VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Segmentace obrazu metodou spektrálního shlukování a difuzního spektrálního shlukování

Image Segmentation Via Spectral Clustering and Diffusion Spectral Clustering

2015 Vojtěch Cima

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Vojtěch Cima**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Segmentace obrazu metodou spektrálního shlukování a difuzního spektrálního shlukování
Image Segmentation Via Spectral Clustering and Diffusion Spectral Clustering

Zásady pro vypracování:

Cílem diplomové práce je kriticky prověřit možnost segmentace obrazu metodou spektrálního shlukování a difuzního spektrálního shlukování. V diplomové práci proveďte následující:

1. Seznamte se s výše uvedenými metodami a naimplementuje je.
2. Prozkoumejte, jak se uvedené metody chovají při segmentaci vhodně zvolené testovací množiny obrazů.
3. Pokuste se odhalit a detailně popsat zdroj případných potíží.
4. Na základě znalosti výše uvedeného se můžete pokusit navrhnout vlastní vylepšení.

Programátorské práce provádějte v C/C++.

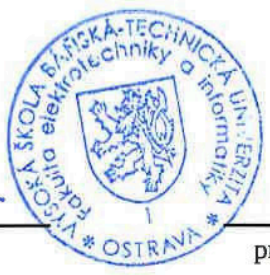
Seznam doporučené odborné literatury:

Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Dr. Ing. Eduard Sojka**

Datum zadání: 01.09.2013
Datum odevzdání: 07.05.2015



doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 16. dubna 2015 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 16. dubna 2015 .....

Rád bych na tomto místě poděkoval doc. Dr. Ing. Eduardovi Sojkovi za trpělivé vedení, podporu a podnětné připomínky při vzniku této práce.

Abstrakt

Spektrální shlukování v posledních letech našlo svou pevnou pozici mezi obecnými datově segmentačními algoritmy. Použití spektrálního shlukování pro segmentaci, zejména reálných obrazů, otevírá široký prostor k dalším optimalizacím a modifikacím tohoto algoritmu. Tato práce představuje teoretický základ rozdílných konfigurací spektrálního shlukování včetně jeho difuzní varianty. Experimentální část práce se zabývá implementací difuzního spektrálního shlukování s použitím algoritmu *Mean-shift*. Dále, na základě dosažených segmentací, poskytuje objektivní náhled možnosti reálného použití tohoto algoritmu pro segmentaci obrazu s ohledem na rozdílné případy použití.

Klíčová slova: Mean-shift, Segmentace obrazu, Spektrální shlukovaní, Difuzní mapa

Abstract

In recent years, spectral clustering has established itself as an robust segmentation algorithm. Using spectral clustering for, particularly real, image segmentation opens a wide scope to optimize and modify this algorithm further. This thesis introduces the theoretical background of spectral clustering algorithm focusing on its different modifications including diffuse spectral clustering. Experimental part of this thesis focuses on the implementation of spectral diffuse clustering using the Mean-shift algorithm and based on its outputs, using both real and synthetic inputs, it provides a sober perspective of possibilities of using spectral clustering for image segmentation concerning various use cases.

Keywords: Mean-shift, Image segmentation, Spectral clustering, Diffusion map

Seznam použitých zkratek a symbolů

CCD	–	Charge-coupled device
GMS	–	Gaussian Mean-shift
RAM	–	Random Access Memory
CPU	–	Central Processing Unit
DFT	–	Discrete Fourier Transform
CSR	–	Compressed Sparse Row
CSC	–	Compressed Sparse Column

Obsah

Seznam tabulek

Seznam obrázků

List of Algorithms

1 Úvod

Stejně tak, jako se lidé a některá zvířata na základě předchozích zkušeností učí, jak rozpoznávat a reagovat na zrakové podněty, je pro automatizovanou segmentaci obrazu potřeba algoritmus, který definuje, jak má vstupní obrazy zpracovat stroj. Vzhledem k účelu takových algoritmů jsou nejčastěji nazývány jako segmentační algoritmy. Cílem segmentačních algoritmů je rozdělit vstupní obraz na oblasti tak, aby jednotlivé shluky korespondovaly s jeho přirozenými částmi a zachycenými objekty.

Obecné datově segmentační algoritmy nacházely své uplatnění už daleko před samotnou aplikací v oblasti zpracování obrazu a stejně tak mohou algoritmy původně navržené pro tuto inženýrskou disciplínu sloužit v jiných oblastech vědy. Různé segmentační algoritmy zpravidla poskytují rozdílné segmentace, výpočetní složitost a obtížnost samotné implementace. Výpočetní čas segmentačních algoritmů je přímo závislý na velikosti segmentované množiny. V oblasti segmentace obrazu tyto množiny nezřídka čítají miliony prvků v závislosti na počtu pixelů segmentovaného digitálního obrazu. Segmentace obrazu proto stále představuje výpočetní výzvu a často je nutné volit vhodný kompromis mezi kvalitou a rychlostí výsledku. V praxi se proto často používají aproximační verze algoritmů, které jsou schopny získat odpovídající výslednou segmentaci ve zlomku času výpočtu původního algoritmu.

Základní charakteristikou, dle které jsou algoritmy schopny segmentovat digitální obrazy, je pozice a jas, popřípadě barva jednotlivých pixelů. V takovém případě je splněno mnoho podmínek segmentace tak, jak ji intuitivně chápou živé bytosti, tj. navzájem si blízké pixely s podobnou či dokonce stejnou barvou bývají vyhodnoceny jako totožný segment. V praxi se nicméně vyskytuje mnoho případů, kdy tato jednoduchá a intuitivní metrika selže, ať už z důvodů ošidnosti vstupu, nebo působením dalšího nežádoucího vlivu - velmi často šumu. Z tohoto důvodu se hledají transformace vstupních dat do prostoru jiných vlastností (*feature space*), ve kterém jsou segmenty algoritmem snadněji rozpoznatelné.

Segmentace obrazu je důležitým krokem předzpracování obrazu pro počítačové vidění a rozpoznání objektů. V tomto kontextu jsou segmenty zjednodušenou aproximací zpracovávané scény. Další metriky, například tvar, velikost a umístění segmentů pak mohou být použity pro jejich klasifikaci.

Tato práce prezentuje segmentaci obrazu na základě spektra Laplaceovy matice grafu obrazu použitím standardního shlukovacího algoritmu *Mean-shift*. Tato metoda je převážně založena na obecné teorii grafů, konkrétně na převedení problému segmentace obrazu na problém nalezení optimálních podgrafů v matici sousednosti obrazu. Součástí práce je implementace algoritmu v jazyce *C++*, jehož popis a experimentální výsledky jsou prezentovány v dalších kapitolách. Mimo implementace standardního algoritmu se práce soustředí na implementaci a evaluaci jeho difuzní varianty.

Text práce je rozdělen do několika kapitol. Kapitola 2 čtenáře seznamuje se základními pojmy problematiky a základním vztahem mezi lidským a počítačovým viděním.

Kapitola 3 popisuje dlouhodobě ověřené segmentační algoritmy a metriky. Zvýšený důraz je kladen zejména na skupinu shlukovacích algoritmů, jež jsou úzce spojeny s té-

matem této práce.

Kapitola 4 objasňuje teoretický základ algoritmu spektrálního shlukování, jeho souvislost s problematikou grafových řezů a vlastnosti spektra Laplaceovy matice grafu. Dále jsou v této kapitole podrobněji definovány vztahy pro sestrojení matic potřebných v jednotlivých krocích výpočtu. V neposlední řadě se kapitola věnuje popisu difuzních souřadnic a výpočtu difuzní mapy vstupní datové množiny.

Kapitola 5 představuje implementační část této práce - aplikaci řešící segmentaci obrazu metodou spektrálního shlukování s podporou difuzních souřadnic. Shlukovací část algoritmu je v této práci řešena algoritmem *Mean-shift* namísto obvykle používaného *k-means*. Mimo jiné jsou zde popsány vstupy, výstupy a parametrizace algoritmu včetně nástrojů a externích knihoven použitých při implementaci.

Kapitola 6 prezentuje dosažené výstupy algoritmu na syntetických a reálných vstupech z pohledu kvality segmentace a výpočetního času. Dále porovnává výsledky bez a s použitím difuzních souřadnic a ukazuje vliv inicializační parametrizace a rozměrů vstupního obrazu na výše zmíněná kritéria. Blíže jsou ověřeny vlivy hodnot difuzního parametru, nastavení podobnostní funkce, nastavení algoritmu *Mean-shift* a počet vlastních vektorů použitých ke shlukování.

2 Obraz a jeho reprezentace

Zrak je jeden z nejdůležitějších z pěti základních smyslů člověka. Tímto smyslem lidé zpracovávají přibližně 70-80 % všech informací [4]. Lidské oko je přizpůsobeno ke vnímání určitého spektra světla, které lidský mozek na základě předchozí zkušenosti vyhodnocuje jako nám známé tvary, barvy, objekty a scény. Na rozdíl od člověka a dalších živočichů, kteří mají tuto úlohu značně zautomatizovanou, představuje reprezentace digitálních obrazů nespočet výpočetních výzev a kompromisů.

Tato kapitola vysvětluje základní pojmy spjaté s vnímáním světla, tvorbou obrazu, obrazovou funkcí a reprezentací obrazu v digitálním i reálném světě. Cílem je poukázat na souvislosti a odlišnosti mezi vnímáním a zpracováním obrazového signálu člověkem a počítačem. Stejně tak jsou představena určitá omezení a kompromisy digitalizace.

Použití těchto základních pojmů lze později najít i v dalších sekcích této práce.

2.1 Vnímání světla

Viditelné světlo je elektromagnetické záření o vlnové délce přibližně 390 nm - 790 nm. Světlo do lidského oka vstupuje čočkou, která světlo rozprostře na sítnici. Fotosenzitivní buňky sítnice, tyčinky a čípky, dopadající záření zpracují na elektrické impulsy, které jsou nervovou soustavou přeneseny do mozku. Mozek tyto impulsy vyhodnotí a umožní nám vnímat výsledný obraz.

Na tomto principu je založena většina dnešních obrazových záznamových zařízení. Srdcem všech takových zařízení je fotosenzitivní prvek zastupující funkci sítnice v lidském oku. V digitálních přístrojích je takovým prvkem CCD čip. Světlo dopadající optickým aparátem na tento optický prvek je vzorkováno v závislosti na rozlišovací schopnosti čipu a následně v digitální formě uloženo na příslušné médium.

Na rozdíl od lidského oka mohou CCD čipy podle potřeby zaznamenávat i obrazy vzniklé snímáním jiného světelného spektra. Jako příklad lze použít zaznamenávání infra červeného záření pro účely zaznamenání tepelné stopy scény. Infračervené záření svou vlnovou délkou navazuje na viditelné spektrum světla (700 nm) a pokračuje až k vlnové délce 1 mm. Takto vzniklé obrazy mohou být použity pro tepelné analýzy, noční vidění či záchranné účely.

2.2 Obrazová funkce

Mějme spojitou funkci $f(x)$ reprezentující jednorozměrný signál. Takto definovanou funkci lze vnímat jako reprezentaci jednorozměrného spojitého obrazu. Obrazy jsou ze své podstaty častěji vnímány jako dvourozměrné, proto jsou intuitivněji reprezentovatelné dvourozměrnou funkcí $f(x, y)$, kde x, y jsou prostorové koordináty obrazu.

Spojitá reprezentace obrazových signálů není vhodná pro počítačové zpracování, je proto nutné tento signálovou funkci digitalizovat, nejčastěji vzorkovacím procesem. Vzorkování je metoda diskretizace spojitých funkcí na základě odebírání vzorků spojitého signálu v definovaných krocích. Vzorkování spojitého jednorozměrného signálu si lze představit jako jeho rozdělení na stejně velké diskrétní úseky, které lze dále vnímat jako

jednotlivé hodnoty. V případě dvojrozměrného spojitého signálu se jeho funkční hodnoty, popřípadě jejich interpolace, promítají do uzlů symetrické mřížky. Takto získanou diskrétní reprezentaci obrazu lze snadno dále zpracovávat jako diskrétní pole hodnot či vektorů.

Se vzorkováním signálu je obecně spjata nevyhnutelná ztráta informace původního signálu, a tudíž dochází ke zkreslení signálu. S rostoucí velikostí vzorkovacího kroku roste i zkreslení signálů (roste počet potenciálně zanedbaných, popřípadě interpolovaných hodnot). V ideálním případě je z pohledu kvality výsledného signálu ideální volit nekonečně malý vzorkovací krok. V takovém případě je získán nekonečný počet vzorkovaných diskrétních hodnot potřebující své místo k uložení. Stejně tak, jako rostou výpočetní a paměťové možnosti počítačů, zvyšuje se i rozlišovací schopnost a kvalita obrazových snímačů. Tvorba a zpracování digitálních obrazů je tak stálým kompromisem mezi kvalitou a rychlostí jejich zpracování.

2.3 Digitální obraz

Digitálním obrazem rozumíme 2-rozměrný rastr hodnot reprezentovaný obrazovými body, pixely. Celkový počet pixelů v obraze n je dán vztahem $n = h \times w$, kde h je počet pixelů ve vertikálním směru (výška obrazu) a w je počet pixelů ve směru horizontálním (šířka obrazu). Dnes běžně používané obrazové snímače jsou schopny zaznamenat obrazy čítající řádově milióny těchto obrazových bodů. (Rozlišení kamer a fotoaparátů je často uváděno v "megapixelech"- Mpix.)

V závislosti na barevnosti rozlišujeme mezi následujícími základními typy obrazů:

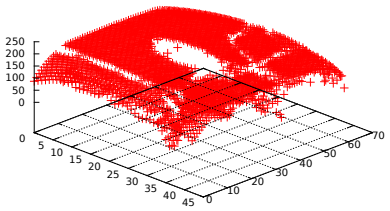
- Černobílý obraz
- Obraz v odstínech šedi
- Barevný obraz

Černobílý, monochromatický obraz lze zakódovat jedním bitem pro každý pixel. Bílé barvě odpovídá hodnota 1 a černé hodnota 0. Obrazy v odstínech šedi jsou nejčastěji kódovány 8 bity / pixel. Takto lze zaznamenat 256 odstínu šedi hodnotami 0 až 255, kde 0 reprezentuje černou a 255 bílou barvu.

Barevné obrazy, nehledě na zvolený grafický model, lze reprezentovat třemi hodnotami. V případě modelu RGB kódujeme každou z barev R (červená), G (zelená) a B (modrá) jedním bajtem. Tento bajt hodnotami 0 až 255 vyjadřuje zastoupení dané barvy v pixelu.

V praxi se pak pro uložení obrazů v digitálním formátu používá komprese. Mezi nejznámější komprimované obrazové formáty patří JPEG pro statické obrazy a MPEG pro video sekvence. V obou případech se jedná o ztrátové kompresní algoritmy umožňující vynecháním nedůležitých částí informace zredukovat velikost na zlomek velikosti původního souboru. V případě dvou výše zmíněných formátů jsou filtrací odstraněny vysoké frekvence frekvenčního spektra obrazu získaného diskrétní *Fourierovou transformací*

(a) Digitální obraz.



(b) Diskrétní obrazová funkce.

Obrázek 1: Ukázka digitálního šedotónového obrazu (1a) a jeho diskrétní obrazové funkce (1b).

(*DFT*). Vzhledem k tomu, že tato práce popisuje zpracování obrazu jako nekomprimovaného datového pole znázorňujícího jednotlivé pixely v mřížce, není věnována další, hlubší pozornost těmto kompresním formátům.

2.3.1 Maticová reprezentace obrazu

S ohledem na dvourozměrnou povahu obrazů se v praxi používá jejich maticová reprezentace, kde jednotlivé prvky zastupují barvu pixelů. Z předchozího rozdělení je zřejmé, že tyto hodnoty lze reprezentovat skalárem v případě obrazů v odstínech šedi, popřípadě trojrozměrným vektorem $v = (r, g, b)$ v obrazech barevných. Zápisem $a_{i,j}$ je obecně označován pixel na i-tém řádku a j-tém sloupci obrazu O.

$$O = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,w} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,w} \\ \vdots & \vdots & \ddots & \vdots \\ a_{h,1} & a_{h,2} & \cdots & a_{h,w} \end{pmatrix} \tag{1}$$

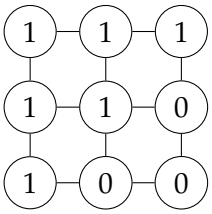
2.3.2 Grafová reprezentace obrazu

Pro účely segmentace lze na obrazy nahlížet jako na neorientované grafy. Graf $G = (V, E)$ je definován množinou vrcholů V a množinou hran E, které vrcholy spojují. Obraz reprezentovaný maticí lze převést na graf tak, že prvky matice (pixely) tvoří vrcholy a hrany pak reprezentují jejich vzájemnou sousednost v původní mřížce. Hrany mohou být doplněny o váhy, které na základě váhové funkce a vzájemné vzdálenosti vyjadřující podobnost či rozdílnost dvou vrcholů hranou spojených.

Pro přiblížení grafové reprezentace definujme triviální binární obraz O o rozměrech 3×3 pixely ($n = 9$) jako rastr reprezentovaný následující maticí

$$O = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}. \tag{2}$$

Jeho grafem rozumíme neorientovaný graf G zobrazený na obrázku 2. Počet vrcholů $|V|$ grafu G odpovídá počtu prvků n původní matice O. Platí tedy vztah $|V| = n = 9$. Počet hran $|E|$ grafu G je dán rozměry původního obrazu a typem použité sousednosti. V případě čtyř-sousednosti platí $|E| = 2|V| - w - h$. V případě osmi-sousednosti pak platí $|E| = 4|V| - 3(w + h) + 2$, kde w je šířka a h výška obrazu.



Obrázek 2: Ukázka reprezentace obrazu grafem.

3 Segmentace obrazu

Tato kapitola hlouběji přibližuje problematiku segmentace obrazu a její současné uplatnění. Vysvětluje s tímto tématem spjaté pojmy, podobnost a vzdálenost, které jsou použity i v dalších kapitolách tohoto textu. V neposlední řadě čtenáře seznamuje se základním rozdělením a průřezem často používaných segmentačních metod. Zvýšená pozornost je věnovaná zástupcům shlukovacích segmentačních metod, jež jsou použity v implementační části této práce.

Segmentací obrazu se rozumí proces rozdělení obrazu na celky, segmenty. V digitálním obraze jsou takové segmenty tvořeny skupinou vzájemně podobných pixelů, které mohou být pro další zpracování vnímány jako souvislá oblast či objekt. Segmentace obrazu umožňuje zjednodušit vstupní obraz tak, že jsou mezi obrazovými body definovány podobnostní závislosti. Tyto závislosti přidávají k množině diskrétních pixelů často kritickou informaci o významu vstupního obrazu. Stejně tak, jako lidský mozek dokáže rozpoznat objekty ve scéně na základě jejich tvaru a barevného vyznění, je pro účely počítačového vidění a autonomního rozpoznání objektů kritická informace o jednotlivých obrazových segmentech.

Segmentační metody jsou algoritmy, jejichž vstupem je obraz určený k segmentaci a výstupem popis reprezentace jeho segmentů. Jedním z problémů segmentace obrazu je ne-jednoznačnost této úlohy. Lze si představit nepřeberné množství obrazů, u kterých neexistuje jen jedna jediná správná segmentace. Stejně tak jako různí lidé mohou vnímat stejné obrazy odlišně, mohou různé, popřípadě různě parametrizované segmentační algoritmy ke stejným vstupům poskytnout rozdílný výstup. Neexistuje tedy ideální segmentační algoritmus vhodný pro libovolnou úlohu, stejně tak nelze hovořit o ideální parametrizaci libovolné z metod. Segmentační metody i jejich parametrizace musí být pro dosažení ideálního výsledku vybrány na míru konkrétním úlohám a očekávaným výstupům.

3.1 Současné aplikace segmentačních metod

Vhodná segmentace obrazu je základním prvkem počítačového vidění. Automatizace založená na těchto disciplínách již pronikla do mnoha odvětví průmyslu a stále častěji se s ní lze setkat i ve všedních aplikacích. V lékařství jsou snímky a video sekvence analyzovány za účelem odhalení anomálií a správného určení diagnózy. S rostoucí databází takto získaných, lékařsky potvrzených a klasifikovaných dat roste i schopnost algoritmů se z těchto dat učit a určit správnou diagnózu, popřípadě navrhnout léčbu autonomně bez zásahu člověka. Počítačové vidění zažívá svůj obrovský rozmach v robotice a dopravě. Dnešní auta jsou s pomocí kamer a senzorů schopna se nejen vyhnout či upozornit na potenciální překážky a nebezpečí, ale dokonce se zcela autonomně pohybují po běžných pozemních komunikacích, což z hlediska počítačového vidění představuje mnoho výzev. Za všechny pak uveďme detekci a rozpoznání dopravních značení, detekci chodců a jiných volně se pohybujících objektů, rozpoznání dopravní signalizace. Detekce obrazu má nepochybně své místo ve strážných systémech, kde slouží k rozpoznávání osob, obličejů, vozidel a dalších entit. Tato data mohou být použita jak k vyhodnocení případné

bezpečnostní hrozby, tak například k dalšímu statistickému zpracování a vyhodnocení. Vzhledem k relativní dostupnosti digitálních kamer a fotoaparátů a vysokému výkonu čipů a procesorů lze najít segmentační algoritmy prakticky kdekoliv.

3.2 Vzdálenost a podobnost

Vzdálenost a podobnost jsou základní pojmy, které si našly své místo také ve zpracování obrazu. Tyto pojmy tvoří základ pro mnoho obecných datově segmentačních metod. Vzdálenost dvou bodů lze v tradičním smyslu chápat jako nezápornou skalární veličinu často označovanou jako δ popisující vzájemnou odlehlost dvou prvků. V oblasti zpracování obrazu, kde jsou body, pixely, mimo jejich pozičních koordinátů definovány i jejich barevností či jasem, lze najít mnoho různě dimenzionálních prostorových reprezentací téhož obrazu. Jednotlivé obrazové body x_i jsou reprezentovány vektory eukleidovského prostoru R^d. Vzdáleností dvou bodů tedy v tomto textu obecně označujeme jejich eukleidovskou vzdálenost podrobněji rozebranou v následují sekci 3.2.1.

Pojem podobnosti se často vztahuje na porovnávání dvou obrazů. Porovnávání podobnosti dvou obrazů je složitá úloha už z její velmi subjektivní podstaty. Za podobné obrazy lze označit obrazy se stejným barevným tónem, průběhem histogramu, členitosti hran a mnoha dalších faktorů. Bez větší námahy najdeme pro každé z takových kritérií dvojici obrazů, které jsou vyhodnoceny jako podobné ba i dokonce shodné, nicméně vyhodnocení člověkem bude opačné. Jako příklad zvažme dvojici binárních obrazů o stejných rozměrech a stejném poměru černých a bílých pixelů. Nehledě na jejich rozmístění, tedy nehledě na výsledný obraz, jsou histogramy těchto dvou obrazů totožné a podobnost založená na jejich porovnání stoprocentní.

Termín "podobnost bodů", ve smyslu v jakém je uváděna v této práci, odkazuje na vzájemnou podobnost dvou diskrétních obrazových bodů založenou na jejich vzdálenosti. Čím jsou si body v R^d blíže, tím jsou si podobnější. Podobnost je vyjádřena zvolenou váhovou funkcí, přijímající jako vstupní parametr vzdálenost porovnávaných bodů. Váhové funkce jsou blíže popsány v kapitole 3.2.2.

3.2.1 Eukleidovská vzdálenost

Euklidovská vzdálenost je metrika umožňující v eukleidovském prostoru $\mathbb{R}^d$ měřit vzdálenost dvou bodů. Mějme dva body x a y reprezentovány vektory $\vec{x} = (x_1, ..., x_d)$ a $\vec{y} = (y_1, ..., y_d)$. Jejich eukleidovská vzdálenost δ je dána vztahem

$$\delta = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2} = \sqrt{(x_1 - y_1)^2 + ... + (x_d - y_d)^2}. \tag{3}$$

Tato vzdálenost je stejně tak vyjádřením normy, velikosti, vektoru $\vec{z}$ vzniklého vzájemným odečtením jednotlivých vektorů $\vec{z} = \vec{x} - \vec{y} = (x_1 - y_1, ..., x_d - y_d)$. Kombinací

výše zmíněných faktů získáváme následující vztah

$$||\vec{z}|| = \sqrt{\sum_{i=1}^{d}(z_i)^2} = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2} = \delta. \tag{4}$$

Vzhledem k tomu, že výpočet pracuje s kvadrátem rozdílu vektorů, není zapotřebí zohledňovat pořadí vektorů při jejich odčítání tedy $||\vec{x} - \vec{y}|| = ||\vec{y} - \vec{x}||$.

3.2.2 Váhová funkce

Definujme váhovou funkci $f(\delta)$ jako jednorozměrnou reálnou funkci popisující podobnost dvou bodů na základě jejich vzájemné vzdálenosti δ. Tato funkce nejčastěji nabývá funkční hodnoty z intervalu $< 0, 1 >$ tak, že podobnost s narůstající vzdálenosti δ klesá. Dva nekonečně blízké body jsou ohodnoceny hodnotou jedna a naopak nulou jsou ohodnoceny body nekonečně vzdálené. Podobnost lze v určitém smyslu chápat jako pravděpodobnost, že dva body k sobě vzájemně patří, tj. tvoří jeden stejný segment. Mezi populární váhové funkce patří Gaussova funkce.

3.2.2.1 Gaussova funkce Gaussova funkce nejčastěji definována jako reálná funkce jedné proměnné x popsaná vztahem

$$f(x) = \alpha \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right), \tag{5}$$

kde α, μ a σ jsou reálné parametry ovlivňující její tvar. Parametr α definuje maximum funkce, μ posouvá maximum podél osy x a parametr σ určuje strmost růstu a klesání.

Normalizovaná Gaussova funkce popisuje normální rozdělení spojité náhodné veličiny. Normalizací rozumíme přizpůsobení parametru α, tak aby určitý integrál této funkce nad celým jejím definičním oborem byl roven jedné dle následujícího vztahu

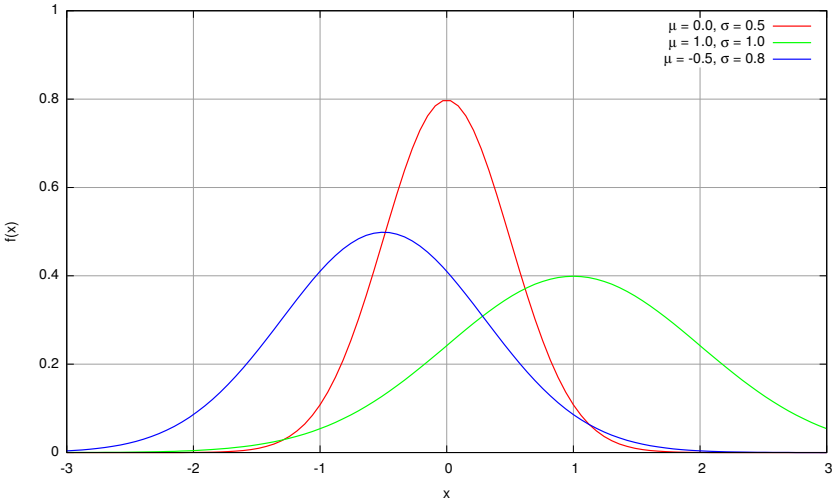
$$\int_{-\infty}^{+\infty} f(x)\mathrm{d}x = 1. \tag{6}$$

Tuto podmínku obecně splňuje $\alpha = \frac{1}{\sigma\sqrt{2\pi}}$. V případě normalizované Gaussovy funkce je parametr μ střední hodnotou náhodné veličiny a σ směrodatná odchylka. Obrázek 3 zobrazuje průběhy rozdílně parametrizované normalizované Gaussovy funkce.

Normální rozdělení často aproximuje velkou část náhodných jevů vyskytujících se v reálném světě, a často se proto požívá ke zpracování rozsáhlých statistický souborů. Použití Gaussovy funkce, včetně její dvourozměrné varianty, se osvědčilo i v oblasti zpracování obrazu a signálu obecně.

Zmiňujeme-li použití Gaussovy funkce ve spojitosti s popisem podobnosti dvou bodů, volíme $\alpha = 1$, $\mu = 0$. Dosadíme-li za parametr x vzdálenost dvou bodů δ, získáme vztah

$$f(\delta) = \exp\left(-\frac{\delta^2}{2\sigma^2}\right). \tag{7}$$



Obrázek 3: Ukázka normálního rozdělení s různou parametrizací.

V případě $\delta = 0$ je funkční hodnota $f(\delta) = 1$. S rostoucí vzdálenosti δ funkční hodnota $f(\delta)$, tedy i vzájemná podobnost klesá.

3.3 Rozdělení segmentačních metod

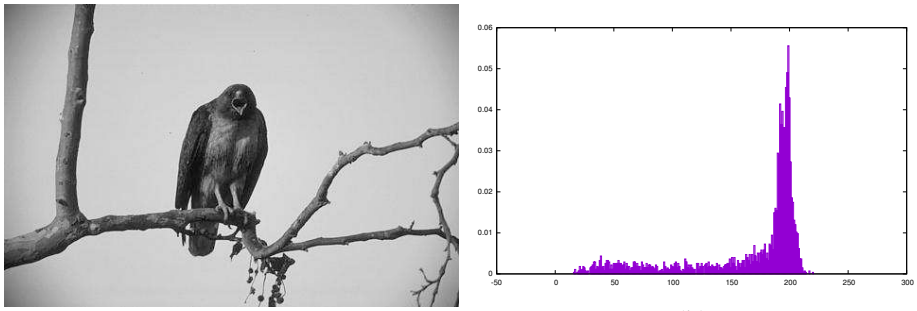
S rozvojem výpočetní techniky se stále rozšiřuje už tak velmi pestrá paleta datově segmentačních algoritmů nacházejících své uplatnění i ve zpracování a analýze obrazu. Tyto metody se široce liší v jejich výpočetní složitosti, implementační složitosti, reprezentací vstupních dat a v neposlední řadě výslednou segmentací. Rozdílně segmentované obrazy mohou být použity k různým typům dalšího zpracování a vyhodnocení. Vzhledem k rozsáhlosti celé problematiky se následující odstavce s ohledem na téma práce věnují pouze těmto třem nejznámějším typům segmentačních metod:

- Metody založené na histogramu
- Metody založené na detekci hran
- Shlukovací metody

Tyto základní skupiny a jejich konkrétní zástupci mají své pevné místo v oblasti zpracování obrazu a v tomto textu jsou s jejich pomocí definovány další důležité pojmy této problematiky.

3.3.1 Metody založené na histogramu

3.3.1.1 Histogram Histogram je grafickým znázorněním distribuce dat. Jedná se o sloupcový graf, kde jednotlivé sloupce reprezentují třídy dat a výška jednotlivých sloupců



(a) (b)

Obrázek 4: Ukázka šedotónového obrazu (4a) a jeho histogramu (4b).

vyjadřuje jejich četnost. Ve světě digitálních obrazů histogram znázorňuje míru zastoupení jednotlivých jasů. V případě barevných obrazů se setkáváme s několika histogramy znázorňující míry jasů jednotlivých barevných složek.

3.3.1.2 Prahování Prahování je jednou z nejstarších segmentačních metod. Tato metoda za jistých okolností umožňuje efektivně oddělit pozadí od objektu našeho zájmu. Definujme prahovací funkci

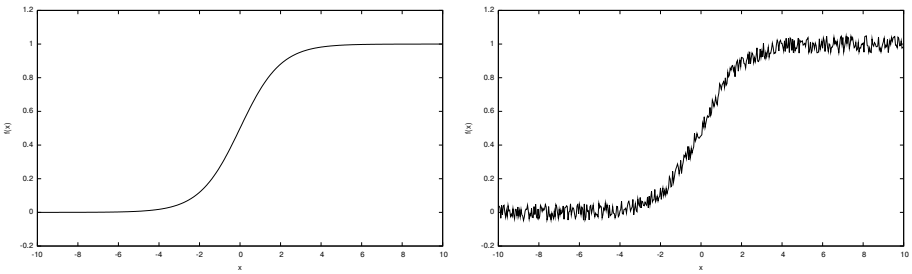
$$f(x) = \begin{cases} 1 & \text{pokud } f(x) \leq t \\ 0 & \text{pokud } jinak \end{cases}. \tag{8}$$

Tato funkce na základě zvoleného prahu t vrací jednu ze dvou hodnot pro libovolnou proměnnou x, tedy klasifikuje data do dvou tříd. V segmentaci obrazu je prahem t hodnota jasu oddělující pozadí od popředí. K určení této hodnoty se často používá histogram, který vizuálně znázorňuje zastoupení jednotlivých jasů. Volba prahu pak připadá na místa vizuálně oddělující hojně zastoupené skupiny (údolí v grafu). Prahovací funkci lze podle potřeby snadno rozšířit o více prahů, což přináší rostoucí obtížnost jejich správné volby. V případě volby k prahů budou data segmentovány do $k + 1$ tříd.

3.3.2 Metody založené na detekci hran

Detekce hran je jednou z elementárních úloh segmentačních úloh. Hranou rozumíme souvislou linii oddělující dvě jasově výrazně odlišné oblasti. Jednotlivé segmenty v segmentovaném obraze jsou tvořeny segmenty ohraničenými právě takto vzniklými hranami. Výsledným obrazem těchto algoritmů je černobílý obraz, kde jedna z barev znázorňuje hrany v původním obraze. Mezi populární metody detekce patří metody založené na první či druhé derivaci funkce obrazového signálu, použití konvoluce s vhodnou konvoluční maskou, popřípadě velmi oblíbený a robustní Cannyho detektor hran. Následující odstavce čtenáři poskytnou náhled na tyto algoritmy.

Definujme hranu jako místo oddělující oblasti s výrazně odlišnými jasy. Obrázek 5 představuje model hrany v ideálním i zašuměném signálu.



(a) Ideální model hrany. (b) Hrana podrobená šumu.

Obrázek 5: Ukázka ideální (a) a zašuměné hrany (b) v jednorozměrném spojitém signálu.

3.3.2.1 Derivace obrazové funkce Hrany, linie v obraze vnímané jako kontury kontrastních oblastí, lze lokalizovat pomocí metod založených na derivaci obrazové funkce. Intuitivně si lze derivaci obecné funkce f' vyložit jako funkci popisující míru změny v průběhu původní funkce f. Pokud je hodnota derivace funkce v bodě malá, nedochází v okolí daného bodu původní funkce k výrazné změně. Naopak, je-li hodnota derivace funkce velká, ke změně funkčních hodnot v původní funkci v okolí bodu dochází. Ve spojitosti s obrazovou funkcí často o existenci hrany rozhoduje hodnota velikosti gradientu v daném bodě. Detekci hran lze tedy vnímat jako úlohu prahování s tím rozdílem, že namísto prahování přímo vstupní funkce f je práh aplikován na její derivaci f'.

Hrany lze detekovat i na základě druhé derivace obrazové funkce f'', konkrétně pak v bodech, pro které platí $f''(x) = 0$. Tuto podmínku obecně splňují inflexní body původní funkce $f(x)$. S ohlédnutím na ideální model hrany, je to právě inflexní bod, který tvoří její střed. S rostoucím počtem derivací obrazové funkce roste i náchylnost těchto metod k chybné detekci hran v obrazech obsahující šumy.

3.3.2.2 Konvoluce Konvoluce je matematický operátor zpracovávající dvě funkce f a g. V odvětví zpracování signálu je funkce f vnímána jako funkce signálu a funkce g aplikovaný filtr, často také označovaný jako konvoluční jádro. Konvoluční jádro lze vnímat jako signálový filtr. Výsledkem konvoluce je nová funkce představující filtrovaný vstupní signál. Konvoluce jednorozměrných spojitých funkcí f a g je definována jako

$$(f * g)(x) = \int_{-\infty}^{\infty} f(x-a)g(a)\mathrm{d}a. \tag{9}$$

Pro zpracování digitálním obrazů se s ohledem na povahu dat používá dvourozměrná diskrétní varianta konvoluce definovaná následujícím vztahem

$$(f * g)(m,n) = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1} f(m-i, n-j)g(i,j), \tag{10}$$

kde $f(m,n)$ je diskrétní obrazová funkce a $g(m,n)$ konvoluční maska představující obrazový filtr.

K detekci hran v obraze se jako konvoluční masky používají například *Prewittové* či *Sobelův operátor*. V obou případech jsou aproximovány derivace obrazové funkce. Body s vysokou hodnotou derivace jsou označeny jako hrany.

Konvoluční masky S_x a S_y *Sobelova operátoru* detekují hrany v horizontálním, respektive vertikálním směru. Tyto masky jsou definovány jako

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}. \tag{11}$$

Konvolucí obrazové funkce f s uvedenými maskami získáme obrazy $G_x = S_x * f$ a $G_y = S_y * f$, které jsou aproximací parciálních derivací $\frac{\partial f}{\partial x}$, $\frac{\partial f}{\partial y}$. Jejich prahováním lze označit hrany v příslušných směrech. Kombinací G_x a G_y lze na základě aproximovaných parciálním derivací vypočíst velikost gradientu G dle vztahu $G = \sqrt{G_x^2 + G_y^2}$. *Sobelův operátor* lze použít i k určení směru detekovaných hran, respektive úhlu, který svírají s osami souřadného systému. Směr gradientu Φ je definován jako $\Phi = arctan(G_x, G_y)$. Směr hrany je k tomuto směru kolmý.

Metody detekce hran založené na gradientu jsou negativně ovlivněny obrazovými šumy. Mimo operátorů určených pro detekci hran se lze často setkat s operátory rozostření sloužící k redukci šumu. Tyto jsou jsou aplikovány před vlastní detekcí hran. Za všechny pak jmenujme Gaussův operátor vycházející ze stejně pojmenované funkce provádějící Gaussovo rozostření. Gaussův operátor může být aproximován jako

$$K = \frac{1}{159} \begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}. \tag{12}$$

3.3.2.3 Cannyho hranový detektor *Cannyho hranový detektor* je široce používaný algoritmus, který se stal standardem pro nalezení hran v obraze. Canny při návrhu algoritmu definoval následující 3 základní kritéria detektoru:

- Minimalizace chybné detekce, tj. označit místo jako hranu pouze v případě, kdy hranou skutečně je.
- Maximalizace přesnosti určení polohy hrany, tj. označení hrany co nejblíže její skutečné polohy.
- Jednoznačnost, tj. označení hrany pouze jednou.

Hlavní myšlenka, která je skryta za za návrhem *Cannyho hranového detektoru*, je maximalizovat podíl signálu a šumu (*SNR*) hranového detektoru, kde šum představuje pravděpodobnost chybné detekce hrany. Bylo dokázáno, že této maximalizace včetně splnění dalších výše popsaných vlastností lze docílit následujícími kroky:

1. Eliminace šumu v obraze aplikací Gaussova operátoru definovaného vztahem 12.
2. Určení gradientu obrazu na základě parciálních derivací obrazové funkce.
3. Ztenčení hran označením hrany pouze v místě lokálního maxima gradientu.
4. Eliminace nevýznamných hran prahováním pomocí dvojice prahů t_{min}, t_{max}. Pokud je hodnota gradientu vyšší než horní práh t_{max} je toto místo označeno jako hrana. Pokud je hodnota gradientu nižší než dolní práh t_{min} hrana uznána není. Pokud je hodnota gradientu mezi dvěma stanovenými prahy, hrana je uznána pouze v případě, kdy tento bod přímo sousedí s již označenou hranou.

3.3.3 Shlukovací metody

Shlukovací metody mají své nezastupitelné místo v obecných datově segmentačních úlohách. Cílem těchto algoritmů je shlukovat vzájemně blízké, podobné datové body do segmentů, které pak lze reprezentovat vhodným zástupcem umístěným z pravidla v jejich centru. Ukázalo se, že při vhodné interpretaci obrazových dat lze shlukovací metody použít i pro efektivní segmentaci obrazu. Výhodou této skupiny segmentačních metod je jejich zanedbatelná závislost výpočetní složitosti na dimenzi segmentovaných dat. Proto touto reprezentací mohou být jak přímo body v obrazovém prostoru snadno definované pouze svou pozicí a jasem, stejně tak i více sofistikované (a více rozměrné) vektory získané předchozím zpracováním.

3.3.3.1 Mean-shift *Mean-shift* je bezparametrická shlukovací metoda nevyžadující předchozí znalosti o počtu shluků ani jejich tvaru. Tato metoda iterativně hledá maxima funkce hustoty pravděpodobnosti vstupních dat [1]. Vstupními daty v tomto případě označujeme diskrétní množinu $S = \{x_1, x_2, ..., x_n\}$, kde $x_i \in \mathbb{R}^d$ je d-rozměrný bod. Definujme váhovou funkci $K(x, x_i)$, kernel. Tato funkce určuje s jakou váhou se dvojice bodů ovlivňuje. Vážená střední hodnota hustoty $m(x)$ nad výpočetním oknem definovaným daným kernelem K je daná vztahem

$$m(x) = \frac{\sum_{x_i \in S} K(d(x, x_i))x_i}{\sum_{x_i \in S} K(x, x_i)}. \tag{13}$$

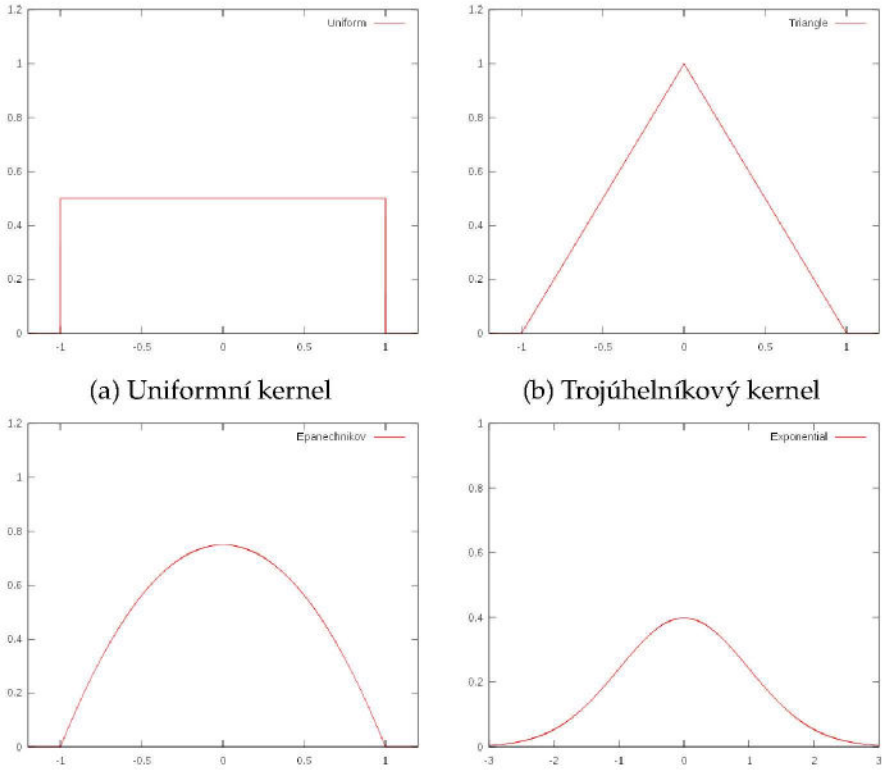
Vizuálně lze na hodnotu $m(x)$ nahlížet jako na centrum shluku, do kterého se bod x přesunul. Body se takto iterativně pohybují do míst s větší hustotou, dokud nedoputují ke svému konečnému atraktoru, ve které je hustota maximální. Každý segment je pak tvořen body, které se přesunuly do stejného atraktoru.

Kernelem K rozumíme radikální symetrickou funkci $k(\delta)$, kde vzdálenost δ je nejčastěji *eukleidovskou vzdáleností* dvojice bodů $\delta = ||x_i - x_j||$. S rostoucí vzdáleností δ klesá váha s jakou se dva dané body ovlivňují (tedy i pravděpodobnost, že náleží stejnému shluku) [2].

V tabulce 1 je uvedeno několik často používaných funkcí kernelu. Jejich průběhy jsou znázorněny na obrázku 6.

Uniformní kernel	$K(x) = \frac{1}{2} 1_{\{\lvert x \rvert \leq 1\}}$
Trojúhelníkový kernel	$K(x) = (1 - \lvert x \rvert) 1_{\{\lvert x \rvert \leq 1\}}$
Epanečnikův kernel	$K(x) = \frac{3}{4}(1 - x^2) 1_{\{\lvert x \rvert \leq 1\}}$
Gaussův kernel	$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$

Tabulka 1: Typy používaných kernelů.



(a) Uniformní kernel (b) Trojúhelníkový kernel

(c) Epanečníkův kernel (d) Gaussův kernel

Obrázek 6: Ukázka kernelových funkcí.

Jedním ze základních používaných kernelů je Uniformní kernel připomínající prahovací funkci. Pokud je vzdálenost bodů menší než stanovený práh t, pak se body ovlivňují s konstantní váhou $w = k$. Takovou funkci lze vyjádřit následujícím vztahem

$$K(x) = \begin{cases} k & \text{pokud } \delta \leq t \\ 0 & \text{pokud } jinak \end{cases}. \tag{14}$$

Takto definovaný kernel nebere v potaz možné rozdíly mezi body splňující podmínku zadaného prahu. Často se na této pozici proto můžeme setkat s jinými, více sofistikovanými funkcemi, které tento nedostatek odstraňují.

Je zřejmé, že volba kernelu ovlivňuje výsledný tvar segmentů i celkový výpočetní čas. Ve spojitosti s Gaussovým kernelem, jehož definiční obor není nijak omezen, a pro libovolnou vstupní vzdálenost vrací nenulovou hodnotu, je vhodné zvážit jeho simplifikaci. Malé funkční hodnoty blížící se nule lze od určitého prahu t vnímat jako hodnoty nulové. Definujme funkci $g'(x)$, která představuje takto "ořezanou"variantu Gaussovy funkce $g(x)$ vztahem 15.

$$g'(x) = \begin{cases} g(x) & \text{pokud } |x| \leq t \\ 0 & \text{pokud } jinak \end{cases} \tag{15}$$

Stejně tak volba kernelu ovlivňuje počet kroků a délku výpočtu algoritmu. V případě segmentace n datových bodů s použitím Gaussova kernelu je časová složitost $O(n^2)$. Přesněji lze obecně počet kroků algoritmu získat vztahem $i \times n \times m$, kde i je počet iteračních kroků, n počet prvků segmentované množiny a m je velikost okolí, které pokrývá kernelová funkce. V případě použití funkce s neomezeným definičním oborem se stává okolím celá vstupní datová množina proto počet kroků algoritmu lze aproximovat jako $i \times n^2$.

Jednotlivé kroky *Gaussian Mean-shift* jsou popsány algoritmem 1.

Algorithm 1 Mean-shift algoritmus zapsaný v pseudokódu.

for $i \in 1, \cdots, n$ **do**
  $x \leftarrow x_i$
  $\forall i : p(i|x) \leftarrow \frac{\exp\left(-\frac{1}{2}||\frac{x-x_i}{\sigma}||^2\right)}{\sum_{i'=1}^{n} \exp\left(-\frac{1}{2}||\frac{x-x_{i'}}{\sigma}||^2\right)}$
  repeat
    $x \leftarrow \sum_{i=1}^{N} p(i|x)x_i$
    $\forall n : p(n|x) \leftarrow \frac{\exp\left(-\frac{1}{2}||\frac{x-x_i}{\sigma}||^2\right)}{\sum_{i'=1}^{n} \exp\left(-\frac{1}{2}||\frac{x-x_{i'}}{\sigma}||^2\right)}$
  until x's update < tol
  $z_i \leftarrow x$
end for

3.3.3.2 K-means

K-means je jedno parametrická shlukovací metoda k nalezení k shluků v n vzorcích. Vzorkem se rozumí vstupní množina dat $S = \{x_1, x_2, ..., x_n\}$, kde $x_i \in \mathbb{R}^d$ je

d-rozměrný vektor reprezentující bod. Algoritmus iterativně hledá shluky tak, aby součet vzdálenosti bodů v rámci jednoho shluku od jeho centra (též označováno jako "mean") byly nejmenší možné, tedy řeší

$$\arg\min_{S} \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_i - \mu_j||^2, \tag{16}$$

kde x_i reprezentuje segmentovaný bod a μ_j pozici centra daného shluku.

Standardně lze pro *k-means* použít aproximační heuristický *Lloydův algoritmus*. Mějme pevně definován požadovaný počet výsledných shluků (segmentů) k. Ve vstupní datové množině je náhodně rozmístěno k centroidů. Body vstupní množiny jsou v jednotlivých iteracích přiřazeny jejich nejbližšímu centroidu. Po každé iteraci se nová pozice centroidů získá interpolací souřadnic jejich přiřazených bodů. Data se považují za segmentovaná, pokud ani jedno z center mezi dvěma následujícími iteracemi nezmění svou pozici. Výsledné segmenty jsou dány body, které náleží jednotlivým centrům po skončení výpočtu.

Stejně jako *Mean-shift*, tak i *k-means* patří k algoritmům založených na metodě "učení bez učitele". Neexistuje tedy žádná zpětná vazba, která by mohla vést k lepším výstupům v závislosti na evaluaci výsledků předchozích.

V případě použití metody *k-means* se očekává alespoň částečná znalost segmentovaných dat, tak aby mohl být alespoň odhadnut očekávaný počet segmentů. Nevhodně zvolená hodnota parametru k může negativně ovlivnit výstup algoritmu.

Z implementačního hlediska je také vhodné se pozastavit náhodným počátečním rozmístěním centrem. Různá počáteční rozmístění mohou vézt k různým výstupům. Snadno si lze představit příklady, ve kterých jsou některá centra rozmístěna tak, že pro žádný z bodů nejsou nejblíže. V takových případech je v praxi vhodné počáteční inicializaci center provést opakovaně.

Problém nalezení globálního optima funkce popsané vztahem 16 je NP-těžký. Nicméně při použití i iterací standardního *Lloidova algoritmu* je časová složitost výpočtu $O(i \times k \times n)$.

4 Segmentace obrazu metodou spektrálního shlukování

Tato kapitola se podrobněji věnuje teoretickému základu spektrálního shlukování a jeho použití pro segmentaci obrazu. Následující odstavce popisují klíčové kroky algoritmu a představují několik různých matic potřebných k jeho běhu. Jádrem algoritmu je spektrální rozklad Laplaceovy matice grafu obrazu. Výpočet různých typů Laplaceových matic včetně výčtu několika jejich důležitých vlastností je popsán v sekci 4.4. S Laplaceovou maticí jsou úzce spojeny také matice sousednosti a matice stupně vrcholů grafu obrazu, kterým je věnován prostor v sekcích 4.2, respektive 4.3. Dále se tato kapitola věnuje samotnému spektrálním rozkladu a vlastnostem spektra Laplaceovy matice, jehož vlastní čísla a odpovídající vlastní vektory jsou použity jako vstupní data pro samotné shlukování. Sekce 4.8 představuje pojem difuzní vzdálenosti a popisuje jak a za jakých okolností je vhodné tuto vzdálenost použít v kontextu spektrálního shlukování.

Některé typy shluků je obtížné detekovat přímo ve vstupních datech. Hlavní myšlenkou shlukování na základě spektra je změna reprezentace segmentovaných abstraktních bodů x_i na body $y_i \in \mathbb{R}^k$. Tato změna často vede k redukci dimensionality shlukovacího problému. Cílem této transformace je stejná data reprezentovat tak, aby shluky byly zřejmější a snadněji lokalizovatelné, čehož může být docíleno reprezentací dat právě spektrem jejich Laplaceovy matice.

Je třeba podotknout, že nalezení vlastních čísel a vektorů je výpočetně náročná úloha. Zvláště pak v oblasti zpracování obrazu zvážíme-li velikosti daných matic, je vhodné se zamyslet, zda je pro segmentaci kritické aby byla data shlukována v prostoru spektra namísto přímého shlukování v prostoru obrazu. Stejná nebo dostatečně podobná segmentace může být v takovém případě přímého shlukování poskytnuta výrazně rychleji. Naopak v určitých případech shlukování na základě spektra poskytuje výrazně přirozenější segmentace a nachází si tak uplatnění i přes svou vyšší výpočetní náročnost.

4.1 Popis algoritmu

V literatuře ([7], [10], [9]) se lze setkat s několika různými algoritmy řešícími problém segmentace obrazu pomocí spektrálního shlukování. Největším rozdílem je použití různých typů Laplaceovy matice a normalizace vektorových hodnot. Následující popis algoritmu je velmi obecný a jednotlivým odlišnostem se pak věnují konkrétní sekce tohoto textu, zvláště pak kapitola 4.4 popisující známé varianty Laplaceovy matice.

Mějme vstupní množinu n bodů $S = \{b_1, b_2, ...b_n\}$, reprezentující data, která chceme segmentovat. Následující kroky vedou k její segmentaci metodou shlukování spektra Laplaceovy matice:

1. Sestrojme matici sousednosti $A \in \mathbb{R}^{n \times n}$ popsanou v kapitole 4.2.
2. Sestrojme diagonální matici $D \in \mathbb{R}^{n \times n}$ popsanou v kapitole 4.3.
3. Sestrojme Laplaceovu matici $L \in \mathbb{R}^{n \times n}$ popsanou v kapitole 4.4.
4. Sestrojme matici $X \in \mathbb{R}^{n \times k}$, jejíž sloupce jsou tvořeny vlastními vektory odpovídající prvním k vzestupně řazeným vlastním číslům $\lambda_0 \leq \cdots \leq \lambda_k$ Laplaceovy matice.

5. Jedním ze shlukovacích algoritmů najděme v matici $X^{n\times k}$ shluky tak, že shlukujeme jednotlivé řádky $X^{n\times k}$ tvořící body v prostoru $\mathbb{R}^k$.

6. Pokud i-tý řádek matice X byl přiřazen segmentu j, pak vstupní bod S_i patří k segmentu j.

4.2 Matice sousednosti

Matice sousednosti A je reálná čtvercová matice vyjadřující podobnost mezi přímo sousedícími vrcholy grafu G v závislosti na jejich vzdálenosti. S rostoucí vzdáleností dvou vrcholů, podobnost, definovaná jednou z váhových funkcí, klesá. Obecně lze ke grafu $G = (V, E)$, kde $||V|| = n$, sestrojit matici sousednosti $A \in \mathbb{R}^{n\times n}$, pro jejíž prvky a_{ij} platí

$$a_{ij} = \begin{cases} w(d(V_i, V_j)) & \text{pokud } i \neq j \\ 0 & \text{pokud } jinak, \end{cases} \tag{17}$$

kde funkce $d(i, j)$ představuje eukleidovskou vzdálenost mezi jednotlivými vrcholy a funkce $w(x)$ je jednou z váhových funkcí vyjadřující míru jejich podobnosti. Zvažujeme-li neorientovaný graf, pak je matice sousednosti symetrická a pro vzdálenost $d(V_i, V_j)$ mezi libovolnými dvěma vrcholy V_i, V_j platí $d(V_i, V_j) = d(V_j, V_i)$.

Počet prvků matice A je kvadrátem počtu vrcholů původního grafu. Z důvodu redukce výpočetní a prostorové složitosti se v souvislosti se spektrálním shlukováním v praxi široce používá několik technik, jak tuto konstrukci zjednodušit. Žádnou z takových konstrukcí nezískáme ideální matici vhodnou pro libovolnou úlohu a často je nutné najít vhodný kompromis mezi kvalitou výsledné segmentace a její rychlostí. Jedním z takových zjednodušení je zvažovat sousednost pouze takových dvojic vrcholů, jejichž vzájemná vzdálenost je menší než pevně definovaný práh. Další, jiná možnost je vytvořit matici sousednosti tak, že každému vrcholu grafu zvažujeme pouze jeho k-nejbližších vrcholů, kde k je konstanta určující jejich počet.

V úlohách v souvislosti se zpracováním a segmentací obrazu se často zkoumá čtyř a osmi-okolí jednotlivých obrazových bodů. Čím menší je vzdálenost mezi jednotlivými pixely, tím spíše budou tyto pixely tvořit jeden segment ve výsledném obraze. Zaměříme se proto na porovnávání pouze přímo přiléhajících pixelů. Čtyř-okolí bodu je definováno jeho přímo přiléhajícími pixely v horizontálním a vertikálním směru. Analogickým rozšířením čtyř-okolí o zbývající přiléhající diagonální pixely získáme osmi-okolí daného bodu. Hraniční body obrazu nemají všechny z těchto sousedících bodů, takové případy z důvodu zjednodušení v tomto textu nepopisujeme. Konstrukce matice sousednosti založená na jednom z těchto přístupů jsou jedny z nejhojněji používaných pro segmentační úlohy založené na spektrálním rozkladu. Jednou z výhod takto získané matice je její řídkost tj. jedná se o matici, která má většinu jejich prvků nulových. Zvažujeme-li obraz o n pixelech, pak jeho matice sousednosti A celkově čítá n^2 prvků. Konstrukcí této matice pomocí čtyř-okolí je maximální počet nenulových prvků $4n$ a $8n$, zvažujeme-li osmi-okolí.

4.2.1 Zápis řídkých matic

Řídké matice jsou v digitální podobě nejčastěji uloženy ve formátech, které zaznamenávají pouze jejich nenulové prvky společně s informací o jejich pozici tak, aby bylo možné původní matici zrekonstruovat. Řídkou matici snadno zapíšeme v souřadnicovém formátu, pro další zpracování se však častěji používá *CSR / CSC* formát.

V souřadnicovém formátu je každý nenulový prvek a_{ij} reprezentován jako trojice $[a_{ij}, i, j]$. K uložení k nenulových prvků tedy potřebujeme $3k$ hodnot. Tento typ zápisu je velmi snadný z pohledu konstrukce, nicméně není tak vyhovující k dalším maticovým výpočtům.

CSC (Compressed Sparse Rows) formát pro uložení matice používá tři pole. První obsahuje hodnoty jednotlivých nenulových prvků a_{ij}. Ve druhém poli, stejně velkém poli jsou uloženy sloupcové indexy j odpovídající prvků a_{ij} seřazené po řádcích. Třetí pole obsahuje indexy prvků prvního pole, které v původní matici zahajují další řádek. Celkově tedy pro uložení k prvků matice $A \in \mathbb{R}^{n\times n}$ ve formátu CSC je potřeba $2k+n$ hodnot. Analogicky lze sestrojit zápis matice ve formátu CSC (compressed sparse columns) s tím rozdílem, že namísto sloupcových indexů jsou ve druhém poli uloženy indexy řádkové a třetí pole obsahuje ukazatele na indexy prvních prvků jednotlivých sloupců původní matice.

Je vhodné zmínit, že v případě symetrických matic je zbytečné ukládat všechny prvky. Bez ztráty informace lze uložit pouze prvky horní, popřípadě dolní trojúhelníkové matice, čímž se počet uložených hodnot zredukuje přibližně na polovinu.

4.3 Matice stupně vrcholů

K sestrojení Laplaceovy matice grafu G je společně s jeho maticí sousednosti popsanou v kapitole 4.2 zapotřebí i diagonální matice D nesoucí informaci o stupních jednotlivých vrcholů grafu. Ke grafu $G = (V, E)$, kde $||V|| = n$ jsou jednotlivé prvky matice $D \in \mathbb{R}^{n\times n}$ definovány vztahem

$$d_{ij} = \begin{cases} deg(V_i) & \text{pokud } i = j \\ 0 & \text{pokud } jinak \end{cases}, \tag{18}$$

kde $deg(V_i)$ je stupeň vrcholu V_i.

V tomto případě zanedbáváme možné váhové ohodnocení hran mezi jednotlivými vrcholy. Proto se často můžeme setkat s funkcí

$$deg'(V_i) = \sum_{i=1}^{n} A_{i,j}, \tag{19}$$

kde A je matice sousednosti grafu G. Diagonální prvky d_{ii} matice D jsou definovány jako součet vážených hran e_{ij} přiléhajících vrcholu V_i.

4.4 Laplaceova matice

Laplaceova matice grafu je základním stavebním kamenem spektrální teorie grafů. V různých textech lze najít její lišící se definice přizpůsobené konkrétním úlohám. Stejně

tak se liší definice Laplaceovy matice v různých textech a její jediná správná definice neexistuje. Pro účely segmentace hledáme Laplaceovu matici, která co možná nejlépe popisuje jednotlivé komponenty původního grafu, popřípadě nastiňuje, jak takové souvislé grafy najít tak, aby vyjadřovaly shluky. V ideálním případě má graf právě tolik komponent, kolikrát se vyskytla nula jako vlastní číslo jeho Laplaceovy matice. V tomto textu se konkrétně zaměříme na obecnou Laplaceovu matici a její normalizovanou formu.

4.4.1 Obecná Laplaceova matice

Obecnou Laplaceovou maticí L grafu G se rozumí matice definovaná vztahy 20 a 21, kde A je matice sousednosti a D matice stupně vrcholu grafu G.

$$L = D - A, \tag{20}$$

$$l_{ij} = \begin{cases} d_{ij} & \text{pokud } i = j \\ -a_{ij} & \text{pokud } jinak \end{cases}. \tag{21}$$

Laplaceova matice grafu a její spektrum prozrazuje mnoho dalších vlastností původního grafu [8]. Mezi nejdůležitější vlastnosti ovlivňující spektrum obecné Laplaceovy matice L patří:

- Pro každý vektor $\vec{y} \in \mathbb{R}^n$ platí

$$\vec{y}^T L \vec{y} = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(y_i - y_j)^2 \tag{22}$$

- L je symetrická, singulární a positivně semi-definitní.
- Nejmenší vlastní číslo λ_1 matice L je 0, tomuto číslu odpovídající vlastní vektor $\vec{u_1}$ je konstantní jednotkový vektor.
- L má n nezáporných reálných vlastních čísel $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$.

První z výše jmenovaných vlastností úzce souvisí s počtem nezávislých komponent grafu a grafovými řezy. Je-li graf tvořen jedinou souvislou komponentou, pak

$$\min_{\vec{y} \in \mathbb{R}^n} f(\vec{y}) = \vec{y}^T L \vec{y} = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(y_i - y_j)^2 = 0. \tag{23}$$

Jelikož je váha hrany w_{ij} nezáporná, výše zmíněný výraz platí, pokud je $\vec{y}$ konstantní. Těmto podmínkám vyhovuje vlastní číslo $\lambda_1 = 0$ a odpovídající konstantní vlastní vektor.

Vycházíme-li z naivní představy, ve které je každý segment v obraze reprezentován jako nezávislá komponenta jeho grafu, lze segmentaci provést na základě vlastních vektorů odpovídajících pouze násobnému nulovému vlastnímu číslu. Lze předpokládat, že jednotlivé obrazové segmenty nemusí nutně být nezávislými komponentami grafu

vstupního obrazu, nicméně vazby mezi vrcholy různých segmentů jsou slabší než vazby mezi vrcholy uvnitř jednoho segmentu. Další vlastní vektory odpovídající nenulovým vlastním číslům umožňují najít grafové řezy vedoucí k logickému rozdělení původního grafu. Prvnímu nejmenšímu nenulovému vlastnímu číslu odpovídá tzv. *Fiedlerův vlastní vektor*, na jehož základě lze vrcholy jedné grafové komponenty (grafu) logicky rozdělit do dvou skupin.

4.4.2 Normalizovaná Laplaceova matice

Normalizovanou Laplaceovou maticí L_{sym} grafu G definujeme jako

$$L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}, \tag{24}$$

kde A je matice sousednosti a D matice stupně vrcholu grafu G.

Popřípadě lze použít normalizovanou Laplaceovou maticí L_{rw} definovanou jako

$$L_{rw} = D^{-1} L = I - D^{-1} A. \tag{25}$$

L_{rw}, na rozdíl od L_{sym}, není symetrická matice a její vlastní čísla a vektory lze získat řešením zobecněného vztahu

$$Lu = \lambda D \vec{u}, \tag{26}$$

kde D je obecná Laplaceova matice a D matice stupně vrcholu grafu G.

4.5 Spektrální rozklad

Spektrálním rozkladem lineárního operátoru (matice) se rozumí výpočet jeho vlastních čísel a vlastních vektorů.

Definujme matici A. Vlastní vektor matice A je nenulový vektor $\vec{u}$, pro který platí

$$A\vec{u} = \lambda \vec{u}, \tag{27}$$

kde λ je vlastní číslo odpovídající vektoru $\vec{u}$. Vlastní číslo je koeficientem s jakým se mění velikost vlastního vektoru při transformaci. Samotný směr vlastního vektoru je po aplikaci transformace neměnný.

Výpočet vlastních čísel lze zapsat následující soustavou rovnic, pro které hledáme nenulové řešení

$$(A - \lambda I)\vec{u} = 0, \tag{28}$$

kde I je jednotková matice stejné dimenze jako matice A.

Nenulové řešení této soustavy získáme, vyřešíme-li rovnici $det(A - \lambda I) = 0$. (Takto získaný mnohočlen na levé straně rovnice je polynomem v praktických úlohách často velmi vysokého řádu, proto nalezení všech kořenů této rovnice je výpočetně velmi náročná úloha.) Jednotlivé kořeny rovnice $\lambda_1, \lambda_2, \cdots \lambda_n$ jsou vlastními čísly matice A. Množina všech těchto vlastních čísel je označována jako její spektrum $\sigma(A)$.

Nalezení příslušných vlastních vektorů $\vec{u}$ je nyní otázkou vyřešení soustavy rovnic vzniklé dosazením jednotlivých vlastních čísel λ do vztahu 28.

V praxi se k výpočtu vlastních čísel namísto výše uvedeného výpočetně náročného postupu často používají rychlejší aproximační metody založené na *Arnoldiho metodě*. Hlavní myšlenkou *Arnoldiho metody* je nahradit výpočet vlastních čísel původní řídké matice A výpočtem vlastních čísel její ortogonální projekce do *Krylovova prostoru*. Vlastní čísla čísla této projekce reprezentované menší Hessenbergovou maticí H jsou pak považována za aproximaci vlastních čísel původní matice A. Lze se také setkat s *Lanczoszovou metodou*, která je vnímána jako zjednodušení *Arnoldiho metody* pro symetrické matice.

4.6 Transformace datových souřadnic

K Laplaceově matici $L \in \mathbb{R}^{n \times n}$ sestrojme matici $X \in \mathbb{R}^{n \times k}$ tak, aby jednotlivé sloupce matice X byly tvořeny vlastními vektory odpovídající vzestupně řazeným vlastním číslům spektra $\sigma(L) = \{\lambda_1 \leq \cdots \leq \lambda_k\}$.

Vycházíme-li z předpokladu, že vlastnímu číslu λ_i odpovídá vlastní vektor $\vec{u_i} = (u_{i,1}, u_{i,2}, \cdots, u_{i,n})$ lze matici X zapsat následujícím vztahem

$$X = (\vec{u_1}^T, \vec{u_2}^T, \cdots, \vec{u_k}^T) = \begin{pmatrix} u_{1,1} & u_{2,1} & \cdots & u_{k,1} \\ u_{1,2} & u_{2,2} & \cdots & u_{k,2} \\ \vdots & \vdots & \ddots & \vdots \\ u_{1,n} & u_{2,n} & \cdots & u_{k,n} \end{pmatrix} \tag{29}$$

.

S ohledem na původní, obecnou vstupní množinu dat $S = \{x_1, x_2, \cdots, x_n\}$ jsou tyto body nyní reprezentovány novými body $x_i \in \mathbb{R}^k$ odpovídající jednotlivým řádkům matice X. Algoritmus tedy hledá shluky v n k-dimenzionálních bodech.

4.7 Shlukování spektra

Shlukováním spektra je myšleno nalezení shluků na základě vhodně uspořádaných hodnot vlastních vektorů náležejících vybraným vlastním číslům spektra Laplaceovy matice. Použití omezeného počtu vlastních vektorů často vede k redukci dimenzionality shlukovacího problému.

Samotné shlukování bodů, získaných transformací blíže popsanou v předchozí kapitole 4.6, může být provedeno libovolným shlukovacím algoritmem, který dokáže hledat shluky v množině S k-dimenzionálních bodů $S = x_1, x_2, \cdots, x_n$, kde $x_i \in \mathbb{R}^k$. Typickými zástupci těchto algoritmů jsou *k-means* a *Mean-shift*, jejichž podrobnějšímu popisu je věnována sekce 3.3.3. Převážná většina současných prací zaměřených na shlukování spektra používají ke shlukování metodu *k-means* vzhledem k její účelnosti a snazší implementaci. V implementační části této práce se dále zaměříme na použití metody *Mean-shift*.

Jednoznačnou výhodou použití metody *Mean-shift* oproti *k-means* je fakt, že *Mean-shift* nevyžaduje jako vstupní parametr počet předpokládaných shluků. Lze proto předpokládat, že použití *Mean-shiftu* je univerzálnější bez ohledu na povahu a dopřednou znalost vstupních dat.

4.8 Difuzní spektrální shlukování

Difuzí se nejčastěji rozumí proces pohybu látky z oblasti s vyšší koncentrací do oblasti s koncentrací nižší. Tento jev lze obecně použít i k modelování jiných systémů například k modelu šíření tepla materiálem, šíření náboje prostředím nebo šíření energie v uzavřeném elastickém systému. Dle druhého termodynamického zákona difuzí dochází ke zvyšování entropie a tedy k dosažení nejnižší možné vnitřní energie daného systému.

Obecný proces difuze lze modelovat parciální diferenciální rovnicí ve tvaru

$$\frac{\partial f(x,t)}{\partial t} = \nabla \cdot (g(f(x,t),x)\nabla f(x,t)). \tag{30}$$

Tato rovnice popisuje stav systému popsaného funkcí $f(x)$ v čase t. Funkce g je difúzním koeficientem, který se obecně rovněž může vyvíjet s časem. V našem případě rozumíme difuzí vyrovnávání koncentrací barev (jasů) obrazové funkce f. Difúzní koeficient definovaný na základě vah podobnostního grafu obrazu zůstává s časem konstantní, proto lze rovnici 30 zapsat jako

$$\frac{\partial f(x,t)}{\partial t} = \alpha\nabla^2 f(x,t) = Lf(t), \tag{31}$$

kde α je koeficient rychlosti šíření látky v celém systému a ∇^2 je v Laplaceův operátor L (v našem diskrétním případě Laplaceova matice) definující rychlost šíření látky v mezi jednotlivými částmi systému. Často se lze s rovnicí 31 setkat jako s rovnicí popisující šíření tepla materiálem. Vektor f(t), jehož prvky odpovídají původním v čase t, lze vyjádřit jako

$$f(t) = H(t)f(0), \tag{32}$$

kde $f(0)$ je původní vstupní funkce $f(x,t)$ v čase $t = 0$ a matice $H(t)$ je difuzní operátor, jehož prvky lze vypočítat následujícím vztahem

$$H(t) = \sum_{k=1}^{n} e^{-\lambda_k t} u_k u_k^T, \tag{33}$$

kde λ_k je k-té vlastní číslo a $u_{i,k}$ odpovídající vlastní vektor operátoru L. Difúzní reprezentaci obrazových bodů x_i v čase t lze na základě výše uvedených vztahů 32 a 33 zapsat jako

$$x_i(t) = (e^{-\lambda_1 t}u_{i,1}, e^{-\lambda_2 t}u_{i,2}, \cdots, e^{-\lambda_n t}u_{i,n}). \tag{34}$$

Matice, jejíž řádky jsou tvořeny takto získanými vektory, tvoří tzv. difuzní mapu systému v čase t. Je vhodné se zamyslet nad vlivem parametru t na difúzní koordináty. Mějme vektor $x_i(t)$ definovaný vztahem 34. Jeho j-tý prvek $x_{i,j}(t)$ v čase t je dán vztahem $x_{i,j}(t) = e^{-\lambda_j t}u_{i,j}$. Je zřejmé, že s rostoucím časem t absolutní hodnota celého členu klesá tj.

$$\lim_{t\to\infty} e^{-\lambda_j t}u_{i,j} = 0. \tag{35}$$

Vzhledem k vzestupnému uspořádání vlastních čísel v tomto vektoru víme, že členy odpovídající vyšším vlastním číslům k nule konvergují rychleji než členy odpovídající vlastním číslům nejmenším. První, konstantní, vlastní vektor odpovídající vlastnímu číslu $\lambda_1 = 0$ zůstává konstantní pro libovolnou hodnotu t. Proto lze stav, kdy jsou všechny ostatní členy s postupně narůstajícím časem vynulovány chápat jako stav látky s konstantní, nejnižší vnitřní energií. Je zřejmé, že hledání shluků v prostředí s takto vysokou entropií je nemožné, jelikož všechna energie vyjadřující jejich rozdíly byla vstřebána. Volba vhodného parametru t a tedy i stavu ve kterém chceme energetický systém (obraz) segmentovat se tedy stává dalším nelehkým úkolem segmentátora.

Difuze v segmentaci obrazu zvažuje vzdálenosti všech možných cest mezi jednotlivými dvojicemi pixelů. Ačkoliv tento fakt na první pohled pohled vypadá pro hledání obrazových segmentů prospěšně, experimenty v kapitole 6 popisují také její nevýhody závisející na tvaru segmentů. Obecně lze říci, že difuzní vzdálenost dvou pixelů roste s klesající šířkou segmentu a naopak [11]. Toto má za důsledek možné nežádoucí rozdělení úzkých obrazových segmentů.

5 Implementace

Vzhledem k výpočetnímu charakteru úlohy je praktickým výstupem této práce multiplatformní konzolová aplikace "SCISA"(Spectral Clustering Image Segmentation Algorithm) segmentující vstupní obraz metodou shlukování spektra. Implementace je provedena v jazyce *C++* s použitím několika externích knihoven blíže popsaných v sekci 5.1. Technická dokumentace aplikace je uvedena v příloze A.

5.1 Použité nástroje a knihovny

Aplikace ke svému běhu vyžaduje následující nadstandardní knihovny:

- OpenCV
- Arpack++

OpenCV (Open source Computer Vision) je multiplatformní svobodná knihovna určená pro zpracování obrazu a počítačové vidění. OpenCV poskytuje širokou škálu obrazově segmentačních metod včetně algoritmů pro základní obrazové transformace a manipulace. Vzhledem ke kompatibilitě této knihovny se všemi známými obrazovými formáty je v práci použita výhradně pro vstupně-výstupní operace, tj. pro načtení obrazu ze souboru do dvourozměrného pole a zpětného uložení z pole do souboru známého formátu. Mezi obrazové formáty podporované OpenCV patří například PNG, JPEG, BMP, TIF a mnoho dalších.

ARPACK je knihovna navržená k numerickému výpočtu vlastních čísel a vektorů matic dle uživatelem stanovených kritérií. V případě řídkých a strukturovaných matic je k výpočtu použita metoda IRAM (Implicitly restarted Arnoldi method). V případě symetrických matic jsou použity varianty *Lancoszova algoritmu* [6].

5.2 Moduly aplikace

Běh programu lze rozdělit do několika následujících základních fází:

- Před-zpracování dat
- Výpočet spektra
- Shlukování
- Po-zpracování dat

Vzhledem k obecné výpočetní povaze problému jsou fáze před-zpracování a po-zpracování dat zaměřeny především na převod obrazových dat z proprietárních datových struktur a typů OpenCV na standardní a naopak.

Jednotlivé kroky segmentace obrazu metodou difuzního spektrálního shlukování jsou popsány v algoritmu 2. Tento algoritmus přesně koresponduje s jeho implementací v této práci sloužící pro experimenty.

Algorithm 2 Algoritmus difuzního spektrálního shlukování.

1. Ke vstupnímu obrazu sestroj graf a matici podobnosti pomocí Gaussovy funkce.
2. Sestroj normalizovanou Laplaceovu matici L_{sym}.
3. Vypočti k prvních vlastních vektorů $u_1, \cdots, u_k$ matice L_{sym}.
4. Sestroj matici $X \in \mathbb{R}^{n \times k}$ seřazením vlastních vektorů do sloupců.
5. Hodnoty řádků X převeď na difuzní X'.
6. Normuj hodnoty řádků X' na jednotkovou délku.
7. Metodou *GMS* najdi shluky v $\mathbb{R}^{n \times k}$

Je vhodné upozornit, že použití normalizace difuzních souřadnic (krok 6 algoritmu 2) se v dostupné literatuře liší a obecně tento krok není považován za standardní součást algoritmu difuzního spektrálního shlukování tak, jak byl popsán v kapitole 4.1. V této práci je tento krok z výzkumných důvodů použit.

6 Experimenty

Tato kapitola prezentuje dosažené segmentace použitím implementovaného algoritmu na syntetických a reálných obrazech. Dále je předvedeno, jak volbou následujících vstupních parametrů ovlivnit rychlost a kvalitu výpočtu.

- Vztah parametru σ_{af} Gaussovy funkce použité pro výpočet Matice sousednosti a výsledné segmentace. (Parametr σ uvedený ve vztahu 7.)
- Vztah parametru σ_{ms} modifikovaného Gaussova kernelu segmentační metody *Mean-shift* a výsledné segmentace. (Parametr σ uvedený ve vztahu 15.)
- Vztah počtu vlastních vektorů n_e a výsledné segmentace. (Počet sloupců matice X uvedené ve vztahu 29.)
- Vztah difuzního parametru t_d a výsledné segmentace. (Parametr t uvedený ve vztahu 34.)

Ve všech případech je použita symetrická Laplaceova matice L_{sym} definovaná v sekci 4.4.2.

Všechny experimenty byly provedeny na notebooku Lenovo Thinkpad E530c (Intel Pentium Dual Core (B970) 2.3GHz CPU, 4GB (1600Mhz) DDR3 RAM).

6.1 Segmentace syntetického obrazu

Ukázka segmentace syntetického obrazu demonstruje možnosti použití spektrálního shlukování v poměrně snadno segmentovatelných obrazech. Předpokládá se tedy, že jednotlivé segmenty a objekty jsou člověku zřejmé na první pohled a není proto nutné diskutovat o kvalitě výsledné segmentace. Stejně tak dobře lze na syntetických obrazech ukázat vlastnosti algoritmu s různou parametrizací.

Obraz 7 je vzorovým zástupcem kategorie syntetických obrazů - zachycuje objekt neurčitého tvaru umístěný na kontrastním pozadí. V ideálním případě je žádoucí, aby algoritmus označil objekt jako jeden a pozadí jako druhý segment. Vzhledem k nehomogenitě barvy objektu se lze spokojit se segmentací, kde je objekt reprezentován dvěma segmenty odpovídající jeho rozdílným barvám.

Obrázek 7: Syntetický obraz (101 $\times$ 101 px)

6.1.1 Kvalita segmentace syntetického obrazu v závislosti na parametrech t_d a n_e

σ_{af}	0.03
σ_{ms}	0.40
n_e	2, 3, 6, 10
t_d	0, 1000, 10000

Tabulka 2: Parametrizace algoritmu

Obrázek 8 zobrazuje několik vybraných výsledných segmentací syntetického obrazu v závislosti na různé hodnotě difuzního parametru t_d a počtu vlastních vektorů n_{eigvec}, na základě kterých byly segmenty vytvořeny. Jednotlivé řádky obrázku 8 představují hodnoty parametru $t_d = 1, 1000, 10000$. Sloupce reprezentují parametr $n_e = 2, 3, 5, 10$. Parametry podobnostní funkce σ_{af} a Gaussova kernelu σ_{ms} jsou konstantní viz. tabulka 2. Ve všech případech, kde počet vlastních vektorů odpovídá odhadnutému počtu segmentů (dva, popřípadě tři, schválíme-li rozdělení dvoubarevného objektu) lze segmentaci označit za ideální.

S rostoucím počtem vlastních vektorů roste i dimenze jednotlivých bodů určených k segmentaci. Každý vlastní vektor přidává informaci o potencionálních segmentech. Jejich počet tedy ovlivňuje pravděpodobnost rozdělení jednoho segmentu na menší celky. Zejména v případě $t_d = 0$, kdy jsou hodnoty vlastních vektorů normovány bez dalšího zpracování. se tento fakt ve výsledné segmentaci výrazně projevuje rostoucím počtem segmentů. Ačkoliv další segmenty vznikají v místech, kde to lze intuitivně očekávat (úžiny, popřípadě středy segmentů), jsou z našeho pohledu nadbytečné a mohou mít negativní důsledek pro další zpracování. Je nutno zmínit, že segmenty jsou rozděleny v úzkých místech, kde to lze intuitivně očekávat. Tento trend je jasně patrný v segmentacích zachycených v jednotlivých sloupcích.

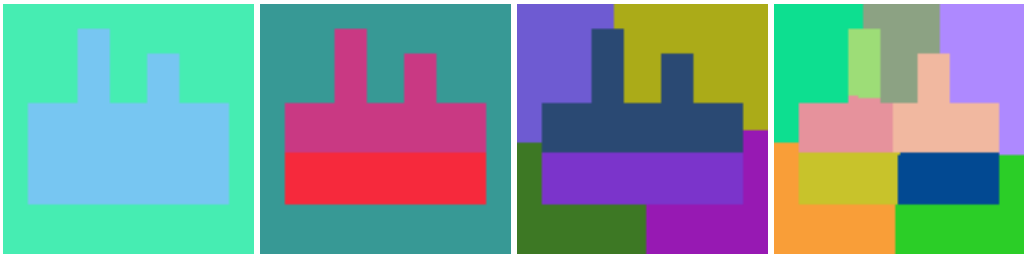
Použitím difuzních souřadnic lze tento trend částečně eliminovat. Dle vztahu 34, rostoucí hodnota difuzního parametru t_d má za důsledek upřednostnění vlivu několika prvních vlastních vektorů a tedy potlačení informace o potencionálně nedůležitých segmentech. Difuze umožňuje docílit kvalitnější segmentace s eliminací nadbytečných segmentů i bez předchozího odhadu počtu výsledných segmentů i v případě použití více vlastních vektorů.

Ve výběru výsledných segmentací na obrázku 8 lze pozorovat pět shodných "ideálních"(8b, 8f, 8f, 8j, 8l) docílených různou parametrizací a tudíž v různých výpočetních časech.

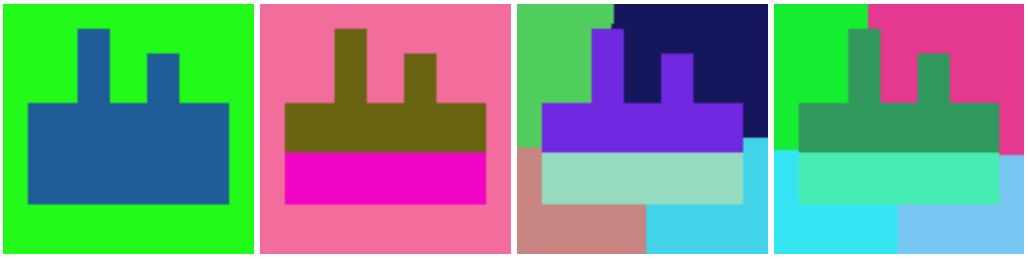
6.1.2 Výkon segmentace syntetického obrazu v závislosti na parametrech t_d a n_e

Vliv parametrů n_e a t_d na dobu výpočtu algoritmu a jeho jednotlivých fází je zaznamenán v tabulkách 3, 4, 5, popřípadě odpovídajících grafech 9, 10, 11.

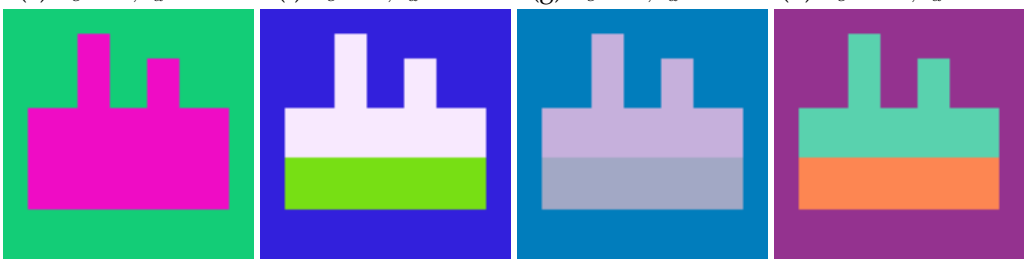
S rostoucím počtem požadovaných vlastních vektorů paradoxně klesá délka jejich výpočtu (viz. tabulka 3). Zatímco v případě $n_e = 2$ byly požadované vstupní vektory vypočteny za přibližně 5 minut, v případě $n_e = 10$ trval výpočet 17 sekund. Sloupec $n_e = 1$



(a) $n_e = 2, t_d = 0$ (b) $n_e = 3, t_d = 0$ (c) $n_e = 6, t_d = 0$ (d) $n_e = 10, t_d = 0$



(e) $n_e = 2, t_d = 10^3$ (f) $n_e = 3, t_d = 10^3$ (g) $n_e = 6, t_d = 10^3$ (h) $n_e = 10, t_d = 10^3$



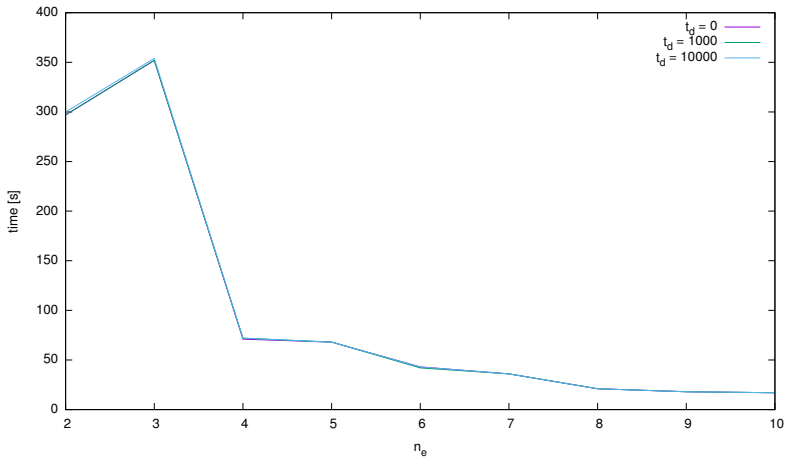
(i) $n_e = 2, t_d = 10^4$ (j) $n_e = 3, t_d = 10^4$ (k) $n_e = 6, t_d = 10^4$ (l) $n_e = 10, t_d = 10^4$

Obrázek 8: Vliv hodnot parametru n_e a t_d na výslednou segmentaci syntetického obrazu.

t_d \ n_e	2	3	4	5	6	7	8	9	10
0	297	352	71	68	43	36	21	18	17
1000	297	352	72	68	42	36	21	18	17
10000	300	354	72	68	43	36	21	18	17

Tabulka 3: Doba výpočtu spektra syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

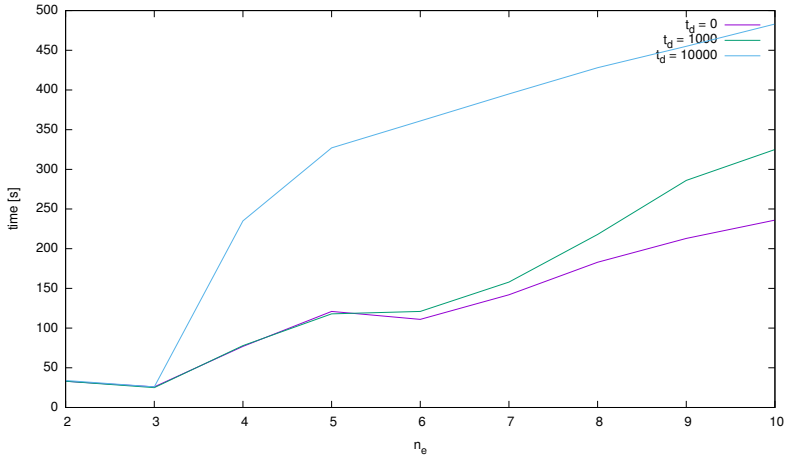
je v tabulce vynechán záměrně vzhledem k faktu, že ve většině případů nebyl algoritmus schopen v maximálním stanoveném počtu iterací dostatečně aproximovat požadované vlastní číslo. Jednotlivé řádky tabulky 3 představují rozdílné hodnoty parametru t_d. Jelikož je difuzní parametr aplikován až na výstup spektrálního rozkladu, výsledné časy nejsou tímto parametrem nijak ovlivněny. Velmi mírné zaznamenané rozdíly v jednotlivých sloupcích jsou tedy odchylky vzniklé vnějšími vlivy ovlivňující výkon CPU stroje použitého pro výpočet a ne důsledek variace t_d.



Obrázek 9: Doba výpočtu spektra syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

t_d \ n_e	2	3	4	5	6	7	8	9	10
0	33	26	77	121	111	142	183	213	236
1000	33	25	78	118	121	158	218	286	325
10000	34	26	235	327	361	395	428	455	483

Tabulka 4: Doba shlukování spektra syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

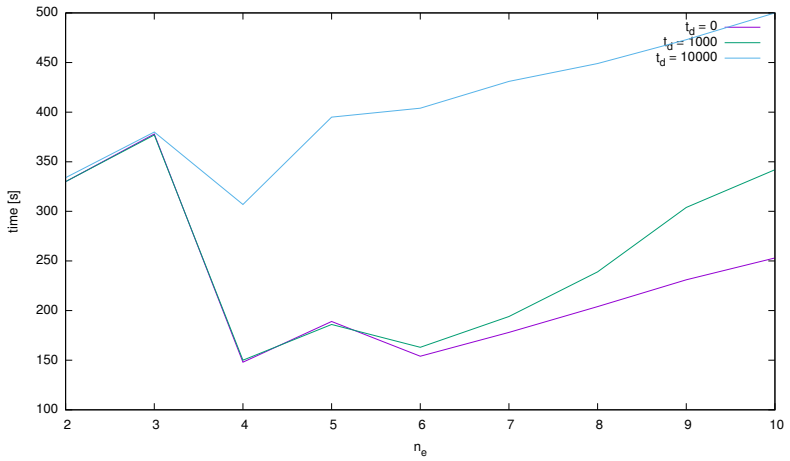


Obrázek 10: Doba shlukování spektra syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

Doba shlukovací fáze roste s rostoucí hodnotou t_d. Difuze jednotlivé podobné obrazové body v prostoru vlastních vektorů vzájemně přiblíží, upravený Gaussův kernel GMS proto do výpočtu s nenulovou váhou zvažuje nezanedbatelně větší počet vzorků. V uvedených příkladech, kde $n_e > 3$ je rozdíl výkonu mezi $t_d = 0$ a $t_d = 1000$ více než dvojnásobný (viz. tabulka 4). Lze také pozorovat korelaci mezi dimensionalitou problému a dobou shlukování. Mimo případ, kdy $n = 2$ s rostoucí dimensionalitou roste i celková doba shlukování GMS.

t_d \ n_e	2	3	4	5	6	7	8	9	10
0	330	378	148	189	154	178	204	231	253
1000	330	377	150	186	163	194	239	304	342
t10000	334	380	307	395	404	431	449	473	500

Tabulka 5: Celková doba segmentace syntetického obrazu v závislosti na parametrech t_d a n_e. [s]



Obrázek 11: Celková doba segmentace syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

Hodnoty v tabulce 5 zobrazují celkovou dobu výpočtu, tj. součet doby výpočtu spektra a doby shlukování. Ačkoliv aplikace mimo tyto dvě fáze provádí i další (např. výpočet Laplaceovy matice, načtení a uložení obrazu, atd), vzhledem k jejich zanedbatelnému konstantnímu trvání nejsou v celkovém čase výpočtu zahrnuty.

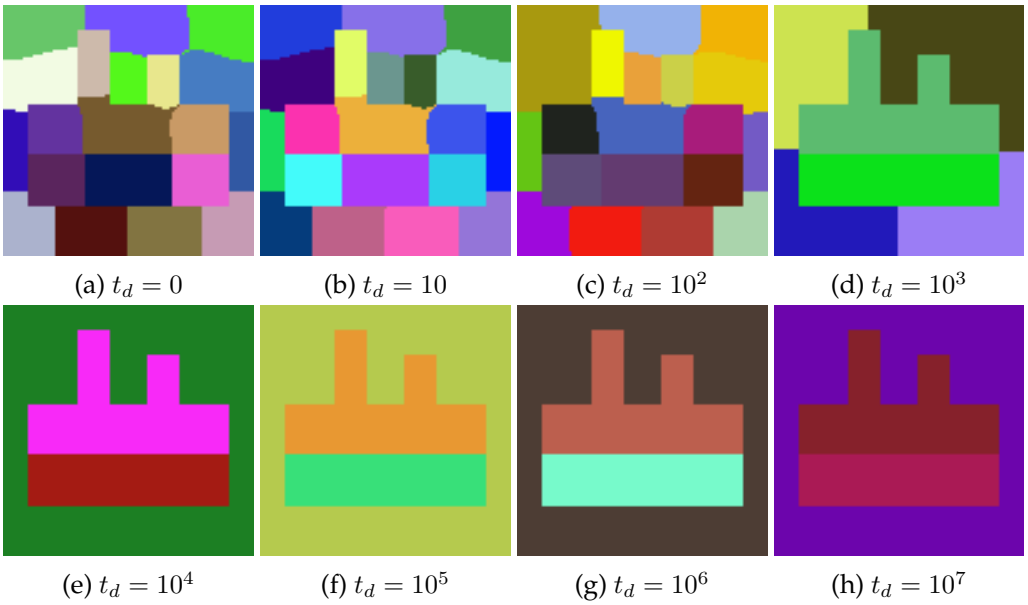
Vzhledem k opačnému trendu doby trvání obou fází v závislosti na parametru n_e dochází k mírnému nárůstu celkového výpočetního času algoritmu společně s rostoucím n_e. Sestupný trend trvání fáze spektrálního rozkladu je převážen vzestupným trendem trvání shlukování (viz. 11). Celková doba segmentace vzorového syntetického obrazu se tak pohybuje od 148 sekund ($t_d = 0$ a $n_e = 4$) do 500 sekund ($t_d = 10000$ a $n_e = 10$).

6.1.3 Důsledek difuze na kvalitu segmentace syntetického obrazu

Segmentace uvedené na obrázku 12 zobrazují výraznější důsledek použití difuze na segmentaci syntetického vstupu 7. Mimo difuzní parametr t_d jsou ostatní parametry konstantní (viz. tabulka 6). Zatímco bez použití difuze je referenční obraz segmentován do nesmyslných 20 shluků (12a), segmentací s dostatečně velkou hodnotou t_d (přibližně $t_d \geq 10^4$) lze docílit segmentací splňující očekávání (12e, 12f, 12g, 12h).

σ_{af}	0.04
σ_{ms}	0.40
n_e	20
t_d	$0, 10, 10^2, 10^3, 10^4, 10^5, 10^6, 10^7$

Tabulka 6: Parametrizace algoritmu pro demonstraci důsledku použití různých hodnot t_d.



(a) $t_d = 0$ (b) $t_d = 10$ (c) $t_d = 10^2$ (d) $t_d = 10^3$

(e) $t_d = 10^4$ (f) $t_d = 10^5$ (g) $t_d = 10^6$ (h) $t_d = 10^7$

Obrázek 12: Vliv difuzního parametru t_d na výslednou segmentaci syntetického obrazu.

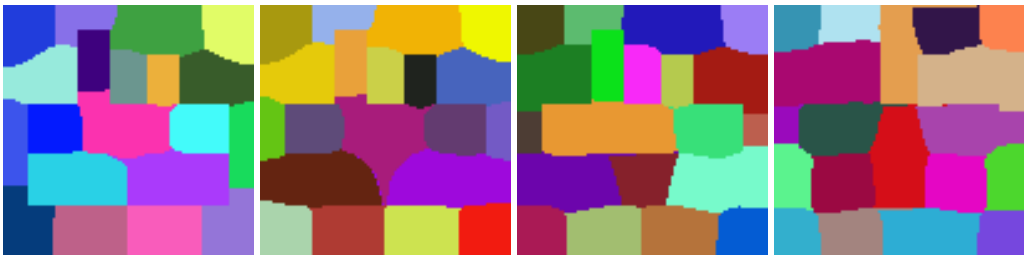
6.1.4 Kvalita segmentace syntetického obrazu v závislosti na parametrech σ_{af} a σ_{ms}

Segmentace uvedené no obrázku 13 zobrazují vliv parametru σ_{af} na segmentaci syntetického obrazu. Hodnoty ostatních použitých parametrů jsou uvedeny v tabulce 7. Parametr σ_{af} ovlivňuje strmost klesání Gaussovy podobnostní funkce použité pro tvorbu podobnostní matice obrazu, tedy míru podobnosti jednotlivých pixelů v jasové doméně. Pomineme-li nadbytečnou segmentaci, lze na obrázku 13a jasně rozpoznat ostré hrany

objektu a jeho částí. S rostoucí hodnotou parametru σ_{af} roste i benevolence podobností funkce. To má za důsledek ztrátu informace o segmentu nejprve spodní části objektu, která má jas k pozadí podobnější než část druhá (viz. segmentace 13b a 13c). Na segmentaci 13d lze pozorovat částečné zaniknutí i horní části objektu.

σ_{af}	0.15, 0.20, 0.25, 0.30
σ_{ms}	0.40
n_e	20
t_d	0

Tabulka 7: Parametrizace algoritmu pro demonstraci důsledku použití různých hodnot σ_{af}.



(a) $\sigma_{af} = 0.15$ (b) $\sigma_{af} = 0.20$ (c) $\sigma_{af} = 0.25$ (d) $\sigma_{af} = 0.30$

Obrázek 13: Vliv parametru σ_{af} na výslednou segmentaci syntetického obrazu.

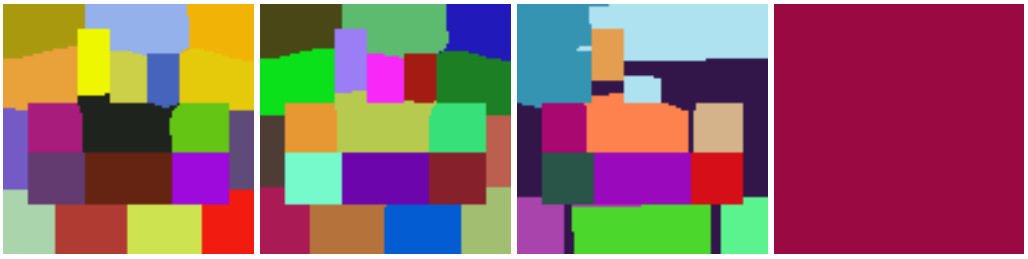
Segmentace uvedené na obrázku 14 představují vliv parametru σ_{ms}, tedy tvar Gaussova kernelu metody *Mean-shift*. Změnou parametru σ_{ms} lze ovlivnit míru podobnosti bodů v doméně vlastních vektorů. S rostoucí hodnotou σ_{ms} lze na obrázku 14c nejprve pozorovat výrazné zhoršení segmentace. Takto vzniklé segmenty jsou na první pohled velmi zmatečné a jsou důsledkem toho, že shlukování není provedeno v jasové doméně vstupního obrazu. Volba příliš velké hodnoty σ_{ms} má za důsledek vyhodnocení vstupu jako jediného segmentu (viz. segmentace 14d).

σ_{af}	0.04
σ_{ms}	0.40, 0.45, 0.50, 0.55
n_e	20
t_d	0

Tabulka 8: Parametrizace algoritmu pro demonstraci důsledku použití různých hodnot σ_{ms}.

6.2 Segmentace syntetického obrazu s aditivním šumem

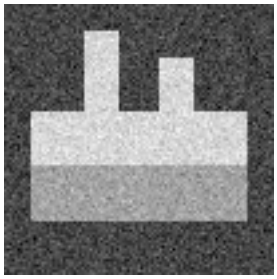
Následující výsledky zobrazují chování algoritmu v případě, kdy je do vstupního obrazu uměle přidán výrazný Gaussův šum ($\sigma = 0.1$). Aby bylo možné výsledky porovnat s



(a) $\sigma_{ms} = 0.40$ (b) $\sigma_{ms} = 0.45$ (c) $\sigma_{ms} = 0.50$ (d) $\sigma_{ms} = 0.55$

Obrázek 14: Vliv parametru σ_{ms} na výslednou segmentaci syntetického obrazu.

předchozím syntetickým případem bez šumu je vstupem algoritmu obraz 16 lišící se od originálního obrazu 7 pouze přidaným šumem. Za ideální segmentaci lze stále považovat oddělení zachyceného objektu od jeho pozadí (dva segmenty), popřípadě možného rozdělení dvojbarevného objektu na dva menší (tři segmenty).



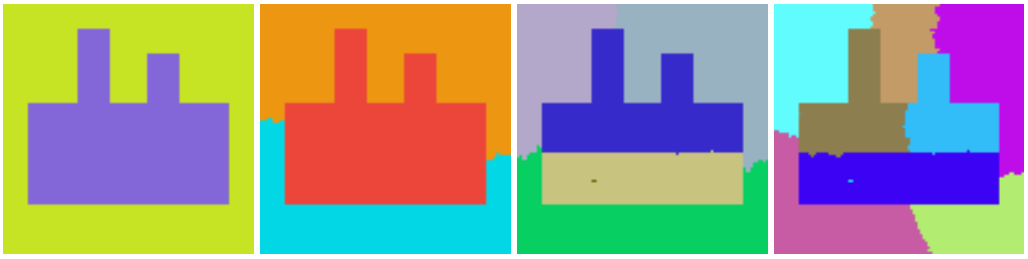
Obrázek 15: Syntetický obraz podrobený šumu (101 $\times$ 101 px)

6.2.1 Kvalita segmentace obrazu s aditivním šumem v závislosti na parametrech t_d a n_e

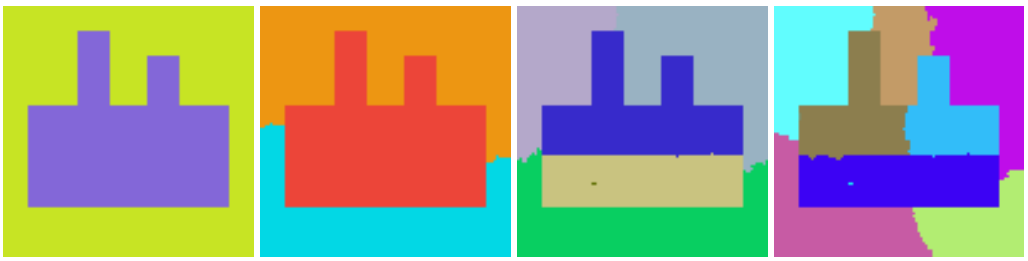
Vybrané segmentace zachycené na obrázku 16 odpovídají jednotlivým parametrizacím popsaných v tabulce 2. V případě shlukování provedeného na základě dvou vlastních vektorů ($n_e = 2$) jsou výsledné segmentace 16a, 16e a 16i shodné se segmentacemi nezašuměného obrazu. V těchto případech, nehledě na hodnotu difuzního parametru, je i přes přidaný šum jasně oddělen objekt a jeho pozadí a lze tedy segmentaci označit za ideální. V případě $n_e = 3$ lze na segmentacích 16b, 16f, 16j pozorovat rozdíly vzhledem k ideálnímu případu. Ačkoliv je objekt reprezentován jedním, jasně patrným segmentem, jeho pozadí je v úzkých místech rozděleno. Stejná parametrizace algoritmu v případě syntetického obrazu bez přidaného šumu vedla k segmentacím, kde je pozadí reprezentováno jedním segmentem a objekt je rozdělen na základě rozdílu jasu jeho částí (8b, 8f, 8j). S dalším rostoucím počtem vlastních vektorů (n_e) je patrné, že přidaný šum má za důsledek lehce rozdílné segmentace, nicméně, hlavní objekt je svými jasnými obrysy stále zřejmý. Z pohledu vlivu počtů vlastních vektorů na počet výsledných segmentů dochází k podobnému nárůstu jako v případě segmentace obrazu bez přidaného šumu; tedy k

trendu zbytečného rozdělení segmentů v úžinách. Lze si všimnout, že vlivem šumu jsou hrany takto oddělených segmentů, na rozdíl od hran ideálních segmentů, výrazně nepravidelné.

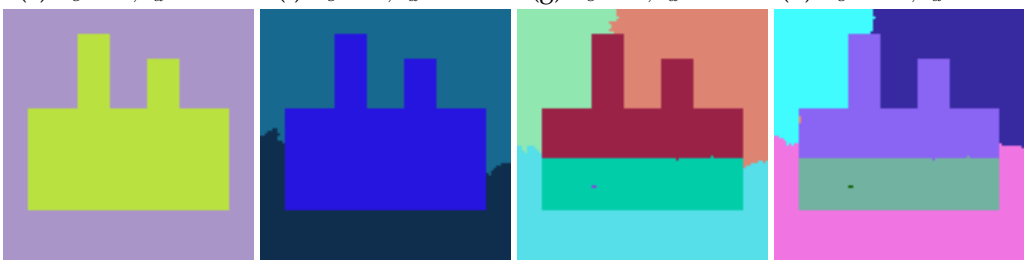
Stejně jako v předchozím případě je míra tvorby přebytečných segmentů potlačena s narůstající hodnotou difuzního parametru t_d. Vzhledem k faktu, že přidaný šum se projevuje zejména na hodnotě prvků vlastních vektorů odpovídajících vyšším vlastním číslům, lze potlačením jejich vlivu docílit kvalitnějších segmentací.



(a) $n_e = 2, t_d = 0$ (b) $n_e = 3, t_d = 0$ (c) $n_e = 6, t_d = 0$ (d) $n_e = 10, t_d = 0$



(e) $n_e = 2, t_d = 10^3$ (f) $n_e = 3, t_d = 10^3$ (g) $n_e = 6, t_d = 10^3$ (h) $n_e = 10, t_d = 10^3$



(i) $n_e = 2, t_d = 10^4$ (j) $n_e = 3, t_d = 10^4$ (k) $n_e = 6, t_d = 10^4$ (l) $n_e = 10, t_d = 10^4$

Obrázek 16: Ukázka vlivu hodnot parametrů n_e a t_d na výslednou segmentaci uměle zašuměného syntetického obrazu.

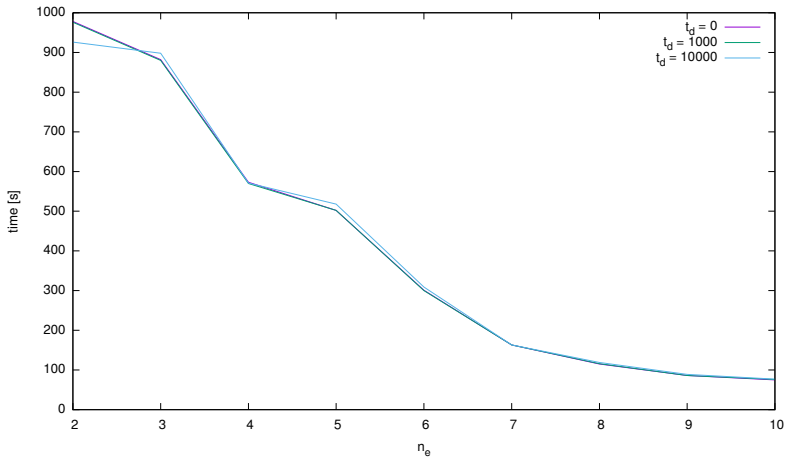
6.2.2 Výkon segmentace obrazu s aditivním šumem v závislosti na parametrech t_d a n_e

Tabulka 9 zachycuje vliv počtu vlastních vektorů n_e ($2 \leq n_e \leq 10$) na dobu trvání výpočtu spektra zašuměného syntetického obrazu. Dle odpovídajícího grafu zachyceného na obrázku 17 je možné pozorovat pokles výpočetního času s rostoucí hodnotou parametru n_e. Ve srovnání s předchozím případem je doba trvání výpočtu několikanásobně

vyšší nehledě na parametrizaci. Nejdéle, 978 sekund trval výpočet dvou vlastních vektorů. V případě nezašuměného obrazu trval výpočet dvou vlastních vektorů v nejhorším případě 300 sekund. Nejrychleji, za 75 sekund bylo vypočteno deset vlastních vektorů. Stejný počet byl v předchozím případě vypočten za 17 sekund.

t_d \ n_e	2	3	4	5	6	7	8	9	10
0	978	882	573	502	301	163	115	86	75
1000	976	880	570	502	300	163	116	86	76
10000	926	898	571	518	309	163	119	89	77

Tabulka 9: Doba výpočtu spektra zašuměného syntetického obrazu v závislosti na parametrech t_d a n_e. [s]



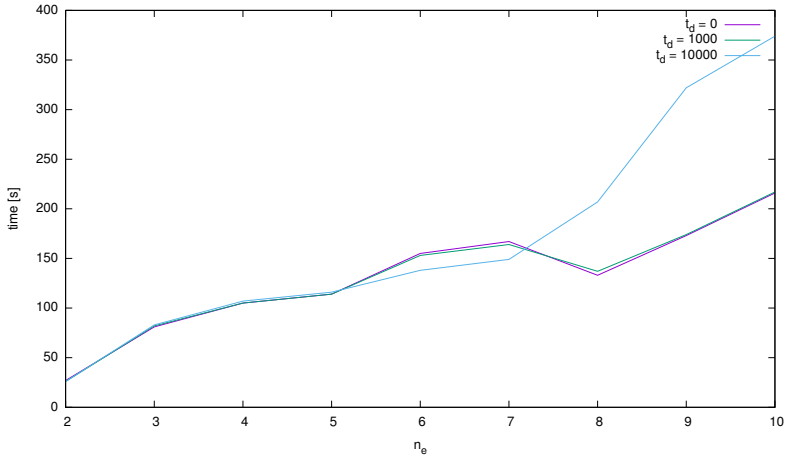
Obrázek 17: Doba výpočtu spektra zašuměného syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

Tabulka 10 a graf na obrázku 18 zachycují vliv parametrů n_e a t_d na trvání shlukování spektra pomocí *GMS*. Výpočetní čas roste přímo s hodnotou obou uvedených parametrů. Zejména při vyšších hodnotách obou parametrů byla samotná shlukovací fáze rychlejší v případě zašuměného obrazu. Doba shlukování se v tomto případě pro $t_d = 0$ v závislosti na dimenzi pohybuje od 27 sekund do 216 sekund, pro $t_d = 1000$ od 26 sekund do 217 sekund a pro $t_d = 10000$ od 34 sekund do 374 sekund.

Celkový čas v závislosti na parametrech n_e a t_d je zobrazen v tabulce 11. Opačné chování obou fází výpočtu v závislosti na počtu vlastních vektorů se v konečném důsledku s rostoucím n_e projevuje nejprve výrazným poklesem celkové doby výpočtu a následným mírným nárůstem zapříčiněným rostoucí dimenzí shlukovacího problému. Podobný i když ne tak výrazný trend byl pozorován i v případě shlukování syntetického obrazu bez přidaného šumu. Zatímco rozdíly v řádech desítek sekund ve shlukovací fázi lze z pohledu celkové doby segmentace zanedbat, několikanásobné rozdíly v době výpočtu

t_d \ n_e	2	3	4	5	6	7	8	9	10
0	27	81	105	114	155	167	133	173	216
1000	26	82	105	114	153	164	137	174	217
10000	26	83	107	116	138	149	207	322	374

Tabulka 10: Doba shlukování spektra zašuměného syntetického obrazu v závislosti na parametrech t_d a n_e. [s]



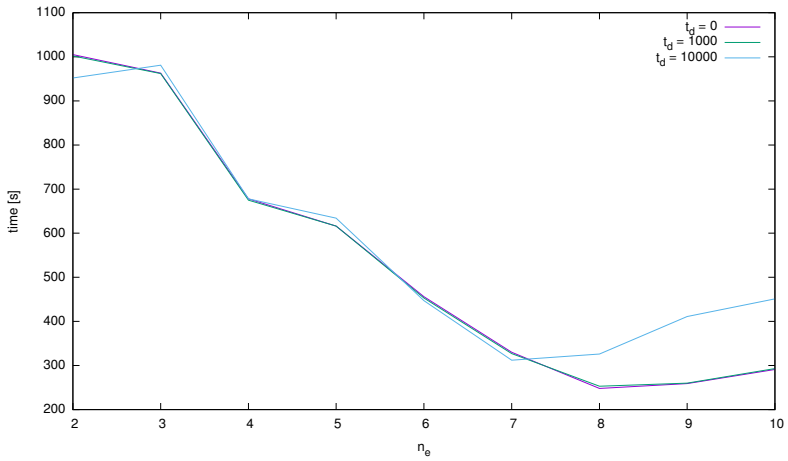
Obrázek 18: Doba shlukování spektra zašuměného syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

spektra mají za důsledek výrazně pomalejší segmentaci v případě obrazu podrobeného šumu.

t_d \ n_e	2	3	4	5	6	7	8	9	10
0	1005	963	678	616	456	330	248	259	291
1000	1002	962	675	616	453	327	253	260	293
10000	952	981	678	634	447	312	326	411	451

Tabulka 11: Celková doba segmentace zašuměného syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

Přidaný šum se na segmentaci syntetického obrazu projevil negativně jak z pohledu výsledných segmentací tak z hlediska výkonu algoritmu. Samotný segmentační algoritmus v závislosti na parametrech n_e a t_d projevil podobné vlastnosti jako v případě segmentace syntetického obrazu bez šumu. Potvrdilo se, že počet nadbytečných segmentů lze do jisté míry redukovat zvýšením hodnoty difuzního parametru t_d. Ačkoliv jsou linie objektu a jeho částí na uvedených segmentacích patrné, bylo v tomto případě pozoro-



Obrázek 19: Celková doba segmentace zašuměného syntetického obrazu v závislosti na parametrech t_d a n_e. [s]

váno mírné zkreslení částí některých hran. Šum se výrazně negativně projevil na výkonu fáze výpočtu spektra, což má za důsledek nárůst celkového výpočetního času algoritmu.

6.3 Segmentace reálného obrazu

Segmentace reálných obrazů, nejčastěji získaných fotoaparátem, ukazuje schopnosti algoritmu segmentovat reálné objekty. Ty jsou často zachyceny v různých prostředích, ovlivněny světelnými vlivy a zatíženy určitou mírou šumu. Snadno lze najít reálné obrazy, ve kterých jsou objekty naprosto skryty a stejně tak lze najít reálné obrazy svým charakterem, kde je objekt vysoce kontrastní s jeho pozadím, připomínající spíše obrazy syntetické. Obraz 20 byl vybrán jako referenční zástupce reálných obrazů pro podrobnější analýzu vlastností implementovaného algoritmu neboť zachycuje reálný objekt - ptáka v jeho reálném prostředí - větev stromu. Díky kontrastnímu pozadí lze však poměrně snadno rozpoznat kvalitní a nekvalitní segmentace. Ideální segmentace tohoto obrazu jsou pak celky reprezentující ptáka, větve stromu a oblohy v pozadí.

Obrázek 20: Referenční šedotónový reálný obraz (128 x 85 px).

6.3.1 Kvalita segmentace reálného obrazu v závislosti na parametrech t_d a n_e

σ_{af}	0.05
σ_{ms}	0.40
n_e	5, 10, 15, 20
t_d	0, 1000, 10000

Tabulka 12: Parametrizace algoritmu pro segmentace zobrazené na obrázku 21.

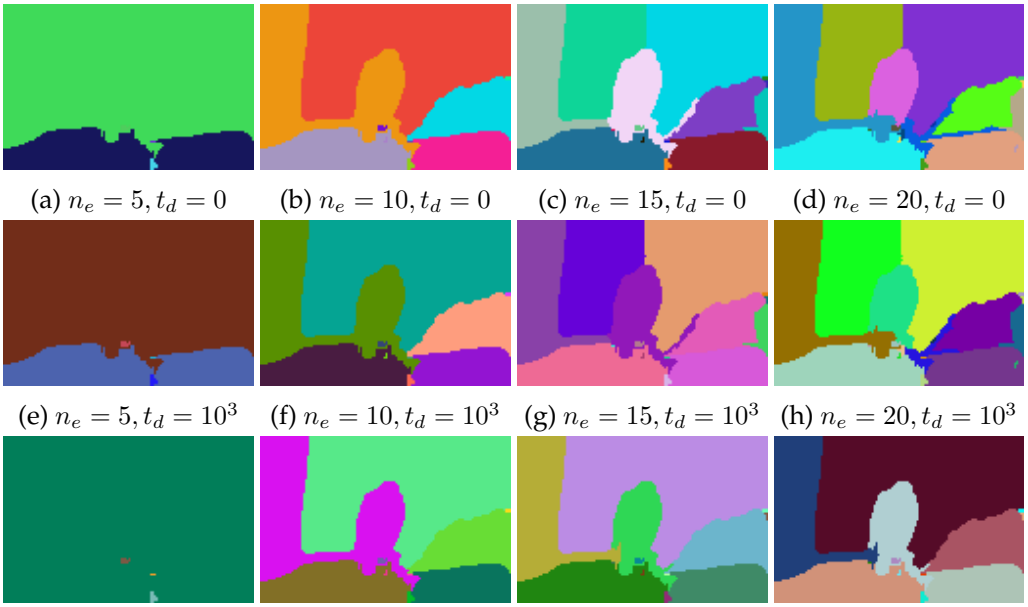
Segmentace na obrázku 21 ukazují vliv parametrů t_d a n_t na počet a tvar výsledných segmentů. Konkrétní použité hodnoty jednotlivých parametrů algoritmu jsou zobrazeny v tabulce 12.

Malý počet vlastních vektorů ($n_e = 5$) má v tomto případě za důsledek zcela nevyhovující segmentace (viz. segmentace 21a, 21e, 21i). Ačkoliv byl v těchto případech obraz segmentován na dvě poloviny na základě výrazné kontury horizontální větve, hlavní podstatu obrazu tyto segmentace nezobrazují. V případě $n_e = 10$ výsledné segmentace (21b, 21f, 21j) mnohem více korespondují s lidskou intuicí. Jsou zde patrné segmenty reprezentující oblohu odděleny větvemi. Většina samotných větví však jako samostatné segmenty není zobrazena. Stejně tak je patrná kontura ptáka, nicméně ne jako samostatný segment, ale pouze jako část podobně zbarvené větve. V případě $n_e = 15$ dochází k dalšímu růstu počtu segmentů, což má za důsledek bližší upřesnění tvaru požadovaných segmentů stejně tak jako nechtěné rozdělení větších celků v úžinách, které bylo zřetelné i u segmentace syntetických obrazů. Na obrázcích 21d, 21h, 21l lze vidět segmentace získané na základě dvaceti vlastních vektorů ($n_e = 20$). V těchto případech lze segmenty označit za poměrně vyhovující. Na těchto segmentacích je patrný lehce zkreslený segment reprezentující objekt společně s několika segmenty oblohy a hrubších části větví stromu. Ačkoliv některé, zejména tenké části větví nejsou zobrazeny jako samostatné segmenty, je jejich vliv patrný na rozdělení segmentů oblohy.

Zvýšením difuzního parametru t_d opět dochází k patrnému vylepšení výsledných segmentací.

6.3.2 Výkon segmentace reálného obrazu v závislosti na parametrech t_d a n_e

Na základě tabulky 13a odpovídajícího grafu 22 lze vidět kritický rozdíl doby výpočtu spektra zejména v závislosti na hodnotě parametru n_e. V omezeném počtu iterací (1000000) nebylo možné dostatečně aproximovat méně než čtyři vlastní vektory. Doba výpočtu spektra se i v případě reálného obrazu snižuje s rostoucí hodnotou n_e. Zatímco čtyři vlastní vektory ($n_e = 4$) byly vypočteny přibližně za 40 minut, výpočet dvaceti ($n_e = 20$) trval 27 sekund (tedy 94.7× rychleji). Výrazný pokles výpočetního času je více patrný zejména mezi výpočty několika prvních vlastních vektorů. Na obrázku 22 lze pozorovat postupné klesání rozdílů mezi sousedícími funkčními hodnotami. Přestože stejně jako v předchozích případech parametr t_d nemá přímý vliv na tuto fázi výpočtu, jsou pro úplnost řádky tabulky zachovány tak, aby byly patrné mírné výkonnostní rozdíly mezi jednotlivými běhy algoritmu.



(a) $n_e = 5, t_d = 0$ (b) $n_e = 10, t_d = 0$ (c) $n_e = 15, t_d = 0$ (d) $n_e = 20, t_d = 0$

(e) $n_e = 5, t_d = 10^3$ (f) $n_e = 10, t_d = 10^3$ (g) $n_e = 15, t_d = 10^3$ (h) $n_e = 20, t_d = 10^3$

(i) $n_e = 5, t_d = 10^4$ (j) $n_e = 10, t_d = 10^4$ (k) $n_e = 15, t_d = 10^4$ (l) $n_e = 20, t_d = 10^4$

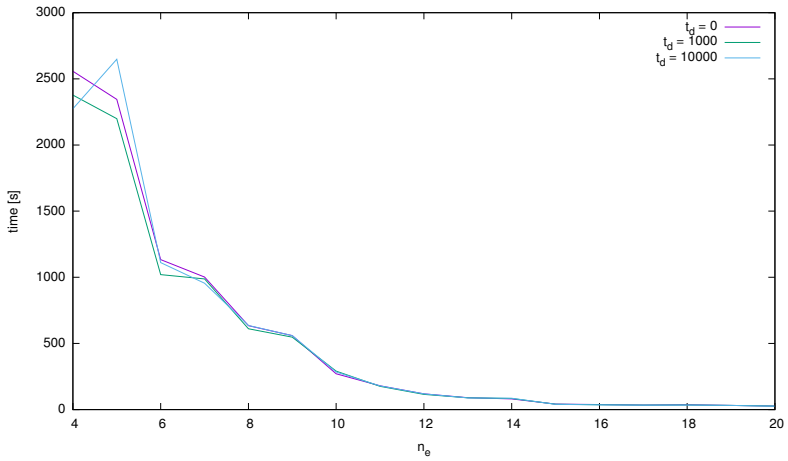
Obrázek 21: Ukázka vlivu hodnot parametrů n_e a t_d na výslednou segmentaci reálného obrazu obrazu.

t_d \ n_e	4	6	8	10	12	14	16	18	20
0	2557	1134	634	271	119	81	38	37	27
1000	2377	1020	611	291	115	84	37	35	27
10000	2274	1111	636	284	119	86	36	34	26

Tabulka 13: Doba výpočtu spektra reálného obrazu v závislosti na parametrech t_d a n_e. [s]

Tabulka 14 a graf 23 zachycují dobu shlukování spektra reálného obrazu v závislosti na parametrech t_d a n_e. Ve všech případech bylo shlukování provedeno v řádech několika minut - 99 sekund v nejlepším případě a 371 sekund v nejhorším. Z uvedených výsledků není na první pohled patrný trend výkonu shlukování v závislosti na dimenzi problému dané parametrem n_e. Výkon se v tomto případě výrazně nemění ani s narůstající hodnotou t_d.

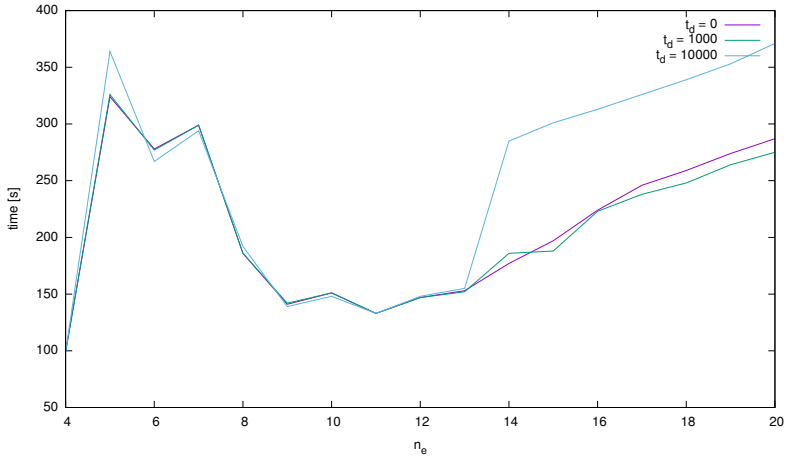
Celková doba segmentace reálného obrazu je zejména pro malé hodnoty n_e více než jindy ovlivněna délkou výpočtu spektra. Vzhledem k chování obou fází lze v tabulce 15 a na obrázku 15 pozorovat minimum pro $n_e = 12$ (262 sekund) následované mírným nárůstem způsobeným rostoucím vlivem shlukovací fáze. Stejně tak lze pozorovat nárůst výpočetního času společně s rostoucí hodnotou difuzního parametru t_d.



Obrázek 22: Doba výpočtu spektra reálného obrazu v závislosti na parametrech t_d a n_e. [s]

t_d \ n_e	4	6	8	10	12	14	16	18	20
0	99	278	186	151	147	177	224	259	287
1000	99	277	186	151	147	186	223	248	275
10000	99	267	192	148	148	285	213	339	371

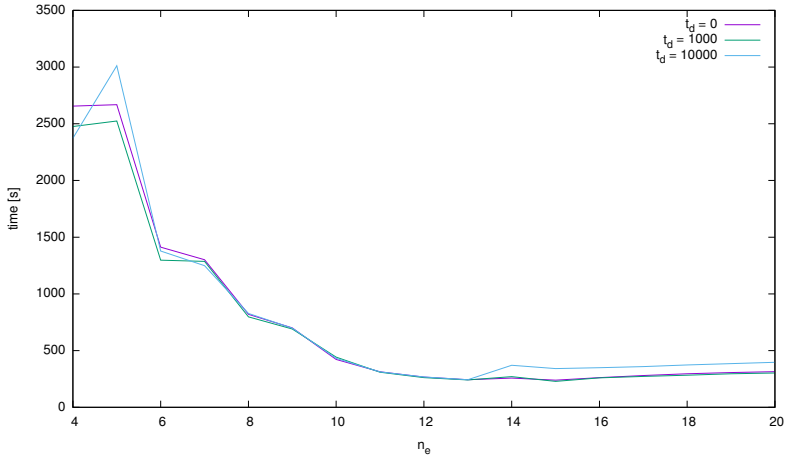
Tabulka 14: Doba shlukování spektra reálného obrazu v závislosti na parametrech t_d a n_e. [s]



Obrázek 23: Doba shlukování spektra reálného obrazu v závislosti na parametrech t_d a n_e. [s]

t_d \ n_e	4	6	8	10	12	14	16	18	20
0	2656	1412	820	422	266	258	262	296	314
1000	2476	1297	797	442	262	270	260	283	302
10000	2373	1378	828	432	267	371	349	373	397

Tabulka 15: Celková doba segmentace reálného obrazu v závislosti na parametrech t_d a n_e. [s]



Obrázek 24: Celková doba segmentace reálného obrazu v závislosti na parametrech t_d a n_e. [s]

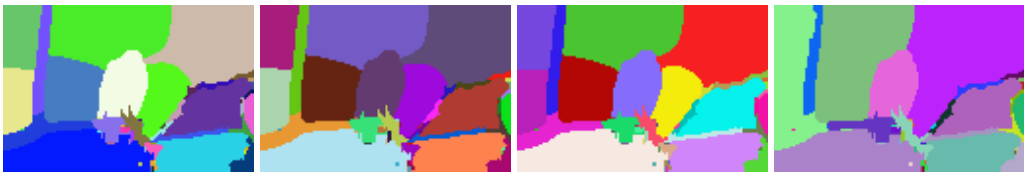
6.3.3 Důsledek difuze na kvalitu segmentace reálného obrazu

Důsledek difuze při segmentaci reálného obrazu je více patrný na segmentacích uvedených na obrázku 25. Tyto segmentace byly docíleny na základě 50 vlastních vektorů použitím exponenciálně narůstající hodnoty t_d (viz. tabulka 16).

V případě segmentací 25a, 25b, 25c, 25d lze pozorovat klesající vliv vysokého počtu použitých vlastních vektorů. Ačkoliv jsou tyto segmentace velice detailní a spousta segmentů je nadbytečných, je možné si všimnout segmentů odpovídajících i velice tenkým větvím v obraze. Stejně tak byla v tomto případě segmentovaná i výrazná svislá větev, která díky svému nasvícení nemá velmi výraznou hranu s levým segmentem oblohy. Segmentace 25e, 25f ($t_d = 10^4$ a $t_d = 10^5$) jsou až na drobné vzniklé artefakty vyhovující. Naopak segementace 25g, 25h jsou naprosto nepoužitelné vzhledem k faktu, že v těchto případech byly vlastní vektory v omezeném počtu iterací aproximovány jen velmi přibližně.

σ_{af}	0.04
σ_{ms}	0.40
n_e	50
t_d	$0, 10, 10^2, 10^3, 10^4, 10^5, 10^6, 10^7$

Tabulka 16: Parametrizace algoritmu pro demonstraci důsledku použití různých hodnot t_d pro reálný obraz.



(a) $t_d = 0$ (b) $t_d = 10$ (c) $t_d = 10^2$ (d) $t_d = 10^3$

(e) $t_d = 10^4$ (f) $t_d = 10^5$ (g) $t_d = 10^6$ (h) $t_d = 10^7$

Obrázek 25: Vliv difuzního parametru t_d na výslednou segmentaci reálného obrazu.

6.3.4 Kvalita segmentace reálného obrazu v závislosti na parametrech σ_{af} a σ_{ms}

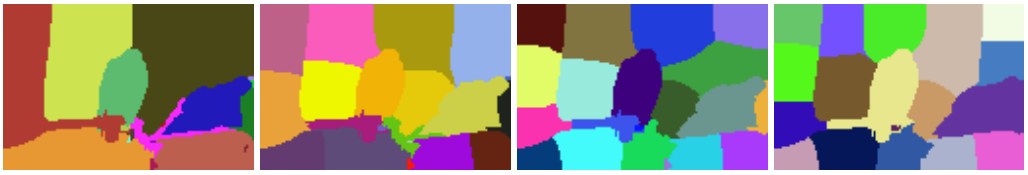
Stejně tak jako v případě segmentace syntetického obrazu je pro úplnost demonstrován vliv parametrů σ_{af} a σ_{ms} na segmentaci reálného obrazu 20.

Volba parametru σ_{af} ovlivňuje výslednou segmentaci reálného obrazu podobně jako v případě segmentace syntetického obrazu. Jak je patrné na segmentacích na obrázku 26 s rostoucí hodnotou σ_{af} se nejprve začne vytrácet informace o sounáležitosti podobně barevných segmentů a následně začnou splývat i kontrastnější objekty. Je zřejmé, že volbou příliš vysoké hodnoty σ_{af} algoritmus ztrácí schopnost rozlišovat míru podobnosti jasu jednotlivých pixelů původního obrazu.

σ_{af}	0.05, 0.07, 0.09, 0.11
σ_{ms}	0.4
n_e	20
t_d	0

Tabulka 17: Parametrizace algoritmu pro demonstraci důsledku použití různých hodnot σ_{af}.

Segmentace reálného obrazu se v závislosti na parametru σ_{ms} vyvíjí podobně jako v případě použití syntetického vstupu. Na obrázku 27 lze pozorovat nejprve úbytek podstatných segmentů (27c) následovaný jejich naprostým splynutím (27c). Normalizace da-



(a) $\sigma_{af} = 0.05$ (b) $\sigma_{af} = 0.07$ (c) $\sigma_{af} = 0.09$ (d) $\sigma_{af} = 0.11$

Obrázek 26: Vliv parametru σ_{af} na výslednou segmentaci reálného obrazu.

tových bodů před segmentací má za důsledek, že hodnota parametru σ_{ms}, od které se segmentace stávají nepoužitelnými, je velmi podobná jako v případě testů pro syntetický obraz bez přidaného šumu.

σ_{af}	0.04
σ_{ms}	0.45, 0.50, 0.55, 060
n_e	20
t_d	0

Tabulka 18: Parametrizace algoritmu pro demonstraci důsledku použití různých hodnot σ_{ms}.

(a) $\sigma_{ms} = 0.45$ (b) $\sigma_{ms} = 0.50$ (c) $\sigma_{ms} = 0.55$ (d) $\sigma_{ms} = 0.60$

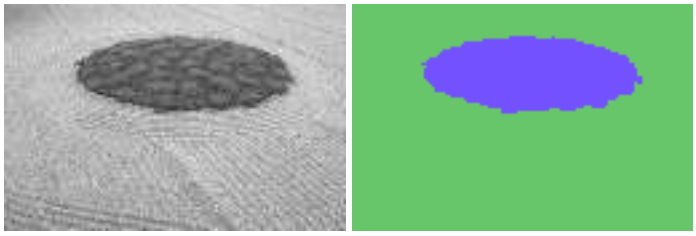
Obrázek 27: Vliv parametru σ_{ms} na výslednou segmentaci reálného obrazu.

6.4 Další ukázkové segmentace

Tato sekce představuje několik dalších experimentálních segmentací reálných obrazů. Pro stručnost je pro každý z uvedených případů zobrazen pouze původní obraz a jeho nejlepší dosažená segmentace. Dále jsou uvedeny informace o celkové době trvání výpočtu a krátký komentář popisující jednotlivé segmentace.

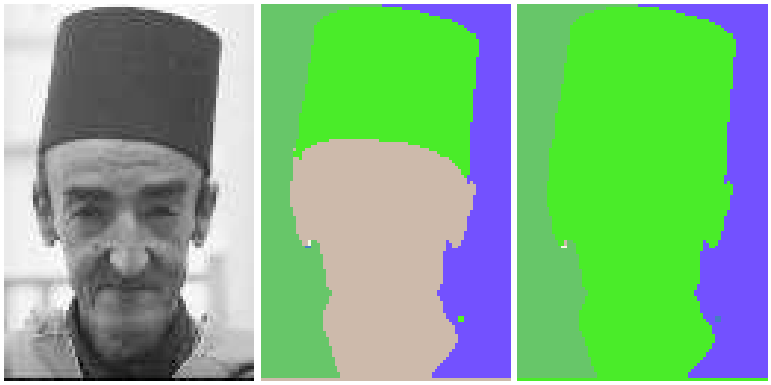
Obraz 28a zachycuje malý jasně ohraničený les uprostřed pole. Jak je patrné na segmentaci 28b, algoritmus oba celky jednoznačně oddělil i přes patrnou vlnitou strukturu pole. Výpočet v tomto případě trval přibližně 26 minut.

Obraz 29 zachycuje mužský obličej a klobouk. V tomto případě byl algoritmus schopný oddělit zmíněné objekty od segmentů pozadí. V závislosti na hodnotě t_d byl obličej s kloboukem vyhodnocen jako jeden (obr. 29b), popřípadě jako dva segmenty (obr. 29c). Celková doba segmentace byla v tomto případě přibližně 10 minut.



(a) (b)

Obrázek 28: Digitální obraz 28a (128 × 85 px) a jeho segmentace 28b ($n_e = 20, t_d = 5000, \sigma_{af} = 0.05, \sigma_{ms} = 0.4$).



(a) (b) (c)

Obrázek 29: Digitální obraz 29a (85 × 128 px) a jeho segmentace 29b ($n_e = 20, t_d = 5000, \sigma_{af} = 0.05, \sigma_{ms} = 0.4$) a 29b ($n_e = 20, t_d = 10000, \sigma_{af} = 0.05, \sigma_{ms} = 0.4$).

Obraz 30a zachycuje letoun na obloze. Vzhledem k nasvícení a tvaru letounu se nepodařilo docílit takové segmentace, aby byl letoun reprezentován jedním segmentem. Jak je patrné na segmentaci 30b, algoritmus byl schopen rozpoznat přední část trupu a křídlo, nicméně ocasní plochy letounu už jsou reprezentovány jiným segmentem. Segment pro střední část trupu není patrný vůbec. Hlavní příčinou této nezdařilé segmentace je nasvětlená horní hrana střední částí splývající se zbytkem scény více než ostatní části letounu. Výpočet této segmentace trval přibližně 21 minut.

Obraz 31a zachycuje houbu v lese. Pozadí tohoto obrazu je mimo jiné tvořeno rozmanitým větvovím jehličnanu. Algoritmem nebylo možné docílit takové segmentace, aby byla houba celá zachycena na jednom segmentu. Na segmentaci 31b lze jasně pozorovat segment odpovídající horní části houby. Segment odpovídající třeni houby naprosto chybí, ačkoliv lze na jeho místě pozorovat nežádoucí rozdělení segmentu pozadí. Je nutno podotknout, že algoritmus relativně dobře vyhodnotil komplikované pozadí tohoto obrazu. Výpočet uvedené segmentace trval přibližně 12 minut.

Jako další ukázku nevyhovující segmentace lze použít obraz surfaře 32a. Testované parametrizace algoritmu pro tento vstup vedly pouze k segmentacím podobným té, uve-

(a) (b)

Obrázek 30: Digitální obraz 30a (128 × 85 px) a jeho segmentace 30b ($n_e = 20, t_d = 1000, \sigma_{af} = 0.04, \sigma_{ms} = 0.4$).

(a) (b)

Obrázek 31: Digitální obraz 31a (85 × 128 px) a jeho segmentace 31b ($n_e = 20, t_d = 7000, \sigma_{af} = 0.04, \sigma_{ms} = 0.4$).

dené na obrázku 32b. Je patrné, že algoritmus v tomto případě nebyl schopný rozlišit horní část plachty splývající s pozadím. V ostatní částech obrazu byla správně oddělena obloha od vodní hladiny. Ačkoliv je patrný obrys spodní části surfaře, výslednou segmentaci nelze považovat za vyhovující. Výpočet této segmentace trval přibližně 12 minut.

(a) (b)

Obrázek 32: Digitální obraz 32a (128 × 85 px) a jeho segmentace 32b ($n_e = 20, t_d = 10000, \sigma_{af} = 0.04, \sigma_{ms} = 0.4$).

7 Závěr

Jádrem této práce je objasnění teoretického základu segmentace obrazu metodou difuzního spektrálního shlukování stejně tak jako samotná implementace této metody v jazyce *C++*. V práci jsou prezentovány experimentální segmentace syntetických i reálných vstupních obrazů dosažené různou parametrizací implementovaného algoritmu. Dále jsou v této práci popsány základní pojmy, terminologie a další metody používané k segmentaci obrazu.

Zatímco většina současných prací zabývajících se spektrálním shlukováním pro samotné shlukování předpokládá použití algoritmu *k-means*, tato práce prezentuje možnosti použití metody *Mean-shift*. *Mean-shift*, na rozdíl od *k-means*, nevyžaduje bližší specifikaci počtu požadovaných shluků a z tohoto důvodu má předpoklady pro univerzálnější použití a potencionální autonomní nasazení bez ohledu na vstupní obraz.

V závislosti na parametrizaci implementovaného algoritmu se projevilo několik zajímavých segmentačních vlastností. S rostoucím počtem vlastních vektorů použitých k segmentaci je pozorován nárůst počtu výsledných segmentů. Ačkoliv jsou tyto segmenty od určité hodnoty nadbytečné a negativně ovlivňují výslednou segmentaci, vznikají zejména v úzkých místech větších segmentů. Použití difuzního spektrálního shlukování ukazuje, jak lze tento trend částečně eliminovat. Zvýšením hodnoty difuzního parametru lze zvýšit vliv vlastních vektorů odpovídajících malým vlastním číslům a potlačit vliv ostatních. Zatímco prvky vektorů odpovídající nejmenším vlastním číslům zůstávají pro segmentaci difuzí téměř nezměněny, absolutní hodnota prvků ostatních vlastních vektorů klesá s rostoucí hodnotou jejich vlastního čísla. Důsledkem této vlastnosti je kvalitnější segmentace při použití většího počtu vektorů. Stejně tak se použití difuze projevilo kladně v případě segmentace obrazu podrobeného šumu. Zatímco na hranách segmentů vypočtených bez difuze lze pozorovat určité zkreslení, zvýšením difuzního parametru dochází k jeho redukci. V případě syntetického obrazu lze od určité hodnoty difuzního parametru pozorovat ustálení výsledných segmentací na ideálních. Artefakty v reálném obraze zapříčinily rapidní zhoršení segmentace pro $t \geq 10^5$. Ve většině případů bylo nehledě na počet vlastních vektorů dosaženo velmi dobrých segmentací s hodnotou difuzního parametru $t_d = 10^4$.

Z hlediska výkonu algoritmu byly pozorovány zejména následující vlastnosti. S narůstajícím počtem vlastních vektorů dochází k výraznému poklesu výpočetního času fáze spektrálního rozkladu. Naopak s takto rostoucí dimenzí mírně narůstá výpočetní čas samotného shlukování. Vzhledem ke kvadratické výpočetní složitosti metody *Gaussian Mean-shift* jsou pro shlukovací fázi kritické zejména rozměry vstupního obrazu. Byl také zaznamenán mírný nárůst celkového výpočetního času s rostoucím difuzním parametrem. Ukázalo se, že volbou většího počtu vlastních vektorů společně s použitím difuze lze implementovaný algoritmus značně akcelerovat.

V práci byla jednoznačně potvrzena spojitost mezi hodnotami vlastních vektorů Laplaceovy matice grafu vstupního obrazu a jeho jednotlivými segmenty. Prezentované experimentální segmentace ukazují přednosti i slabiny implementovaného algoritmu. Vhodnou parametrizací lze pro poměrně jednoduché obrazy docílit ideálních segmentací. Stejně tak bylo ukázáno, že pro některé případy jsou segmentace implementova-

ného algoritmu zcela nevyhovující. Jako problém se projevila závislost difuzní vzdálenosti na tvaru segmentů, což má za důsledek mimo jiné malou schopnost algoritmu kvalitně segmentovat úzké podlouhlé objekty. Stejně tak bylo prokázáno zhoršení segmentačních vlastností u vstupních obrazů obsahující šum, popřípadě jiné výrazné artefakty. Ačkoliv je popsaná metoda založená na robustním teoretickém základu grafových řezů, zvážíme-li její schopnost dobře segmentovat především poměrně jednoduché vstupní obrazy, které mohou být stejně tak dobře segmentovány rychlejšími metodami, je vhodné se zamyslet nad možnostmi uplatnění tohoto algoritmu v praxi. Z pohledu výkonu jsou největšími nedostatky této metody již zmíněné vysoké výpočetní složitosti výpočtu spektra a shlukování *GMS*. Obě tyto fáze představují další implementační výzvu zejména v ohledu jejich paralelizace.

Vojtěch Cima

8 Reference

[1] CARREIRA-PERPIÑÁN, Miguel Á. *Fast nonparametric clustering with Gaussian blurring mean-shift.* In: Proceedings of the 23rd international conference on Machine learning. ACM, 2006. p. 153-160.

[2] COMANICIU, Dorin; MEER, Peter. *Mean shift: A robust approach toward feature space analysis.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2002, 24.5: 603-619.

[3] COMANICIU, Dorin; MEER, Peter. *Mean shift analysis and applications.* In: Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on. IEEE, 1999. p. 1197-1203.

[4] HAUPT, Corinna; HUBER, Andrea B. *How axons see their way–axonal guidance in the visual system.* Front Biosci, 2008, 13: 3136-3149.

[5] CHENG, Yizong. *Mean shift, mode seeking, and clustering.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 1995, 17.8: 790-799.

[6] LANCZOS, Cornelius. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators.* United States Governm. Press Office, 1950.

[7] VON LUXBURG, Ulrike. *A tutorial on spectral clustering.* Statistics and computing, 2007, 17.4: 395-416.

[8] MOHAR, Bojan; ALAVI, Y. *The Laplacian spectrum of graphs.* Graph theory, combinatorics, and applications, 1991, 2: 871-898.

[9] NG, Andrew Y., et al. *On spectral clustering: Analysis and an algorithm.* Advances in neural information processing systems, 2002, 2: 849-856.

[10] SHI, Jianbo; MALIK, Jitendra. *Normalized cuts and image segmentation.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2000, 22.8: 888-905.

[11] SOJKA, Eduard; GAURA, Jan. *A Modification of Diffusion Distance for Clustering and Image Segmentation.* In: Advanced Concepts for Intelligent Vision Systems. Springer International Publishing, 2013. p. 480-491.

A Technická dokumentace aplikace

Tato příloha popisuje vstupy a výstupy implementovaného algoritmu difuzního spektrálního shlukování, jež je součástí digitální přílohy této práce. Stejně tak tato kapitola popisuje technické detaily parametrizace této aplikace.

Vstup aplikace:

- Obraz určený k segmentaci.
- Parametrizace spektrálního rozkladu:
  - Parametr $\sigma_a f$ Gaussovy podobností funkce pro výpočet Laplaceovy matice.
  - Počet požadovaných vlastních čísel a vektorů n_e.
  - Difuzní parametr t_d.
- Parametrizace algoritmu *Gaussian Mean-shift*:
  - Parametr σ Gaussova kernelu.

Výstup aplikace:

- Segmentovaný obraz pojmenovaný ve formátu:
  - [timestamp]_sa[val]_sm[val]_n[val]_t[val]_[input_name]
- Textový soubor obsahující metadata: parametrizaci a délku výpočtu jednotlivých fází segmentace.

Syntaxe: scisa -f *source_image_path* [-sa *val* -sm *val* -n *val* -t *val*]

- -f: Povinný parametr následovaný systémovou cestou ke zdrojovému obrazu.
- -sa: Volitelný reálný parametr určující hodnotu parametru σ Gaussovy podobnostní funkce použité pro výpočet Laplaceovy matice. (Výchozí hodnota $sa = 0.5$, pokud nespecifikováno.)
- -sm: Volitelný reálný parametr určující hodnotu parametru σ Gaussova kernelu algoritmu *Mean-shift*. (Výchozí hodnota $sm = 0.4$, pokud nespecifikováno.)
- -n: Volitelný celočíselný parametr určující počet vlastních vektorů. (Výchozí hodnota $n = 10$, pokud nespecifikováno.)
- -t: Volitelný reálný parametr určující počet vlastních vektorů. (Výchozí hodnota $t = 0$, pokud nespecifikováno.)

B Příloha na CD/DVD

Přiložené CD/DVD obsahuje:

- Soubory se zdrojovými kódy implementovaného algoritmu.
- Experimentální vstupní obrazy a jejich výsledné segmentace.