VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SEGMENTACE OBRAZU POMOCÍ NEURONOVÉ SÍTĚ

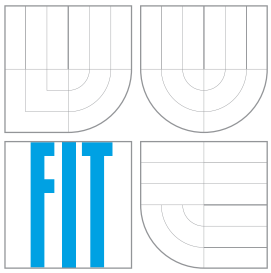
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PAVLA BROMOVÁ

BRNO 2010

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SEGMENTACE OBRAZU POMOCÍ NEURONOVÉ SÍTĚ

NEURAL NETWORK BASED IMAGE SEGMENTATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. PAVLA BROMOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MIROSLAV ŠVUB

BRNO 2010

Abstrakt

Tato práce se zabývá aplikací neuronových sítí na problém segmentace obrazových dat. První část práce je věnována úvodu do problematiky zpracování obrazu a neuronových sítí, v druhé části je popsán vytvořený segmentační systém a jsou prezentovány výsledky provedených experimentů. Vytvořený segmentační systém umožňuje použití různých typů klasifikátorů, extrakci různých příznaků pro klasifikaci a také vyhodnocení úspěšnosti segmentace. Byly vytvořeny dva klasifikátory – neuronová síť (samoorganizační mapa) a algoritmus K-means. Pro klasifikaci byly použity barevné (RGB a HSV) a texturní příznaky a jejich kombinace. Texturní příznaky byly získány pomocí sady Gáborových filtrů. Byly provedeny experimenty s vytvořenými klasifikátory a extraktory a porovnány výsledky.

Abstract

This paper deals with application of neural networks in image segmentation. First part is an introduction to image processing and neural networks, second part describes an implementation of segmentation system and presents results of experiments. The segmentation system enables to use different types of classifiers, various image features extraction and also to evaluate the success of segmentation. Two classifiers were created – a neural network (self-organizing map) and an algorithm K-means. Colour (RGB and HSV) and texture features and their combinations were used for classification. Texture features were extracted using a set of Gabor filters. Experiments with designed classifiers and feature extractors were carried out and results were compared.

Klíčová slova

zpracování obrazu, segmentace obrazu, neuronové sítě, samoorganizační mapa, Kohonenova mapa, K-means, Gáborovy filtry

Keywords

image processing, image segmentation, neural networks, self-organizing map, Kohonen map, K-means, Gabor filters

Citace

Pavla Bromová: Segmentace obrazu pomocí neuronové sítě, diplomová práce, Brno, FIT VUT v Brně, 2010

Segmentace obrazu pomocí neuronové sítě

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracovala samostatně pod vedením pana Miroslava Švuba. Uvedla jsem všechny literární prameny a publikace, z kterých jsem čerpala.

.....
Pavla Bromová
26. května 2010

Poděkování

Velmi ráda bych poděkovala vedoucímu mé diplomové práce panu Miroslavu Švubovi za jeho vedení, pomoc a čas, které mi věnoval při konzultacích. Také bych chtěla poděkovat svému příteli za jeho pomoc, trpělivost a podporu.

Obsah

Kapitola 1

Úvod

Lidé vnímají okolní svět pomocí zraku a dokážou interpretovat vizuální informace okolo nich. Tyto schopnosti se snažíme propůjčit strojům, aby dokázaly interpretovat vizuální informace obsažené v obrazech, grafice a videu. Snahou dosáhnout efektů lidského vidění se zabývá vědní obor počítačové vidění. Počítačové vidění je rozmanité a relativně nové odvětví. Mnoho metod a aplikací je stále ještě ve fázi základního vývoje, ale stále více metod už nachází uplatnění v komerčních produktech, kde často tvoří část většího systému, který dokáže řešit komplikované úkoly. Jednou z nejdůležitějších fází zpracování obrazu je segmentace.

Při interpretaci obrazu naráží počítačové vidění na problémy, které lidé řeší automaticky na základě znalostí a zkušeností. Snahou dát strojům schopnost rozumět tomu, co pozorují, se zabývá umělá inteligence už několik desetiletí a za tu dobu došlo v této oblasti k obrovskému pokroku. V současnosti většina aplikací počítačového vidění využívá počítače, které jsou předprogramovány k řešení konkrétního úkolu, ale stále více se uplatňují metody založené na učení, které nacházejí inspiraci v biologických systémech, a mezi něž patří i umělé neuronové sítě.

Tato práce se zabývá aplikací neuronových sítí na problém segmentace obrazových dat. Cílem je prozkoumat možnosti využití neuronové sítě v segmentaci a srovnání této metody s některým z tradičních segmentačních algoritmů. K tomuto účelu byl vytvořen segmentační systém, který umožňuje použití jak různých typů klasifikátorů, tak příznaků pro klasifikaci, a také vyhodnocení úspěšnosti segmentační metody.

První část práce je věnována teoretickému úvodu do problematiky a popisu oblastí, kterých se práce týká, druhá část pak popisuje vytvořený segmentační systém. Následuje stručný přehled jednotlivých kapitol.

Kapitola 2 je úvodem do zpracování obrazu. Popisuje základní principy zpracování obrazu, možné způsoby reprezentace obrazu a uvádí několik příkladů aplikací. Obsahuje popis nejčastějších metod a podrobněji se věnuje segmentaci.

Třetí kapitola popisuje neuronové sítě a jejich základní principy. Obsahuje příklady využití neuronových sítí, vysvětluje jejich souvislost s biologickými systémy, popisuje princip učení sítě a podrobněji se zabývá několika základními typy sítí.

Čtvrtá kapitola už se věnuje popisu praktické části a podrobně popisuje vytvořený segmentační systém.

Pátá kapitola slouží k prezentaci dosažených výsledků a obsahuje také přehled provedených experimentů.

Závěrečná kapitola obsahuje zhodnocení dosažených výsledků a další možná rozšíření.

Kapitola 2

Zpracování obrazu

Tato kapitola slouží k zasazení segmentace do širšího kontextu. Popisuje základní principy zpracování obrazu, zmiňuje několik oblastí využití a konkrétních příkladů aplikací. Zabývá se problematikou reprezentace obrazu – je zde uvedena matematická definice obrazu a popis různých úrovní reprezentace. Dále se kapitola zabývá nejčastějšími metodami zpracování obrazu, mezi něž patří i segmentace. Blíže se podíváme na metody filtrace obrazu, které byly použity v praktické části práce, a poslední část kapitoly je věnována segmentaci, která je hlavním předmětem práce.

2.1 Úvod

Zpracování obrazu zkoumá metody uložení, zpracování, přenosu, rozpoznávání a interpretace vizuálních scén. Klasickým problémem počítačového vidění a zpracování obrazu je rozpoznávání objektů nebo vlastností obrazu – metody se snaží získat z obrazu informace (např. materiál povrchu nebo tvar objektu) a pomocí nich najít shodu mezi objekty v obrazu a modely objektů reálného světa [7]. Často se lze setkat s následující posloupností kroků [19, 1]:

1. Prvním krokem je digitalizace obrazu. Obraz je zachycen senzorem (např. fotoaparátem) a digitalizován.

2. Někdy je součástí obrazu šum, jehož příčinou může být např. špatné zaostření fotoaparátu, mlha, nebo relativní pohyb mezi objektem a fotoaparátem. V takovém případě potřebujeme vhodné techniky pro dosažení lepší vizuální kvality obrazu.

3. Dalším krokem je segmentace obrazu, jejímž cílem je rozlišit objekty v obrazu od pozadí a od sebe navzájem. Obraz je rozdělen na několik oblastí, z nichž každá reprezentuje nějaký objekt. Oblasti jsou souvislé a nepřekrývají se, takže každý pixel obrazu patří do právě jedné oblasti. Segmentace je jednou z nejdůležitějších částí analýzy obrazu. V této fázi jsou z obrazu získány objekty nebo jiné významné prvky pro další zpracování obrazu (např. rozpoznávání). Například v případě leteckého snímku obsahujícího oceán a souš je úkolem segmentovat obraz nejdříve na dvě části – oceán a souš. Až potom mohou být správně segmentovány a následně klasifikovány objekty v oblasti souše.

4. Po rozlišení jednotlivých segmentů je dalším úkolem extrakce významných rysů jako textura, barva a tvar. Jsou to důležité prvky, které určují různé vlastnosti segmentů

obrazu. Příkladem texturových vlastností je drsnost, hladkost, pravidelnost, atd., běžným popisem tvaru je např. délka, šířka, plocha, obvod, atd. Každý segment obrazu může být charakterizován množinou takových rysů.

5. Nakonec je každý segmentovaný objekt na základě množiny těchto rysů klasifikován do jedné z tříd. Například v obrazu oceánu mohou být tyto třídy lodě a voda. Segmentace a klasifikace jsou dvě spojené oblasti počítačového vidění. Ukazuje se, že tyto úkoly celkem efektivně zvládají expertní systémy, sémantické sítě a systémy založené na neuronových sítích.

Jiná oblast počítačového vidění se zabývá kompresí a kódováním vizuálních dat. Praktická část této práce se však těmito metodami nezabývá, a proto vynecháme jejich podrobnější popis, který lze najít v [19] a [1].

Dát počítačům schopnost vidět není snadné. Jedním z hlavních problémů je ztráta informace při projekci 3D → 2D. Žijeme v trojrozměrném světě, ale typické přístroje zachycující obraz (např. fotoaparáty, kamery) mají obvykle pouze dvourozměrný výstup. Tato projekce na menší počet rozměrů způsobuje obrovskou ztrátu informací (např. měřítko, hloubka).

2.2 Aplikace zpracování obrazu

Existuje velké množství aplikací zpracování obrazu v různých oblastech lidských činností. Stručně se zmíníme pouze o některých z nich [1, 13]:

Zpracování medicínských dat – je jednou z nejvýznamnějších oblastí aplikace počítačového vidění. Metody zpracování obrazu se využívají zejména pro účely medicínské diagnózy. Obrazová data jsou získávána pomocí různých zobrazovacích technik jako počítačová tomografie, ultrazvuk, magnetická rezonance atd.

Výrobní procesy – zpracování obrazu se zde využívá hlavně při řízení výroby (např. zjištění pozice a orientace výrobku pro správné nastavení robotického ramena, navigace robotických vozidel nebo manipulátorů) a kontrole kvality (detekce chybných produktů). Systémy pro automatickou kontrolu kvality jsou asi nejúspěšnější oblastí aplikace počítačového vidění ve výrobě [7].

Vojenské aplikace – jsou asi jednou z největších oblastí počítačového vidění. Příkladem je detekování nepřátelských vojáků a vozidel nebo navigace řízených střel. Se stále dokonalejšími zbraněmi se zvyšují nároky na rychlou a přesnou identifikaci a sledování cílů. Hlavním cílem této oblasti je zvýšení přesnosti rozpoznávání a sledování více cílů současně [7].

Analýza leteckých a satelitních snímků – obrazy zemského povrchu jsou analyzovány a získané informace jsou použity např. při plánování měst, regulaci povodní, kontrole zemědělské produkce, atd. Analyzují se snímky městských oblastí - velká část výzkumu se věnuje detekci budov [11].

3D modely – množství aplikací jako např. simulace, virtuální realita nebo pokročilá kartografie využívá komplikované 3D modely a terény.

Komprese obrazu a videa – je aktuální oblastí aplikace zpracování obrazu. Požadavky na velikost přenášených a ukládaných dat rostou exponenciálně a tím pádem rostou i nároky na kompresní metody.

2.3 Reprezentace obrazu

2.3.1 Obraz jako funkce dvou proměnných

Pro počítačové zpracování obrazu je nutné obraz reprezentovat nějakým matematickým modelem. Obraz je zde signál, což je nejčastěji dvourozměrná funkce dvou souřadnic v rovině [22]:

$$f(x, y) : \mathbb{R} \times \mathbb{R} \rightarrow V \tag{2.3.1}$$

kde x, y jsou souřadnice a V je obor hodnot obrazu (např. barva).

Tato definice obrazu však předpokládá, že velikost obrazu je neomezená. V praxi se však velikost obrazu omezuje na $w \times h$, kde w je šířka obrazu a h jeho výška. Obrazová funkce pak nabývá tvaru [22]:

$$f(x, y) : \langle 0, w) \times \langle 0, h) \rightarrow V \tag{2.3.2}$$

Pro popis černobílého obrazu postačuje skalární funkce (obor hodnot V je skalár), zatímco barevný obraz složený ze tří barevných složek je potřeba reprezentovat vektorovou funkcí. Funkce dále může být spojitá nebo diskrétní. V reálném světě obraz nabývá hodnot ze spojitého intervalu, což je však pro počítačové zpracování nepraktické, protože spojitá funkce není obecně reprezentovatelná v počítači. Proto je potřeba obraz reprezentovat pomocí diskrétních vzorků (pixelů), což je úkolem digitalizace obrazu.

2.3.2 Úrovně reprezentací obrazu

Porozumění obrazu počítačem může být chápáno jako snaha najít vztah mezi vstupním obrazem a předem vytvořeným modelem reálného světa. Během převodu obrazu na model se redukují informace obsaženy v obrazu na informace relevantní pro oblast aplikace a stále více se uplatňují sémantické znalosti o interpretaci obrazových dat. Tento proces je většinou rozdělen do několika kroků a využívá několika úrovní reprezentace obrazu. Nižší úrovně obsahují hrubá obrazová data a vyšší úrovně tato data interpretují.

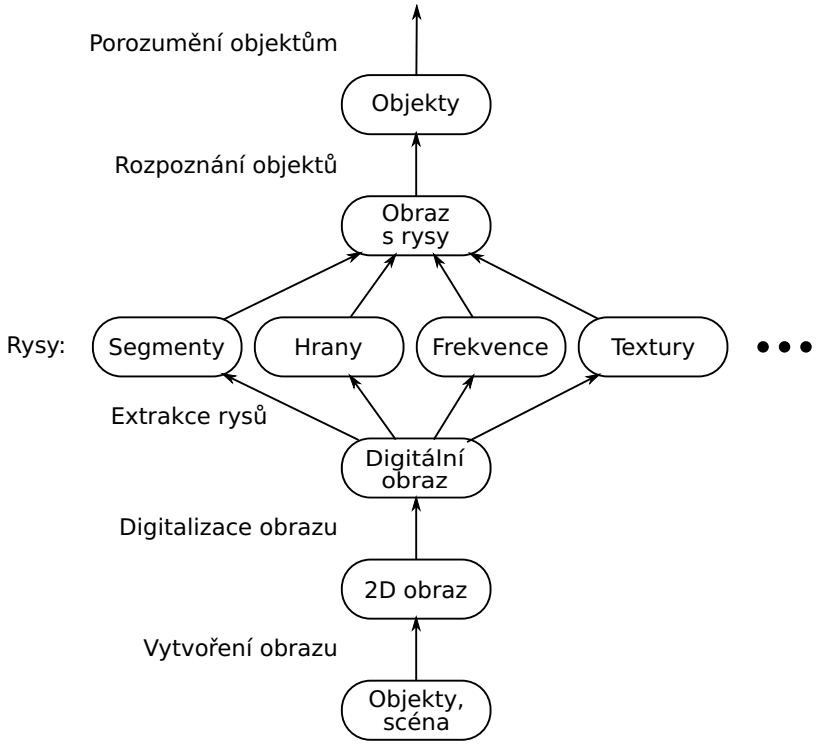
Reprezentace obrazu mohou být na základě uspořádání dat rozděleny do několika úrovní, viz obrázek 2.1. Hranice mezi jednotlivými úrovněmi jsou neurčité a používá se i podrobnější dělení. Tyto úrovně reprezentace jsou řazeny od signálů s téměř žádnou úrovní abstrakce až po vysoce abstraktní popis potřebný k porozumění obrazu. Tok informací mezi jednotlivými úrovněmi může být obousměrný a pro některé účely mohou být některé reprezentace vynechány.

Úrovně reprezentací obrazu lze shrnout do čtyř základních skupin [19]:

1. **úroveň – původní obrazy** – zahrnuje obrazy obsahující původní data – číselné matice tvořené údaji o jasu pixelu. Tyto obrazy jsou také výstupem metod předzpracování obrazu (např. filtrace nebo zvýraznění hran), které se používají ke zvýraznění některých rysů obrazu významných pro další zpracování.

2. **úroveň – segmentované obrazy** – obrazy jsou rozděleny na oblasti, které reprezentují objekty v obrazu.

3. **úroveň – geometrické reprezentace** – uchovávají informace o 2D a 3D tvarech. Kvantifikace tvaru je velmi obtížná, ale také velmi důležitá. Geometrické reprezentace jsou



Obrázek 2.1: Možné úrovně reprezentace obrazu vhodné pro detekci a klasifikaci objektů. Reprezentace jsou znázorněny ovály. Zdroj: [19].

užitečné u náročných simulací vlivu osvětlení a pohybu na reálné objekty. Jsou potřeba také pro převod mezi rastrovými obrazy (pořízenými např. fotoaparátem) a daty používanými v počítačové grafice (CAD).

4. úroveň – relační modely – umožňují efektivnější zpracování dat na vyšší úrovni abstrakce. Pracují s informacemi o vzájemném vztahu objektů. Často se zde používají metody umělé inteligence.

2.4 Metody zpracování obrazu

Zpracování obrazu lze rozlišit na dvě hlavní kategorie – nízkoúrovňové a vysokoúrovňové. Toto rozdělení je uvedeno např. v [19], odkud byly také čerpány informace k této podkapitole.

2.4.1 Nízkoúrovňové metody

Nízkoúrovňové metody obvykle používají jen velmi málo informací o významu obrazu. Snaží se získat z obrazu informace nebo jej nějak upravit, ale porozumění obrazu už je předmětem vysokoúrovňových metod.

Stručně si popíšeme nejčastěji používané metody:

Digitalizace – je prvním krokem potřebným k dalšímu zpracování obrazu počítačem. Obraz musí být reprezentován diskrétními hodnotami ve vhodné datové struktuře, např. matici. Obraz zachycený senzorem je dvourozměrný signál, který lze vyjádřit spojitou

funkcí $f(x, y)$ dvou souřadnic. Digitalizace znamená, že funkce $f(x, y)$ je navzorkována do matice o požadovaném počtu sloupců a řádků a každému vzorku je přiřazena celočíselná hodnota - funkce je diskretizována [13]. Čím je vzorkování jemnější (tzn. čím je větší matice), tím přesnější je získaný obraz.

Metody předzpracování obrazu – předzpracování je souhrnný název pro metody zpracování obrazu na nejnižší úrovni abstrakce. Vstupem i výstupem jsou obrazy intenzity často reprezentované maticí hodnot obrazové funkce (jasu). Obvykle omezují množství informací v obrazu, což může být užitečné v různých situacích, protože lze potlačit ty informace, které nejsou podstatné pro daný úkol. Toho se často využívá např. pro potlačení šumu v obrazu. Cílem předzpracování je vylepšení obrazových dat potlačením nežádoucích vad, nebo naopak zdůrazněním rysů podstatných pro další zpracování (příkladem je detekce hran). Mezi základní metody předzpracování obrazu patří filtrace, která je klíčovou součástí této práce a proto si ji podrobněji popíšeme v podkapitole 2.5.

Segmentace – je jedním z nejdůležitějších kroků zpracování obrazu. Hlavním cílem je rozdělit obraz na části odpovídající objektům reálného světa. Každému pixelu je přiřazen segment reprezentující určitý objekt v obrazu. Pixely jsou rozděleny do segmentů na základě určitých vlastností, na nichž potom závisí kvalita výsledné segmentace. Příkladem těchto vlastnosti může být textura okolí nebo barva. Výsledky segmentace jsou předmětem dalšího zpracování metod vyšší úrovně. Segmentací se blíže zabývá podkapitola 2.6.

Extrakce rysů, klasifikace – jsou vybrány vlastnosti objektů, které je nějakým způsobem charakterizují. Tyto vlastnosti se nazývají rysy. Na základě těchto rysů jsou objekty reprezentované segmenty rozděleny (klasifikovány) do disjunktních množin nazývaných třídy. Toto rozdělení provádí klasifikátor. Například k rozlišení oceli od písku se použijí vlastnosti jako textura, hmotnost, nebo tvrdost. Pokud jsou zvoleny vhodné vlastnosti, objekty v rámci jedné třídy si budou podobnější (na základě daných vlastností) než objekty z různých tříd. Pro výběr vhodných rysů existuje množství algoritmů.

Většina současných metod nízkoúrovňového zpracování obrazu byla navrhnuta v 70. letech nebo dříve. Poslední výzkumy se snaží najít efektivnější a obecnější algoritmy a implementují je na technologicky vyspělejších nástrojích. Příkladem je použití paralelních strojů ke zpracování obrovského množství výpočetních operací prováděných na obrazových datech. Potřeba lepších a rychlejších algoritmů je poháněna technologií poskytující stále větší obrazy (s lepším rozlišením a barvou).

Komplikovaným a zatím nevyřešeným problémem zůstává, jak uspořádat posloupnost nízkoúrovňových kroků tak, aby řešily konkrétní problém. Obvykle je potřeba, aby tuto posloupnost operací našel člověk pomocí intuice, zkušeností a znalostí daného problému.

2.4.2 Vysokoúrovňové metody

Vysokoúrovňové metody jsou založeny na znalostech, cílech a plánech, jak těchto cílů dosáhnout. Často využívají techniky umělé inteligence. Vysokoúrovňové počítačové vidění se snaží napodobit lidské poznání a schopnost rozhodovat se vzhledem k informacím obsaženým v obrazu.

Vysokoúrovňové metody vycházejí z nějakou formou formálního modelu světa, s tímto modelem porovnává "realitu" ve formě digitálních obrazů a snaží se najít shodu. Pokud narazí na nějaké odlišnosti, je potřeba aktualizovat model. Aby byly získány informace potřebné pro aktualizaci modelu, přejde se na nízkoúrovňové metody zpracování obrazu. Tento proces je potom opakován iterativně – je vytvořena zpětná vazba, kde částečné výsledky vysokoúrovňového zpracování vytváří úkoly pro nízkoúrovňové zpracování obrazu. Porozumění obrazu se tak stává výsledkem spolupráce přístupů shora-dolů a zdola-nahoru.

Nízkoúrovňové a vysokoúrovňové zpracování se liší v typu používaných dat. Nízkoúrovňová data jsou tvořena původními obrazy, které jsou reprezentovány maticemi tvořenými např. hodnotami jasu, zatímco vysokoúrovňová data pocházejí také z obrazu, ale jsou použita pouze data relevantní pro vysokoúrovňové cíle (porozumění obrazu), čímž je značně sníženo množství dat. Vysokoúrovňová data reprezentují znalost významu obrazu, např. velikost a tvar objektů a vzájemný vztah mezi objekty v obrazu. Jsou obvykle reprezentována symboly.

2.5 Filtrace obrazu

Tato podkapitola se zabývá filtrací obrazu a vysvětluje některé metody filtrace, které byly v práci použity.

Filtrace patří mezi základní metody předzpracování obrazu. Transformuje vstupní obraz na výstupní tak, že pro výpočet nové hodnoty pixelu výstupního obrazu je použito blízké okolí odpovídajícího pixelu ve vstupním obrazu. Typickou realizací filtru nad obrazem je lineární konvoluční filtr [22].

2.5.1 Lineární konvoluční filtr

Lineární konvoluční filtr je filtr popsaný konvolučním vzorem. Výpočet výsledku filtrace probíhá vzorek po vzorku (pixel po pixelu) s tím, že každý pixel výsledného obrazu se vypočítá jako konvoluce okolí pixelu v původním obrazu s konvolučním vzorem, což je obvykle malá obdélníková nebo čtvercová plocha. Obvyklá velikost konvolučního vzoru bývá 3x3, 5x5 nebo 7x7 pixelů, ale nemusí to být pravidlem. Protože se jedná o diskrétní obrazy, lze konvoluci nahradit sumou. Výsledné pixely výstupního obrazu se vypočítají podle vzorce:

$$p_{2(x,y)} = \sum_{i=-i_{max}}^{i_{max}} \sum_{j=-j_{max}}^{j_{max}} p_{1(x+i,y+j)} \cdot k_{(i,j)} \tag{2.5.1}$$

kde:

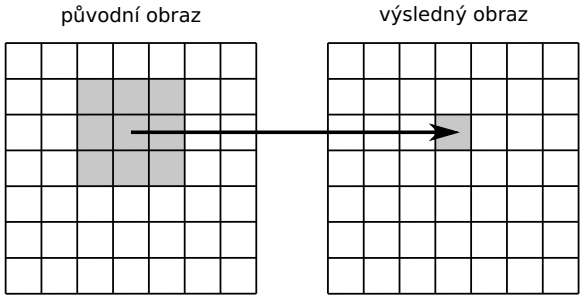
p_1 je hodnota vstupního pixelu,

p_2 je hodnota výstupního pixelu,

k je konvoluční vzor (maska),

x, y jsou souřadnice pixelu v obrazu,

i_{max}, j_{max} jsou maximální indexy konvoluční matice (šířka, resp. výška matice je rovna $2i_{max} + 1$, resp. $2j_{max} + 1$).



Obrázek 2.2: Situace při výpočtu konvolučního filtru.

Znázornění situace při výpočtu konvolučního filtru je na obrázku 2.2.

Filtrace se často používá za účelem odstranění šumu v obrazu – k tomu slouží průměrovací filtry. Výsledná hodnota pixelu je získána jako aritmetický průměr hodnot pixelů v jeho okolí. Konvoluční maska k pro okolí velikosti 3x3 vypadá takto:

$$k = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \tag{2.5.2}$$

Informace o konvolučním filtru byly čerpány z [22].

Gaussovský filtr

Mezi průměrovací filtry patří také Gaussovský filtr, který je použit v této práci pro úpravu textur.

Gaussovský filtr nepočítá prostý aritmetický průměr, ale průměr vážený Gaussovou funkcí. Míra, kterou se na výsledné odezvě pixel podílí, je Gaussovou funkcí jeho vzdálenosti od zpracovávaného pixelu. Je tak zesílen význam pixelu ve středu konvoluční masky a čím dále je pixel od středu, tím menší vliv má na výslednou hodnotu. Konvoluční maska Gaussova filtru může vypadat např. takto [19]:

$$k = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \tag{2.5.3}$$

Na obrázku 2.3 je ukázka použití Gaussovského filtru na obrazu.

Obrázek 2.3: Ukázka použití Gaussovského filtru. Vlevo je původní obraz, vpravo výsledek filtrace.

Gáborův filtr

Gáborův filtr je také lineární konvoluční filtr, ale už nepatří mezi základní filtrační metody používané v předzpracování obrazu. Používá se hlavně k detekci hran. Je také hojně využíván v segmentaci obrazu pro reprezentaci textury díky dojmu, že modeluje některé aspekty lidského vidění, a také díky několika atraktivním výpočetním vlastnostem.

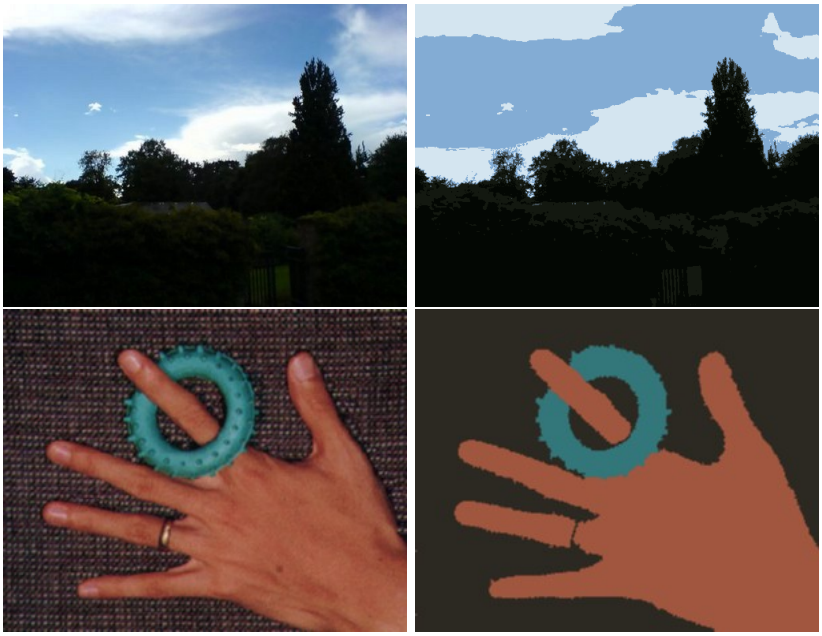
Z těchto důvodů si o tomto filtru řekneme více až při popisu konkrétního využití v části 4.3.1.

2.6 Segmentace

V této podkapitole se blíže podíváme na segmentaci, která je hlavním předmětem této práce.

Segmentace obrazu je klíčovým krokem většiny systémů pro zpracování obrazu. Jejím úkolem je rozlišení jednotlivých objektů v obrazu – obraz je rozdělen na oblasti korespondující s konkrétními objekty v obrazu. Každému pixelu je přiřazen segment reprezentující určitý objekt v obrazu. Pixely jsou rozděleny do segmentů na základě nějakých jejich vlastností. Tyto vlastnosti jsou klíčové pro kvalitu výsledné segmentace. Mezi tyto vlastnosti obvykle patří černobílá textura okolí nebo barva. Výsledky segmentace jsou dále zpracovány vyššími metodami při úkolech jako detekce přítomnosti daného objektu nebo nalezení a klasifikace objektů v obraze.

Ukázky segmentace jsou na obrázcích 2.4 a 2.5. Obrázek 2.4 zobrazuje segmentaci založenou na oblastech, na obrázku 2.5 jsou příklady segmentace založené na hranách. Principy obou metod jsou popsány v sekci Rozdělení segmentačních metod.



Obrázek 2.4: Segmentace obrazu založená na oblastech. Vlevo je původní obraz, vpravo segmentovaný.

Obrázek 2.5: Segmentace obrazu založená na hranách.

2.6.1 Definice segmentace

Podle obecné definice je segmentace proces dělení obrazu na části, které korespondují s konkrétními objekty v obrazu. Segmentaci však lze definovat i matematicky:

Segmentace obrazu $f(x, y)$ je jeho rozdělení na podobrazy $R_1, R_2, ..., R_n$ tak, že podobrazy splňují následující kritéria:

1. $\bigcup_{i=1}^{n} R_i = f(x, y)$
2. $R_i \cap R_j = \emptyset, i \neq j$
3. Každý podobraz R_i splňuje nějaké tvrzení, popř. množinu tvrzení, např:
   - všechny pixely podobrazu R_i mají stejnou úroveň šedi,
   - úroveň šedi všech pixelů podobrazu R_i se liší maximálně o předepsanou hodnotu,
   - standardní odchylka úrovni šedi všech pixelů poddobrazu R_i je dostatečně malá, apod.

2.6.2 Rozdělení segmentačních metod

Segmentační metody lze rozdělit do několika skupin [17]:

Metody založené na hranách (detekce hran) – metody se snaží detekovat významné hrany v obrazu tak, že hledají body, u kterých se ostře mění jas. V ideálním případě jsou výsledkem souvislé křivky, které značí hranice objektů nebo oblastí. Detekcí hran dochází ke značnému zmenšení objemu dat a filtrují se tak informace významnější pro daný účel. Úspěšná detekce hran může podstatně zjednodušit následnou interpretaci obrazu. Je ale problém najít správné hrany. Obrysy často nejsou spojité, některé úseky chybí, nebo naopak obsahují křivky, které neodpovídají hledaným objektům. [13]

Metody založené na oblastech – obraz je rozdělen na samostatné oblasti na základě nějaké jeho vlastnosti (např. jas, barva, textura, atd.) tak, že všechny pixely jedné oblasti mají podobné vlastnosti. Oblasti jsou souvislé a nepřekrývají se, takže každý pixel obrazu patří do právě jedné oblasti.

Metody založené na oblastech i metody založené na hranách řeší v podstatě stejný problém, jelikož každá oblast může být reprezentována uzavřenými hranicemi, a každá uzavřená hranice definuje oblast. Každá skupina však dává jiné výsledky a informace, proto lze oba typy kombinovat.

Statistické metody – jsou založeny na statistické analýze obrazových dat (nejčastěji to jsou hodnoty nějakých vlastností pixelů). Strukturní informace jsou obvykle zanedbávány. Mezi statistické metody se řadí **shluková analýza**, která se používá ke klasifikaci objektů. Shlukovací metody třídí objekty do skupin (shluků) na základě určitých vlastností. Příkladem shlukovací metody je algoritmus k-means, který je popsán v další podkapitole.

Hybridní metody – kombinují přístupy výše zmíněných metod. Mezi hybridní metody patří mimo jiné i metody využívající k segmentaci neuronové sítě. Tyto metody mají velmi blízko ke statistickým metodám. Vycházejí ze statistické analýzy obrazu, jejíž výsledky slouží jako vstupy neuronové sítě. Podrobněji se těmto metodám věnuje kapitola 4.

Metody založené na znalostech – znalosti o vlastnostech segmentovaných objektů (tvar, barva, struktura, apod.) mohou segmentaci značně usnadnit. Metody založené na znalostech využívají atlas modelů segmentovaných objektů (v případě medicínských dat např. atlas lidských tkání). Atlas může být generován automaticky z množiny trénovacích dat, nebo do něj mohou být informace vloženy ručně, na základě lidské zkušenosti. V průběhu segmentace algoritmus hledá transformaci známých objektů z atlasu na objekty nalezené v obraze.

2.6.3 Algoritmus K-means

Algoritmus K-means třídí n objektů do k shluků, $k < n$. Každý shluk je určen středem. Předpokládejme, že máme nějakou metriku, která definuje vzdálenost mezi dvěma objekty, např. euklidovská vzdálenost:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2.6.1}$$

Princip algoritmu je pak následující [14]:

Na začátku jsou středy shluků nějakým způsobem inicializovány, např. náhodně. Potom se opakují dva kroky: Nejdřív je každý objekt přiřazen do shluku s nejbližším středem. Potom jsou středy všech shluků přepočítány tak, aby odpovídaly skutečným středům objektů v daném shluku.

Algoritmus:

1. **Inicializace** – inicializace k středů $s_1, ..., s_k$ na náhodné hodnoty.
2. **Přiřazení** – každý objekt $x^{(n)}$ je přiřazen shluku S_k s nejbližším středem. Označme nejbližší střed s^*. Pak platí:
$$d(x^{(n)}, s_*) \leq d(x^{(n)}, s_i), i \in 1...k \tag{2.6.2}$$
3. **Aktualizace** – středy shluků jsou aktualizovány tak, aby odpovídaly skutečným středům objektů v daném shluku:
$$s_i = \frac{1}{|S_k|} \sum_{x_j \in S_k} x_j \tag{2.6.3}$$
4. Kroky 2 a 3 jsou opakovány, dokud se mění příslušnost objektů do shluků, nebo dokud míra změny polohy středů neklesne pod určitou hodnotu.

Kapitola 3

Neuronové sítě

Tato kapitola popisuje neuronové sítě a jejich základní principy. Uvedeme si několik oblastí využití neuronových sítí, ukážeme si jejich souvislost s biologickými nervovými systémy a popíšeme základní principy učení. Nakonec se blíže podíváme na některé typy neuronových sítí a podrobněji si popíšeme síť, která byla použita v praktické části.

3.1 Úvod

Neuronové sítě vznikly jako součást výzkumu umělé inteligence. Jsou výsledkem části výzkumu založené na modelování mikroskopických detailů nervového systému. Umělá neuronová síť se snaží simulovat strukturu a funkci biologických nervových systémů. Je tvořena skupinou propojených prvků (neuronů) pracujících paralelně.

Neuronová síť je obvykle adaptivní systém, který se učí na příkladech. Během učení mění svou strukturu tak, aby se co nejlépe přizpůsobila řešení daného problému. Naučenou neuronovou síť lze považovat za experta v řešení daného problému. Neuronové sítě jsou abstrakcí lidského mozku, hodí se proto k řešení problémů, které dokážou vyřešit lidé (např. rozpoznávání vzorů). Tradiční výpočetní metody mají s těmito úkoly problémy.

Tyto informace byly čerpány z [20].

3.1.1 Porovnání s tradičními výpočetními metodami

Neuronové sítě řeší problémy jinak než tradiční výpočetní metody. Tradiční metody používají k řešení problémů algoritmy – řídí se předem danými instrukcemi. Dokud počítač nezná všechny potřebné kroky, nedokáže daný problém vyřešit. To omezuje použití tradičních metod na problémy, kterým už rozumíme, a víme, jak je řešit.

Neuronové sítě zpracovávají informace podobně jako lidský mozek. Je obtížné je naprogramovat na provádění konkrétního úkolu, učí se na příkladech. Je důležité vybrat správné příklady, protože na nich silně závisí úspěšnost naučené neuronové sítě. Nevýhodou je, že jelikož síť sama najde řešení problému, její chování může být nepředvídatelné.

Neuronové sítě a tradiční výpočetní metody se navzájem doplňují. Pro některé úkoly se lépe hodí algoritmický přístup (např. pro aritmetické výpočty), pro jiné zase neuronové sítě. Často se také tyto dva přístupy kombinují (běžně se tradiční metody používají k učení neuronové sítě).

Tyto informace byly čerpány z [5].

3.2 Využití neuronových sítí

Neuronové sítě mají široké využití v oblasti obchodu. Díky své vynikající schopnosti rozpoznávat vzory se hodí tam, kde je třeba předpovídat další vývoj, např. předpověď tržeb, kontrola výrobních procesů, průzkum zákazníků, ověřování údajů, řízení rizika, cílový marketing, apod. [5].

Neuronové sítě byly úspěšně aplikovány i v mnoha jiných oborech, např. [13]:

- Matematika – aproximace funkcí, regresní analýza
- Zpracování obrazu – klasifikace, rozpoznávání vzorů a objektů (obličejů, řeči, gest, ručně psaného písma)
- Zpracování dat – čištění, shlukování, třídění, komprimace
- Robotika – navigace robotických ramen, číslicově řízené stroje

3.3 Biologický a umělý neuron

Abychom dokázali modelovat biologické procesy, je nutné porozumět detailům biologických systémů, v nichž tyto procesy probíhají. Základem je pochopit, jak funguje neuron – základní stavební jednotka biologických nervových systémů.

3.3.1 Biologický neuron

Biologický neuron komunikuje s ostatními neurony pomocí elektrochemických signálů [13]. Skládá se ze čtyř základních částí:

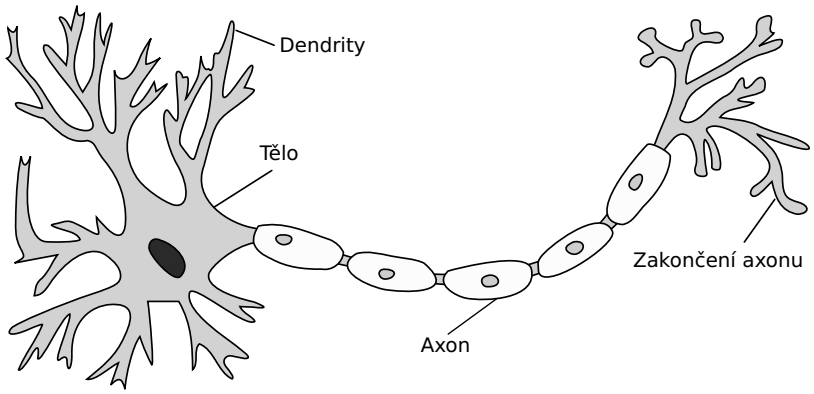
- tělo
- dendrity – vstupy
- axon – výstup
- synapse – rozhraní mezi axonem jednoho a dendritem druhého neuronu

Neuron přijímá signály od ostatních neuronů pomocí dendritů a vysílá impulsy do ostatních neuronů pomocí axonu. Axon se na konci větví do tisíců výběžků zakončených synapsemi, které spojují axon neuronu s dendrity ostatních neuronů. Zjednodušená podoba neuronu je na obrázku 3.1.

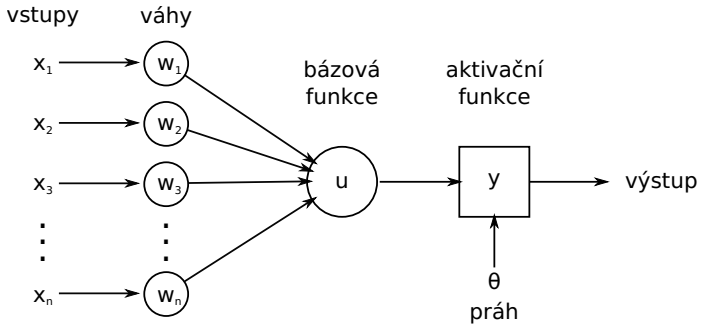
3.3.2 Umělý neuron

Umělý neuron je abstrakcí biologického neuronu [13]. Stejně jako biologický neuron má vstupy a výstup, pomocí nichž komunikuje s ostatními neurony. Vstupů je obvykle více, výstup je pouze jeden. V neuronové síti jsou neurony propojeny tak, že výstup jednoho neuronu je zapojen na vstup dalšího neuronu. Každý vstup je násoben malou hodnotou – váhou. Tělo neuronu je reprezentováno dvěma matematickými funkcemi: bázovou a aktivační [21]. Struktura umělého neuronu je na obrázku 3.2.

Bázová funkce – označuje se u, parametry jsou vstupy a váhy. Může být lineární nebo radiální [21]:



Obrázek 3.1: Biologický neuron a jeho základní části. Převzato z [13].



Obrázek 3.2: Struktura umělého neuronu.

- Lineární – výstupem je vážený součet vstupů daný následujícím vztahem:

$$u = \sum_{i=1}^{n} w_i x_i \tag{3.3.1}$$

- Radiální – výstupem je euklidovská vzdálenost vektoru vstupů od vektoru vah popsaná následovně:

$$u = ||\vec{x} - \vec{w}|| = \sqrt{\sum_{i=1}^{n} (x_i - w_i)^2} \tag{3.3.2}$$

Aktivační funkce – označuje se y, parametrem je výstup bázové funkce. Používají se různé aktivační funkce, nejjednodušší je skoková, nejčastější je sigmoida [16, 21]:

- Skoková – má binární výstup. Používá se v binárních klasifikátorech, které klasifikují vzory do dvou skupin. Výstup může vypadat např. takto:

$$y = \begin{cases} 0 & \text{pro } u \leq \theta \\ 1 & \text{pro } u > \theta \end{cases} \tag{3.3.3}$$

  kde θ je nějaký práh.

- Sigmoida – má esovitý tvar, běžně se používá funkce daná vztahem:

$$y = \frac{1}{1 + e^{-\lambda u}} \tag{3.3.4}$$

kde λ určuje strmost funkce.

Násobením vstupů vahami je dosaženo toho, že jednotlivé vstupy mají různý vliv na výstup neuronu. Jaký vliv bude mít vstup na výstup neuronu, závisí na váze daného vstupu. Neuron má schopnost přizpůsobit se konkrétní situaci změnou vah a prahu – to probíhá ve fázi učení.

3.4 Učení

Učení probíhá během trénovací fáze návrhu sítě. Konkrétní postup použitý pro trénování se nazývá trénovací pravidlo. Rozlišují se dva základní typy učení – s učitelem a bez učitele [20]:

Učení s učitelem – je vytvořena množina trénovacích vzorů reprezentujících reálná vstupní data řešeného problému. Tato množina se nazývá trénovací množina. Každý trénovací vzor je tvořen jedním vstupem a požadovaným výstupem neuronové sítě. Během učení jsou trénovací vzory postupně předkládány neuronové síti a váhy sítě jsou upravovány pomocí trénovacího pravidla tak, aby daný vstup vyvolal požadovaný výstup. Váhy postupně konvergují k hodnotám, které vyvolají správné výstupy u všech trénovacích vzorů.

Učení bez učitele – od učení s učitelem se liší v tom, že trénovací data neobsahují požadované výstupy. Při použití této metody neuronová síť hledá v datech shluky podobných vzorů a každému shluku přiřadí nějaký uzel (jeden z neuronů). Uzlem se stává ten neuron, který na vstupy daného shluku vyvolává nejsilnější odezvu. Tomuto procesu se také říká samoorganizace [8].

Dalším rozdílem je, že při učení s učitelem je fáze učení oddělená od fáze, kdy už síť řeší konkrétní problém. Při učení bez učitele tyto dvě fáze probíhají současně – síť se učí během celého svého chodu [5].

Pro trénování neuronových sítí existuje množství různých metod. Většina je založena na metodě *největšího spádu*, která minimalizuje chybu sítě při učení s učitelem. Využívá ji např. metoda *zpětného šíření chyby*, což je nejznámější trénovací metoda [2].

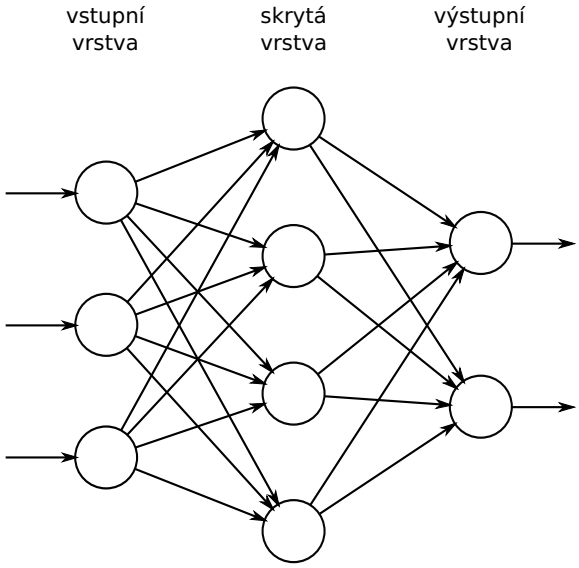
Metoda zpětného šíření chyby Nejdříve je vypočítána chyba sítě na základě porovnání výstupů sítě s požadovanými výstupy. Chyba je potom různými metodami přenášena sítí nazpět (od výstupních neuronů ke vstupním) – podle ní jsou postupně upravovány váhy u všech spojení tak, aby se chyba snížila. Tento proces se opakuje tak dlouho, dokud není chyba dostatečně malá. Podrobněji se touto metodou nebudeme zabývat, jelikož v této práci se nevyužívá. Podrobnější popis lze najít v [2, 13].

3.5 Typy neuronových sítí

Existuje mnoho typů neuronových sítí a lze je kategorizovat podle různých vlastností, např. architektury, způsobu učení, aplikace nebo způsobu výpočtu [21]. Zmíníme zde pouze dopřednou neuronovou síť coby nejjednodušší typ sítě a podrobněji popíšeme samoorganizační mapu, která byla použita pro experimenty.

3.5.1 Dopředná neuronová síť

Dopředná neuronová síť byla prvním a asi nejjednodušším typem neuronové sítě. Neurony jsou uspořádány do vrstev a tok informací probíhá pouze jedním směrem – od vstupních neuronů k výstupním (dopředu). Síť neobsahuje žádné cykly ani smyčky [13]. Ukázka architektury je na obrázku 3.3.



Obrázek 3.3: Příklad dopředné neuronové sítě. Počet vrstev může být libovolný stejně jako počet neuronů v nich.

Dopředné neuronové sítě lze rozdělit podle počtu vrstev neuronů na dva typy – jednovrstvé a vícevrstvé [13]:

Jednovrstvá dopředná síť – je nejstarším typem neuronové sítě. Skládá se z jediné (výstupní) vrstvy neuronů – vstupy jsou zapojeny přes váhy přímo do výstupů. Jednovrstvá síť má binární výstup, např. 0/1 nebo -1/1. V každém uzlu se spočítá vážený součet vstupů a pokud je hodnota nad nějakou hranicí, výstupem je 1 (neuron se aktivuje), jinak 0 (neuron se deaktivuje). Vrstva obvykle obsahuje více neuronů. Síti obsahující pouze jeden neuron se říká perceptron.

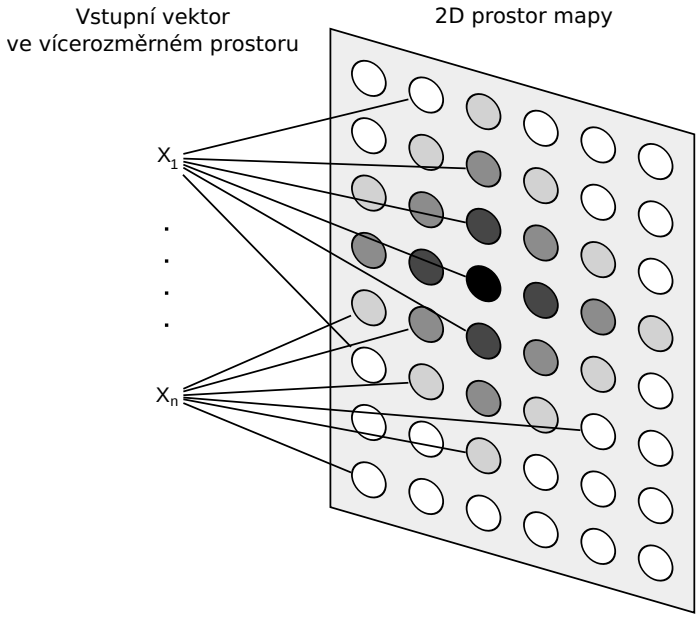
Vícevrstvá dopředná síť – skládá se z několika vrstev neuronů zapojených dopředně – výstupy neuronů jedné vrstvy jsou zapojeny na vstupy neuronů následující vrtsvy. Aktivační funkcí neuronů je často funkce sigmoida. Pro učení vícevrstvých sítí se používají různé metody, nejpopulárnější je metoda zpětného šíření chyby.

3.5.2 Samoorganizační mapa (SOM)

Samoorganizační mapa, nazývaná také Kohonenova mapa (podle T. Kohonena), je nejznámějším zástupcem neuronových sítí založených na učení bez učitele. Není závislá na informaci o příslušnosti vzorů do tříd, ale je schopná se samoorganizovat tak, že vzory rozpozná automaticky. Využívá se pro vizualizaci a analýzu vícerozměrných dat, kdy potřebujeme snížit počet rozměrů, abychom mohli s daty dále pracovat a byli schopní je interpretovat.

Samoorganizační mapa mapuje vstup o velkém počtu rozměrů do prostoru sítě s menším počtem rozměrů. Každý vstupní vektor je tak klasifikován do jedné z několika výstupních tříd. Klasifikace probíhá tak, že se najde neuron, jehož váhový vektor nejlépe reprezentuje vstupní vektor, a vstupní vektor je namapován na tento neuron, tzn. je klasifikován do třídy reprezentované tímto neuronem.

Samoorganizační mapa je tvořena dvourozměrným polem neuronů, z nichž každý je n váhami plně propojen s n-rozměrným vstupním vektorem. Struktura sítě je znázorněna na obrázku 3.4.



Obrázek 3.4: Samoorganizační mapa - propojení neuronů se vstupním vektorem.

Učení probíhá tak, že síti jsou postupně předkládány vzory z trénovací množiny a vybere se vždy neuron, jehož váhy nejlépe odpovídají vstupu - tento neuron se označuje jako vítězný. Váhy vítězného neuronu a sousedních neuronů v určitém jeho okolí jsou potom upraveny tak, aby se rozdíl mezi váhami a vstupem ještě zmenšil. Příště tak dá tento neuron na stejný vstup ještě silnější odezvu. Síť tak provádí shlukování – podobné vstupy produkují podobné výstupy.

Pro určení sousedních neuronů existuje více metod. Lze použít pevnou vzdálenost, kdy se stejným způsobem upraví váhy všech neuronů do určité vzdálenosti, nebo Gaussovu funkci, která nejvíce přiblíží vstupu střední neuron a míra úpravy vah okolních neuronů postupně klesá se vzdáleností od středu. Velikost okolí se v průběhu učení obvykle postupně zmenšuje.

Výstup samoorganizační mapy se velmi liší od výstupu dopředné sítě. V případě dopředné sítě s pěti výstupními neurony obdržíme výstup složený z pěti hodnot. U samoorganizační mapy ve skutečnosti produkuje výstup pouze jeden neuron. Nezajímá nás hodnota výstupu, ale číslo neuronu, který výstup vyprodukoval.

Algoritmus lze popsat následovně [19]:

1. Inicializuj váhy na náhodné hodnoty (nebo hodnoty náhodně vybraných vstupních vektorů).

2. Pro každý vstupní vektor v_i z trénovací množiny:

(a) Urči vítězný neuron:

$$j^* = \arg\max_j \sum_i w_{ij} v_i$$

kde w_{ij} je váha od i-tého vstupu do j-tého neuronu.

(b) Všem neuronům n_j v okolí do vzdálenosti d neuronu n_j^* uprav váhy:

$$w_{ij} = w_{ij} + \alpha(v_i - w_{ij})$$

kde α je konstanta učení, obvykle se volí jako malé číslo z intervalu $(0, 1)$ a v průběhu učení se postupně snižuje.

3. Zmenši vzdálenost d a konstantu učení α.

4. Opakuj od kroku 2, dokud chyba sítě není dostatečně malá.

Informace o samoorganizační mapě byly čerpány z [4, 19], kde lze také nalézt více podrobnějších informací.

Kapitola 4

Segmentační systém

Cílem této práce bylo vytvořit systém pro segmentaci obrazů pomocí neuronové sítě, který by umožňoval srovnání výsledků se segmentací pomocí tradičního algoritmu používaného k tomuto účelu.

Tato kapitola se zabývá podrobným popisem vytvořeného segmentačního systému. Nejdříve nabídne ucelený pohled na celkovou strukturu systému – jeho základní prvky, vztahy mezi nimi a data, s kterými tyto prvky pracují. Další podkapitoly jsou věnovány podrobnému popisu těchto prvků a procesů, které v nich probíhají. Nakonec zmíníme několik základních informací o implementaci.

4.1 Struktura systému

Systém se skládá ze čtyř hlavních prvků:

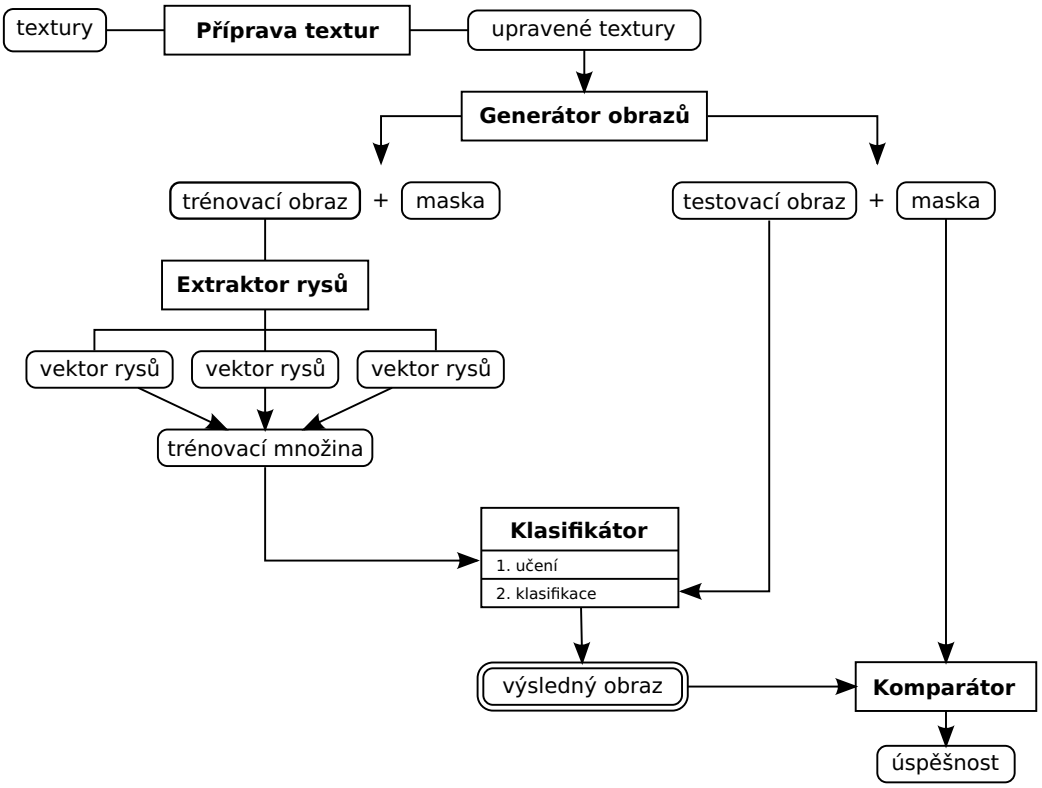
- generátor obrazů
- extraktor rysů
- klasifikátor
- komparátor

Generátor obrazů slouží k vytvoření vstupních obrazů pro klasifikátor. Obrazy jsou vytvořeny z jednotlivých textur, které jsou nejdříve upraveny. Vstupní obrazy jsou rozděleny na trénovací a testovací. Trénovací slouží pro naučení klasifikátoru a na testovacích je poté otestována jeho úspěšnost.

Jádro segmentačního systému tvoří extraktor rysů a klasifikátor. Pomocí extraktoru rysů jsou z obrazu získávány informace o jednotlivých pixelech (vektory rysů), z nichž je vytvořena trénovací množina pro klasifikátor. Klasifikátor je na těchto datech natrénován a poté je použit k vlastní segmentaci testovacích obrazů.

Pomocí komparátoru je následně vyhodnocena úspěšnost segmentace. Úspěšnost je vyhodnocena na základě porovnání výsledného segmentovaného obrazu a vzorové segmentační masky.

Celou strukturu systému a segmentační proces shrnuje obrázek 4.1.



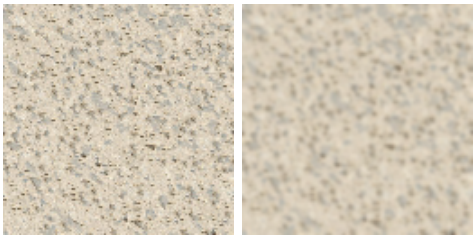
Obrázek 4.1: Celková struktura segmentačního systému.

4.2 Vstupní data

Pro správné naučení neuronové sítě je potřeba použít při trénování velké množství dat. Je problém získat takové množství reálných obrazů, proto pro trénování budeme používat syntetická data. Za tímto účelem byl vytvořen generátor obrazů, který generuje obrazy tvořené segmenty z různých textur. Generování je popsáno v podkapitole 4.2.2.

4.2.1 Příprava textur

Textury byly ještě před generováním upraveny tak, aby se potlačily některé nežádoucí vlastnosti jako velké rozdíly v barevnosti nebo příliš výrazný texturní vzor. K potlačení těchto vlastností byl použit Gaussovský filtr popsaný v části 2.5.1. Ukázka textury před a po filtraci je na obrázku 4.2.



Obrázek 4.2: Ukázka textury před a po použití Gaussovského filtru. Vlevo je původní obrázek, vpravo filtrovaný.

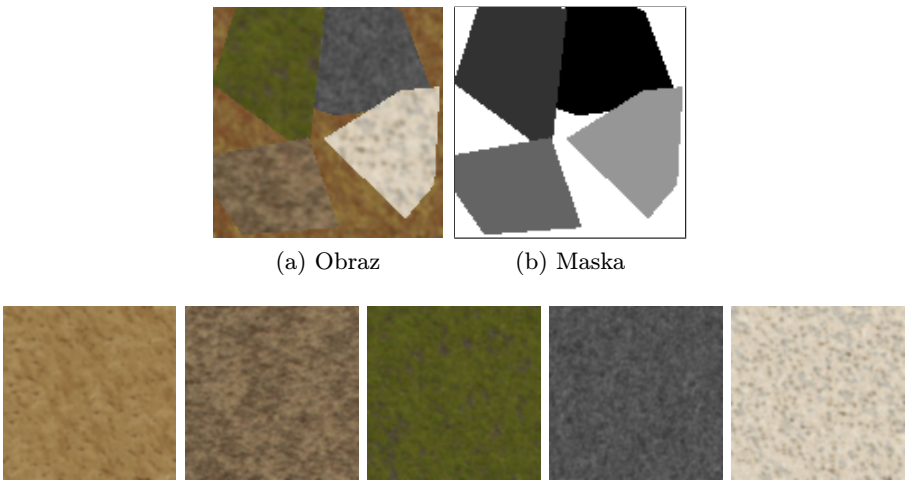
4.2.2 Generátor obrazů

Generátor obrazů slouží k vytvoření syntetických obrazů pro trénování neuronové sítě.

Generátor generuje z dané množiny textur obrazy obsahující různý počet segmentů, každý s jinou texturou. Segmenty se od sebe liší kromě textury také tvarem a velikostí. Segmenty mají tvar mnohoúhelníku, jehož tvar je určen středem a určitým počtem vrcholů, které jsou rozmístěny po obvodu kružnice okolo středu segmentu. Víceméně rovnoměrného rozložení vrcholů je dosaženo tak, že kružnice je rozdělena na několik segmentů a v každém segmentu je náhodně vygenerován jeden vrchol. Podobně je určena i poloha segmentů v obrazu – středy segmentů jsou generovány po obvodu kružnice okolo středu obrazu. Počet segmentů je generován náhodně ze zadaného intervalu.

Spolu s každým obrazem je vygenerována i jeho maska, která představuje ideálně segmentovaný obraz. Segmenty v masce jsou odlišeny barvou, přičemž však přiřazení barvy určitému segmentu není pevně dáno, tzn. nezáleží na tom, který segment má jakou barvu. Podstatné je pouze odlišení různých segmentů od sebe. Maska slouží pro vyhodnocení úspěšnosti segmentace popsané v následující podkapitole.

Ukázka vygenerovaného obrazu a jeho masky spolu s použitými texturami je na obrázku 4.3.



(a) Obraz (b) Maska

(c) Textury

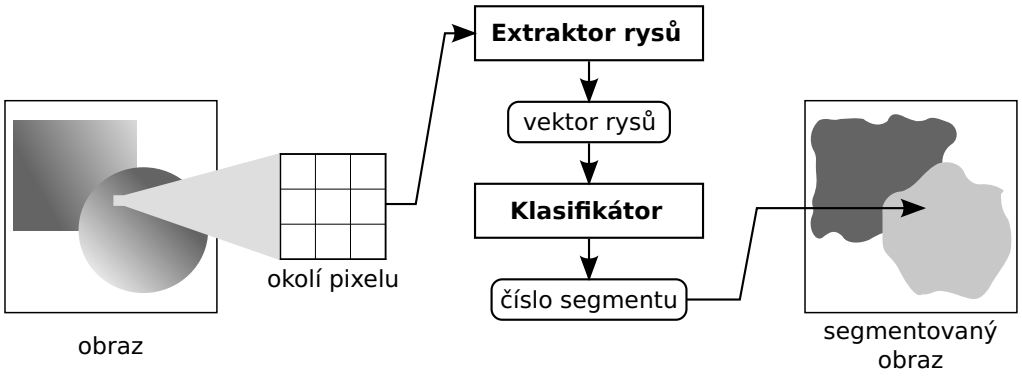
Obrázek 4.3: Ukázka vygenerovaného obrazu a jeho masky. Dole jsou použité textury.

4.3 Jádro segmentačního systému

Jádro segmentačního systému tvoří extraktor rysů a klasifikátor. Každý z nich reprezentuje příslušnou fázi zpracování obrazu při segmentaci. Extraktor i klasifikátor operují vždy pouze nad jedním pixelem nebo popřípadě nad jeho okolím (v případě práce s texturními příznaky). Segmentace obrazu tedy probíhá tak, že se obraz prochází pixel po pixelu, a pomocí extraktoru a klasifikátoru jsou vytvořeny výstupní pixel, které tvoří výsledný segmentovaný obraz.

Prvním krokem zpracování pixelu je extrakce rysů. Extraktor vypočítá hodnoty reprezentující požadované informace o zpracovávaném pixelu a vytvoří tzv. vektor rysů. Tento

vektor rysů je vstupem klasifikátoru, který mu přiřadí jeden ze segmentů na základě naučených vzorů (viz učení samoorganizační mapy 3.5.2). Každému segmentu ve výsledném obrazu odpovídá nějaká barva tak, aby bylo možné jednotlivé segmenty jednoduše rozlišit. Tedy už jen zbývá určit barvu pixelu odpovídající přiřazenému segmentu. Znázornění celého procesu je na obrázku 4.4.



Obrázek 4.4: Proces zpracování obrazu po pixelech.

V následujících podkapitolách si blíže popíšeme prvky tvořící jádro systému.

4.3.1 Extraktor rysů

Extraktor rysů (příznaků) slouží k získání vektoru rysů pixelu, na základě něhož je daný pixel přiřazen klasifikátorem do jedné z cílových tříd. Bylo vytvořeno více typů extraktoru, z nichž každý umožňuje získat hodnoty jiných vlastností pixelu. Hlavním cílem práce bylo porovnání výsledků segmentace na základě barvy a textury, byly proto vytvořeny extraktory pro získání jak barevných, tak texturních příznaků a také jejich kombinace. Následující seznam popisuje použité příznakové vektory:

Barevné RGB příznaky - vektor je tvořen 3 složkami barevného modelu RGB - Red (červená), Green (zelená), Blue (modrá).

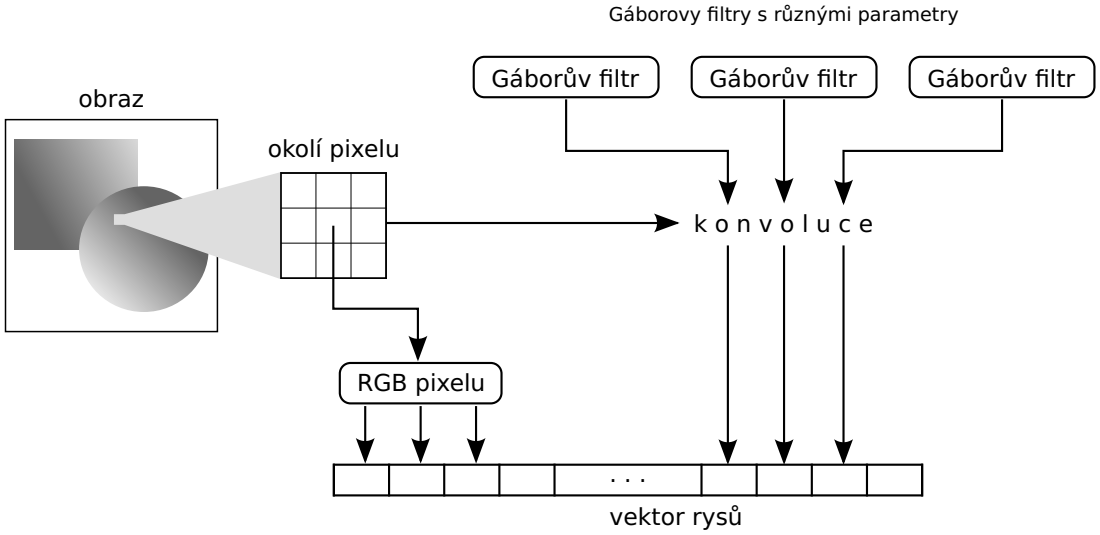
Barevné HSV příznaky - vektor je tvořen 3 složkami barevného modelu HSV - Hue (odstín), Saturation (sytost barvy), Value (hodnota jasu), známém také jako HSB - Hue, Saturation, Brightness (jas).

Texturní příznaky - složkami vektoru je 24 výstupů Gáborových filtrů (popsány v následující podkapitole).

Kombinované texturní + barevné RGB příznaky - vektor je vytvořen spojením texturních a barevných RGB příznaků, jeho velikost je tedy $24 + 3 = 27$.

Kombinované texturní + barevné HSV příznaky - vektor je vytvořen spojením texturních a barevných HSV příznaků, jeho velikost je tedy opět 27.

Proces vytvoření příznakového vektoru znázorňuje obrázek 4.5.



Obrázek 4.5: Proces vytvoření příznakového vektoru.

Gáborovy filtry

Abychom mohli textury porovnávat, musíme je popsat číselnými hodnotami, které je budou charakterizovat. Matematických modelů pro popis textury existuje celá řada. Rozhodli jsme se pro použití Gáborových filtrů z následujících důvodů:

Gáborovy filtry jsou hojně využívány pro popis textury díky reprezentaci frekvence a orientace, která je podobná lidskému vnímání textury, a také díky několika atraktivním výpočetním vlastnostem. Dvourozměrný Gáborův filtr je používán k detekci frekvencí v různých směrech. Pomocí různých parametrů lze navrhnout sadu Gáborových filtrů tak, aby pokryly celou frekvenční doménu obrazu [4].

Jádro dvourozměrného Gáborova filtru tvoří Gaussova funkce, která je modulovaná sinusoidou. Třída Gáborových funkcí je definována následovně [6]:

$$G(x,y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\cos\left(\frac{2\pi x'}{\lambda} + \psi\right) \tag{4.3.1}$$

kde

$$x' = x\cos\theta + y\sin\theta$$

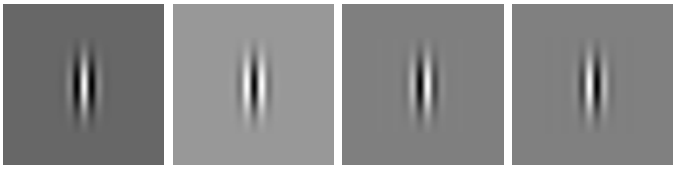
$$y' = -x\sin\theta + y\cos\theta$$

Ve vztahu figuruje pět parametrů, které ovlivňují tvar Gáborovy funkce. Tvar funkce určuje, jaké frekvence a orientace textury lze tímto filtrem zachytit. Při experimentování jsme se snažili nalézt takové kombinace hodnot parametrů, které by co nejlépe zachytily vlastnosti textur, což je důležité pro jejich správnou klasifikaci. Proto si parametry a jejich vliv na tvar funkce vysvětlíme podrobněji. Pro znázornění byly použity obrázky z [18].

Vlnová délka λ - vlnová délka sinusoidy. Hodnota je v našem případě specifikována v pixelech. Platnými hodnotami jsou reálná čísla v intervalu $\langle 2, \infty)$. Ukázka Gáborových funkcí s různou vlnovou délkou je na obrázku 4.6.

Fázový posun ψ - fázový posun sinusoidy v stupních. Ukázka je na obrázku 4.7.

Obrázek 4.6: Jádra dvourozměrného Gáborova filtru s vlnovými délkami $\lambda = 5$, 10 a 15 v pořadí zleva doprava.

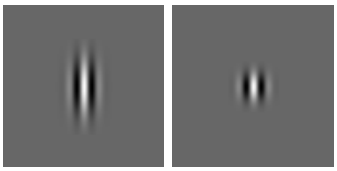


Obrázek 4.7: Gáborovy filtry s fázovým posunem $\psi = 0°, 180°, -90°$ a $90°$ v pořadí zleva doprava.

Obrázek 4.8: Gáborovy filtry s orientací $\theta = 0°, 45°$ a $90°$ v pořadí zleva doprava.

Orientace θ - tento parametr určuje orientaci souběžných čar Gáborovy funkce, udává se v stupních. Ukázka je na obrázku 4.8.

Poměr stran γ - tento parametr určuje eliptičnost Gaussovy funkce. Pro $\gamma = 1$ má funkce kruhový tvar, pro $\gamma < 1$ je prodloužena ve směru souběžných čar. Demonstrace hodnot je na obrázku 4.9.



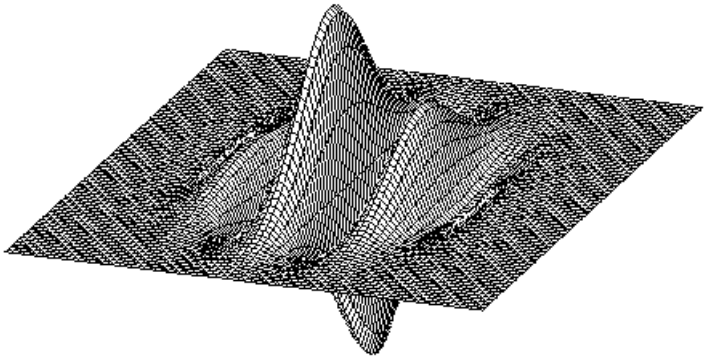
Obrázek 4.9: Gáborovy filtry s parametrem poměru stran $\gamma = 0.5$ a 1 v pořadí zleva doprava.

Rozpětí σ - určuje šířku Gaussovy funkce. Čím větší σ, tím širší gaussián a tím větší počet viditelných souběžných čar. Ukázka je na obrázku 4.10.

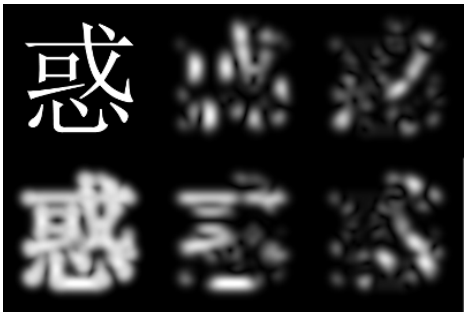
Pro lepší představu si ukážeme znázornění dvourozměrného Gáborova filtru ve třech rozměrech. Příklad je na obrázku 4.11.

Každý Gáborův filtr je naladěný k detekci pouze určitého lokálního vzoru dané frekvence, orientovaného pod daným úhlem. Abychom z obrazu získali užitečné rysy, je potřeba sada Gáborových filtrů s různými hodnotami parametrů. Ukázka takové sady filtrů je na obrázku 4.12.

Obrázek 4.10: Gáborovy filtry s parametrem rozpětí σ klesajícím zleva doprava.



Obrázek 4.11: Ukázka dvourozměrného Gáborova filtru v 3D.



Obrázek 4.12: Ukázka sady Gáborových filtrů aplikovaných na čínský znak. Vpravo jsou výstupy jednotlivých filtrů s orientací $\theta = 0°$, $45°$, $90°$ a $135°$. Vlevo je původní obrázek a pod ním superpozice všech čtyř filtrů [13].

Pro experimenty v této práci bylo použito 24 Gáborových filtrů s kombinacemi parametrů následujících hodnot:

Vlnová délka λ : 2, 3, 4

Fázový posun ψ : $0°$, $90°$

Orientace θ : $0°$, $45°$, $90°$, $135°$

Poměr stran γ : 1

Rozpětí σ : 0.56λ (odpovídá vlnovému rozsahu $b = 1$, blíže viz [18])

Výstup Gáborova filtru je z obrazu vypočítán pomocí konvoluce. Velikost konvolučního vzoru je vypočítána na základě parametrů σ a γ a konvoluční matice je následně naplněna

hodnotami podle vzorce 4.3.1. Výstupní hodnota pixelu se vypočítá jako konvoluce okolí pixelu původního obrazu s konvoluční maticí podle vztahu 2.5.1 uvedeného v části 2.5.1.

Konvolucí všech 24 Gáborových filtrů tak získáme pro každý pixel 24-rozměrný vektor rysů, který by měl texturu poměrně dobře charakterizovat.

4.3.2 Klasifikátor

Klasifikátor klasifikuje pixely do výstupních tříd (segmentů) na základě vektorů rysů získaného extraktorem.

Klasifikátorem je v našem případě neuronová síť typu samoorganizační mapa, kterou jsme si popsali v podkapitole 3.5.2. Vstupem této sítě je vektor rysů vytvořený extraktorem, počet vstupních neuronů je tedy roven velikosti vektoru rysů. Vstupní data je nutné před začátkem učení sítě vhodným způsobem normalizovat. Popisu normalizačních metod je věnována podkapitola 4.3.2.

Výstupy sítě reprezentují třídy, do kterých chceme vstupní data klasifikovat – v případě segmentace jsou to segmenty, do kterých chceme obraz rozdělit. Pro každou cílovou třídu je ve výstupní vrstvě přítomen jeden neuron, který je aktivován právě tehdy, pokud vstupní vektor do této třídy náleží. Klasifikace spočívá v prostém výběru výstupního neuronu, na němž byla zjištěna maximální odezva. Index cílového segmentu odpovídá pořadí vítězného neuronu ve výstupní vrstvě.

Počet neuronů sítě je pevně dán při jejím vytvoření a tím je pevně dán i počet cílových tříd. Klasifikátor s pevným počtem výstupních tříd není z hlediska segmentace nejvhodnější, je proto důležité počet tříd vhodně zvolit s ohledem na typ segmentovaných dat – např. podle počtu nejčastěji se vyskytujících objektů v obrazu. Existují metody, které se snaží řešit tuto nevýhodu, těmi se však tato práce nezabývá.

Činnost klasifikátoru je rozdělena do dvou fází – učení a samotná segmentace. Učení probíhá na trénovací množině, která je tvořena vektory rysů získanými extraktorem z pixelů trénovacích obrazů. Jsou dvě možnosti, jak vytvořit trénovací množinu – buď vybrat pixely náhodně ze všech trénovacích obrazů, anebo jen z jednoho aktuálního obrazu určeného k segmentaci. V druhém případě by měla být úspěšnost klasifikátoru vyšší, protože trénovací množina neobsahuje žádné prvky, které by se v segmentovaném obrazu nenacházely – učí se na mnohem přesnějších datech. V experimentech vyzkoušíme obě možnosti.

Jako referenční klasifikátor pro porovnání úspěšnosti neuronové sítě byl zvolen shlukovací algoritmus K-means, popsaný v sekci 2.6.3. Tento algoritmus je velice podobný algoritmu samoorganizační mapy. Počet výstupních tříd zde odpovídá počtu shluků, který je také pevně dán.

Normalizace vstupů neuronové sítě

Ve většině aplikací je nutné před trénováním sítě vstupní data nějakým způsobem předzpracovat. V praxi má volba metody předzpracování často velký vliv na úspěšnost segmentace. Jednou z nejběžnějších metod předzpracování je normalizace. Normalizace je užitečná v případě, kdy různé proměnné nabývají velmi rozdílných hodnot. Například v systému monitorujícím rostliny jsou dvěma vstupy teplota a tlak. V závislosti na použitých jednotkách mohou nabývat hodnot lišících se od sebe o několik řádů. Normalizací lze dosáhnout podobných hodnot u všech vstupů. Více informací o možnostech předzpracování vstupních dat lze najít v [3].

V samoorganizační mapě je nutné vstupy normalizovat na hodnoty v intervalu od -1 do 1 [10]. Někdy je normalizace chápána jako další vrstva sítě. Data lze normalizovat více

způsoby. Uvedeme si dvě nejčastější metody [10, 15], které jsme vyzkoušeli na jednoduchém příkladu – vytvořili jsme obrazy se stejným počtem barevných segmentů, kde každý segment měl jednu z 12 různých barev.

Multiplikativní normalizace - spočívá v podělení všech složek vektoru jeho délkou. Délka vektoru se vypočítá podle vzorce:

$$l = \sqrt{\sum_{i=1}^{n} x_i^2} \tag{4.3.2}$$

kde n je rozměr vektoru a x_i je prvek vektoru.

Normalizační faktor je pak dán vztahem:

$$f = \frac{1}{l} \tag{4.3.3}$$

Normalizovaný vektor získáme jednoduše vynásobením jeho složek normalizačním faktorem.

Tato metoda se však příliš neosvědčila, síť ve většině případů vstupní vektor klasifikovala špatně, i přesto, že obrazy obsahovaly malý počet segmentů.

Z-normalizace - tato metoda je nejběžnější alternativou k jednoduché multiplikativní normalizaci. Název pochází z analogie s počítačovou grafikou, kde se používá imaginární osa Z vycházející ze souřadnicového systému (X, Y) obrazovky směrem do oka pozorovatele. Z-normalizace spočívá ve vytvoření dalšího rozměru v datech. Vstupní vektor je rozšířen o uměle vytvořený vstup, jehož hodnota je vypočítána na základě aktuálních složek vektoru. Tato hodnota je vybrána tak, aby délka rozšířeného vektoru byla konstantní.

Normalizační faktor f a umělá vstupní hodnota s jsou vypočítány následovně:

$$f = \frac{1}{\sqrt{n}} \tag{4.3.4}$$

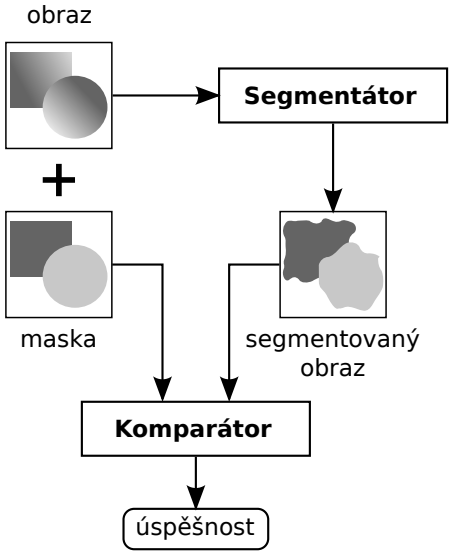
$$s = f\sqrt{n - l^2} \tag{4.3.5}$$

kde n je rozměr vektoru a l je délka vektoru definovaná v (4.3.2). Normalizovaný vektor získáme opět vynásobením jeho složek normalizačním faktorem a přidáním umělého vstupu do vektoru. Jak vidíme, normalizační faktor zde narozdíl od multiplikativní metody nezávisí na samotných datech, ale pouze na velikosti vstupu, která je konstantní. Díky tomu je zachována informace o absolutní velikosti vektoru. Protože tuto informaci většinou potřebujeme, je tato metoda poměrně oblíbená.

Výsledky segmentace potvrdily výhody této metody. Narozdíl od multiplikativní metody síť nyní ve většině případů přiřadila pixelu správný segment, a to i v případě velkého počtu segmentů v obraze.

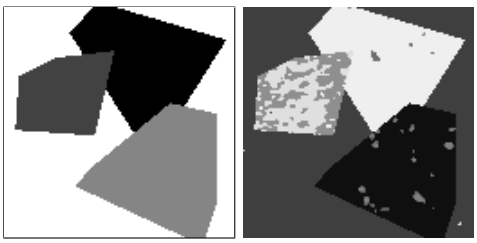
4.4 Komparátor

Úkolem komparátoru je vyhodnotit úspěšnost segmentace. Úspěšnost je vyhodnocena na základě porovnání segmentovaného obrazu a jeho masky, vytvořené generátorem zároveň se vstupním obrazem. Proces porovnání je znázorněn na obrázku 4.13.



Obrázek 4.13: Proces porovnání úspěšnosti segmentace.

Za předpokladu, že by měly odpovídající segmenty v obou obrazech stejnou barvu, stačilo by spočítat počet shodných pixelů. To však není náš případ. Barva pixelu segmentovaného obrazu je určena číslem segmentu, které odpovídá indexu výstupního neuronu v neuronové síti. Samoorganizační mapa nemá předem daný požadovaný výstup, mapování vstupního vektoru na výstupní neuron je vytvořeno během učení sítě a není tedy pevně dáno. Proto nelze očekávat, že odpovídající segmenty porovnávaných obrazů budou mít stejnou barvu. Znázornění této situace je na obrázku 4.14.



Obrázek 4.14: Ukázka situace, kdy odpovídající si segmenty v porovnávaných obrazech mají jinou barvu. Vlevo je maska, vpravo segmentovaný obraz.

Pro vyhodnocení úspěšnosti byla proto navrhnuta následující metoda:

Nejdříve musíme zjistit, které barvy výsledného obrazu a masky si odpovídají, tzn. která barva segmentovaného obrazu se namapovala na který segment v masce. To zjistíme tak, že vyvoříme korespondenční matici, jejíž rozměry odpovídají počtu segmentů v každém obrazu, a prvek na pozici (i, j) obsahuje počet pixelů, které mají v jednom obrazu barvu i a v druhém j. V případě dokonalé segmentace by se všechny hodnoty nacházely na diagonále a ostatní prvky by byly rovné 0. To je však velmi nepravděpodobné a proto musíme zjistit, které řádky a sloupce tvoří odpovídající si dvojici. To zjistíme tak, že najdeme v matici prvek s maximální hodnotou a odstraníme daný řádek i a sloupec j, kde (i, j) je pozice prvku. Odstraněný řádek a sloupec odpovídá korespondující dvojici segmentů. Znovu vyhledáme největší prvek a odstraníme odpovídající řádek a sloupec. Proces opakujeme tak dlouho,

dokud matice obsahuje nějaké řádky i sloupce. Pokud má např. řádky, ale žádné sloupce, znamená to, že klasifikátor našel v obrazu více segmentů než v něm skutečně bylo.

Jakmile známe odpovídající si dvojice segmentů, stačí spočítat počet pixelů odpovídající danému mapování (jakoby to byl počet shodných pixelů) a spočítat jejich součet pro všechny dvojice. Tento součet odpovídá počtu shodných pixelů v případě, že by měly odpovídající segmenty stejnou barvu. Nyní už stačí pouze vydělit toto číslo celkovým počtem pixelů v obrazu a po vynásobení stem získáme procentuální úspěšnost.

Pro větší přehlednost si uvedeme algoritmus výpočtu:

1. Vytvoříme korespondenční matici. Oba obrazy procházíme po pixelech a inkrementujeme prvek matice na pozici (i, j), kde i je barva segmentovaného obrazu a j je barva masky.
2. Dokud jsou v matici řádky i sloupce:
   (a) Najdeme prvek s maximální hodnotou.
   (b) Hodnotu přičteme k počtu odpovídajících si pixelů.
   (c) Odstraníme řádek i a sloupec j, kde (i, j) je pozice prvku.
3. Počet odpovídajících si pixelů vydělíme celkovým počtem pixelů v obrazu.

4.5 Implementace

Zde uvedeme několik základních informací o implementaci systému a knihovně použité pro implementaci neuronové sítě.

Celý systém je naprogramován v jazyce Java, což zaručuje jeho bezproblémovou přenositelnost.

Klíčovým prvkem celého systému je neuronová síť, která byla vytvořena pomocí knihovny Encog ([12]). Encog je otevřená knihovna pro neuronové sítě a umělou inteligenci, jejímž autorem je Jeff Heaton. Je dostupná pro platformy Java, .Net a Silverlight. Umožňuje vytvářet velké množství různých typů neuronových sítí a také poskytuje nástroje pro zpracování dat pro tyto sítě.

Objektový návrh systému odpovídá schématu na obrázku 4.1, uvedeného na začátku kapitoly. Program obsahuje čtyři hlavní moduly – generátor, extraktor, klasifikátor a komparátor:

Generátor - obsahuje třídy potřebné pro tvorbu obrazů obsahujících segmenty z jiných obrazů a jejich masky.

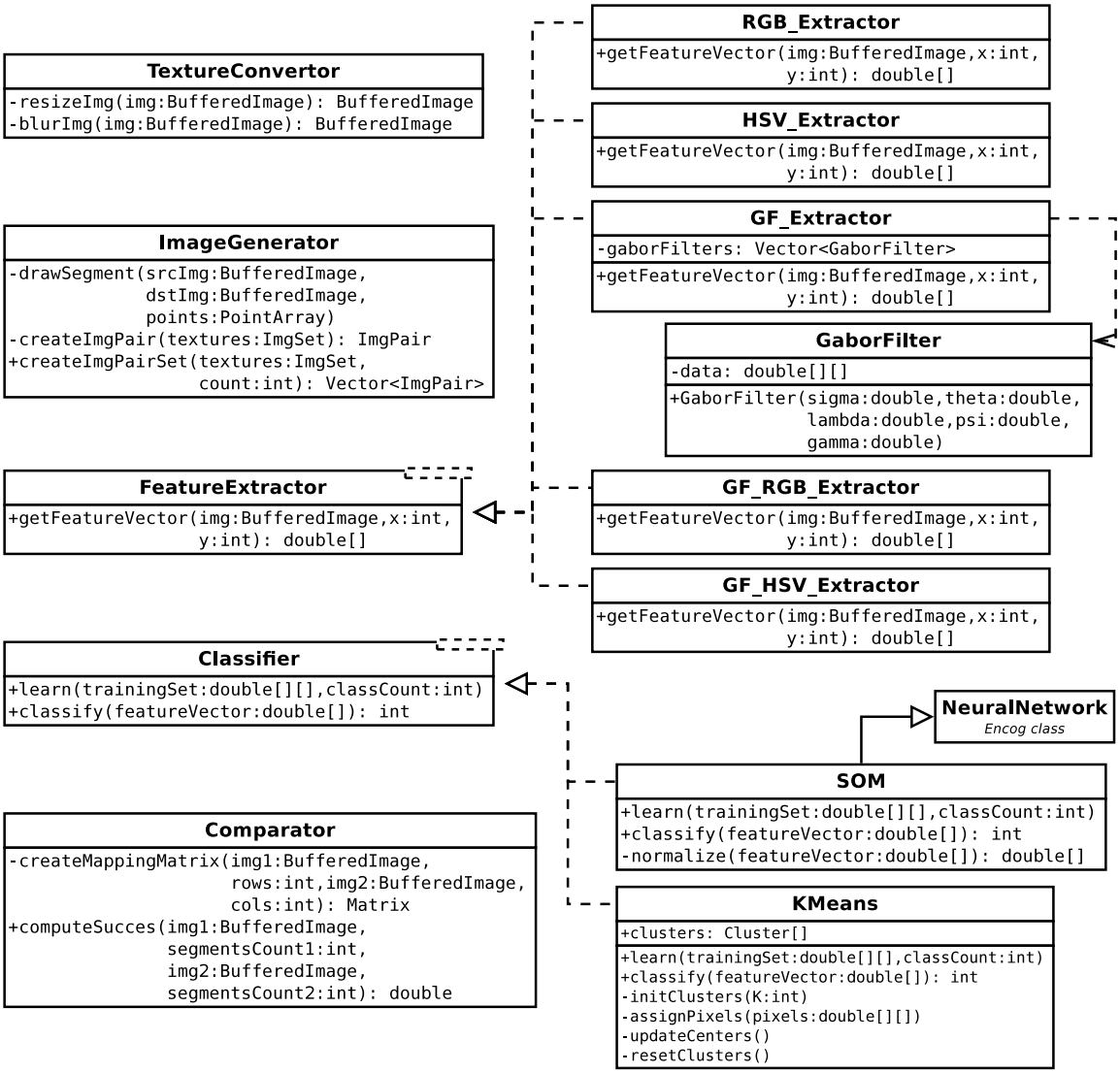
Extraktor - obsahuje rozhraní pro třídy reprezentující vytvořené typy extraktorů uvedené v předchozím textu a třídu pro Gáborův filtr.

Klasifikátor - obsahuje rozhraní pro třídy reprezentující různé typy klasifikátorů, v našem případě neuronovou síť a algoritmus K-means.

Komparátor - obsahuje třídu reprezentující matici korespondujících segmentů a třídu s metodami pro výpočet úspěšnosti.

Dále program obsahuje modul pro úpravu textur, pomocné třídy pro operace s obrazem a několik dalších pomocných tříd. Zmíníme z nich třídu reprezentující pixel, která obsahuje různé hodnoty, jimiž lze pixel reprezentovat (např. hodnoty barevných složek R, G a B, jejich uložení v poli, jejich reprezentace celočíselnou hodnotou, hodnota jasu, aj.). Důležitá je také třída reprezentující Gáborův filtr.

Návrh tříd je znázorněn v UML diagramu na obrázku 4.15. Diagram obsahuje pouze nejdůležitější třídy, jejich atributy a metody.



Obrázek 4.15: UML diagram nejdůležitějších tříd v programu.

`FeatureExtractor` (extraktor rysů) je rozhraní pro pět tříd, jejichž název odpovídá extrahovaným příznakům (GF je zde zkratka pro Gáborův filtr). Třída `GF_Extractor` využívá pro výpočet příznaků třídu `GaborFilter`. Třídy `GF_RGB_Extractor` a `GF_HSV_Extractor` vytvoří příznakový vektor spojením vektorů vytvořených třídami `GF_Extractor` a `RGB_Extractor`, resp. `HSV_Extractor`. Tato závislost není v diagramu znázorněna z prostorových důvodů a také kvůli přehlednosti. `Classifier` (klasifikátor) je rozhraním pro dvě třídy – `KMeans` a `SOM` (samoorganizační mapa). `SOM` navíc dědí z třídy `NeuralNetwork`, což je třída knihovny

Encog pro neuronovou síť. Mimo jiné poskytuje metody pro inicializaci, trénování sítě a určení vítězného neuronu.

Kapitola 5

Experimenty

Tato kapitola slouží hlavně k prezentaci výsledků dosažených segmentací pomocí neuronové sítě a porovnání úspěšnosti s algoritmem K-means. Byly provedeny experimenty i s jinými parametry segmentace, jejichž výsledky si ukážeme v jednotlivých podkapitolách. Nejdříve si uvedeme seznam aspektů segmentace, které byly porovnávány:

- Typ dat:
  - syntetická (obrazy vygenerované generátorem trénovacích dat)
  - reálná (fotografie)
- Klasifikátor:
  - neuronová síť (samoorganizační mapa)
  - algoritmus K-means
- Příznaky použité pro klasifikaci:
  - barevné RGB
  - barevné HSV
  - texturní
  - texturní + barevné RGB
  - texturní + barevné HSV
- Trénovací množina:
  - vytvořená ze všech vygenerovaných obrazů
  - vytvořená pouze z aktuálního obrazu určeného k segmentaci (klasifikátor je natrénován na každý obraz zvlášť)
- Počet cílových tříd:
  - roven počtu použitých textur
  - roven předem zvolené hodnotě (u reálných obrazů většinou nemáme k dispozici informaci o počtu rozlišitelných segmentů)

Výsledky jsou rozděleny do dvou hlavních kategorií – segmentace syntetických dat a segmentace reálných dat:

5.1 Segmentace syntetických dat

Na syntetických datech jsme otestovali, jak je neuronová síť schopna naučit se na konkrétní vzory.

Pro tento účel byl sestaven soubor 17 textur. Z těchto textur bylo vygenerováno 50 trénovacích a 20 testovacích obrazů o velikosti 128×128 pixelů obsahujících 3 až 5 segmentů odlišných textur. Do trénovací množiny bylo vybráno 100 náhodných pixelů z každého trénovacího obrazu v případě trénování na celé množině a 1000 pixelů v případě trénování na jednom obrazu. Počet výstupních tříd je roven počtu textur, v případě klasifikace na předem zvolený počet cílových tříd jsme tuto hodnotu zvolili jako průměrný počet segmentů v obrazu, tedy 4. S tímto nastavením byl každý experiment zopakován 6x, všechny hodnoty v následujících tabulkách jsou tedy průměrem z 6 hodnot.

Všechny tabulky obsahují hodnoty pro každý použitý příznakový vektor zvlášť, jelikož výsledná hodnota úspěšnosti segmentace se pohybuje v různých rozmezích v závislosti na použitých příznacích. Pro srovnání vlivu různých příznakových vektorů na segmentaci tedy nebudeme vytvářet zvláštní tabulku – rozdíly je možné vidět v tabulkách v každém z následujících experimentů.

Následující tabulky nezahrnují výsledky zcela všech experimentů, které byly provedeny. Kompletní výsledky, z nichž byly hodnoty čerpány, jsou pro úplnost uvedeny v příloze A.

5.1.1 Srovnání neuronové sítě s algoritmem K-means

Výsledky jsou rozděleny do dvou tabulek v závislosti na počtu cílových tříd klasifikace. V tabulce 5.1 jsou výsledky segmentace, kdy počet cílových tříd je roven počtu použitých textur, tedy 17, tabulka 5.2 zobrazuje výsledky klasifikace do 4 cílových tříd. Trénovací množina byla v obou případech vytvořena ze všech obrazů.

Použité příznaky	**Klasifikátor**	
	Neuronová síť	K-means
barevné RGB	75%	86%
barevné HSV	82%	90%
texturní	40%	60%
texturní + barevné RGB	70%	86%
texturní + barevné HSV	81%	91%

Tabulka 5.1: Výsledky segmentace pomocí neuronové sítě a algoritmu K-means s použitím různých příznaků. Počet cílových tříd klasifikace je roven počtu použitých textur.

Použité příznaky	**Klasifikátor**	
	Neuronová síť	K-means
barevné RGB	75%	76%
barevné HSV	74%	78%
texturní	65%	67%
texturní + barevné RGB	68%	78%
texturní + barevné HSV	73%	80%

Tabulka 5.2: Výsledky segmentace pomocí neuronové sítě a algoritmu K-means s použitím různých příznaků. Počet cílových tříd klasifikace = 4.

Z obou tabulek 5.1 a 5.2 vidíme, že algoritmus K-means byl ve všech případech úspěšnější než neuronová síť. Nicméně v případě klasifikace na menší počet tříd se úspěšnost sítě velmi blíží algoritmu K-means.

5.1.2 Srovnání způsobů vytvoření trénovací množiny

Výsledky jsou v tabulce 5.3. Pro získání těchto hodnot byla při segmentaci použita neuronová síť a v obou případech klasifikace do 4 cílových tříd.

Použité příznaky	**Trénovací množina**	
	všechny obrazy	pouze aktuální obraz
barevné RGB	75%	68%
barevné HSV	74%	72%
texturní	65%	52%
texturní + barevné RGB	68%	69%
texturní + barevné HSV	73%	70%

Tabulka 5.3: Výsledky segmentace s použitím různých příznaků a různých způsobů vytvoření trénovací množiny.

Z tabulky 5.3 vidíme, že úspěšnost segmentace při trénování na samotném obrázku byla u většiny příznaků menší než při trénování na všech obrazech. Tyto výsledky jsou poněkud překvapující – předpokládali jsme, že trénování na jednom obrazu bude přesnější, vzhledem k tomu, že trénovací množina v tomto případě neobsahuje žádné příznaky, které se v obrazu nenacházejí; síť se tedy učí pouze na těch datech, která budou skutečně použita k segmentaci, narozdíl od druhého způsobu trénování. Poznamenejme ještě jednou, že počet cílových tříd byl v obou případech stejný (4), právě z důvodu lepšího srovnání.

Pro vysvětlení těchto výsledků a návrh lepšího řešení bychom této tematice zřejmě museli věnovat ještě další rok nebo dva, než bychom tuto problematiku úplně pochopili. Bylo by potřeba provést více experimentů a podrobněji se zabývat možnostmi trénování samoorganizační mapy. Zřejmě by bylo potřeba tuto neuronovou síť vlastnoručně naimplementovat, abychom mohli experimentovat s různými technikami učení a klasifikace. V této práci jsme implementaci sítě ponechali na knihovně, což neposkytuje tak široké možnosti při řešení tohoto problému, jak bychom potřebovali.

5.1.3 Srovnání různých počtů cílových tříd

Výsledky jsou v tabulce 5.4. Pro získání těchto hodnot byla při segmentaci použita neuronová síť a trénovací množina byla vytvořena ze všech vygenerovaných obrazů.

Klasifikace do menšího počtu výstupních tříd má podle očekávání většinou horší výsledky, lepší jsou pouze v případě texturní segmentace. V našem případě 17 textur je rozděleno do 4 segmentů, tzn. průměrně 4 textury se namapují na 1 segment. Je tedy poměrně velká šance, že v jednom obrazu se bude nacházet více než 1 textura z těch, které se mapují na 1 segment. Jiné by to bylo v případě použití menšího počtu textur – čím méně textur, tím méně se jich namapuje na jeden segment a klasifikace by tak měla být přesnější. Tento experiment by mohl být předmětem dalšího rozšíření práce.

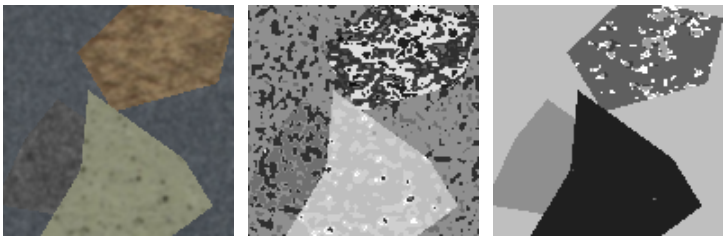
Použité příznaky	**Počet tříd**	
	17 (počet textur)	4
barevné RGB	75%	75%
barevné HSV	82%	74%
texturní	40%	65%
texturní + barevné RGB	70%	68%
texturní + barevné HSV	81%	73%

Tabulka 5.4: Výsledky segmentace s použitím různých příznaků a různých způsobů vytvoření trénovací množiny.

5.1.4 Srovnání použitých příznaků

Zde už není potřeba další tabulky – stačí se podívat na kteroukoli z výše uvedených tabulek (5.1 – 5.4), z nichž každá obsahuje hodnoty pro všechny použité příznaky. Ze všech těchto tabulek je vidět, že nejlépe dopadla segmentace s použitím pouze barevných příznaků a nejhůře s použitím pouze texturních příznaků.

Dále je vidět, že lepší výsledky dává použití barevného modelu HSV než RGB, a to v případě jak pouze barevné, tak i kombinované segmentace. Pro zajímavost si ukážeme výsledek segmentace s použitím RGB a s použitím HSV příznaků. Ukázka je na obrázku 5.1.



(a) Obraz (b) RGB segmentace (c) HSV segmentace

Obrázek 5.1: Ukázka RGB a HSV segmentace syntetických dat.

Na obrázku je vidět, že v případě HSV příznaků byly 3 ze 4 segmentů rozpoznány téměř bezchybně, narozdíl od RGB příznaků. I čtvrtý segment v pravém horním rohu byl v případě HSV klasifikován s větší přesností než v případě RGB.

5.2 Segmentace reálných dat

Kromě syntetických obrazů byla neuronová síť otestována i na reálných obrazech. V tomto případě nelze objektivně určit úspěšnost segmentace, protože nemáme k dispozici masku se vzorovými segmenty. Bylo by potřeba masku vytvořit ručně ke každému obrazu zvlášť, což by bylo velmi pracné a v praxi nepoužitelné. Proto si zde pouze ukážeme několik příkladů provedených segmentací.

Co se týče parametrů segmentace, trénování klasifikátoru probíhalo pouze na jednom obrazu, tzn. trénovací množina byla vytvořena z aktuálního obrazu určeného k segmentaci, z kterého bylo vybráno 1000 pixelů. Výsledek segmentace je silně závislý na zvoleném počtu výstupních tříd. Počet požadovaných tříd se však může velmi lišit v závislosti na

typu zpracovávaných dat, proto jsme tuto hodnotu odhadli na základě použitých obrazů na 4 třídy podle nejčastěji se vyskytujících objektů. Nejvhodnější pro segmentaci na základě textury jsou přírodní scenérie, které budeme nejčastěji používat – v těchto obrazech chceme obvykle rozlišit stromy, oblohu, trávu a popřípadě nějaké další objekty.

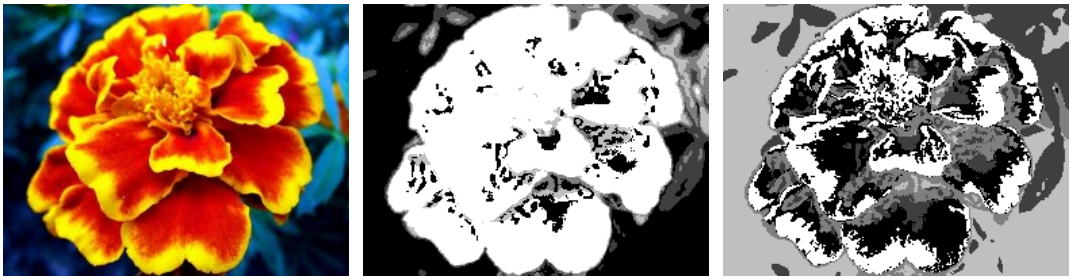
Barevná segmentace

Ukážeme si příklady segmentace na základě příznaků tvořených barevnými složkami RGB i HSV. Ukázky jsou na obrázcích 5.2 a 5.3. Vlevo je vždy původní obraz, uprostřed segmentace pomocí neuronové sítě a vpravo segmentace pomocí algoritmu K-means.

(a) Obraz (b) Neuronová síť (c) K-means

Obrázek 5.2: Ukázka barevné RGB segmentace reálných dat.

V ukázce na obrázku 5.2 vidíme, že segmentace s použitím neuronové sítě a algoritmu K-means dopadla velmi podobně. Obloha i cesta byly v obou případech rozeznány dokonale. Také byly poměrně úspěšně odlišeny tráva a květiny. Strom v obou případech víceméně splynul s trávou díky stejné barvě, ale neuronové síti se navíc podařilo rozeznat i kmen stromu od koruny a také stín stromu.



(a) Obraz (b) Neuronová síť (c) K-means

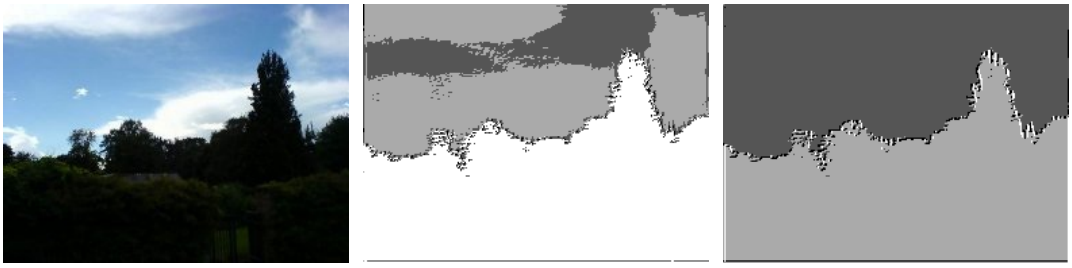
Obrázek 5.3: Ukázka barevné HSV segmentace reálných dat.

Na obrázku 5.2 je příklad ne moc úspěšné segmentace. Neuronová síť víceméně správně odlišila květinu od pozadí, až na několik chybných částí, které jsou způsobeny stínem okvětních lístků. Na pozadí však lze rozeznat až čtyři segmenty, což určitě není ideální. Naproti tomu K-means rozlišil na pozadí celkem vhodně dva segmenty, ale v květině jich našel až pět, což také moc neodpovídá skutečnosti. Ideální by bylo rozlišit dva segmenty v květině a dva na pozadí – to bychom ale museli klasifikátoru zadat přesný počet požadovaných segmentů ke každému obrazu zvlášť. To je nevýhoda obou metod (neuronové sítě i algoritmu K-means) – výsledná segmentace je silně závislá na zvoleném počtu požadovaných

segmentů, který musíme zadat dopředu.

Texturní segmentace

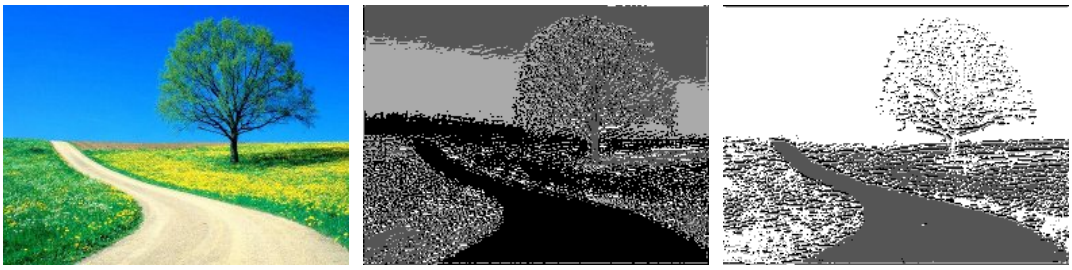
Pro demonstraci segmentace na základě textury jsou ideální přírodní scenérie – lze v nich najít plochy s výrazným texturním charakterem. Ukázky jsou na obrázcích 5.4 a 5.5.



(a) Obraz (b) Neuronová síť (c) K-means

Obrázek 5.4: Ukázka texturní segmentace reálných dat.

Na obrázku 5.4 je ukázka segmentace, kdy byla neuronová síť úspěšnější než algoritmus K-means. Stromy správně rozlišily obě metody, ale neuronová síť navíc rozeznala mraky na obloze. Hranice segmentů s mraky nejsou sice ideální, ale to je způsobeno hlavně velmi nejasnou hranicí mezi mraky a oblohou v samotném původním obrázku. Odlišné segmenty na hranici stýkajících se ploch mezi stromy a oblohou a na dolním okraji obrázku jsou způsobeny charakterem Gáborových filtrů, které počítají výslednou hodnotu na základě určitého okolí pixelu. Na přechodu dvou výrazně se lišících ploch zasahují do okolí jednotlivých pixelů pixely s výrazně odlišnou texturní charakteristikou, proto i výsledná odezva Gáborových filtrů je jiná než ve většině obrazu – právě díky této vlastnosti jsou Gáborovy filtry využívány také pro detekci hran. Na okraji obrazu nastává stejný efekt z podobných důvodů – pokud totiž okolí počítaného pixelu přesahuje mimo obraz, chybějící hodnoty pixelů jsou nahrazeny nulami, což má stejný efekt, jako by tam byla plocha s odlišnou texturou.



(a) Obraz (b) Neuronová síť (c) K-means

Obrázek 5.5: Další ukázka segmentace na základě textury.

Obrázek 5.5 ukazuje nepříliš úspěšnou texturní segmentaci už známého obrázku stromu. Cesta byla rozeznána správně oběmi metodami, ale s ostatními objekty byl problém. K-means ještě správně rozeznal oblohu, nicméně trávě přiřadil stejný segment. Neuronová síť měla problém i s barevným přechodem oblohy. V obou obrázcích je určitý náznak segmentu

odpovídajícímu žlutému květinovému pásu, ale strom se už nepodařilo rozeznat, splývají v něm segmenty z celého obrázku.

Kombinovaná segmentace

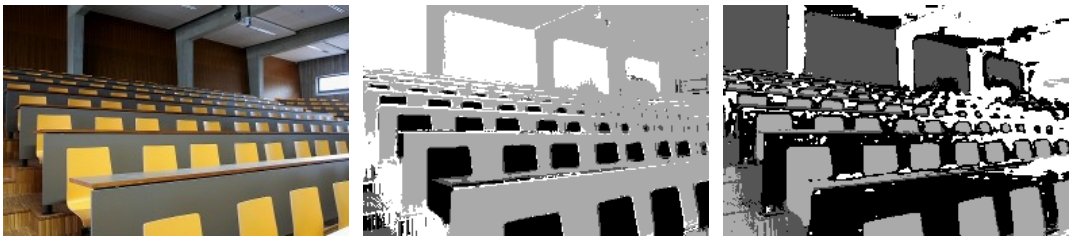
Na obrázku 5.6 je ukázka poměrně úspěšné segmentace přírodní scenérie na základě příznaků složených z texturních a barevných HSV příznaků.

(a) Obraz (b) Neuronová síť (c) K-means

Obrázek 5.6: Ukázka kombinované texturní a barevné HSV segmentace.

Neuronové síti i algoritmu K-means se zde podařilo odlišit strom od pozadí. Neuronová síť lépe rozpoznala plochu s oblohou, avšak na úkor hůře segmentovaného stromu, v němž našla tři různé segmenty. Dva z nich véceméně odpovídají kmenu a koruně, ale třetí, bílý segment je tam očividně navíc – je to pravděpodobně způsobeno jemnějším texturním vzorem tvořeným tenkými koncemi větví, který už síť označila jako odlišnou texturu. Naproti tomu K-means měl větší problém s jasnější oblastí oblohy v pravém dolním rohu, zato strom klasifikoval přesněji pouze na dva segmenty odpovídající koruně a kmenu. Travnatá plocha v dolní části obrázku byla v obou případech klasifikována stejně jako strom pravděpodobně díky tomu, že na daném počtu cílových tříd nebylo možné rozlišit tak jemné rozdíly v barvě a textuře.

Na obrázku 5.7 si pro změnu ukážeme jednu ukázku segmentace obrazu jiného charakteru – jedná se o přednáškovou místnost.



(a) Obraz (b) Neuronová síť (c) K-means

Obrázek 5.7: Ukázka kombinované texturní a barevné HSV segmentace.

Neuronová síť byla v tomto případě jednoznačně úspěšnější. Zcela správně rozeznala židle a lavice, které sice splývají se stropem, ale kvůli velmi podobné barvě i malým rozdílům v textuře je těžké tyto objekty rozlišit. Velmi dobře se síť vypořádala s výrazným přechodem jasu na hnědé zadní stěně, které přiřadila jeden segment až na malou osvětlenou plochu vpravo, kde už jasové hodnoty překročily určitý limit. Menší problémy měla pouze s oknem a schody, což měl i algoritmus K-means. Ten měl však větší problémy i s lavicemi, kde se

poněkud nelogicky střídají černé a bílé segmenty. Vpravo splývá bílý segment na lavicích se stropem stejně jako u neuronové sítě, ale na stropě se objevují nežádoucí černé artefakty, stejně jako na zadní stěně.

5.3 Shrnutí

Při experimentech se segmentací syntetických obrazových dat jsme zjistili, že algoritmus K-means je při použití popsaných příznaků vhodnější než neuronová síť SOM. Příčin neúspěchu sítě může být více:

- SOM redukuje dimenzionalitu vstupního prostoru. Je možné, že při použití příznakového vektoru délky 100 a výše už by se výhody SOM oproti klasickému shlukování projevily.
- Je možné, že SOM potřebuje pro dobré nastavení mřížky větší objem dat než K-means na výpočet centroidů.

Těmito experimenty se lze zabývat v dalším pokračování práce. Tímto směrem lze zaměřit další výzkum v této oblasti.

Při testování vlivu způsobu vytvoření trénovací množiny na výsledek segmentace jsme zjistili poněkud překvapující výsledky, které nečekaně hovoří v neprospěch trénování na jediném obrazu. Zde by bylo potřeba provést více experimentů a podrobněji se zabývat touto problematikou, abychom mohli navrhnout lepší řešení nebo alespoň vysvětlit dosažené výsledky. Nicméně cílem bylo srovnání vlivu tohoto aspektu na segmentaci a to bylo splněno.

Naproti tomu v testu srovnávajícím úspěšnost segmentace z hlediska počtu cílových tříd výsledky odpovídají očekávání. Klasifikace do menšího počtu tříd by se mohla zdát na první pohled výhodná z toho pohledu, že neuronová síť se nemusí učit "tolik" vzorů a stačí, když každý obraz rozdělí na daný počet segmentů. Což je také případ segmentace reálných dat, kde požadovaný počet rozlišitelných tříd neznáme dopředu. Ovšem v provedeném testu na syntetických datech, kdy počet tříd známe předem, je v tomto případě segmentace logicky úspěšnější. V případě klasifikace do omezeného počtu tříd je síť nucena namapovat více textur do jedné třídy a když se pak několik z těchto textur objeví v jednom obrazu, síť není schopna je od sebe odlišit.

Při porovnání výsledků z hlediska použití různých příznaků jsme zjistili, že největší úspěšnost má segmentace na základě pouze barevných příznaků. Tento výsledek také zcela neodpovídá očekávání – teoreticky by měla být nejúspěšnější segmentace s použitím barevných i texturních příznaků zároveň, vzhledem k tomu, že tato kombinace poskytuje síti nejvíce informací o obrazu. Problém celkově menší úspěšnosti segmentace s použitím texturních příznaků je zřejmě ve výběru těchto texturních příznaků. Na experimentech s reálnými daty jsme zjistili, že Gáborovy filtry jsou pravděpodobně vhodnější na detekci hran než na segmentaci založenou na oblastech. Úspěšnost tohoto typu segmentace by tedy pravděpodobně bylo možné zvýšit volbou vhodnějších texturních příznaků (např. Haralick [9]), kterými se zde však už nebudeme zabývat.

V segmentaci reálných dat byla neuronová síť poměrně úspěšná. Ohledně těchto experimentů nemáme k dispozici žádné statistiky, protože k reálným obrazům nemáme k dispozici masky, podle nichž by bylo možné vypočítat úspěšnost segmentace. Posouzení úspěšnosti je tedy ponecháno vizuálnímu porovnání výsledných segmentovaných obrazů s původními. Na základě tohoto srovnání jsou výsledky neuronové sítě srovnatelné s výsledky algoritmu

K-means. V některých případech jsou horší, ale v mnoha případech naopak lepší. Ovšem k získání objektivního hodnocení by bylo nutné zavést nějaká statistická měřítka.

Kapitola 6

Závěr

Cílem této práce bylo prozkoumat a zhodnotit možnosti a omezení segmentační metody založené na neuronové síti SOM (samoorganizační mapa) a srovnání této metody s tradičním segmentačním algoritmem. K tomuto účelu byl vytvořen segmentační systém, který je popsán v kapitole 4. Pomocí tohoto systému byly provedeny experimenty se segmentací obrazových dat, jejichž výsledky jsou prezentovány v kapitole 5. Tato kapitola shrnuje dosažené výsledky a navrhuje další možný směr výzkumu v této oblasti.

Výsledky experimentů ukázaly, že metoda je v praxi použitelná, nicméně větší úspěšnost shlukovacího algoritmu K-means při segmentaci syntetických dat ukazuje, že metodu je nutné dále rozšířit. Další výzkum by se mohl týkat použití většího příznakového vektoru pro klasifikaci nebo většího objemu dat při trénování.

Dále byly provedeny experimenty ohledně trénování sítě a způsobu vytvoření trénovací množiny, z nichž překvapivě vyplývá, že trénování na celé množině obrazů dává lepší výsledky než trénování pouze na aktuálním obrazu. Tento výsledek by bylo vhodné dále zanalyzovat a zjistit příčinu tohoto chování metody, popřípadě navrhnout lepší řešení.

Test segmentace z hlediska počtu cílových tříd dopadl podle očekávání. Segmentace s větším počtem cílových tříd byla úspěšnější než s omezeným počtem tříd. Proto se při segmentaci reálných dat musíme často spokojit s horšími výsledky právě z toho důvodu, že neznáme dopředu počet objektů, které chceme v obrazu rozlišit. To je však problém i u tradičních segmentačních metod.

Při porovnání výsledků z hlediska použití různých příznaků byla zjištěna menší úspěšnost segmentace v případě použití texturních příznaků. Problém je zřejmě ve výběru texturních příznaků a volbou vhodnějších příznaků by mohlo být dosaženo větší úspěšnosti. Hledání vhodnějších příznaků by mohlo být předmětem dalšího pokračování projektu.

Poslední částí experimentů byla segmentace reálných dat, která dopadla velice uspokojivě. Výsledky neuronové sítě byly srovnatelné s výsledky shlukovacího algoritmu K-means. K těmto výsledkům bohužel nejsou k dispozici žádné statistiky; posouzení úspěšnosti je zde ponecháno na vizuálním srovnání původních a výsledných obrazů.

Segmentace obrazových dat je poměrně složitý problém, pro který je obtížné najít obecný algoritmus. Metoda navrhnutá v této práci zahrnuje mnoho dílčích operací, z nichž každá měla spoustu alternativ (jiná normalizační metoda, jiné příznaky atd.). Bohužel je nad rámec této práce vyzkoušet všechny možnosti za účelem nalezení nejideálnější metody. Hledání lepších alternativ jednotlivých operací může být předmětem dalšího výzkumu v této oblasti.

Literatura

[1] Acharya, T.; Ray, A. K.: *Image Processing: Principles and Applications*. Wiley and Sons, říjen 2005, ISBN 978-0-471-71998-4.

[2] Anderson, J. A.: *An Introduction to Neural Networks*. The MIT Press, 1995, ISBN 0-262-51081-2.

[3] Bishop, C. M.: *Neural networks for pattern recognition*. Oxford University Press, 1996, ISBN 0198538642.

[4] Campbell, N. W.; Thomas, B. T.; Troscianko, T.: Segmentation of Natural Images using Self-Organising Feature Maps. In *British Machine Vision Conference*, British Machine Vision Association, září 1996, ISBN 0 9521898 3 6, s. 223–232.
URL http://www.cs.bris.ac.uk/Publications/Papers/1000140.pdf

[5] Christos Stergiou; Dimitrios Siganos: Neural Networks. Technická zpráva, Department of Computing, Imperial College London, 1996.
URL
http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html

[6] Grigorescu, S.; Petkov, N.; Kruizinga, P.: Comparison of texture features based on Gabor filters. *IEEE Trans. on Image Processing*, ročník 11, č. 10, 2002: s. 1160–1167.

[7] Grimson, W. E. L.; Mundy, J. L.: Computer vision applications. *Commun. ACM*, ročník 37, č. 3, 1994: s. 44–51, ISSN 0001-0782.

[8] Gurney, K. N.: *An introduction to neural networks*. CRC Press, 1997, ISBN 1-85728-503-4.

[9] Haralick, R. M.: Statistical and structural approaches to texture. *Proc. IEEE*, ročník 67, č. 5, 1979: s. 786–804.

[10] Heaton, J.: *Introduction to Neural Networks for Java*. Heaton Research, Inc, 2008, ISBN 1-60439-008-5.

[11] Jaynes, C. O.: Computer vision and artificial intelligence. *Crossroads*, ročník 3, č. 1, 1996: s. 7–10, ISSN 1528-4972.

[12] Jeff Heaton: Encog Java and DotNet Neural Network Framework. http://www.heatonresearch.com/encog, květen 2010.

[13] Kolektiv autorů: Wikipedia. http://en.wikipedia.org, prosinec 2009.

[14] MacKay, D. J. C.: *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, 2003, ISBN 0-521-64298-1, 285, 286 s.

[15] Masters, T.: *Practical neural network recipes in C++.* San Diego Academic Press, 1995, ISBN 0124790402.

[16] Mehrotra, K.; Mohan, C. K.; Ranka, S.: *Elements of artificial neural networks.* The MIT Press, 1996, ISBN 0-262-13328-8.

[17] Michal Španěl; Vítězslav Beran: Obrazové segmentační techniky. Technická zpráva, Ústav počítačové grafiky a multimédií, Fakulta informačních technologií, Vysoké učení technické v Brně, 2005.
URL http://www.fit.vutbr.cz/~spanel/segmentace/

[18] N. Petkov; M.B. Wieling: Gabor filter for image processing and computer vision. http://matlabserver.cs.rug.nl/edgedetectionweb/web/edgedetection_params.html, 2008.

[19] Sonka, M.; Hlaváč, V.; Boyle, R.: *Image processing, Analysis, and Machine Vision ISE (International Student Edition).* Nelson Engineering, 2007-04-18, ISBN 0-495-24438-4.

[20] Stomski, P. J.; Elmaghraby, A. S.: Selection of a neural network system for visual inspection. *SIGSIM Simul. Dig.*, ročník 20, č. 2, 1989: s. 8–21, ISSN 0163-6103.

[21] Zbořil, F.: Soft computing, prosinec 2009, přednášky k předmětu.

[22] Zemčík, P.; Španěl, M.: Zpracování obrazu, květen 2010, studijní opora.

Seznam použitých zkratek

RGB – Red (červená), Green (zelená), Blue (modrá)

HSV – Hue (odstín), Saturation (sytost), Value (hodnota)

HSB – Hue (odstín), Saturation (sytost), Brightness (jas)

SOM – Self-organizing map (samoorganizační mapa)

CAD – Computer-aided design (počítačem podporované projektování)

UML – Unified Modeling Language (unifikovaný modelovací jazyk)

Dodatek A

Výsledky segmentace syntetických dat

Zde jsou uvedeny kompletní výsledky všech provedených experimentů se segmentací syntetických dat. Protože nebylo možné přehledně zahrnout všechny parametry do jedné tabulky, jsou výsledky rozděleny do 5 tabulek podle použitých příznaků:

A.1 Barevné RGB příznaky

Výsledky segmentace na základě příznaků tvořených pouze barevnými složkami RGB jsou v tabulce A.1.

Trénovací množina	**Počet tříd**	**Klasifikátor**	
		neuronová síť	K-means
všechny obrazy	počet použitých textur	75%	86%
	předem zvolený	75%	76%
pouze aktuální obraz	předem zvolený	68%	77%

Tabulka A.1: Výsledky barevné RGB segmentace syntetických dat.

A.2 Barevné HSV příznaky

Výsledky segmentace na základě příznaků tvořených pouze barevnými složkami HSV jsou v tabulce A.2.

Trénovací množina	**Počet tříd**	**Klasifikátor**	
		neuronová síť	K-means
všechny obrazy	počet použitých textur	82%	90%
	předem zvolený	74%	78%
pouze aktuální obraz	předem zvolený	72%	77%

Tabulka A.2: Výsledky barevné HSV segmentace syntetických dat.

A.3 Texturní příznaky

Výsledky segmentace na základě texturních příznaků jsou v tabulce A.3.

Trénovací množina	**Počet tříd**	**Klasifikátor**	
		neuronová síť	K-means
všechny obrazy	počet použitých textur	40%	60%
	předem zvolený	65%	67%
pouze aktuální obraz	předem zvolený	52%	65%

Tabulka A.3: Výsledky texturní segmentace syntetických dat.

A.4 Texturní + barevné RGB příznaky

Výsledky segmentace na základě příznaků složených z texturních a barevných RGB příznaků jsou v tabulce A.4.

Trénovací množina	**Počet tříd**	**Klasifikátor**	
		neuronová síť	K-means
všechny obrazy	počet použitých textur	70%	86%
	předem zvolený	68%	78%
pouze aktuální obraz	předem zvolený	69%	78%

Tabulka A.4: Výsledky kombinované texturní a barevné RGB segmentace syntetických dat.

A.5 Texturní + barevné HSV příznaky

Výsledky segmentace na základě příznaků složených z texturních a barevných HSV příznaků jsou v tabulce A.5.

Trénovací množina	**Počet tříd**	**Klasifikátor**	
		neuronová síť	K-means
všechny obrazy	počet použitých textur	81%	91%
	předem zvolený	73%	80%
pouze aktuální obraz	předem zvolený	70%	82%

Tabulka A.5: Výsledky kombinované texturní a barevné HSV segmentace syntetických dat.