# 7. Segmentace. Algoritmy formalizující segmentaci jako problém minimalizace řezu grafem.

# Segmentace obrazu

Segmentace obrazu je metoda, nebo spíše skupina metod postavených na různých principech, digitálního zpracování obrazu, která slouží k automatickému **rozdělení** vlastního **obrazu** na **oblasti** se **společnými vlastnostmi** a které obvykle mají nějaký smysluplný význam. Typickým cílem segmentace obrazu je identifikace **popředí** a určení oblastí v obraze odpovídajícím významnému prvku zachycené scény.

Výsledky segmentace jsou využitelné například v počítačovém vidění, zpracování lékařských obrazových dat (Medical Imaging) nebo při analýze obrazů získaných při dálkovém průzkumu Země.

**Příklad:**

|  |  |
| --- | --- |
| Částečná anotace vytvořená manuálně | Počítačem odhadnutá segmentace |

Úloha se skládá ze dvou kroků. Prvním krokem je ke každé třídě (popředí, pozadí) vytvořit **statistický** **popis**. Ke statistickému popisu použijeme Gaussovskou směs, kterou odhadneme pomocí algoritmu K-means. Druhým krokem bude rozhodnutí (pro každý pixel) o tom, jestli patří k pozadí nebo popředí.

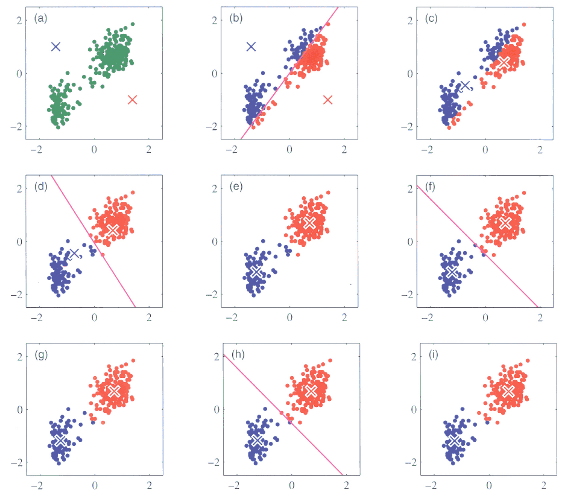
Budeme potřebovat metodu vektorové kvantizace K-means. Algoritmus postupuje takto:

|  |  |
| --- | --- |
| 1. **Vstup**: množina dat X. Každé datum je D-rozměrný vektor. Vektorů je celkem N. 2. Inicializace K středů. **Středy** mohou být zadány na vstupu. Alternativně mohou být vybrány náhodně, např. výběrem ze vstupních dat. 3. Pro každý vektor z X se najde jemu nejbližší střed. Tím dojde k rozdělení vektorů do K podmnožin. 4. Posunutí středů do těžišť podmnožin. Spočítáme těžiště pro každou (i-tou) podmnožinu vektorů a tím dostaneme novou polohu příslušného (i-tého) středu (i = 1..K). Pokud k danému středu nejsou přiřazeny žádné vektory z X, vygenerujte mu novou pozici jako náhodně vybrané datum ze vstupního souboru X. 5. opakování bodů 3 a 4, dokud se označkování bodů z X do podmnožin mění a nebo dokud není překročen zadaný počet iterací. |  |

V algoritmu se bude hodit funkce, která k množině **vektorů** (matice D x N) a množině **středů** (matice D x K) najde index **nejbližšího** **středu**. Implementuje se jako samostatná funkce *nearest*.

Funkce nalezne ke každému sloupcovému vektoru v matici data nejbližší vektor z množiny means. Výstupem funkce budou indexy idxs (matice 1xN) nejbližších středů a dists (matice 1xN) vzdálenosti k nejbližšímu středu.

**Postup:**



## Základní metody segmentace

### Prahování

Prahování (angl. tresholding) je nejjedodušší medota segmentace obrazu založená na hodnocení jasu každého pixelu. Jejím principem je nalezení takové hodnoty (prahu) v **histogramu**, pro kterou bude platit, že všechny hodnoty jasu **nižší** než práh **odpovídají** **pozadí**, zatímco všechny hodnoty **vyšší** než práh **odpovídají** **popředí**. Nehomogenní obraz, obvykle obraz obsahující různě jasné úseky, neumožňuje určit přímo globální práh, je však možné využít metod adaptivního prahování.

V případě, že se rozložení jasu pixelů popředí a pozadí výrazně překrývá, například protože obraz obsahuje výrazný podíl šumu, může být přímé použití prahování prakticky nemožné.

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

### Regionální metody

|  |  |
| --- | --- |
| Regionální metody (angl. region-based methods) jsou metody, které jsou založeny na zjišťování **podobnosti** **pixelů** v nějaké vlastnosti, touto vlastností může být například **jas** nebo **statistické** **vlastnosti** **okolí** **pixelu**.  Podstatou metod je, že konstrukce segmentu segmentu postupuje zdola nahoru, od jednoho pixelu po celý segment. Nejprve jsou nějakým algoritmem v obraze rozmístěny iniciální (semínkové, angl. seed) pixely, obvykle **rovnoměrně** nebo **náhodně**, segment pak vzniká iterativním **rozrůstáním** se okolí iniciálního pixelu.  Při použití metod založených na růstu segmentu není zaručeno, že při různém počtu a rozmístění iniciálních pixelů bude výsledek segmentace identický. Na druhou stranu jsou tyto metody schopny segmentovat i takový obraz, který obsahuje značné množství **šumu**. |  |

## Metody založené na hranici

Metody založené na **hranici** (angl. boundary-based methods) jsou založeny především na detekci regionálních rozdílů ve vlastnostech obrazu.

### Detekce hran

K detekci hran (angl. edge detection) se obvykle používají **gradientní operátory**, např. **Cannyho** **hranový** **detektor** nebo **Sobelův** **filtr**. Protože výstupem gradientního operátoru je obraz, kde jsou sice hrany **zdůrazněny**, ale může obsahovat další artefakty odpovídající lokálním nehomogenitám v obraze, je třeba obraz dále upravit. K odstranění artefaktů vzniklých lokálními malými rozdíly obvykle postačuje **prahování**. Vzhledem ke svým vlastnostem mohou detektory hran vytvářet **přerušované** hranice i **falešné** hranice, obraz hranic je tedy třeba dále zpracovat.

### Sledování hranice

Sledování hranice (angl. boundary tracking) je postup aplikovatelný na obrazy obsahující především informaci o hranicích, například na výsledky použití **gradientních filtrů** (gradientní obraz). Cílem metod je sledovat v gradientním obraze právě jen skutečné hrany a **vyloučit** artefakty. Sledování hranice může selhávat u příliš **zašuměných** obrazů nebo u příliš **komplikovaných** tvarů segmentu.

## Další metody

### Aktivní kontura

Aktivní kontura (angl. active contour, snake) je pokročilá metoda segmentace obrazu. Metoda požaduje, aby jejím vstupem byla **uzavřená** **křivka** přibližně ohraničující segment, který je třeba ohraničit přesně. Na základě fyzikálních analogií se pak definují energie a síly, které v konečném důsledku deformují uzavřenou křivku tak, že se stane hranicí segmentu.

### Segmentace rozvodím

Segmentace rozvodím (angl. watershed segmentation) je metoda založená na názorné představě **zaplavování** **obrazu** **vodou**. Jas pixelu je přitom chápán jako jakási **nadmořská výška**. Jeden segment je pak vlastně oblastí **vyplněnou vodou**, hranice segmentů je místem, kde se stýkají dvě zaplavené oblasti oddělené souvislou skupinou pixelů s vyšší hodnotou jasu.

# Algoritmy formalizující segmentaci jako problém minimalizace řezu grafem

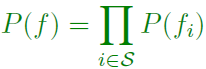
**MRF** (*Markov Random Field*) je teorie s těmi pravděpodobnosti a minimalizací energie, **Graph cut** pak algoritmus, který tu minimalizaci řeší v grafu. Graph cut je obecnější, Grab cut je už pak nějaká konkrétní implementace.

**Labeling:** Každému pixelu v obrázku můžeme přiřadit label, labelů může být několik (např. 0 - pozadí, 1 - objekt).

, f je množina, která vyjadřuje přiřazení labelů všem pixelům, f\_1 označuje label pixelu č. 1, tak např. někde uprostřed obrázku pro pixel s indexem 1000 bude label např. "popředí", tzn. f\_1000 = 1.

 f je tedy funkce, která pixelům z S (site) přiřazuje labely z L (label set).

Různých kombinací jak jednotlivým pixelům přiřadit který label je spousta. Pravděpodobnost, že pixelu i přiřadíme label f\_i za předpokladu, že známe labely ostatních pixelů i' je 

Celková pravděpodobnost pro tu množinu přiřazení f je: 

Tedy pravděpodobnost, že použijeme přiřazení f se rovná tomu, že pronásobíme pravděpodobnosti v jednotlivých pixelech. Jak najdeš správné přiřazení labelů pixelům? Musíš ze všech možných přiřazení, kterých je spousta, vybrat to dobré - to **nejpravděpodobnější**.

**Markov Random Field: **

Toto říká jaká je pravděpodobnost, že pixelu i přiřadíš label f\_i, když znáš jaké labely jsou v okolí (např. 8x8) toho pixelu, prostě P( f\_i | f\_Ni ).

**MRF a Gibbs Random Fields**

|  |  |
| --- | --- |
| Energie U, která vyjadřuje jako by chybu (odchylku) a čím je odchylka **vyšší** tím je **méně** **pravděpodobné**, že f je správné přiřazení, takže musíš najít takové přiřazení labelů f, pro které je energie U nejmenší, abys měl největší pravděpodobnost, že to f je správné.    Takže energie se rovná: pro každý pixel sečteš jeho potencial V1(f\_i) a pro každý pixel sečteš nějaké potenciály pezi pixelem a jiným pixelem v okolí, ty potenciály vyjadřují chybu. |  |

Na cvičení jsme to dělali nějak tak, že pokud jsou dva sousední pixely mají **totožný** **label**, tak V2(f\_i, f\_i') = 0 a pokud měli **různý** **label**, tak V2(f\_i, f\_i') = 1, takže různé labely jsme penalizovali chybou 1, tak si to představ teď v grafu.

Máme pixely, sousední pixely jsou mezi sebou spojeny hranami. Tam , kde pixely mají **stejný label** má hrana cenu 0

a tam kde mají **různý label** má hrana cenu 1, najde se řešení pro **maximální tok** v grafu, takže hranami, které mají větší chybu (větší váhu) proteče víc, hranami bez chyb proteče méně, takže tím maximálním tokem získáš hrany, kterými toho **protéká nejvíc** a to je ten **minimální řez**. A rozřízneš to tam, kde pixely jsou nejvíc různé.

**Princip je teda takovej**:

Hrany, kde pixely jsou nejvíce rozlišné mají největší váhu (největší propustnost), řešíš **maximální tok**, kterým najdeš takové hrany s velkou propustností (velkou různorodostí pixelů) a tam to rozřízneš (oddělíš pixely, které si nejsou podobné).