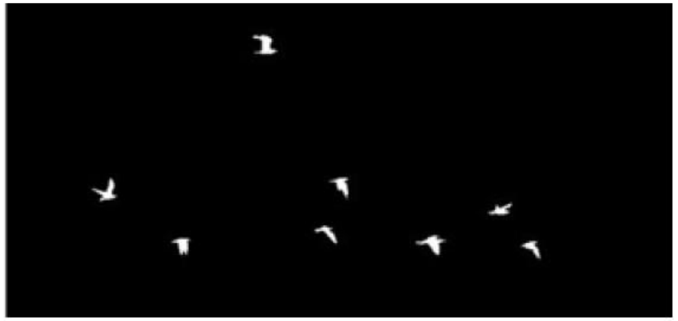
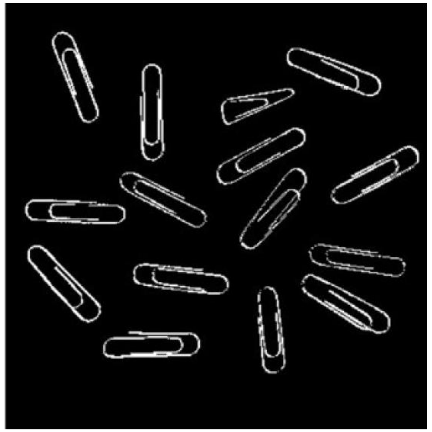
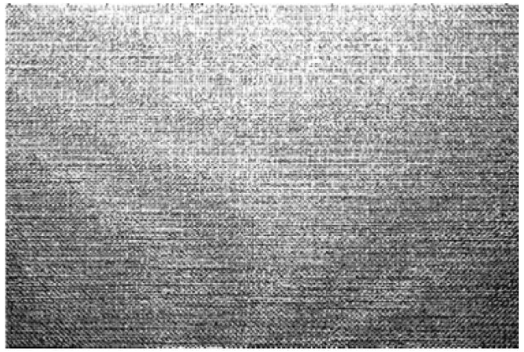
Plošná segmentace

- **Jednoduché prahování** (single thresholding)
  - Při jednoduchém prahování je převáděn šedotónový obrázek na binární.
  - Zvolí se práh *T* a následně se porovná hodnota každého pixelu původního obrázku s tímto prahem.
  - Pokud je hodnota pixelu větší než práh *T*, potom ve výsledném (binárním) obrázku má pixel hodnotu odpovídající bílé barvě, jinak má hodnotu odpovídající černé barvě.
  - Jednoduché prahování se hodí pro oddělování objektů od pozadí.
  - Dokáže zdůraznit na první pohled neviditelné objekty (strukturu) v obrázku – viz příklad prahování obrázku papíru na následujícím slajdu úplně vpravo.
  - Funguje dobře pro kontrastní obrázky.
  - Výsledný binární obrázek může sloužit i jako maska pro „vyřezání“ objektu z původního obrázku.
  - Vhodná metoda např. pro následné spočítání objektů v obraze, výpočet velikostí objektů, tvarových charakteristik apod.

Plošná segmentace

- Jednoduché prahování

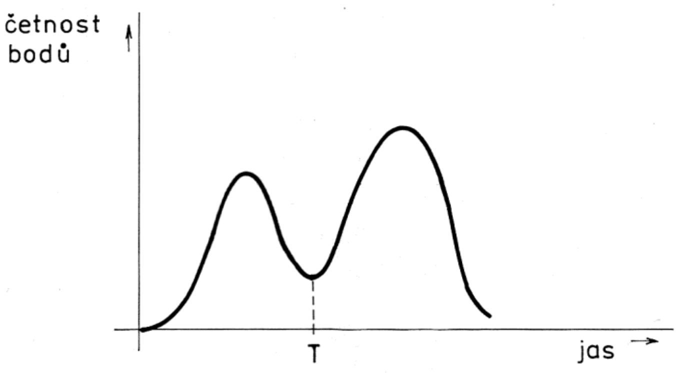




Plošná segmentace

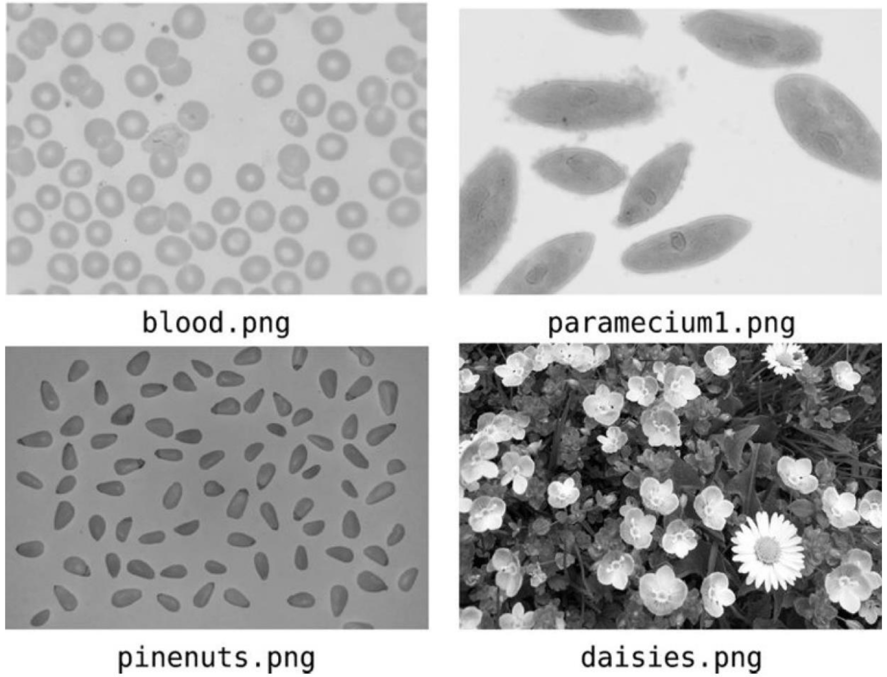
- **Dvojité prahování** (double thresholding)
  - Varianta jednoduchého prahování, kdy používáme dva prahy T_1 a T_2.
  - Pokud je hodnota pixelu mezi hodnotami prahů T_1 a T_2 ($T_1 < T_2$), potom ve výsledném (binárním) obrázku má pixel hodnotu odpovídající bílé barvě, jinak má hodnotu odpovídající černé barvě.

Plošná segmentace

- **Volba vhodného prahu**
  - Využití histogramu pro stanovení vhodného prahu *T* (místa výrazného předělu).
  - Výrazným předělem může být např. minimum zastoupených hodnot mezi dvěma „kopci“ v histogramu.
  - Nevýhodou je, že kopce nemusí být ale výrazné.

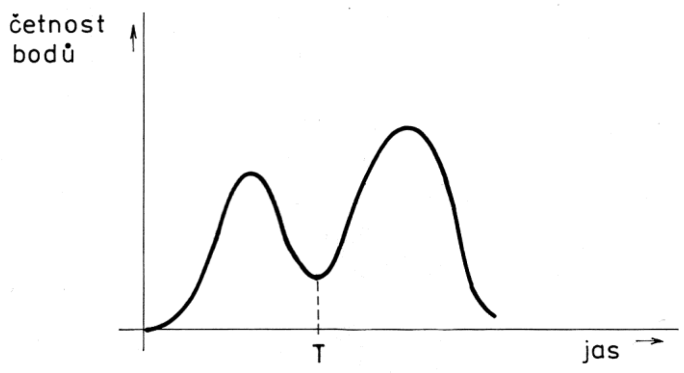


$$g(i,j)=\begin{cases}1 & pro \quad f(i,j)\geq T\\ 0 & pro \quad f(i,j)<T\end{cases}$$

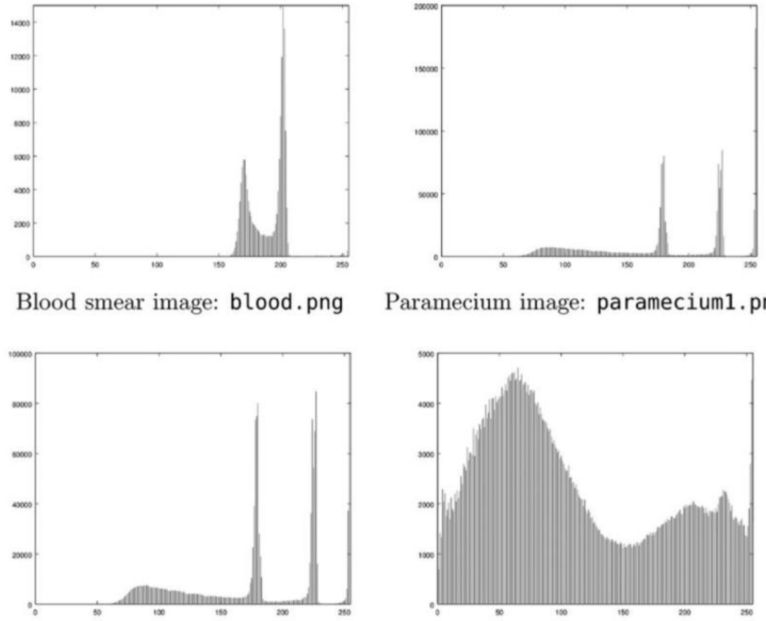


Plošná segmentace

- **Volba vhodného prahu**
  - Využití histogramu pro stanovení vhodného prahu *T* (místa výrazného předělu).
  - Výrazným předělem může být např. minimum zastoupených hodnot mezi dvěma „kopci“ v histogramu.
  - Nevýhodou je, že kopce nemusí být ale výrazné.



$$g(i,j)=\begin{cases}1 & pro \quad f(i,j)\geq T\\ 0 & pro \quad f(i,j)<T\end{cases}$$



Blood smear image: blood.png

Paramecium image: paramecium1.png

Pine nuts image: pinenuts.png

Daisies image: daisies.png

Plošná segmentace

- **Otsuova metoda pro stanovení vhodného prahu**
  - Nobuyuki Otsu, 1979
  - Postavena na **maximalizaci** rozptylu $\sigma_b^2(t)$ **mezi** třídami světlých a tmavých oblastí (segmentovaných prahem *t*) (*between class variance*).

min. $\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$

$$\sigma_b^2(t) = \sigma^2 - \sigma_w^2(t) = \omega_0(\mu_0 - \mu_T)^2 + \omega_1(\mu_1 - \mu_T)^2$$
$$= \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2$$

max. $\sigma_b^2(t) =$ max. $w_f w_b(\mu_f - \mu_b)^2$

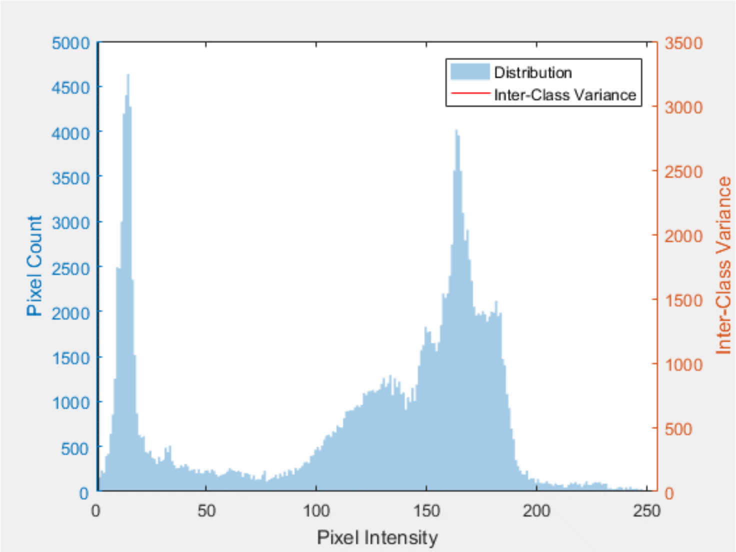
  - kde w_b, resp. w_f jsou poměrná zastoupení pixelů pozadí a popředí v obrázku ($w_b + w_f = 1$) pro danou hodnotu prahu *t*.
  - Hodnoty w_b a w_f se počítají podle vztahů $w_b = \sum_{k=0}^{t-1} p_k$ a $w_f = \sum_{k=t}^{L-1} p_k$
  - kde $p_k = n_k/N$ (pravděpodobnost výskytu pixelu s hodnotou intenzity *k*, *N* je celkový počet pixelů v obrázku a *L* počet odstínů šedi.

Plošná segmentace

- Střední hodnoty μ_b a μ_f se vypočtou ze vztahů $\mu_b = \frac{1}{w_b}\sum_{k=0}^{t-1} kp_k$ a $\mu_f = \frac{1}{w_f}\sum_{k=t}^{L-1} kp_k$

$$w_b = \sum_{k=0}^{t-1} p_k = \sum_{k=0}^{t-1} \frac{n_k}{N} = \frac{1}{N}\sum_{k=0}^{t-1} n_k$$

$$\mu_b = \frac{1}{w_b}\cdot\sum_{k=0}^{t-1} k\cdot p_k = \frac{1}{\cancel{\frac{1}{N}}\sum_{k=0}^{t-1} n_k}\cdot\sum_{k=0}^{t-1} k\cdot\frac{n_k}{\cancel{N}} = \frac{1}{\sum_{k=0}^{t-1} n_k}\cdot\sum_{k=0}^{t-1} k\cdot n_k = \frac{\sum_{k=0}^{t-1} k\cdot n_k}{\sum_{k=0}^{t-1} n_k}$$

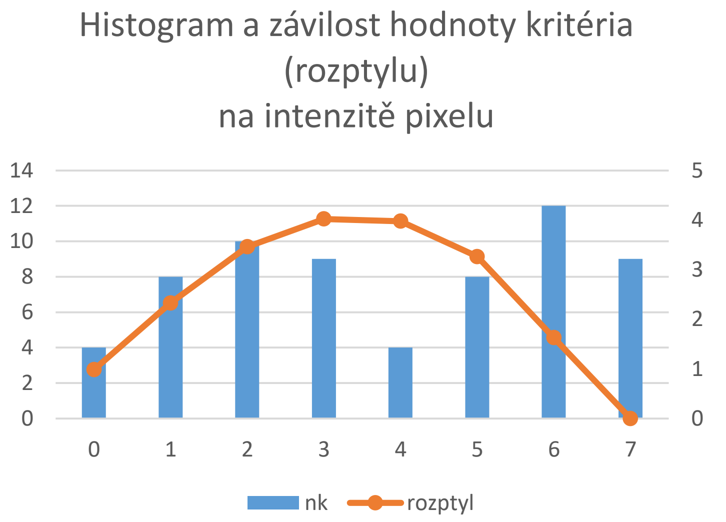


- Optimální hodnota prahu se hledá úplným prohledáváním prostoru všech možností (postupně vypočítáváme hodnoty kritéria pro jednotlivé hodnoty prahu).
- Optimální hodnota prahu je taková hodnota, kdy rozptyl intenzit světlé a tmavé oblasti je největší – oblasti se od sebe nevíce liší, jsou vůči sobě nejvíce kontrastní.

Plošná segmentace

- **Příklad**: Mějme obrázek velikosti 8x8 pixelů s 8 úrovněmi odstínů šedi. Počty pixelů pro jednotlivé úrovně *k* jsou dány v tabulce níže (n_k). Cílem je najít optimální práh pomocí Otsuovy metody.

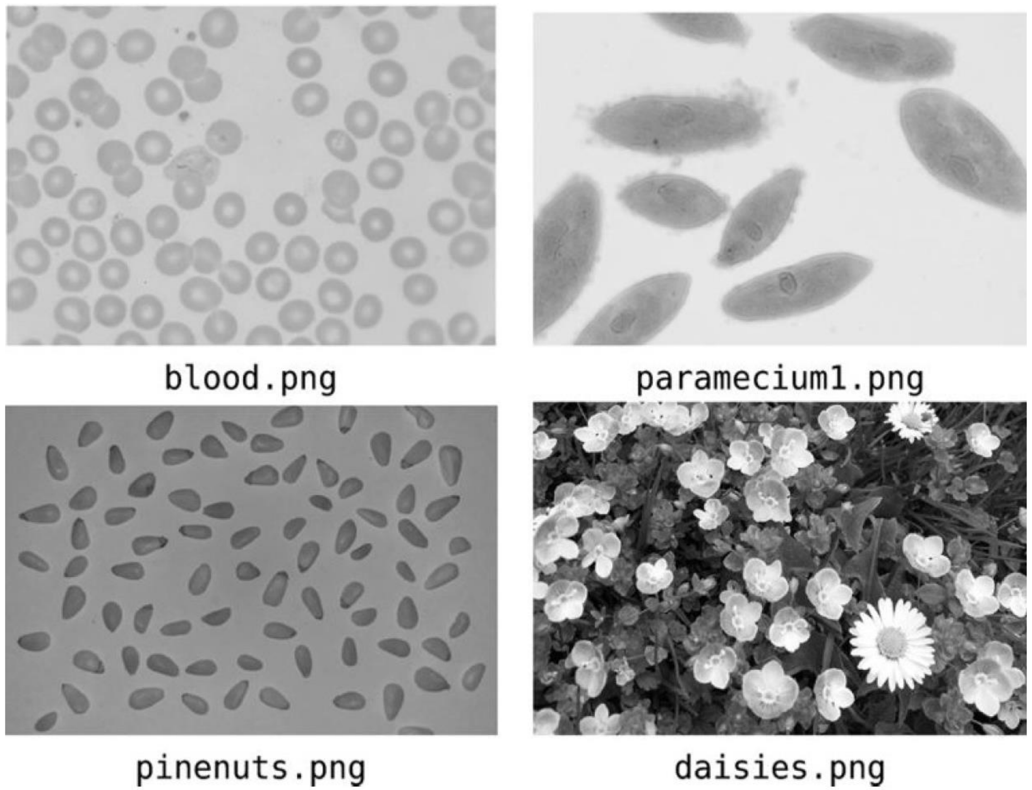
k	n_k	p_k	w_b	w_f	kp_k	$\sum kp_k$	μ_b	μ_f	$\sigma_b^2(t)$
0	4	0.06250	0.06250	0.93750	0.00000	0.00000	0.00000	4.10000	0.98496
1	8	0.12500	0.18750	0.81250	0.12500	0.12500	0.66667	4.57692	2.32935
2	10	0.15625	0.34375	0.65625	0.31250	0.43750	1.27273	5.19048	3.46246
3	9	0.14062	0.48438	0.51562	0.42188	0.85938	1.77419	5.78788	4.02348
4	4	0.06250	0.54688	0.45312	0.25000	1.10938	2.02857	6.03448	3.97657
5	8	0.12500	0.67188	0.32812	0.62500	1.73438	2.58140	6.42857	3.26296
6	12	0.18750	0.85938	0.14062	1.12500	2.85938	3.32727	7.00000	1.63013
7	9	0.14062	1.00000	0.00000	0.98438	3.84375	3.84375	–	–

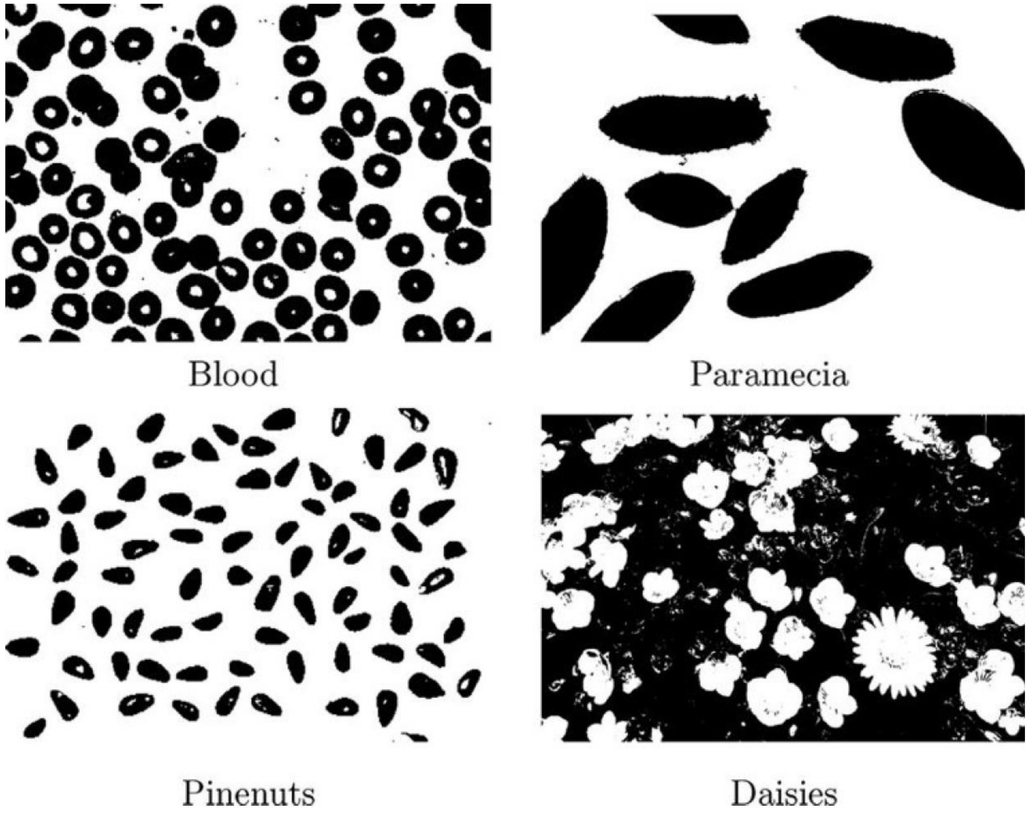


- Maximální hodnota kritéria $\sigma_b^2(t)$ je 4,02348, což odpovídá hodnotě *k* = 3, resp. *t*-1 = 3, tj. **optimální práh je *t* = 4**.

Plošná segmentace

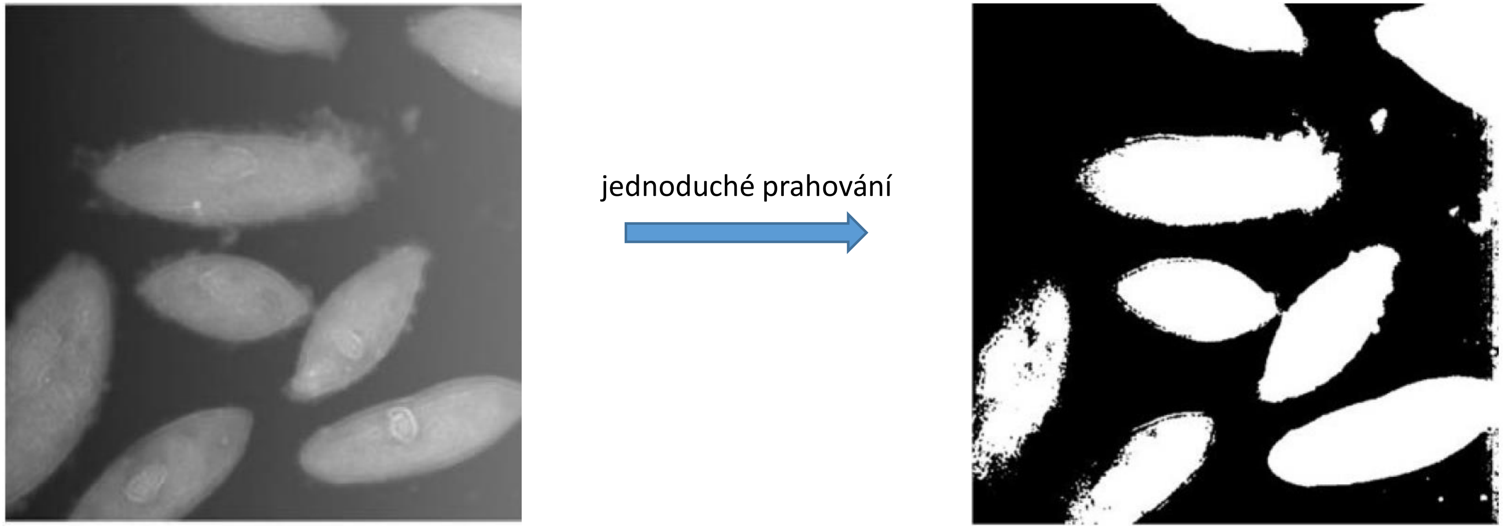
- Příklad prahování pomocí Otsuovy metody





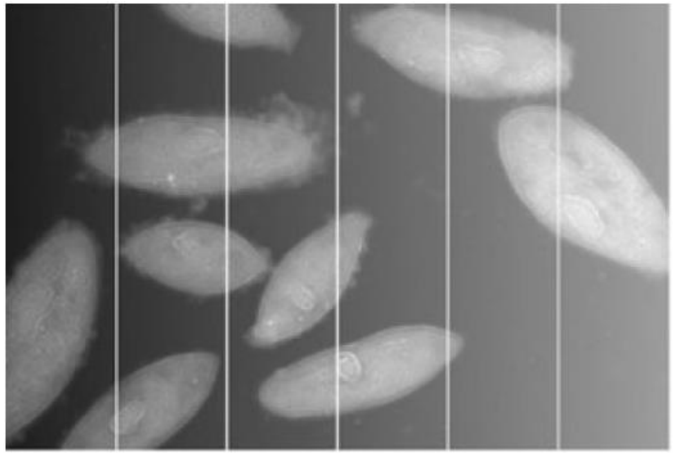
Plošná segmentace

- **Adaptivní prahování**
  - Segmentace pomocí prahu (prahů) nemusí být vždy úspěšná.
  - Např. pro obrázky, kde pozadí mění svoji intenzitu (v jedné části je tmavší a v jiné světlejší) a současně i objekty mají různou intenzitu, nelze využít jednoduchého prahování pro celý snímek, i když byl práh nastaven optimálně, např. pomocí uvedené Otsuově metody.



Plošná segmentace

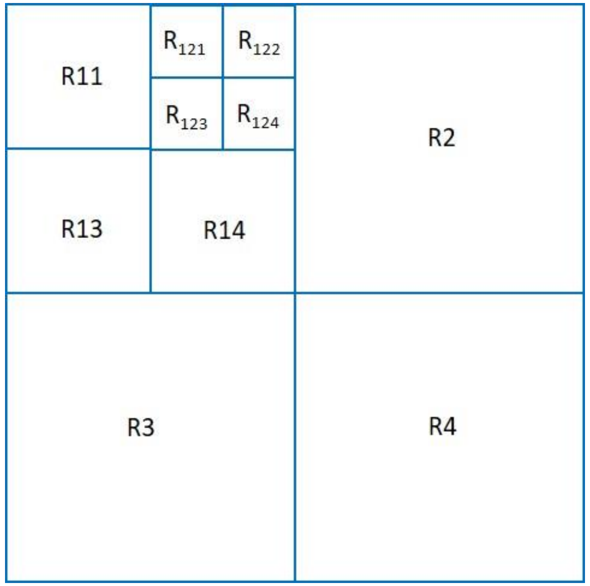
- Řešením je „rozřezat“ obrázek na části a segmentaci (prahování) řešit v každé části obrázku zvlášť.
- Jednoduchým způsobem rozdělením obrázku je rozřezat obrázek na čtvercové oblasti stejné velikosti. Otázkou je, jak velké mají být.
- V případě našeho konkrétního obrázku se intenzita pozadí i objektů mění zleva doprava, takže rozřezání může být po pruzích. Použita byla Otsuova metoda.

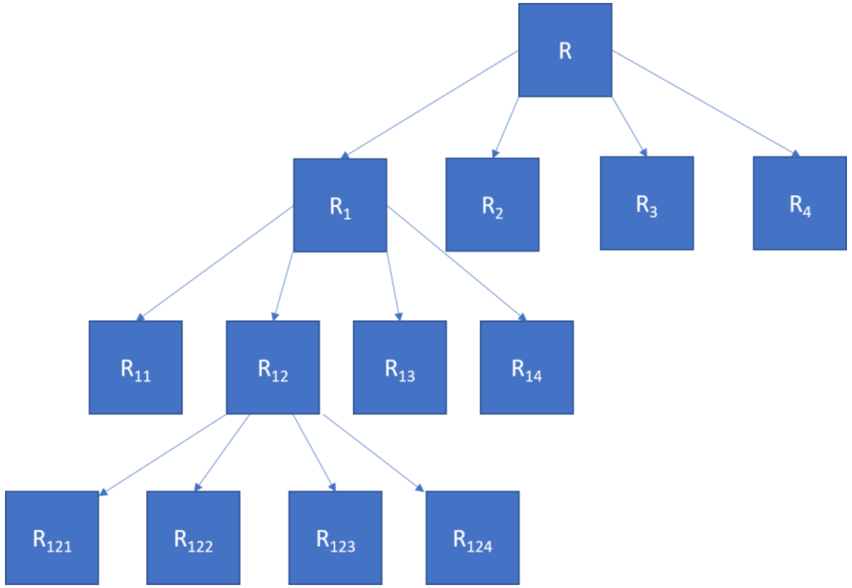


adaptivní prahování

Plošná segmentace

- **Split & merge segmentace**
  - (Split) Obrázek je postupně (rekurzivně) rozdělován na oblasti (kvadranty) na základě definovaného kritéria homogenity. Nehomogenní oblasti jsou dále děleny, dostatečně homogenní oblasti ponechány.
  - (Merge) Sousední podobné (homogenní) oblasti jsou zpětně slučovány dohromady.

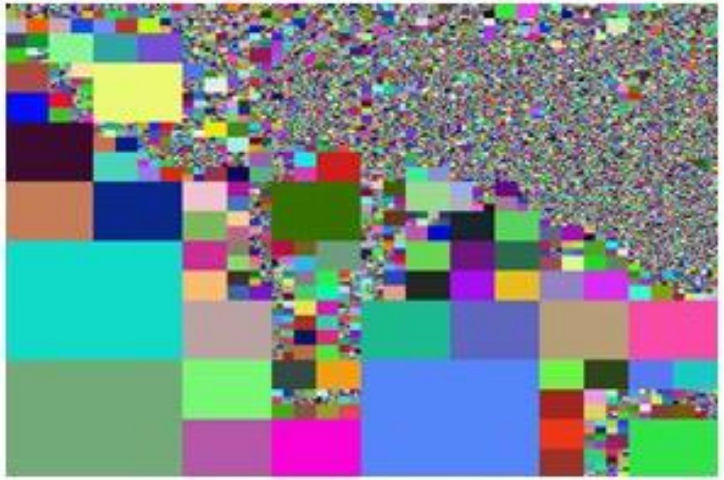


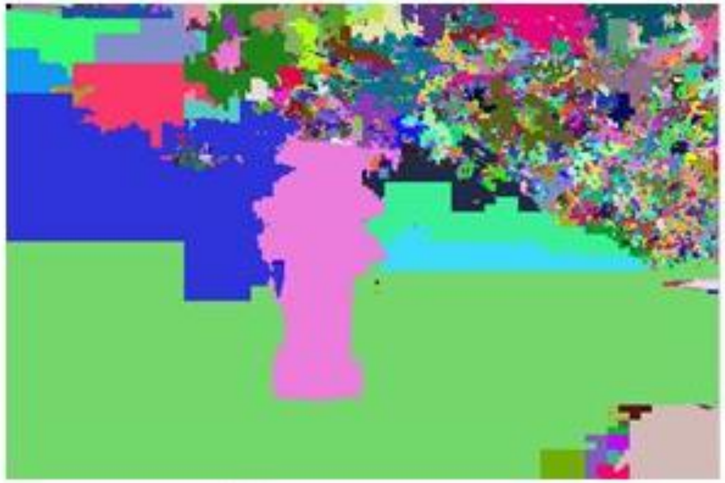


Plošná segmentace

- Příklad split & merge segmentace


původní obrázek


po dělení (split)

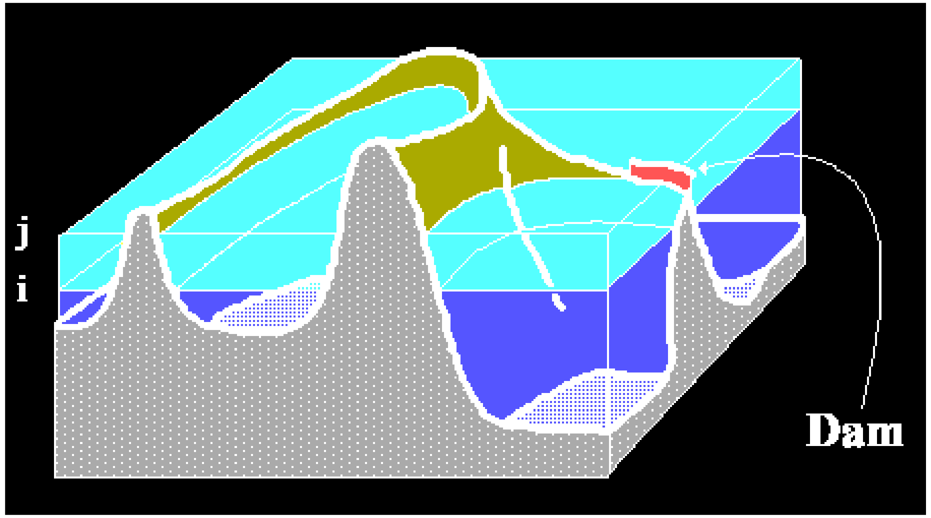

po sloučení (merge)

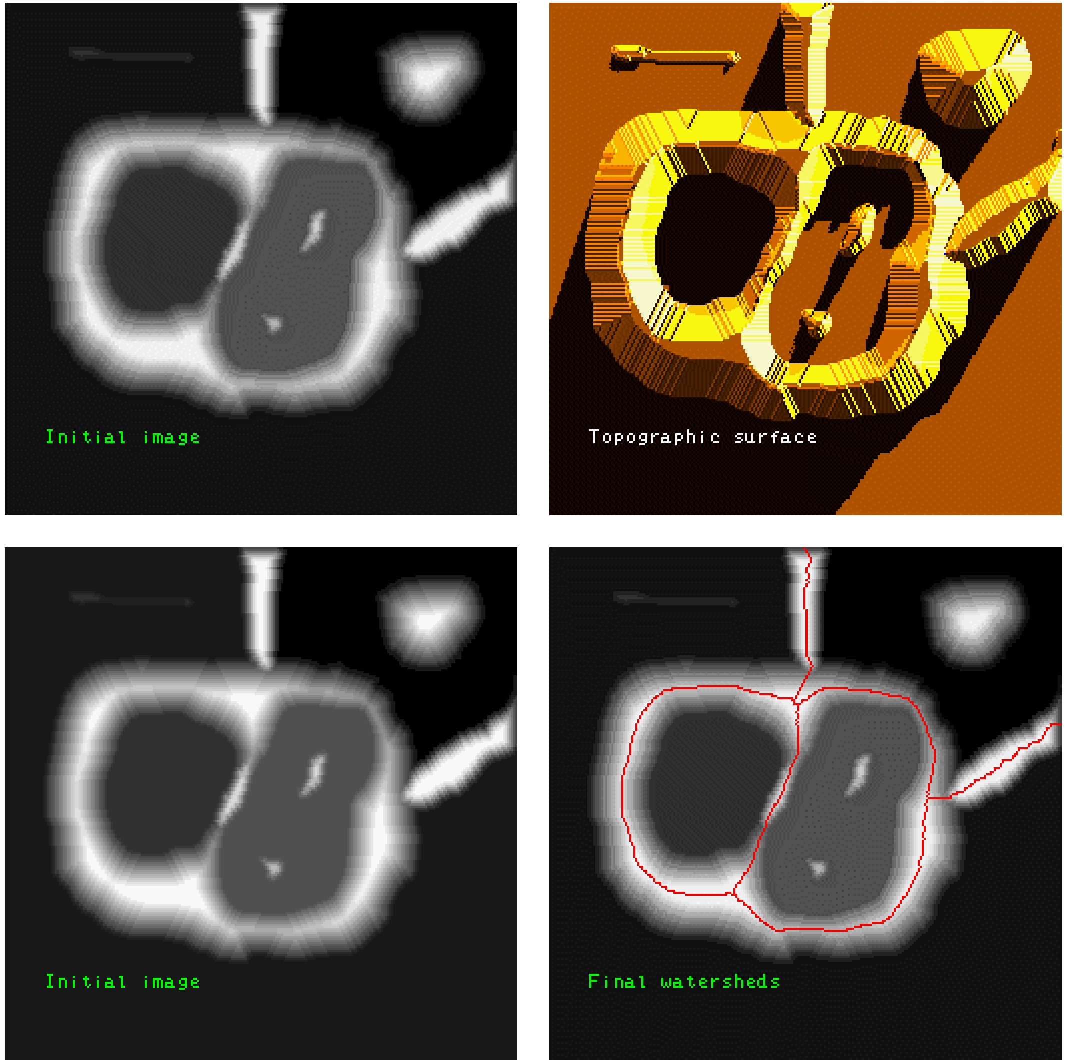
Plošná segmentace

- **Watershed segmentace ("vodní předěl")**
  - každý snímek v odstínech šedi lze považovat za topografickou mapu (hloubková mapa, terén)
  - černá barva = nejvzdálenější/nejhlubší místa, bílá barva = nejbližší/nejvyšší místa

  - Princip segmentace:
    - postupně „zaplavujeme" terén, přičemž zabraňujeme tomu, aby se „slila" jezírka – vytváříme mezi nimi hranici (watershed)
    - rozdělujeme tak terén na dvě oblasti: jezírka a hranice
    - Watershed segmentaci aplikujeme nejčastěji na gradient původního obrazu nebo na obrázek po vzdálenostní transformaci (zvýraznění hranic)

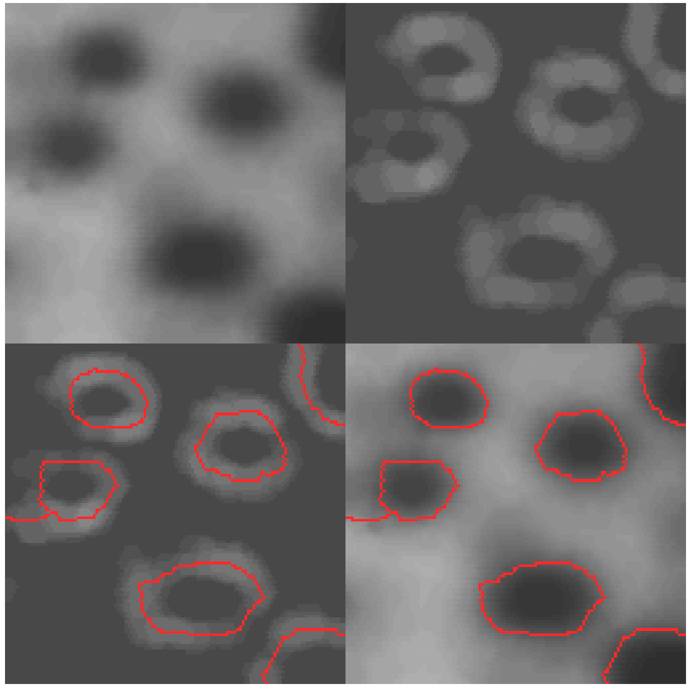
Plošná segmentace

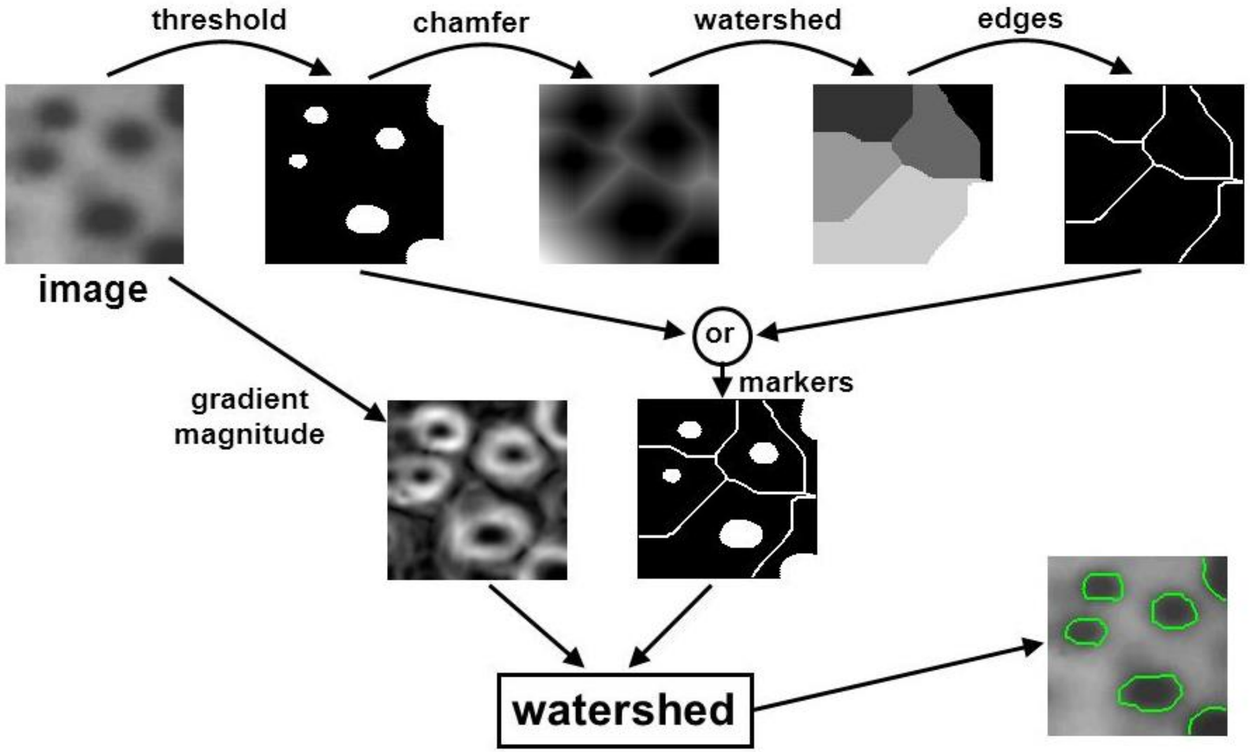




Plošná segmentace

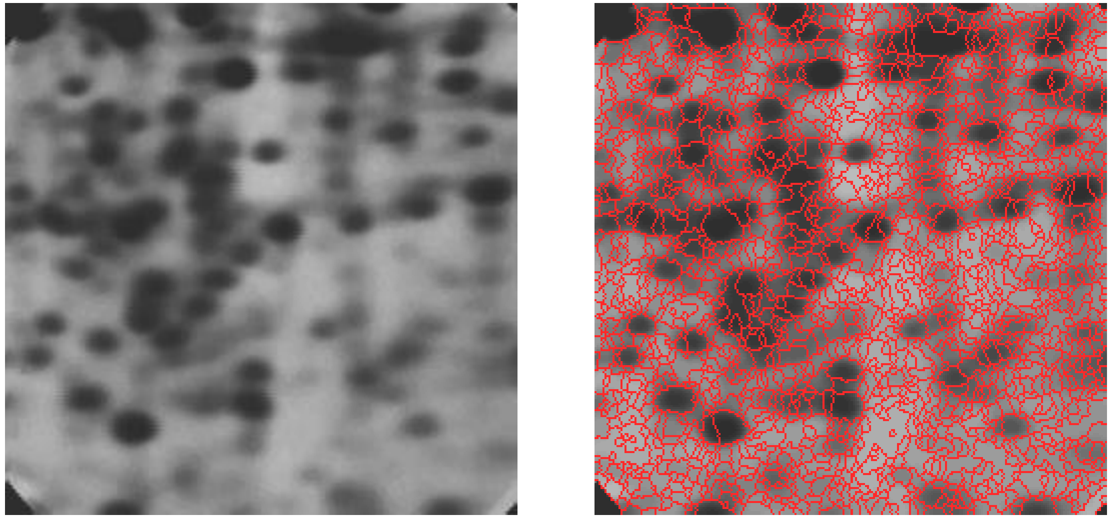
- Původní obrázek
- Gradient/vzdálenostní transf.
- Watershed segmentace
- Finální segmentace





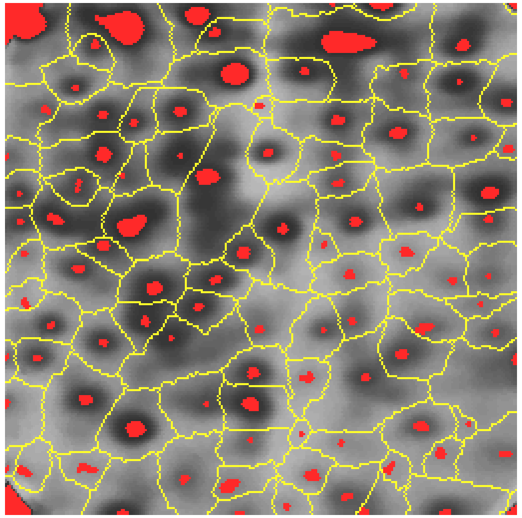
Plošná segmentace

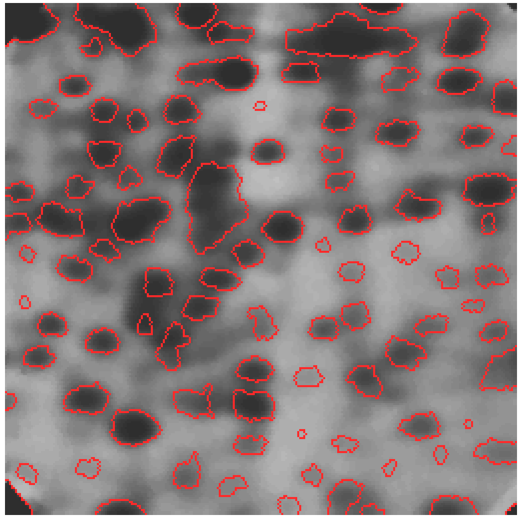
- Metoda je citlivá na šum – dochází k nadbytečné segmentaci (přesegmentování)
  - Řešení: Watershed segmentace s pomocí značek



Plošná segmentace

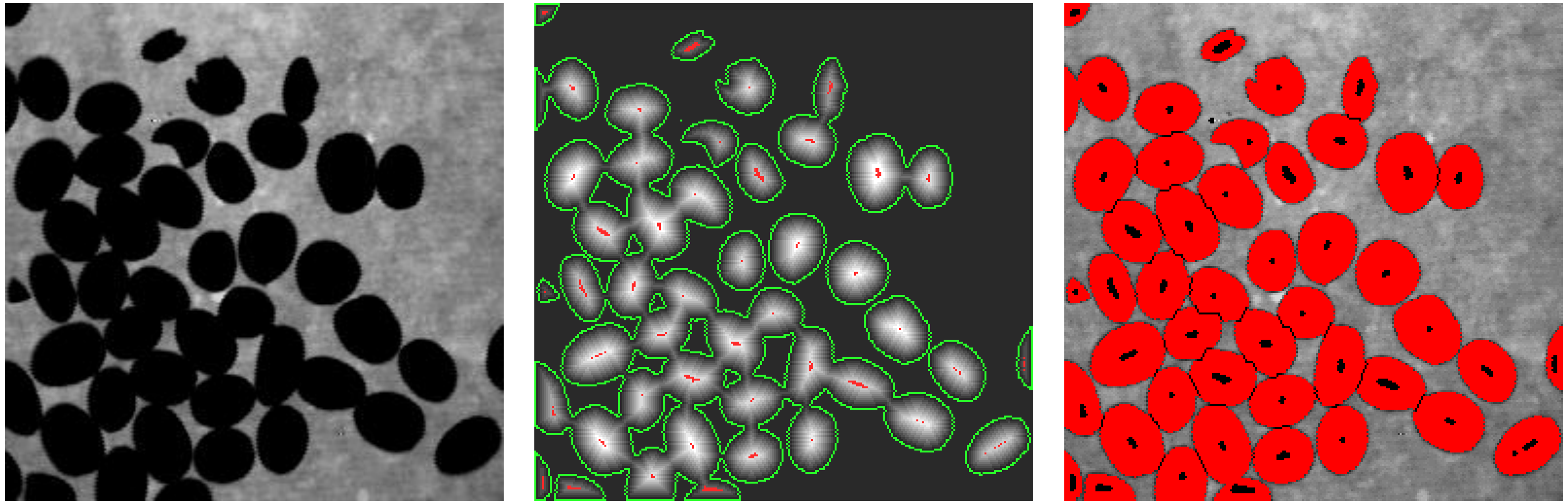
- **Watershed segmentace s pomocí značek** (*Marker-controlled watershed*)
  - předem definujeme místa odkud má začít watershed segmentace, např. jinou metodou segmentace
  - snaha zabránit „přesegmentování“





Plošná segmentace

- **Příklad**: Watershed segmentace kávových zrnek
  - značky byly určeny na základě vzdálenosti od nehomogenní hranice, nikoliv gradientu (ten by nepomohl)



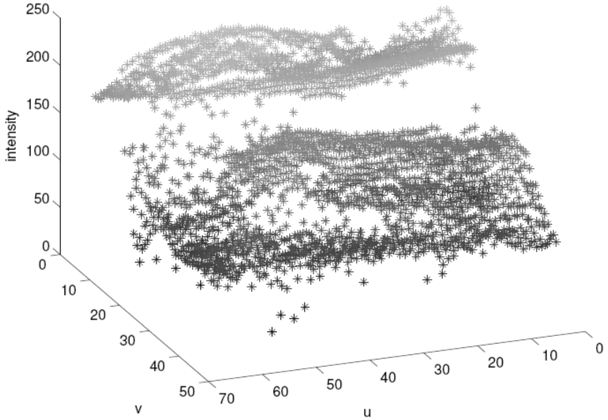
Plošná segmentace

- **Mean shift segmentace**
  - Mean shift segmentace vychází z mean shift filtrace.
  - Mean shift filtrace už téměř připomíná výslednou segmentaci, ale je třeba provést ještě několik úprav. Většinou se jedná o spojení podobných sousedních pixelů a oblastí.
  - Mean shift filtrace/segmentace dává dobré výsledky (zachovává hrany), ale je časově náročná (4 vnořené cykly v sobě).

mean shift segmentace

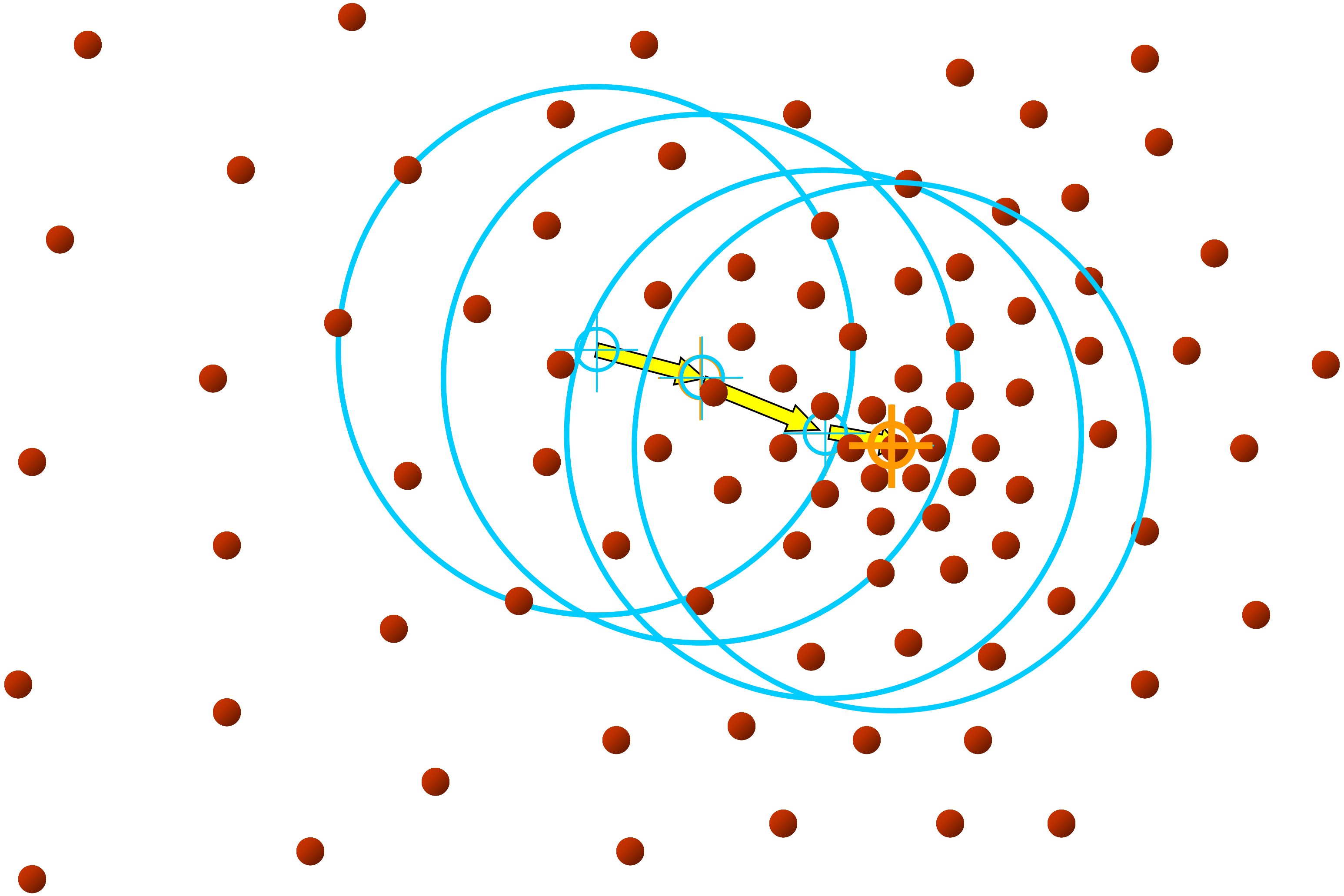
Plošná segmentace

- Základní (a zjednodušený) algoritmus mean shift filtrace/segmentace:
  - Mějme šedotónový obrázek I(x,y). Ten si představme jako bodový 3D graf, kde odstín šedi I(x,y) je funkcí souřadnic (x,y).



  - Postupně budeme procházet všechny pixely obrázku, resp. body v 3D prostoru, [x, y, I(x,y)].
  - Kolem každého 3D bodu definujeme okolí (kouli) o zvolené velikosti a spočítáme průměr (**mean**) ze souřadnic bodů, které se v tomto okolí nachází. Tím určíme nový výchozí bod, do kterého toto okolí posuneme (**shift**) a celý proces opakujeme tak dlouho, dokud se pozice středu nestabilizuje.
  - Poté, co jsme dokonvergovali do stabilní pozice, bude hodnota intenzity pixelu (ze kterého jsme vyšli) v novém (vyfiltrovaném/segmentovaném) obrázku odpovídat výsledné intenzitě v oblasti, kam jsme dokonvergovali (třetí souřadnici středu koule v místě, kam jsme se výsledně posunuli).

Mean-shift analýza



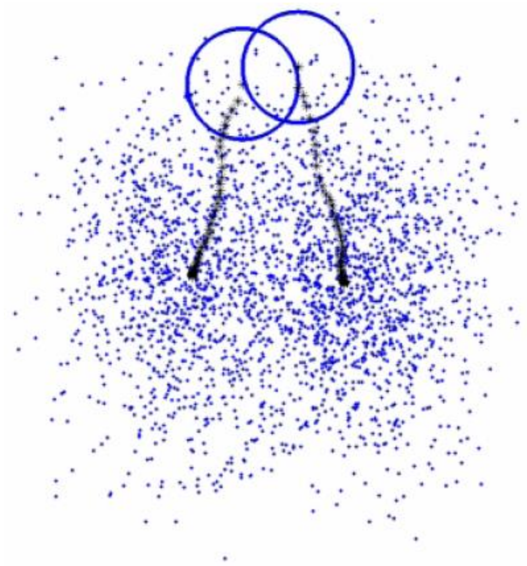
oblast zájmu (kernel)

těžiště

mean-shift vektor

Plošná segmentace

- Několik postřehů k uvedenému algoritmu
  - Shlukuje body obrazu na základě podobnosti jejich vzhledu a blízkosti jejich pozice (hledání nejhustších oblastí – lokálních maxim). Předpokládáme, že blízké pixely o podobné intenzitě náležejí stejnému objektu a budou v prostoru [x, y, I(x,y)] vytvářet shluky.
  - Pro barevné (RGB) obrázky se pohybujeme v 5D prostoru.
  - V každém kroku se posuneme ve směru váženého průměru vzorku (těžiště) z oblasti kolem současné pozice (definované kernelem – radiálně symetrickou funkcí).
  - Je třeba si poradit s tím, že prostorové souřadnice mají jiné jednotky, než hodnoty intenzity. Většinu se to řeší vhodnou přepočetní konstantou. Jinou možností je měřit zvlášť okolí v rovině (*range neighborhood, spatial radius*) a zvlášť povolený rozdíl intenzit (*value difference, range radius*).
  - Okolí (kernel) nemusí být „ostře“ definované koulí, ale spíše se využívá jako kernel Gaussova funkce.



Plošná segmentace

- Symbolicky zapsaný mean-shift algoritmus

Input: grayscale image I, bandwidth parameters h_s and h_r
Output: output image I' resulting from the mean-shift (edge-preserving) filter

for $(x, y) \in I$ **do**
  $(x', y', v') \leftarrow (x, y, I(x, y))$
  repeat
    $num \leftarrow (0, 0, 0)$
    $den \leftarrow 0$
    for $(x_i, y_i) \in I$ **do**
      $w \leftarrow g(((x' - x_i)^2 + (y' - y_i)^2)/h_s^2 + (v' - v_i)^2/h_r^2)$ ← Gaussova funkce
      $num \leftarrow_+ w * (x_i, y_i, I(x_i, y_i))$
      $den \leftarrow_+ w$
    $mean\text{-}shift \leftarrow num/den - (x', y', v')$
    $(x', y', v') \leftarrow num/den$
  until $\textsc{Norm}(mean\text{-}shift) < \tau$
  $I'(x, y) \leftarrow v'$
return I'

Literatura

- McAndrew A., Computational Introduction to Digital Image Processing, CRC Press, 2. vydání, 2016
- Sundararajan D., Digital Image Processing: A Signal Processing and Algorithmic Approach, Springer, 2017
- Birchfield S., Image Processing and Analysis, Cengage Learning, 2016
- Acharya T., Ray A. K., Image Processing: Principles and Applications, Wiley, 2005
- Burger W., Burge M. J., Principles of Digital Image Processing: Fundamental Techniques, Springer-Verlag, 2009
- Beucher S., Watersheds & Waterfalls, 2000, http://cmm.ensmp.fr/~beucher/publi/course2000.pdf