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

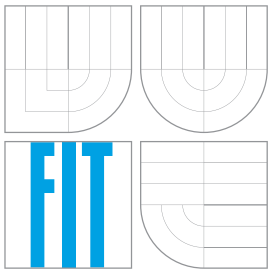
ROZPOZNÁVÁNÍ TEXTU V OBRAZE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JAN BÍLEK
AUTHOR

BRNO 2008

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

ROZPOZNÁVÁNÍ TEXTU V OBRAZE

OPTICAL CHARACTER RECOGNITION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAN BÍLEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ ZUZAŇÁK
SUPERVISOR

BRNO 2008

Abstrakt

Tato práce se zabývá problematikou rozpoznávání textu v obraze a těmi metodami zpracování obrazu, které rozpoznání textu napomáhají. Zvláště se zaměřuje na problematiku obrazového spamu a jeho úpravu pro rozpoznání. Popisuje metody zpracování obrazu, které vedou k takové úpravě obrazu, aby byl v něm obsažený text snadněji rozpoznatelný, a některé metody využívané pro rozpoznávání znaků. V práci je také popsán návrh a implementace systému, který zpracovává obrazový spam a detekuje v něm obsažený text.

Klíčová slova

Rozpoznávání textu v obraze, adaptivní prahování, redukce šumu, histogram, šablona, detekce průsečíků, detekce spamu, spam filtr.

Abstract

This work describes problems of optical character recognition and methods which are used to improve its results. It specializes in problems of image spam and its adjustment for its recognition. This work describes methods of processing an image which leads to better results of text recognition and also some methods used to the character recognition. There is also described a scheme and an implementation of a system which processes an image spam and detects a text within it.

Keywords

Optical character recognition, reduction, image thinning, histogram, template, crossings detection, spam detection, spam filter.

Citace

Jan Bílek: Rozpoznávání textu v obraze, bakalářská práce, Brno, FIT VUT v Brně, 2008

Rozpoznávání textu v obraze

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana inženýra Jiřího Zuzaňáka

.....
Jan Bílek
12. května 2008

Poděkování

Děkuji panu inženýrovi Jiřímu Zuzaňákovi za pomoc a cenné rady při vytváření této práce

© Jan Bílek, 2008.
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

Kapitola 1

Úvod

Rozpoznávání textu v obraze se v odvětví počítačové grafiky může vyskytovat minimálně ve dvou typech případů.

Prvním z těchto typů je ten obecně známější, kdy máme nějaký tištěný, na stroji psaný či ručně psaný text, který jsme převedli do digitální podoby, a z bitové mapy, ve které je zaznamenán, potřebujeme získat informace o znázorněných znacích. Druhým případem je potom ten, kdy text, který získáme ve formě bitové mapy, nemá svůj fyzický vzor, ale byl vytvořen přímo v digitální podobě.

Zde vyvstává otázka, z jakého důvodu by někdo ukládal text v obrazové formě, když má možnost jej zaznamenat přímo textově. Jeden z takových důvodů můžeme nalézt v současných pokusech odesílatelů nevyžádané elektronické pošty. V mnoha případech je totiž možné se setkat s tím, že text, který by e-mail usvědčil jako nevyžádaný, autor záměrně umístí do přílohy právě ve formě obrázku a tím zajistí, že e-mail daným filtrováním spíše projde bez úhony.

Z toho důvodu se jako aktuální jeví vytvoření takového programu, který by byl schopen z obrázku, který má v případě spamu (nevyžádané pošty) velmi charakteristické vlastnosti, text detekovat.

V první kapitole této práce je možno nalézt uvedení do problematiky a zběžný popis struktury tohoto dokumentu. Druhá kapitola se zabývá problematikou spamu, jeho historií a specifickými vlastnostmi obrazů k nevyžádané poště přiložených. Také jsou zde zmíněny některé zajímavé statistiky týkající se obrazového spamu. Ve třetí kapitole jsou teoreticky rozebrány metody zpracování obrazu, které mají souvislost s činnostmi, jež je třeba provést pro úpravu či segmentaci obrazu před samotným rozpoznáváním. Čtvrtá kapitola rozebírá některé metody využívané pro rozpoznávání znaků. Stručně jsou zde popsány principy a cíle genetických algoritmů a neuronových sítí. Obsahem páté kapitoly je samotný návrh systému. Zde jsou zmíněny metody, které byly při návrhu použity, a rozebrány konkrétní specifikace těchto metod, jež byly uzpůsobeny pro co možná nejefektivnější detekci obrazového spamu. V šesté kapitole se nalézají fakta o implementaci. Je popsán nástroj použitý pro tvorbu implementace a návrh rozhraní programu. Sedmou kapitolou je závěr, v němž je možno nalézt vyhodnocení výsledků zracování spamu navrženým systémem i bakalářské práce samotné.

Kapitola 2

Spam

Označení spam se používá pro nevyžádané sdělení masově šířené internetem, které je nejčastěji reklamního charakteru. Evropská unie ho ve svém zákoně č. 480/2004 definuje jako nevyžádané elektronickými prostředky šířené „obchodní sdělení, což jsou všechny formy sdělení určeného k přímé či nepřímé podpoře zboží či služeb nebo image podniku fyzické či právnické osoby.“ Nejčastěji se jedná o nevyžádané e-maily, ale v současnosti se spam vyskytuje i v diskusních fórech a obecně kdekoli na internetu, kde je možnost přidávat příspěvky a není dostatečné zabezpečení.

Samotné označení *spam* má původ v jednom díle britského seriálu Monty Pythonův létající cirkus, který komediálně ztvárnil situaci ve stravování za druhé světové války a krátce po ní, kdy Spam byla značka velmi rozšířené a nepopulární haše.

2.1 Historie spamu

Internet byl ve svých začátcích využíván především vojenskými a vládními organizacemi, proto se zde nevyžádaná reklamní sdělení až do jeho masovějšího rozšíření téměř nevyskytovala. Přesto však existovalo několik výjimek, které je možné označit za úplný počátek spamu.

Historicky první zaznamenaný případ se odehrál 1. května 1978, kdy byla prezentace produktů společnosti Digital Equipment Corporation rozeslána na všechny e-mailové adresy sítě ARPANET (Advanced Research Projects Agency Network – předchůdce internetu). Další případ se objevil na USENETu, kdy student Dave Rhodes (nejspíše se však jednalo o falešné jméno, protože univerzita, na které měl studovat, o něm nemá žádný záznam) masově rozeslal pozvánku do pyramidové hry, předmět zněl: *„MAKE.MONEY.FAST!!“*.

Poprvé však bylo označení *spam* použito až v roce 1993, kdy se Richard Depew rozhodl představit svůj pohled na fungování USENETu a koncept retro-moderace. Jeho program pro mazání příspěvků však obsahoval chybu a zahltil skupinu více než 200 příspěvky. To mnoho uživatelů popudilo a poprvé v historii nazvali tyto zprávy spamem. Z prvních případů ještě za zmínku stojí hromadně rozeslaná zpráva (opět na USENETu) od Clarence L. Thomase IV. Předmět zněl ”Global Alert For All: Jesus Is Coming Soon” a jejím obsahem bylo upozornění na údajně blížící se konec světa v souvislosti s tehdejšími katastrofami.

Nevyžádané zprávy v podobě, jak je známe, byly poprvé rozeslány 5. 3. 1994. Pánové Cantor a Siegel tehdy rozeslali pozvánku na „Green Card Lottery“. Jejich zpráva zahltila 6 000 diskusních skupin.

Podrobnější informace o původu a historii spamu je možné nalézt v [19] nebo [8].

Od té doby se spam rozvinul do podoby, kdy začíná být problémem nejen z hlediska obtěžování adresátů nevyžádanými nabídkami, ale i z hlediska zátěže sítě, kterou díky svému extrémně vysokému počtu způsobuje.

2.2 Filtrování spamu

Z obou výše zmíněných důvodů je vhodné takové zprávy filtrovat a omezovat. V případě veřejně přístupných formulářů je situace relativně jednoduchá, stačí zabránit jejich automatickému vyplňování a množství spamu bude do značné míry omezeno. Při vyplňování formulářů se tak můžeme setkat s nutností opsat kód z obrázku, který je upraven takovým způsobem, aby bylo obtížné text strojově rozpoznat a vyplňovat a odesílat formuláře automaticky,

V případě elektronické pošty však tento princip není možné použít. Proto je využíván způsob, kdy je obsah zpráv vyhodnocován a podle výsledku je snaha co nejpřesněji rozhodnout, jestli se jedná o nevyžádané sdělení, nebo standardní zprávu.

Zprávy obsahující text, který je pro spam typický, jsou v současnosti do značné míry úspěšně filtrovány. Tabulka 2.1 ukazuje situaci, která byla v roce 2006, podle současných průzkumů však podíl e-mailů označených jako spam od té doby stoupl na 78,5% (dle [12]), tedy téměř na dvojnásobek. Za pozornost z dat uvedených v předchozí tabulce stojí určitě také to, že 28% lidí na nevyžádané zprávy reaguje a necelá třetina z nich dokonce objednává inzerované zboží.

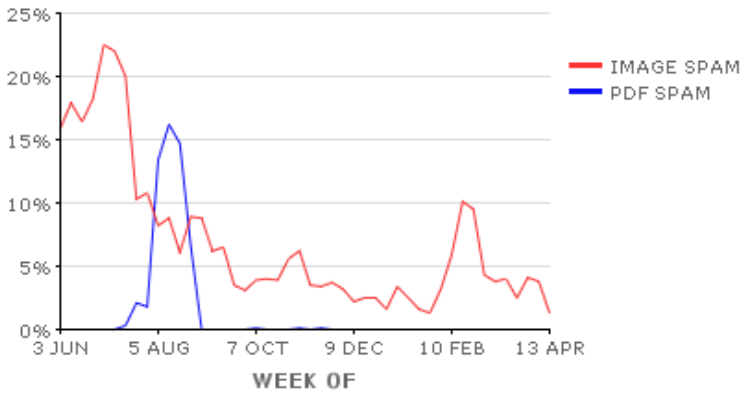
E-mail označený za spam	40%
Počet denně odeslaných e-mailů	12,4 miliard
Uživatelé odpovídající na spam	28%
Uživatelé objednávající zboží inzerované pomocí spamu	8%

Tabulka 2.1: Statistika spamu pro rok 2006 (zdroj: [2])

Současné filtry větší část nevyžádaných zpráv úspěšně rozpoznají a patřičně s ní naloží, proto se autoři spamů snaží tyto filtry obejít. Jednou z cest, jak toho dosáhnout, je ta, kdy text, který by daný e-mail usvědčil, není umístěn v těle zprávy, ale v příloze ve formě obrázku.

datum	typ spamu	velikost	datum	typ spamu	velikost
1. 11. 2007	Léky	9,5 Kb	30. 11. 2007	Léky	14,6 Kb
2. 11. 2007	Léky	9,7 Kb	1. 12. 2007	Léky	25,9 Kb
5. 11. 2007	Léky	11,0 Kb	12. 12. 2007	Léky	11,0 Kb
7. 11. 2007	Léky	9,9 Kb	19. 12. 2007	Léky	26,3 Kb
7. 11. 2007	Léky	14,1 Kb	20. 12. 2007	Léky	7,8 Kb
9. 11. 2007	Léky	10,7 Kb	29. 12. 2007	Léky	8,7 Kb
17. 11. 2007	Léky	13,9 Kb	31. 12. 2007	Léky	8,2 Kb
21. 11. 2007	Léky	14,5 Kb	4. 2. 2008	Léky	78,9 Kb
29. 11. 2007	Léky	11,9 Kb	15. 3. 2008	Léky	11,2 Kb

Tabulka 2.2: Zaznamenané příchozí obrazové spamy



Obrázek 2.1: Nevyžádané zprávy obsahující přílohy

2.3 Obrazový spam

Dle [10] se jeho množství pohybuje kolem hodnoty 7% z celkového množství spamu (podle nejnovější statistiky hodnota klesla dokonce na 1,3%, ale ještě nedávno se pohybovala kolem dvaceti procent – více viz graf na obrázku 2.1). To by se mohlo zdát jako relativně zanedbatelné množství, ale pokud vezmeme na zřetel že

1. míra úspěšnosti spamfiltrů se pohybuje kolem 99,5% a
2. průměrná velikost nevyžádané zprávy je 2,5 Kb, oproti tomu velikost obrazového spamu se pohybuje v rozmezí od 7Kb po 80Kb (viz tabulka 2.2),

je již vidět, že se jedná o závažný problém. Z prvního bodu totiž vyplývá, že i přes malé procentuální zastoupení by se počet nevyžádaných zpráv ve schránkách uživatelů znatelně zvýšil. Z druhého bodu potom můžeme usuzovat, že obrazový spam zatíží síť mnohem více, než by se zdálo, pokud bychom vzali v potaz pouze jeho zastoupení co do počtu kusů.

Řešením, které umožní identifikovat i spam skrývající text v obrázcích, je využití automatického rozpoznávání textu v obraze. Metodám spojeným s úpravou obrazu a rozpoznáváním textu (OCR – Optical Character Recognition) jsou věnovány následující kapitoly.

Kapitola 3

Zpracování obrazu

Existuje velké množství rozmanitých metod, v této práci však budou zmíněny pouze ty, které se uplatňují ve značné míře, byly použity při implementaci nebo jsou považovány za důležité či za zajímavé..

3.1 Prahování

Prahování je jednou z metod segmentace obrazu. Segmentace obrazu je seskupování pixelů do regionů tak, že:

- Každý region splňuje podmínku, že všechny jemu náležící pixely mají nějakou společnou vlastnost.
- Pixely ze sousedních regionů tuto společnou vlastnost nemají.
- Složením všech regionů vzniká kompletní segmentovaný obraz.
- Jednotlivé regiony jsou po dvojicích disjunktní.

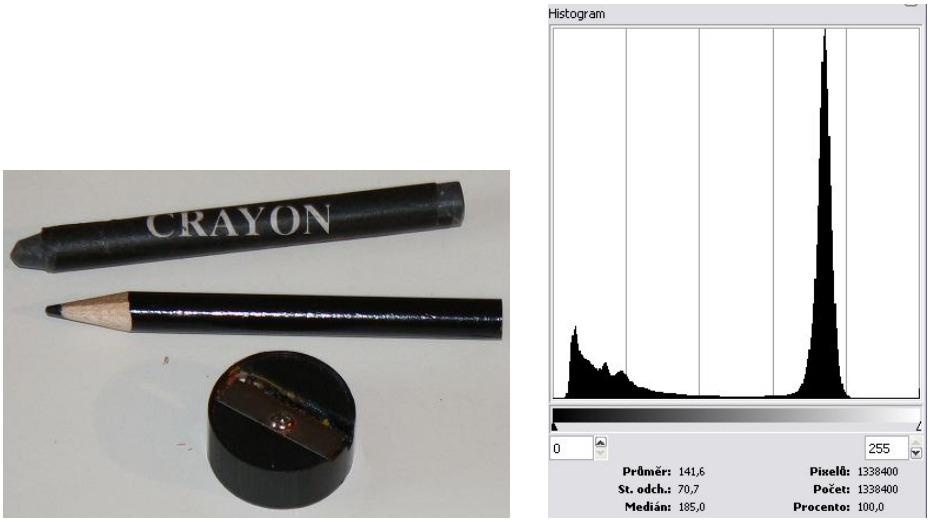
Cílem segmentace je zjednodušit nebo změnit reprezentaci obrazu tak, aby se lépe analyzoval.

Klasickým případem je situace, kdy je barevný obraz obsahující určité objekty převeden na binární (pouze dvě barvy, obvykle černá a bílá), kdy jedna barva reprezentuje objekty a druhá pozadí. Podmínou pro určení, do které ze skupin bude bod náležet, je intenzita jasu daného bodu. Takto upravený obraz je pro mnoho metod rozpoznávání obrazu výrazně přijatelnější variantou, než jeho barevná verze, především díky tomu, že jasně odděluje objekty popředí a pozadí. Ačkoliv je totiž pro člověka snadné odlišit na fotografii zachycené objekty od pozadí, na kterém se nacházejí, pro počítač to může být překvapivě velký problém.

Aby však byly objekty od pozadí správně odděleny, musí být vhodně stanovena podmínka, která rozdělení do těchto skupin určuje. Je nazývána práh a určuje mez hodnot intenzity jasu. Body mající intenzitu rovnou nebo vyšší než práh, jsou převedeny na barvu pozadí, body s nižší intenzitou potom na barvu popředí (případně naopak). Následující rozdílné metody prahování se potom liší především ve způsobu určení prahu.

Globální prahování

Pro globální prahování je charakteristické, že má jednu společnou hodnotu prahu pro celý obrázek. K určení této hodnoty vede několik různých cest.



Obrázek 3.1: Vhodný obrázek pro globální prahování a jeho histogram intenzit jasu pixelů

- Hodnota prahu může být pevně stanovena. Pokud je rozsah možných intenzit bodů 0 až 255, potom můžeme práh určit například na hodnotu 128. Body s intenzitou 0 až 127 budou převedeny na barvu popředí, body s intenzitou 128 až 255 potom na barvu pozadí. Tento způsob je však statický, a pokud by byl použit například na obrázek, který je příliš tmavý, byly by všechny body převedeny na jednu barvu.

- Z předchozího bodu vyplývá potřeba určovat hodnotu prahu pro každý obrázek podle jeho vlastností. Nejjednodušším způsobem je spočítání průměrné hodnoty pixelů a následné stanovení prahu na tuto hodnotu.

$$P = \frac{\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} B[i,j]}{m \cdot n}, \text{ kde m a n jsou rozměry obrázku v pixelech} \qquad (3.1)$$

  To zajistí, že body popředí a pozadí budou zastoupeny v rozumné míře.

- Ještě přesnějšího rozdělení však dosáhneme vyšetřením histogramu intenzity jasu bodů obrázku. Na obrázku 3.1 vidíme obrázek několika tmavých objektů na bílém pozadí a odpovídající histogram intenzit jasu pixelů. V tomto případě jsou hodnoty intenzit jasu popředí a pozadí hojně zastoupeny, mezi nimi vzniká oblast hodnot intenzit, kterých je v obrázku malé množství. Právě nejméně často zastoupená hodnota ležící mezi oběma významně zastoupenými oblastmi je ideálním kandidátem na zvolení za práh. Postup pro výběr prahu může být následující:

  1. Nalezneme dvě nejvyšší lokální maxima g_1 a g_2 v histogramu, která jsou od sebe minimálně vzdálená.
  2. Nalezneme nejnižší bod g_n, který se nalézá v histogramu H mezi hodnotami g_1 a g_2
  3. Nalezneme špičatost definovanou jako $\frac{\min(H(g_1),H(g_2))}{H(g_n)}$

4. Použijeme tu kombinaci g_1 a g_2, pro kterou je špičatost největší. Hodnota g_n je dobrou volbou pro takový práh, který rozděluje objekty.
   Zdroj: [6]

- Další metodou, jak zvolit globální práh, může být *iterativní výběr prahu*. Podmínkou použití tohoto algoritmu je, že z prahovaného obrazu musíme být schopni na základě některé jeho vlastnosti určit vhodnější hodnotu prahu. Postup:

  1. Odhadneme úvodní hodnotu prahu
  2. Provedeme prahování původního obrazu s aktuálně zvoleným prahem
  3. Určíme nový práh
  4. Opakujeme od bodu 2, dokud nejsme s výsledkem prahování spokojeni.

Problém s globálním prahováním nastává v případě, kdy se intenzita jasu bodů popředí, pozadí, nebo obou prvků zároveň nepohybuje ve stejných hodnotách pro celý obrázek. Často takové obrazy vznikají při nerovnoměrném osvětlení či v případě vrhnutých stínů. Objekty popředí jsou sice od pozadí jasně rozpoznatelné, ale protože v jedné části může být hodnota intenzity jasu pozadí srovnatelná s hodnotou intenzity popředí v jiné části, nelze najít takový globální práh, který by objekty rozpoznal v celém obraze. Příklad takové situace můžeme vidět na obrázku 3.2, výsledek nesprávně provedeného prahování potom na obrázku 3.3.

Lokální prahování

Rozdíl mezi globálním a lokálním prahováním spočívá v tom, že v prvním případě je práh určen pro všechny pixely obrazu jednotně, kdežto v druhém případě je hodnota určována pro každý pixel zvlášť podle jeho lokálního okolí.

Různé implementace této metody se mohou lišit ve velikosti, případně tvaru vyšetřovaného okolí i ve vlastnosti, která je vyšetřována, klasicky se však jedná o intenzitu jasu bodů okolí čtvercového tvaru.

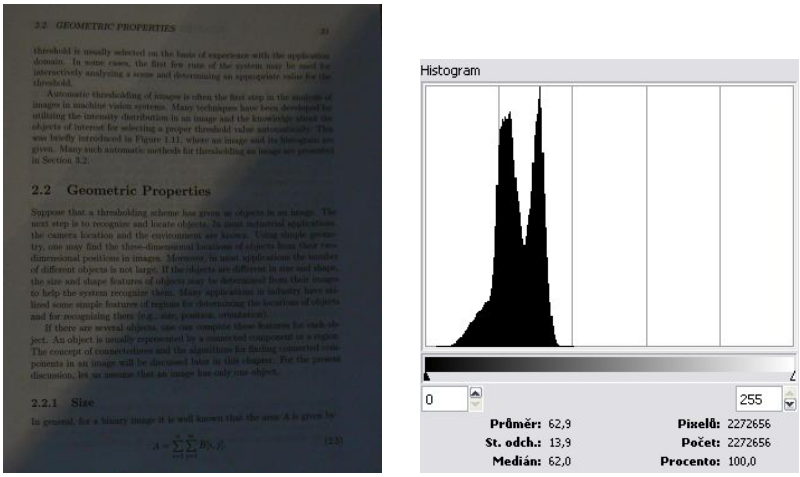
Pro stanovení prahu z vyšetřených hodnot okolí můžeme použít průměrnou hodnotu, medián, nebo například průměrnou hodnotu z nalezené maximální a minimální hodnoty.

Určení ideální velikosti okolí a funkce pro stanovení prahu se pro různé obrazy může lišit. Pokud například zvolíme velikost okolí na 11 x 11 pixelů, může být pro obraz, který obsahuje velké objekty, příliš malé a objekty budou členěny více, než je vhodné. Zato v případě malých objektů a prudkých změn intenzity obrazu by bylo vhodnější použít okolí ještě menší, protože při zvolené velikosti může docházet ke špatnému určení prahové hodnoty a objekty mohou splynout s pozadím.

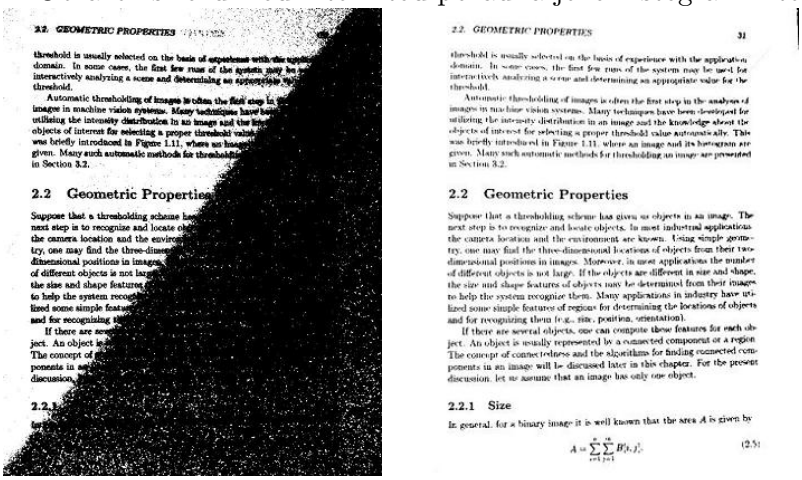
Problémem u lokálního prahování bývají velké oblasti s podobnou intenzitou, protože se při vyhodnocování bodů ležících uvnitř těchto oblastí vyšetřením lokálního okolí omezíme pouze na hodnoty podobné intenzity. Tím je část bodů chybně určena do jiné oblasti, než do které by byla zařazena v případě užití větší oblasti pro vyhodnocení. Tento problém řeší například metoda mean-C, kde je určena zvláštní prahová hodnota C, která je uplatněna právě pro prahování uniformní oblasti.

3.2 Redukce šumu

Pojem šum se používá pro náhodné, nepředvídatelné a nežádoucí signály nebo změny signálů, které zakrývají požadované informace. Například u digitální fotografie se často



Obrázek 3.2: Obrázek s rozdílnou intenzitou pozadí a jeho histogram intenzit pixelů



Obrázek 3.3: Srovnání výsledků globálního a lokálního prahování

vyskytuje u snímků pořízených za špatného osvětlení v podobě zrnitosti.

Ve výše zmíněném případě se jedná o *náhodný šum*, jehož příčinou jsou tepelné změny elektronických součástí ovlivňující výstupní analogový signál a kolísaní množství fotonů dopadajících na buňky snímače. Zdroj: [3]

Gaussův šum a mean filtr

Gaussův šum, také nazývaný *závislý šum*, je šumem, kde je každý pixel obrazu mírně pozměněn dle Gaussova rozložení. Pro jeho odstraňování se osvědčil mean filtr.

Mean filtr nahrazuje hodnotu každého pixelu průměrnou hodnotou určenou z hodnot sousedních pixelů a ze své vlastní. Tímto způsobem dochází k eliminaci hodnot, které s okolními hodnotami nemají žádnou spojitost.

$$h[i,j] = \frac{1}{M} \sum_{(k,l)\in N} f[k,l] \tag{3.2}$$

Výběrem velikosti okolí, ze kterého bude průměr určován, ovlivňujeme nejen rychlost

Obrázek 3.4: Obraz obsahující šum typu sůl a pepř

algoritmu, ale také míru redukce šumu. Při příliš velkém okolí bude obraz rozmazaný, při příliš malém bude šum potlačen v menší míře.

Zdroj: [18],[4]

Šum typu sůl a pepř a medián filtr

Šum typu sůl a pepř se ve své základní podobě v obraze projevuje změnou barvy náhodně rozmístěných pixelů na barvu, která je od okolí pixelu velmi odlišná. Příklad takového šumu můžeme vidět na obrázku 3.4.

Tento šum může být způsoben nechtěně (například vadnými elementy snímačů), nebo přidán záměrně. Pokud se jedná o druhý přípaď, jeho generování většinou probíhá prostým náhodným výběrem určitého množství pixelů a následnou změnou jejich barvy. Pro výběr těchto pixelů může být použit například některý z generátorů pseudonáhodných čísel.

Pro odstranění šumu typu sůl a pepř z obrázku existuje řada postupů fungujících většibnou na velmi podobném principu, z nichž se jako jeden z nejúčinnějších jeví *medián filtr*, který je naopak slabší při odstraňování velkého množství Gaussova šumu.

Medián filtr se svým principem od filtru mean příliš neliší. Také vyšetřuje okolí každého pixelu, ale hodnotu, kterou bude původní hodnota pixelu nahrazena, neurčuje jako průměrnou hodnotu, ale jako medián hodnot vyšetřovaného okolí.

Medián je hodnota, která dělí řadu dle velikosti seřazených hodnot na dvě stejně početné části. Pokud je prvků sudý počet n, medián je určen jako aritmetický průměr hodnot na pozicích $\frac{n}{2}$ a $\frac{n}{2} + 1$.

Důvodem vysoké účinnosti tohoto filtru je fakt, že pixely obsahující šum jsou extrémními hodnotami vzhledem k okolním bodům (příklad v tabulce 3.1) a určení mediánu je jimi ovlivněno pouze nepatrně. Tato skutečnost je prezentována na následujícím příkladu, kde je možné vidět srovnání fitrování stejného bodu pomocí filtru mean a medián.

Zdroj: [17]

Mean filtr provede pro uprostřed umístěný pixel následující výpočet:

12	14	9
13	255	19
2	11	21

Tabulka 3.1: Bílý bod v tmavém okolí

$\frac{12+14+9+13+255+19+2+11+21}{9} = 39,\overline{5}$

Výsledek filtru medián bude pro stejnou situaci následující:

$12, 14, 9, 13, 255, 19, 2, 11, 21 \rightarrow 2, 9, 11, 12, \mathbf{13}, 14, 19, 21, 255$

Výsledná hodnota mediánu (13) odpovídá hodnotám okolních bodů výrazně více, než výsledná hodnota mean filtru (40).

Impulsní šum

Impulsní šum je velice podobný šumu typu sůl a pepř, ale místo tmavých a světlých bodů obsahuje pouze světlé. Při jeho odstraňování se uplatňují stejné principy s velmi podobnými výsledky.

Velikostní filtr

Velikostní filtr již není zaměřený na obecně jakékoliv vstupní obrazy, ale pracuje s obrazy, které prošly prahováním za účelem rozpoznání objektů. Často se při prahování stane, že kromě objektů jsou barvou popředí označeny i jiné části obrazu. Klasickým výskytem této situace je obraz znečištěný šumem.

Protože tyto špatně rozpoznané oblasti mohou znemožňovat další operace s prahovaným obrazem, je třeba je odstranit. Jedná se tedy o konkrétní metodu pro redukci konkrétního typu šumu, která je navržena tak, že počítá s vhodným vstupním formátem obrazu (prahovaný obraz na binární podobu) a šum, který je v ní odstraňován, je předvídán jako osamocené malé objekty o stanovené maximální velikosti. Maximální velikost nežádoucích částic je třeba stanovit tak, aby při jejich odstraňování nebyly odstraněny žádoucí objekty.

Detekce velikosti spojitých oblastí může být prováděna následujícím způsobem.

1. Postupně jsou procházeny všechny pixely obrazu v pořadí od levého horního k pravému spodnímu, dokud se nenalezne pixel s hodnotou barvy popředí, který ještě nebyl přiřazen k žádné oblasti. Je nastavena hodnota délky detekované oblasti na 1 a daný pixel je označen jako přiřazený k oblasti.

2. Jsou prověřeny hodnoty pixelů v osmiokolí vyšetřovaného pixelu. Pokud je nalezen pixel s hodnotou barvy popředí, inkrementuje se délka detekované oblasti, nově nalezený pixel je označen jako přiřazený k oblasti a je pro něj proveden bod 2.

3. Pokud je délka detekované oblasti menší, než zvolená mez, je provedeno její odstranění obdobným způsobem, jakým byla provedena její detekce.

4. Pokračuje se v provádění bodu 1 z místa, kde bylo přerušeno.

Výsledek velikostního filtru je možné si prohlédnout na obrázku 3.5.

Obrázek 3.5: Obraz obsahující šum – po prahování se špatně detekovanými objekty – po redukci velikostním filtrem

3.3 Detekce oblasti znaků

Tímto bodem se dostáváme k metodám zpracování obrazu přímo spojeným s rozpoznáváním textu. První z nich, která bude zmíněna, je metoda pro detekci řádků využívající histogramu.

Vstupními daty této metody jsou obrazy obsahující text, které již prošly prahováním a v případě potřeby i redukcí šumu. Podmínkou pro úspěšné rozpoznání řádků je také, aby obraz obsahoval pouze text. To znamená, že v případech, kdy obsahuje text smíšený s obrázky, musíme tyto obrázky nejprve odstranit.

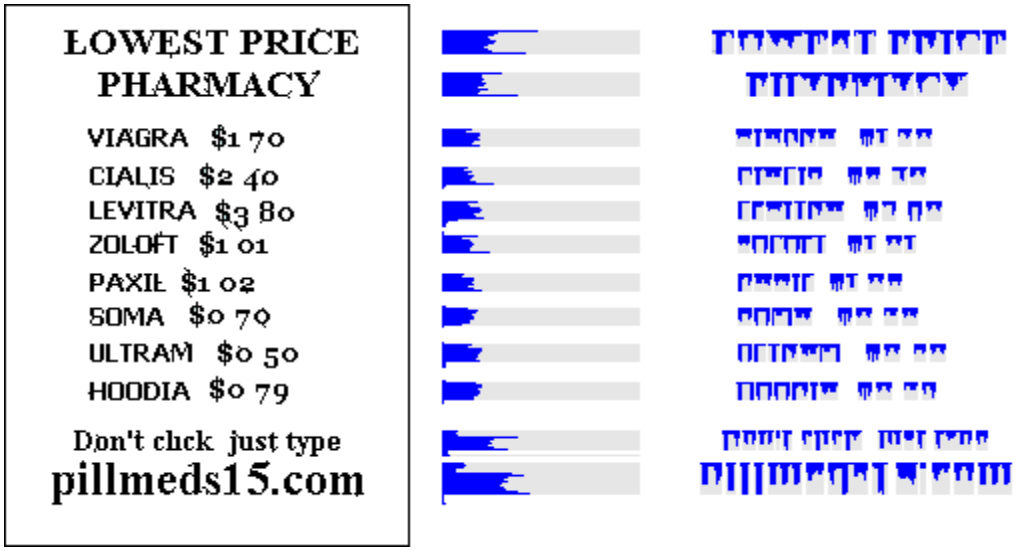
Detekce řádků

Oblast řádku je charakteristická vysokou koncentrací barvy popředí oproti oblastem mezer mezi řádky, kde se barva pozadí vyskytuje minimálně (pouze v případě výskytu nežádoucího šumu). Předpokladem pro funkčnost této metody je správný úhel natočení textu v obraze, tedy takový, aby byly řádky umístěny vodorovně.

Provedeme vyhodnocení množství pixelů intenzit popředí a pozadí pro každý řádek obrazu pomocí histogramu. Ze získaných výsledků detekujeme intervaly obsahující zvýšené množství bodů popředí, jedná se o oblasti, ve kterých se nachází řádky textu. Naopak oblasti mezer mezi řádky mají množství bodů popředí prakticky nulové, a tedy v histogramu bude naprosto převládat hodnota bodů pozadí. Příklad vyšetření histogramů můžeme vidět na obrázku 3.6 v prostřední části.

Právě zde by nastal problém, který by způsobil nefunkčnost metody, pokud by obraz obsahoval text smíšený s obrázkem. Obrázek by smazal rozdíly mezi množstvím bodů popředí v oblasti řádků a v oblasti mezer.

Pokud by byl obraz prahován zcela ideálně, bylo by možné za řádek označit interval řádků obsahující nenulové množství bodů popředí. Protože je však obraz často znečištěn šumem nebo se vyskytnou místa, která jsou při prahování chybně určena jako body popředí, je třeba takovou možnost uvažovat a zvolit práh, který bude oddělovat hodnoty, jez lze považovat za oblast řádku textu, od hodnot, které jsou s větší pravděpodobností výsledkem výše zmíněných poruch.



Obrázek 3.6: Vlevo: prahovaný obraz obsahující text; uprostřed: histogramy řádků; vpravo: histogramy sloupců z detekovaných řádků

Možným problémem tohoto algoritmu je například situace, ve které je na řádku následující text:

moc se nerozpoznal

Zde je totiž jediné písmeno (l) vyšší, než ostatní na řádku, a jediné zasahuje hlouběji (p), než ostatní. Pokud jsme zvolili mez posunutou na nízkou hodnotu bodů popředí, abychom se vyhnuli vlivu případného šumu, pravděpodobně bude oblast řádku rozpoznána bez horní části písmene l a spodní části písmene p.

Další možností je, že se v některé oblasti mezi řádky bude nacházet takové množství šumu, že bude rozpoznáno jako oblast řádku. Tento problém však lze snadno vyřešit, pokud stanovíme minimální velikost řádku. Pokud totiž k chybnému rozpoznání dojde, téměř vždy se jedná o oblast nepatrných rozměrů.

Detekce znaků

Tato metoda je spojena s předchozí. Poté, co máme rozpoznanou oblast řádků, můžeme pro tuto oblast opět vytvořit histogram, tentokrát však ne pro jednotlivé řádky, ale pro sloupce. Z histogramu jsme potom schopni určit, ve které oblasti se na řádku nachází znak. To je možné poznat stejně jako v předchozím případě, tedy tak, že detekujeme spojité oblasti obsahující body popředí. Příklad vyšetření histogramů jednotlivých řádků můžeme vidět na obrázku 3.6 v pravé části.

Problémem této metody jsou situace, kdy dochází k prolínání oblastí znaků (ať již z důvodu užití italiky, typu písma, či přítomného šumu). Potom je více znaků rozeznáno jako jediný.

3.4 Detekce hran

Detekce hran je postup, který slouží k nalezení oblastí pixelů, ve kterých dochází k podstatné změně intenzity jasu.

Jedná se o často využívanou metodu v oblasti počítačového vidění, protože umožňuje rozeznávání objektů – objekty zachycené v obraze mají ve většině případů rozeznatelné hrany, což umožňuje určit jejich pozici, velikost i další atributy. Dalším využitím je doostřování rozmazaného obrazu, to však s tématem rozpoznávání textu v obraze již příliš nesouvisí.

Hrana je takové místo v obraze, kde dochází k výrazné změně v jasové funkci. Derivace jasové funkce je v tomto místě nejvyšší ve směru kolmém na hranu, ale v praxi jsou hrany detekovány pouze v omezeném počtu směrů, typicky ve dvou či čtyřech směrech. Ve většině případů se k aproximaci této derivace využívá metody konvoluce. Konvoluce je matematický operátor zpracovávající dvě funkce. Většinou je v počítačové grafice prováděna tak, že z hodnot okolí pixelu je pomocí koeficientů v konvoluční masce vypočtena výsledná hodnota. Některá z používaných konvolučních jader:

- Robertsův operátor $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$
- operátor Prewitové 3 x 3 $\begin{pmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$
- operátor Prewitové 5 x 5 $\begin{pmatrix} -2 & -2 & -2 & -2 & -2 \\ -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \end{pmatrix}$
- Sobelův operátor $\begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$

Posledním krokem v detekci hran je prahování výsledků dosažených pomocí konvoluce. Pokud je prahová hodnota nastavena na příliš nízkou hodnotu, bude mezi hrany zahrnut i šum, oproti tomu při příliš vysoké hodnotě mohou být ztraceny některé podstatné hrany. Pro odstranění těchto problémů je použito prahování s hysterzí. To se provádí tak, že jsou nastaveny dva prahy. Nejprve jsou převedeny hrany pomocí prvního prahu, následně jsou hodnoty, které na již detekované navazují, detekovány pomocí druhého prahu.

Zdroj: [14], [16]

Kapitola 4

Rozpoznávání znaků

Pro rozpoznávání znaků bylo navrženo mnoho postupů, které využívají velmi rozmanitou škálu metod. V této práci je popsána metoda zužování a použití šablony pro detekci průsečíků se znakem, protože byly uplatněny v implementaci. Také jsou zmíněny neuronové sítě, protože se jedná o velmi zajímavou, variabilní a používanou metodu.

4.1 Zužování

Zužování je metodou zpracování obrazu, ve které je oblast obrazu vyjádřeného v binární podobě zredukována do podoby úseček. Tyto úsečky svou polohou odpovídají takzvaným středovým úsečkám. Také je možno je nazvat kostrou původní oblasti.

Jedná se o algoritmus, který je možno využít v řadě metod sloužících pro rozpoznání textu, protože oblasti obrazu jsou redukovány na jejich základní informaci, která je vhodnou formou pro následnou analýzu za účelem rozpoznání. Tato redukce je vidět na obrázku 4.1.

Důležitou vlastností metody je, že musí zachovávat propojenost oblastí. To znamená, že pokud jsou některé části obrazu před redukcí propojeny, i výsledný zredukovaný tvar by měl pro odpovídající části dodržet propojení.

Další podstatnou informací, kterou zužování musí alespoň přibližně zachovávat, jsou rozměry objektů. Protože algoritmus prahování většinou pracuje na principu odstraňování vnějších pixelů oblastí, je důležité, aby nebyl odstraněn krajní bod úsečky společně s ostatními body.

Nejčastěji užívaný přístup iterativně testuje osmiokolí každého pixelu a odstraňuje ty, které leží na okraji. Proces končí ve chvíli, kdy se během iterace neprovede žádná změna.

Zhang – Suen

Metoda zužování Zhang – Suen je zástupcem výše zmíněného přístupu. Pracuje se s oblastí okolí pixelu o velikosti 3 x 3 pixely a iterativně jsou odstraňovány pixely ležící na okrajích oblasti.

Postup zužování metodou Zhang – Suen:

1. Procházíme všechny pixely obrazu. Pokud narazíme na pixel obsahující hodnotu popředí, přejdeme na bod 2. V případě, že byl vyšetřen poslední pixel obrazu, přejdeme na bod 5.

2. Z hodnot osmiokolí pixelu b určíme hodnotu N, která odpovídá počtu pixelů popředí v daném okolí.

Obrázek 4.1: Různé podoby písmene T v původní podobě a po zúžení

	s	
z	b	**v**
	j	

Tabulka 4.1: Vyšetřovaný bod (b) a jeho čtyřokolí (s, j, v, z)

3. Určíme hodnotu P značící počet přechodů z pixelu s hodnotou pozadí na pixel s hodnotou popředí při procházení okolí pixelu b po směru nebo proti směru hodinových ručiček.

4. Pokud platí $(N \in < 2, 6 >) \wedge (P = 1) \wedge (s \cdot j \cdot v = 0) \wedge (z \cdot v \cdot j = 0)$, označíme vyšetřovaný pixel a pokračujeme v provádění bodu 1 v místě, kde bylo přerušeno.

5. Nastavíme všechny označené pixely na hodnotu pozadí.

6. Znovu procházíme všechny pixely obrazu. Pokud narazíme na pixel obsahující hodnotu popředí, přejdeme na bod 7. V případě, že byl vyšetřen poslední pixel obrazu, přejdeme na bod 10.

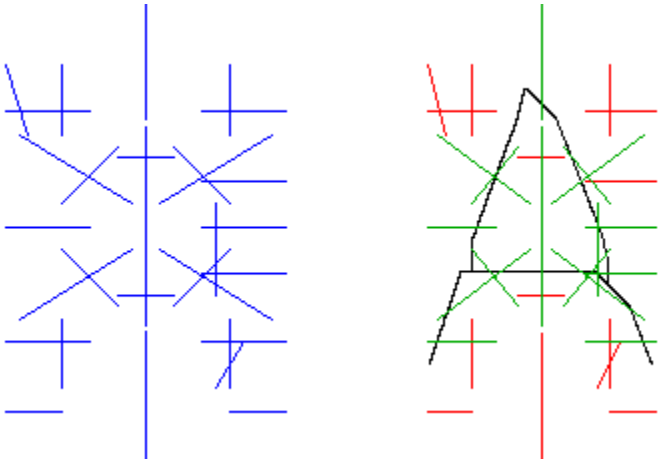
7. Z hodnot osmiokolí pixelu b určíme hodnotu N, která odpovídá počtu pixelů popředí v daném okolí.

8. Určíme hodnotu P značící počet přechodů z pixelu s hodnotou pozadí na pixel s hodnotou popředí při procházení okolí pixelu b po nebo proti směru hodinových ručiček.

9. Pokud platí $(N \in < 2, 6 >) \wedge (P = 1) \wedge (s \cdot j \cdot z = 0) \wedge (z \cdot v \cdot s = 0)$ (s, j, z a v jsou hodnoty intenzity sousedních pixelů znázorněných v tabulce 4.1), označíme vyšetřovaný pixel a pokračujeme v provádění bodu 6 v místě, kde bylo přerušeno.

10. Nastavíme všechny označené pixely na hodnotu pozadí.

11. Pokud byl alespoň jeden pixel změněn, znovu provedeme bod 1, jinak končíme.



Obrázek 4.2: Vlevo: možný návrh šablony pro detekci průsečíků; vpravo: ukázka detekce průsečíků s obrazem znaku *A*

Existuje řada dalších metod zužování, například metoda Stentiford. Některé z nich jsou zaměřeny na konkrétní potřeby zužování, jako například metoda SUSAN (Smallest Univalue Segment Assimilating Nucleus), která je vhodná pro úpravu obrazu po detekci hran.

Zdroj: [6], [7] a [11]

4.2 Detekce průsečíků s využitím šablony

Pokud mají být z obrazu rozpoznány znaky, je třeba získat informace, na základě kterých bude provedeno vyhodnocení. Takovou informací může být přímo rastrové znázornění znaku, ale lepším řešením je zvolit zdroj informací, který i v případě ne zcela totožných tvarů znaků (například z důvodu použití různých typů písma, použití tučného písma nebo různých velikostí písma) je schopen poskytnout alespoň do značné míry shodné informace.

Jedním ze způsobů, jak informace splňující tuto podmínku získat, je detekce průsečíků znaku a šablony, která je za tímto účelem navržena.

Šablona definuje konečné množství úseček umístěných tak, aby po přiložení šablony na oblast znaku, který si přejeme rozpoznat, protnuly či neprotnuly znak v místech, která nám poskytnou informace o podobě znaku. Příklad takové šablony je znázorněn na obrázku 4.2.

Přizpůsobení šablony oblasti znaku

Pro použití šablony je třeba zajistit, aby průsečíky svou pozicí odpovídaly znakům různé velikosti i různého provedení. K tomuto problému existují dva přístupy. Prvním z nich je ten, kdy šablonu necháme v původní velikosti, a přesně opačný, kdy velikost znaku neměníme, ale změníme velikost šablony tak, aby se shodovala s velikostí znaku.

Při prvním přístupu je problematické neporušit tvar písma, jeho zúžení, spojitost a další vlastnosti, které jsou důležité k jeho rozpoznání. Proto se jako vhodnější jeví druhý přístup, kdy měníme velikost šablony. Zde je však třeba dát pozor na situace, kdy je znak příliš malý a po přiložení šablony neskýtá dostatečné množství pixelů pro detekci průsečíku. To nastává v případě, kdy úsečka, pro kterou je detekován průsečík, je příliš krátká. Například úsečka o délce dvou pixelů nemůže v žádném případě detekovat průsečík.

Tuto komplikaci lze řešit spojením obou metod, kdy je v základu použita změna velikosti šablony na velikost vyšetřované oblasti. Pokud je vyšetřovaná oblast příliš malá, je pomocí vhodného algoritmu zvětšena, případně i upravena tak, aby byly eliminovány nežádoucí následky zvětšení a poskytla lepší možnosti rozpoznání průsečíků.

Šablonu je možné přímo vytvářet tak, aby byla v podobě, která je snadno použitelná pro různé velikosti znaků. Úsečky, které mají jejich koncové body vyjádřeny v hodnotě, jež je zároveň procentuálním vyjádřením jejich polohy v rámci šablony, umožňují snadný výpočet jejich polohy pro konkrétní oblast, a proto se jeví jako vhodné řešení.

Genetický algoritmus

Pro dosažení vhodného umístění úseček, dle kterých budou detekovány průsečíky, je možno využít genetického algoritmu. Genetický algoritmus se řadí do skupiny evolučních algoritmů. Evoluční algoritmy se snaží napodobit evoluční procesy známé z evoluční biologie k tomu, aby nalezly řešení náročných a rozsáhlých úloh. V rámci množiny možných řešení vytvářejí nová kombinováním vlastností stávajících a vyřazují ta, která se ukážou být nevhodnými.

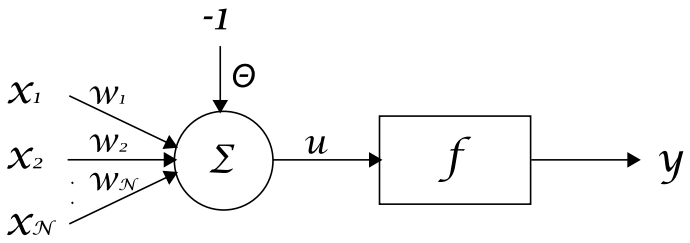
Pro genetický algoritmus je třeba určit:

1. objekt, který bude pomocí genetického algoritmu navržen,
2. vlastnosti objektu, které jsou měněny,
3. geny, které vedou k ovlivnění vlastností objektu, a způsob, jakým k němu dochází,
4. způsob křížení stávajících objektů za účelem vytvoření nové generace,
5. způsob vyhodnocení kvality objektů,
6. pravidla pro třídění objektů na ty, které budou v množině, se kterou se dále pracuje a na ty, které budou vyřazeny.

V případě použití pro návrh šablony pro detekci průsečíků se znaky je objektem přímo tato šablona a vlastnostmi, které jsou algoritmem měněny, jsou pozice koncových bodů úseček tvořících průsečíky s vyšetřovaným znakem. Geny je potřeba určovat takovým způsobem, aby byly obsaženy v takové míře, která dostačuje pro vytváření nových návrhů. Zpravidla je tedy genů vysoký počet a jsou vyjádřeny v jednoduché formě. Počáteční generace je většinou generována náhodně. V důsledku vyřazování neúspěšných návrhů a generování nových z těch, které se ukázaly jako úspěšné, by mělo v průběhu algoritmu dojít k dosažení kombinace genů, která zajišťuje přibližně optimální vlastnosti objektu. Během křížení mohou být prováděny mutace – náhodné změny genů, které nejsou závislé na křížení a mají zabránit, aby se proces evoluce dostal pouze do oblasti lokálního optima. Mutace se většinou uplatňují v nepatrné míře, aby příliš neznehodnotily průběh evoluce. Kvalita výsledku genetického algoritmu by měla obecně stoupat s počtem provedených cyklů, kdy jedním cyklem je myšleno vytvoření nové generace objektů a vyřazení neúspěšných.

Algoritmus tedy může mít následující strukturu:

1. inicializace počáteční generace,
2. provedení křížení,
3. provedení mutace,
4. provedení ohodnocení vygenerovaných řešení,



$$y = f(\sum_{i=1}^{\mathcal{N}} w_i x_i - \Theta)$$

y - výstup neuronu

x_i - vstupy neuronu

u - vnitřní potenciál neuronu

Θ - práh neuronu

w_i - váhy neuronu

f - neuronová aktivační funkce

Obrázek 4.3: Model neuronu ([13])

5. provedení přirozeného výběru,
6. pokud nebylo dosaženo požadovaného výsledku, opakování postupu od bodu 2,
7. ukončení algoritmu a výběr nejlépe ohodnoceného řešení ze stávající populace.

Více o genetických algoritmech je možné nalézt ve zdrojích, z nichž bylo pro tuto práci čerpáno – [9], [15]

4.3 Neuronové sítě

Neuronové sítě jsou typem výpočetních modelů využívaných v oblasti umělé inteligence. Jejich chování napodobuje chování biologických neuronových propojení.

Neuronová síť se skládá z neuronů, které jsou vzájemně propojeny a na základě aktivačních funkcí si předávají signály. Neuron přijímá vzruchy ze vstupů, kterých může mít libovolný počet, a následně dle aktivační funkce rozhodne, zda vyšle signál do svého jediného výstupu, nebo zůstane neaktivní.

Některé aktivační funkce neuronu:

- Sigmoidální funkce: $f_s(u) = \frac{1}{1+e^{-u}}$
- Hyperbolický tangens: $f_h(u) = \text{tgh}(u)$
- Znaménková funkce: $f_h(u) = \text{sgn}(u)$
- Heavisideova funkce: $f_H(u) = \begin{cases} 1 \text{u} > 0 \\ 1 \text{u} \leq 0 \end{cases}$

Neuronové sítě nalézají uplatnění v oblastech rozpoznávání obrazů a zvuků, předvídání vývoje časových řad, aproximace funkce i filtrování spamu. Takto široké spektrum využití je dáno jejich variabilitou a schopností učit se. Právě schopnost učit se je pro neuronové sítě charakteristická. Učení probíhá tak, že se neuronové síti poskytnou učební data, u kterých známe výsledek, který očekáváme. Data necháme sítí vyhodnotit a dle správného nebo špatného výsledku upravíme váhu synapsí tak, aby se snížil rozdíl mezi skutečným a požadovaným výstupem. Tento přístup je nazývaný učením s učitelem (supervised learning). Existuje také přístup učení bez učitele, kdy se váhy synapsí nastavují samy takovým způsobem, aby síť poskytovala stejnou odezvu pro stejné či podobné vstupy.

Neurony jsou většinou uspořádány do několika vrstev. Vstupní a výstupní vrstva je počtem neuronů uzpůsobena požadovanému vstupu a výstupu sítě. Protože je první vrstva uzpůsobena vstupu a její neurony slouží pouze pro rozdělení vstupů na neurony následující vrstvy, do celkového počtu vrstev modelu se nezapočítává. Kromě vstupní a výstupní vrstvy neuronů může síť obsahovat také jednu nebo více skrytých vrstev. Úkolem skrytých vrstev je zlepšit aproximační vlastnosti neuronové sítě. Propojení neuronů se vytvářejí v rámci jedné vrstvy i mezi neurony ležících na jiných (nejčastěji však sousedních) vrstvách.

Sítě lze dělit také z hlediska toku signálů na:

- Dopředné sítě – signál se šíří pouze jedním směrem.
- Zpětnovazební sítě – mezi neurony a vrstvami existují zpětné vazby. Díky zpětným vazbám je u této skupiny těžké definovat vstupní a výstupní vrstvy.

Architektury neuronových sítí

V následujícím textu jsou ve stručnosti zmíněny některé ze známých návrhů neuronových sítí.

- *Síť perceptron*

  Perceptronové sítě jsou jedny z nejznámějších a v praxi nejpoužívanějších neuronových sítí. Síť je tvořena jedním nebo více perceptrony (obrázek 4.3). Perceptron využívá učení s učitelem. Jeden perceptron může klasifikovat pouze do dvou tříd, pokud je potřeba více tříd, využívá se více perceptronů nebo vrstev sítě.

- *Hopfieldova síť*

  Hopfieldova síť je jednovrstvou neuronovou sítí, ve které je každý neuron propojen se všemi ostatními. Počet neuronů v síti je dán počtem vstupů či výstupů (vyšší hodnota ze zmíněných). Každý neuron je zároveň vstupním i výstupním. Výstup neuronů je přiváděn na vstupy dalších – vzniká zpětná vazba. Hopfieldova síť využívá učení bez učitele. Hodnoty na vstupu způsobí reakci na výstupu, která je následně opět přiváděna na vstup až do jejich rovnosti. Tento typ sítí se využívá například při optimalizačních problémech.

- *Kohonenova síť*

  Kohonenova síť obsahuje jedinou vrstvu neuronů – Kohonenovu kompetenční vrstvu. Každý neuron v této vrstvě má informaci o hodnotě každého vstupu. Váha spojení neuronů udává konkrétní polohu neuronu v prostoru. Opět se jedná o síť využívající učení bez učitele, učení je ovlivňováno vzdáleností neuronů a proces probíhá podle předem stanoveného počtu iterací. Kohonenovy sítě se uplatňují při rozpoznávání vzorů a robotice.

- *ART síť*

  ART (Adaptive Resonance Theory) sítě řeší problém proměnné stability (pokud je již naučená síť učena z nového vzoru, riskuje se poškození již naučené informace). Problém je řešen dvěma módy sítě – tvárným a stabilním. Síť je dvouvrstvá (porovnávací a rozpoznávací vrstva) a využívá principu učení bez učitele. Učení probíhá na základě výměny informace mezi vrstvami do doby, než dojde k ustálení stavu.

Zdroj: [13], [5], [1]

Kapitola 5

Návrh systému

Jedním z cílů práce bylo navrhnout takový OCR systém, který je do určité míry schopen odstranit následky úpravy obrazového spamu pro ztížení rozpoznání textu. V následující části je popsán návrh tohoto systému, problémy, které při něm bylo nutné řešit, a z jakého důvodu a jakým způsobem byly použity které metody.

5.1 Prahování obrazu

Metody úpravy i rozpoznávání obrazu, které budou v systému použity, pracují s obrazem v binární podobě, tedy takové, kdy je obraz složen pouze z pixelů majících barvu popředí, nebo pozadí. Protože však naprostá většina obrazů (obrazový spam nevyjímaje) obsahuje výrazně větší množství barev, je třeba provést prahování obrazu, které zajistí převod z vícebarevného originálu na obraz obsahující pouze požadované dvě hodnoty barvy. Prvním krokem, který je v případě prahování prováděn, je převod barevného obrazu na odstíny šedi.

$$r(i,j) = g(i,j) = b(i,j) = 0.3 \cdot r(i,j) + 0.59 \cdot g(i,j) + 0.11 \cdot b(i,j)$$

Indexy i a j jsou indexy pixelu převáděného obrazu získané z jeho pozice v obrazu, r, g a b jsou červená, zelená a modrá složka barvy.

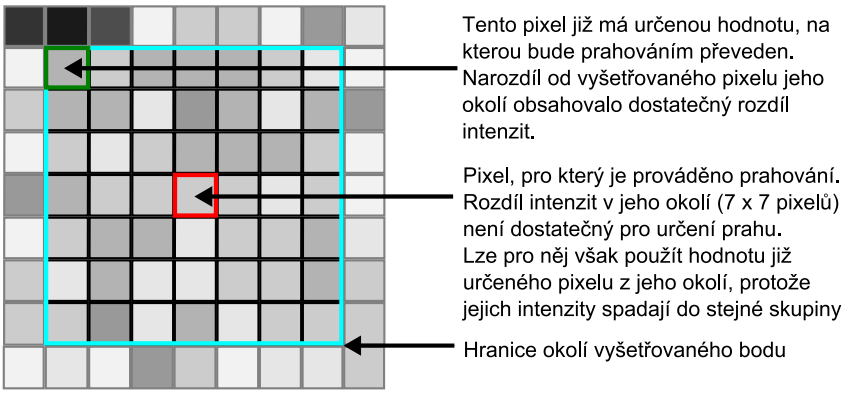
Důležité je zvolit správnou metodu prahování. Přestože obrazový spam zřídkakdy využívá takového stylu pozadí textu, který by měl za následek nepoužitelnost globálního prahování, tato metoda nebyla zvolena a bylo použito lokální prahování.

Toto rozhodnutí má svůj důvod především v jiném charakteristickém rysu obrazového spamu, a to ve vícebarevném textu. Se špatně zvolenou hodnotou globálního prahu by mohlo dojít k situaci, kdy text, který byl napsán příliš světlou barvou, nebude správně převeden na hodnotu popředí, ale protože se intenzita jeho pixelů nachází pod hodnotou stanoveného prahu, bude označen jako oblast pozadí.

Výše popsanou situaci lze vyřešit použitím lokálního prahování, které je sice oproti globálnímu pomalejší, ale při vhodných parametrech dokáže lépe rozlišit objekty v obraze od pozadí.

Jak již bylo zmíněno v kapitole 3, lokální prahování stanovuje prahovou hodnotu pro každý pixel zvlášť dle hodnot v lokálním okolí. Parametry algoritmu, které je třeba zvolit, jsou velikost a tvar vyšetřovaného okolí a způsob určení prahu.

Velikost vyšetřovaného okolí pixelu je třeba volit tak, aby byl vzorek dat dostatečně rozsáhlý pro správné stanovení hodnoty prahu a zároveň algoritmus neztrácel svou výhodu práce pouze s lokálními hodnotami. Pro navrhovaný systém bylo zvoleno čtvercové okolí



Obrázek 5.1: Vyšetřování okolí pixelu pro stanovení hodnoty prahu

o rozměrech 7 x 7 pixelů. Takové okolí se ukazuje být vhodné pro prahování textu v obrazovém spamu vzhledem k jeho velikosti i tloušťce.

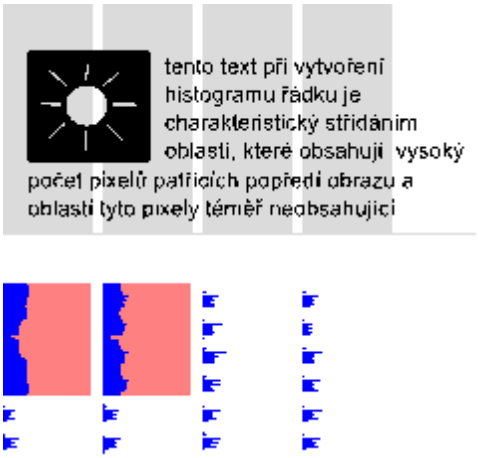
Okolí o takto malé velikosti se však ukazuje být problematické při prahování větších oblastí pixelů podobné intenzity, protože se při výběru prahu vychází pouze z hodnot neobsahujících žádný z pixelů opačné skupiny intenzit. Tento problém je však snadno řešitelný, pokud stanovíme minimální hodnotu, o kterou se musí intenzita pixelů s nejnižší a nejvyšší intenzitou lišit, aby byl z lokálních hodnot stanoven práh. Pokud není dosaženo požadovaného rozdílu, může se provést prahování dle globálně stanoveného prahu, nebo lze využít toho, kdy některý z pixelů, které se nacházejí ve vyšetřovaném okolí, již byl prahován, a použít stejnou hodnotu, která bude použita i pro tento pixel. Tento postup je patrnější na obrázku 5.1.

Hodnota prahu je určena jako aritmetický průměr nejvyšší a nejnižší hodnoty intenzity v okolí vyšetřovaného pixelu. Tato volba vychází z nutnosti prahovat hodnoty na rozmezí znaků a pozadí, na němž se znaky nacházejí, které většinou leží blízko průměrné hodnoty intenzit obou skupin pixelů. V případě, kdy by byly všechny pixely s hodnotou blízkou průměru převedeny na barvu pozadí, by mohla být poškozena podoba znaku, což by vedlo k chybnému rozpoznání. V opačném případě by hrozilo, že znaky budou po prahování příliš silné, což opět vede k možnému poškození podoby znaku, navíc však i k riziku propojení více znaků dohromady. K těmto problémům by mohlo vést jak určení prahu jako průměrné hodnoty ze všech hodnot intenzit v okolí vyšetřovaného pixelu, tak i stanovení prahu jako mediánu z těchto hodnot. Obě možnosti by totiž v případě, kdy by se v obraze vyskytovalo velké množství pixelů jedné skupiny intenzit, vybraly práh ovlivněný tímto množstvím.

Lokální prahování se na testovaných vzorcích osvědčilo, text byl převeden v přijatelné podobě, v případě použití více barev nedošlo ke ztrátě některých textů a pozadí bylo s vysokou úspěšností převedeno na barvu pozadí.

5.2 Odstranění obrázků

Nejen v oblasti obrazového spamu se můžeme velice často setkat s obrazy, které obsahují text i obrázek. Pro lidský mozek je snadné tyto informace od sebe oddělit. V případě počítačového vidění však oddělení těchto dvou složek představuje překvapivě závažný problém.



Obrázek 5.2: Rozlišení oblastí obrázku a textu pomocí histogramů částí obrazu

Již se totiž nejedná o odlišení objektů zachycených v obraze od pozadí, ale o rozlišení částí obrazu obsahujících jeden typ objektu od částí obsahujících jiný.

Pro oddělení obrázků od textu se musíme zaměřit na vlastnosti, kterými se od sebe tyto oblasti liší, a na způsob, jak je detekovat.

- Oblast textu se od oblasti obrázku liší ve velikosti objektů, které se v ní vyskytují. Text je složen ze znaků, které jsou většinou malých rozměrů a příliš se od sebe svou velikostí neliší. Naproti tomu obrázek, který prošel prahováním, ve většině případů obsahuje rozsáhlé spojité oblasti. Spojitost je v tomto případě vhodné ověřovat s využitím osmiokolí.

- Oblast textu je charakteristická tím, že jsou znaky uspořádány v řádcích, a tedy při vyšetření řádků pomocí histogramu dostáváme spojité oblasti s vysokým obsahem bodů popředí a oblasti, které body popředí téměř neobsahují. Naproti tomu u obrázku se většinou takové střídání oblastí nevyskytuje.

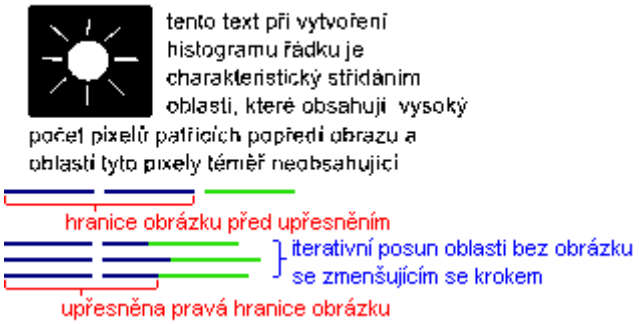
V případě využití první ze zmíněných odlišností je problémem, pokud by bylo písmo podtržené, že spojitá oblast bude příliš velká, bude považována za obrázek a bude odstraněna. To využití způsobu založeného na velikosti objektů do značné míry limituje.

Využití druhé ze zmíněných odlišností problém s podtrženými řádky nemá, ovšem detekce je poněkud obtížnější. Protože se nepodařilo nalézt žádnou metodu, která by dokázala této vlastnosti textu využít pro segmentaci obrazu, byl navržen vlastní postup. Následuje jeho stručný popis.

Vlastní postup pro oddělení obrázků od textu

Principem oddělení je detekce takových částí obrazu, které obsahují pravidelně se střídající oblasti řádků s vysokým a nízkým výskytem pixelů náležících popředí obrazu.

To lze provést například tak, že obraz vertikálně rozdělíme na oblasti stejné šířky (vyznačené oblasti v horní části obrázku 5.2). Pro každou tuto část zobrazíme histogram řádků. Pokud je nalezena spojitá skupina řádků obsahující vyšší koncentraci bodů popředí, která je větší, než stanovená mez, nalézá se na této pozici obrázek (vyznačené oblasti ve spodní části obrázku 5.2).



Obrázek 5.3: Upřesnění hranic obrázku posunem oblasti prahování

Z takto získaných informací jsme schopni s nepřesností závislou na šířce vyšetřovaných oblastí zjistit přibližnou pozici obrázku. Z popisu postupu je zřejmé, že jedním omezením, který tento postup má, je, že detekovaná oblast obrázku je obdélníková, což mnoha obrázkům nemusí odpovídat.

Přesnějšího stanovení levé a pravé hranice obrázku lze dosáhnout podrobnějším vyšetřováním okrajových oblastí, ve kterých byl obrázek rozpoznán. To lze provést iterativním upřesňováním oblasti, která je vyšetřována. Oblast, která následuje za oblastmi detekujícími obrázek, je posunuta o vzdálenost odpovídající polovině velikosti oblasti směrem k detekovanému obrázku. V posunuté oblasti je znovu detekován obrázek použitím histogramu řádků. Pokud je nalezen, je hranice posunuta o polovinu předchozího posunu směrem od detekovaného obrázku. Pokud obrázek nalezen není, hranice se posouvá o polovinu předchozí vzdálenosti směrem k obrázku. Tento postup se opakuje, dokud není velikost posunu dostatečně malá. Funkci tohoto postupu ilustruje obrázek 5.3.

I přes toto vylepšení není postup zdaleka dokonalý.

- Text, který by kopíroval hranice neobdélníkového obrázku, by mohl být zčásti zahrnut do oblasti obrázku.
- Také není zaručeno, že obrázek bude větší, než stanovená mez. Menší obrázky většinou zůstanou nerozpoznány.
- Obrázky, které budou při prahování převedeny takovým způsobem, že budou co do rozložení bodů popředí a pozadí svými vlastnostmi podobné řádkům textu, také nebudou tímto postupem odstraněny.

Přesto však aplikování tohoto postupu na obrazový spam, který obsahoval text smíšený s obrázky, přinesl zlepšení výsledků. V případě potřeby by bylo možno postup dále zdokonalovat a některé ze zmíněných nevýhod odstranit.

Poté, co jsou oblasti obrázků rozpoznány, jsou vyplněny barvou pozadí. Touto úpravou se dosáhne toho, že obrázky neovlivní rozpoznání řádků textu v následující části systému.

Při testování na získaných vzorcích se projevily některé vady navrženého postupu, kdy při členitých, těsně obtékaných obrázcích byla odstraněna i část textu. Rovněž nebyly odstraněny malé obrázky. Přesto však algoritmus výrazně ovlivnil úspěšnost rozpoznání textu u většiny obrazových spamů obsahujících text i obrázky. Upřesňování hranic znatelně zlepšilo detekci rozměrů obrázku a snížilo chybovost algoritmu. Nejúspěšnější byl tento postup u tmavých obrazů obdélníkového tvaru dostatečné velikosti.

5.3 Redukce šumu

Autoři obrazového spamu se často snaží ztížit rozpoznání textu záměrným přidáním šumu do obrazu. Pro lidský mozek není příliš problematické tento šum od samotného textu odlišit, ale v případě počítačového zpracování působí problém hned z několika důvodů.

Protože se při provedení operace prahování z obrazu ztrácí informace o barevnosti jeho jednotlivých částí, což je vhodné pro detekci textu složeného z různě barevných částí, je i značná část šumu převedena na hodnotu barvy popředí a byla by zahrnuta do dalších fází rozpoznávání. Zde by z tohoto důvodu docházelo k nepřesnostem hned ve dvou fázích.

První z nich je rozpoznávání řádků, kdy je využito nahromadění barvy popředí v oblastech textu a naopak jejího nedostatečného výskytu v ostatních oblastech. Pokud však předem neodstraníme šum, který je rozprostřen přibližně rovnoměrně v celé oblasti obrazu, zaplní i oblasti, ve kterých se text nevyskytuje, body popředí a je šance, že řádky textu nebudou vyhodnoceny správně, a tedy nebude rozpoznán ani v řádcích obsažený text.

Druhou závažnou nepřesnost šum způsobuje při samotném rozpoznávání znaků za pomoci šablony, protože pokud se nalézá v oblasti rozpoznaného znaku, může vytvářet průsečíky i v místech, kde by se jinak nenacházely a způsobit chybné určení textu. Zde je třeba rozlišit dvě varianty znečištění oblasti znaku šumem. První z nich nastává tehdy, kdy se částice přidaného šumu sice nalézá v oblasti znaku, ale se znakem samotným nekoliduje, potom je možné za dalších podmínek, které budou popsány dále, částici bezpečně odstranit. Pokud však částice se znakem přímo koliduje a po redukci barevného prostoru s ním zcela splývá, předchozí postup již uplatnit nejde a znak je skutečně šumem poškozen. Avšak ani za této situace samotné rozpoznání nemusí skončit chybou, protože stále existuje možnost, že poškození buď šablonu vůbec neprotne, nebo sehraje dostatečně nepatrnou roli na to, aby bylo písmeno vyhodnoceno rozdílně.

Šum charakteristický pro obrazový spam

Šum v získaných exemplářích obrazového spamu je sice velmi podobný šumu typu *sůl a pepř*, ale jedna podstatná odlišnost existuje. Nejedná se totiž o zcela osamocené pixely, ale o malé shluky pixelů většinou v podobě čárek.

Přes jeho rozdílnou podobu by se však metody pro odstranění šumu typu sůl a pepř (například medián filtr) daly použít (ať již přímo, nebo po drobné úpravě) i zde. I tento typ šumu je totiž při jejich použití úspěšně eliminován.

Problém však nastává s podobou znaků poté, co je filtrace provedena. Text bývá oproti pozadí, na kterém se nachází, velmi kontrastní a filtry redukující šum jej silně poškozují. To by nebylo vážné v případech, kdy jsou znaky dostatečně velké, ale právě v obrazovém spamu, který se většinou nachází v obrázcích malých rozměrů, je takřka vždy písmo velmi malé. To je potom filtrem poškozeno do té míry, že jeho rozpoznání není možné. Z toho důvodu není možné použít mean či medián filtr, ale šum musí být odstraněn použitím velikostního filtru.

Pro správnou funkci velikostního filtru je třeba vhodně nastavit mez, která určuje, zda bude spojitá oblast ponechána, nebo odstraněna. Musí být zvolena tak, aby došlo k odstranění co největšího množství přítomného šumu, ale zároveň nedošlo k odstranění některých písmen.

Zaměření na obrazový spam umožnilo tuto mez definovat o něco výše, než by tomu bylo možné při odstraňování tohoto šumu z libovolného obrázku s textem, protože v případě, že jsou z textu spamu odstraněna interpunkční znaménka a diakritika, prakticky nijak

nedochází k ovlivnění jeho rozpoznávání. Horní hranicí se tedy ukázal být znak *i*, který by byl při příliš vysoké mezi odstraněn jako první.

Po experimentování s dostupnými materiály byla zmiňovaná mez nastavena na hodnotu 6. To znamená, že všechny detekované oblasti, které byly menší než 6 pixelů, jsou odstraněny.

Výsledky redukce pomocí velikostního filtru je možné vidět na obrázku 3.5.

Aplikování velikostního filtru se projevilo především výrazným zlepšením rozpoznání oblasti řádků a znaků. Pokud byl částicí šumu poškozen přímo některý znak, toto poškození však pochopitelně nebylo napraveno.

5.4 Detekce oblasti znaků

Nyní, když je obraz více či méně úspěšně segmentován na pozadí a znaky, můžeme přejít k jejich samotnému rozpoznávání. Proto, abychom mohli znak rozpoznat, musíme nejprve vědět, kde se nachází. A právě proto systém obsahuje část, která detekuje oblasti znaků.

Pro získání oblastí jednotlivých znaků je prvním krokem získání oblastí řádků, ve kterých se znaky nacházejí. Pro jejich detekci je použita metoda popsaná v bodu 3.3, která opět používá pro jejich detekci histogram řádků.

Protože šum, který byl redukován v předchozí metodě, nemusí být odstraněn zcela dokonale, nelze považovat za prostor mezi řádky taková místa, kde je počet bodů náležících popředí nulový, ale ta, kde je tento počet velmi nízký. V bodě 3.3 jsou při popisu metody zmíněny případy, kdy mohou některé části znaků ležet mimo detekovanou oblast. Pro tuto situaci existuje jednoduchá náprava.

Oblast, ve které se znak nachází, shora i zespoda zvětšíme o takový počet pixelů, aby byly tyto případné části znaku do dané oblasti i přesto zahrnuty. Protože se jedná o malý posun a zpravidla je mezi jednotlivými řádky dostatečný prostor, není příliš pravděpodobné, že by došlo k obsažení i části znaků dalšího řádku.

Po detekci řádku je možné přejít k detekci oblastí jednotlivých znaků, které se v tomto řádku nacházejí.

Ta probíhá využitím metody popsané v bodu 3.3. Postup je velmi podobný jako v předcházejícím případě, nyní je však již správnou volbou označit za prostor mezi znaky svislé řady pixelů, ve kterých se nenachází žádný z pixelů náležících popředí.

V případě výskytu šumu mezi jednotlivými znaky, použití italiky, zvláštního typu písma nebo nedostatečné kvality obrazu však může být více písmen detekováno jako jediné. Této situaci se nelze příliš účinně bránit. Pouze v některých případech by mohlo pomoci případné vylepšení postupu o stanovení meze šířky jednoho znaku, kdy po jejím přesažení by byl znak rozdělen v polovině, nebo v místě nejmenšího počtu pixelů popředí. Toto vylepšení by se nejspíš ve většině případů minulo účinkem, ale alespoň částečné zvýšení úspěšnosti rozpoznávání by mohlo přinést. Jedná se však pouze o návrh vylepšení, které nebylo zahrnuto do finální podoby systému.

Detekce oblasti znaků probíhala u většiny textů správně. Někdy však při nevhodné kombinaci znaků na řádku byla výška řádku rozpoznána pouze pro nízké znaky a následně při detekci průsečíků úsečky šablony neodpovídaly zcela svou pozicí, což se negativně projevilo na výsledku rozpoznávání textu. Tuto situaci by bylo možno řešit detailnějším vyhodnocením histogramů řádků nebo odhadem správné výšky řádku z průměrné šířky znaků.

Dalším případem, kdy nebyla oblast znaků rozpoznána správně, je situace, kdy se v důsledku nedostatečného rozlišení, chyby prahování, poškození šumem nebo sklonu znaků

překrývají oblasti dvou či více po sobě jdoucích znaků. Jedná se však o ojedinělé případy a výsledek rozpoznávání byl tímto problémem ovlivněn pouze v omezené míře.

5.5 Detekce průsečíků

Ve chvíli, kdy je určena oblast znaku, je možné přikročit k určení, o který znak se jedná. K tomu je potřeba vycházet z některé z vlastností znaků, které vedou k jejich rozlišení a jsou zpracovatelné pro počítač.

Jednou z variant, jak takové hodnoty získat, je určení průsečíků se šablonou, která obsahuje vhodně umístěné úsečky. Právě tato metoda, jejíž podrobný popis se nachází v bodu 4.2 je využita v navrhovaném systému.

Návrh šablony

Úsečky, které poslouží pro detekci průsečíků, musí být v šabloně vhodně umístěny. Pro jejich pozici by mělo platit, že případné protnutí nebo neprotnutí se znakem v daném místě je pro některou skupinu znaků charakteristické a pro jinou naopak nezvyklé. Správným umístěním více takových úseček potom lze dostat potřebné množství informací pro rozhodnutí, o který znak se jedná.

Prvotní návrh šablony byl sestaven odhadem. Úsečky byly umístěny tak, aby detekovaly průsečíky na místech, která se zdála být dobrými zdroji informací o podobě znaku. Následně byla šablona otestována na příkladech a upravena tak, aby pozice úseček více odpovídala jednotlivým znakům.

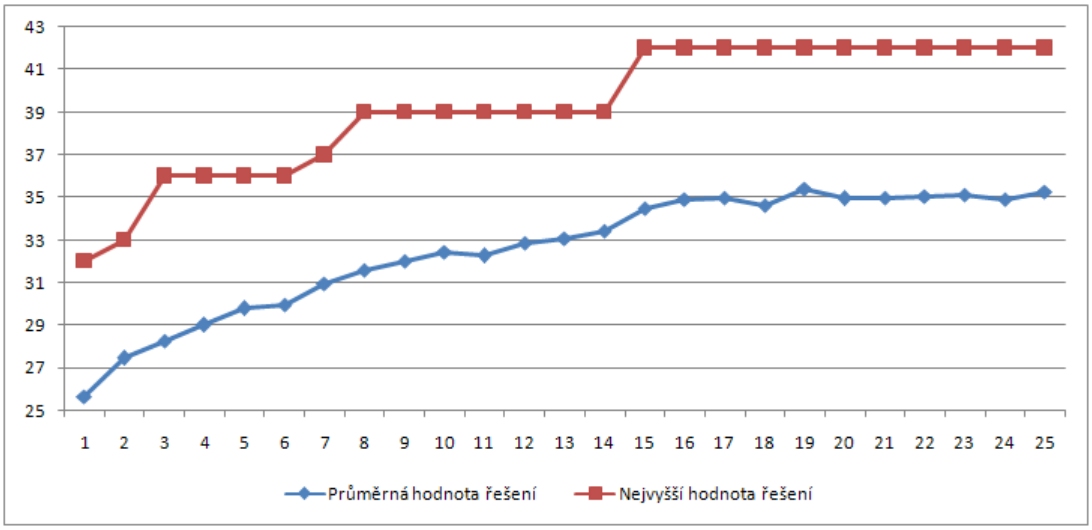
V další úpravě bylo přidáno několik dalších detekčních úseček, které upřesňovaly některé rysy znaků, jako například neuzavřenost písmen G, C a dalších, nebo sloužily k odlišení podobných znaků jako například O a Q.

Další vylepšení šablony přináší použití genetického algoritmu k přesnějšímu stanovení pozic úseček. Princip genetického algoritmu je popsán v bodu 4.2.

Vlastnostmi, které jsou ovlivněny pomocí genů, jsou pozice koncových bodů úseček, počet genů pro jeden bod je stanoven na 30. Hodnoty genů jsou R, L, U, D a X a ovlivňují pozici bodu o jedno procento rozměru šablony ve směrech doleva, doprava, nahoru, dolů a beze změny. Úvodní pozice, od které se počítá konečná pozice po ovlivnění geny, je pevně stanovena z původního návrhu šablony. Geny jsou vybírány z dvojice návrhů vždy ze sady genů pro shodné body.

Úvodní počet populace je stanoven na 20 návrhů šablon, jsou vytvořeny náhodným výběrem genů pro každý koncový bod, který obsahují. Je vytvořeno dalších 80 návrhů křížením genů náhodně vybraných dvojic. Úspěšnost návrhů je vyhodnocena detekcí průsečíků v obraze, u kterého je znám výsledek, který by měl být co nejpřibližněji dosažen, a návrhům je přiřazeno ohodnocení dle jejich výsledků. Následně jsou seřazeny podle úspěšnosti a ponecháno je 20 nejúspěšnějších. S nimi se celý proces padesátkrát opakuje a na závěr je vybrán návrh s nejlepším ohodnocením pro testovaná data.

S těmito hodnotami je genetický algoritmus prováděn ve verzi, která se nachází v příloze. Jedná se o velmi nízké hodnoty, které poskytnou výsledek za krátkou dobu. Přesto však i při těchto hodnotách lze dobře vysledovat trend zkvalitňování populace a výsledek, který je nalezen, je ve výrazné vetšině případů znatelně kvalitnější než původní návrh. Pro vygenerování šablony užívané v programu byly použity výrazně vyšší hodnoty, které však poskytly výsledek až po několika hodinách.



Obrázek 5.4: Graf kvality výsledků genetického algoritmu s 25 iteracemi

Vyhodnocování kvality šablony je prováděno jako porovnávání shody předpokládaného a reálného výstupu rozpoznání textu v obraze za použití dané šablony.

Zvyšování kvality populace i samotného výsledku genetického algoritmu při návrhu šablony je znázorněno v grafu na obrázku 5.4

Z provedených experimentů s genetickým algoritmem bylo vypozorováno následující:

- Růst kvality populace i nejvyššího ohodnocení je nejznatelnější v počátečních stádiích vývoje řešení.
- Čím je populace i nejlepší nalezené řešení kvalitnější, tím menší je pravděpodobnost dalšího zlepšení v následující generaci.
- Algoritmus je velmi účinný pro nalezení vhodného řešení pro novou situaci (například jiný typ písma).
- S vyššími hodnotami počtu návrhů v generaci, počtu potomků i počtu iterací se kvalita výsledného návrhu zvyšuje.

5.6 Určení znaku dle detekovaných průsečíků

Z informací, které jsme získali při detekci průsečíků, je třeba vyvodit, o který znak se jedná.

V návrhu této práce k určení znaku bylo vytvořeno ohodnocení průsečíků pro jednotlivé znaky. Pokud je pravděpodobné, že by se daný znak měl s danou úsečkou protnout, je ohodnocení kladné. Pokud je průsečík nepravděpodobný, je ohodnocení záporné. Pokud na daném průsečíku pro daný znak nezáleží, je ohodnocení nulové. Hodnota se dále liší dle toho, jak je daný průsečík pro znak charakteristický. Například pro znak T je charakteristické, že v levé a pravé spodní části znaku nejsou průsečíky detekovány. Naproti tomu v případě písmene W ve většině případů sice průsečíky v těchto oblastech také nalezeny nebudou, ale jedná se o méně výraznou vlastnost u znaku W, než u znaku T, takže i ohodnocení bude méně výrazné.

Při detekci znaku je pro každý možný znak vypočtena hodnota závislá na ohodnocení detekovaných průsečíků a následně je znak určen jako ten, jehož hodnota je nejvyšší.

Hodnoty pro jednotlivé úsečky a znaky byly určeny odhadem, následně byly několikrát upravovány tak, aby bylo rozpoznávání úspěšnější.

Navržený systém detekuje pouze velké a malé znaky anglické abecedy a číslice. To je však pro úspěšnou detekci spamu dostačující. Kombinace zužování algoritmem Zhang – Suen a detekce znaků s využitím šablony do značné míry ztěžuje detekci velmi úzkých znaků (například *i*, *j*, *l*, *1*, nebo *t*). Ostatní znaky jsou však po úpravě šablony pomocí genetického algoritmu rozpoznávány velmi dobře i při relativně malých velikostech textu.

Genetický algoritmus dokázal šablonu vhodným způsobem upravit nejen v případě rozdílné velikosti znaků, ale i v případě rozdílného typu písma. Příklad tohoto přizpůsobení je možno vidět v tabulce 5.1.

typ písma	Arial	Arial	Times New Roman
velikost (v bodech)	40	16	20
detekované znaky	46 z 53	44 z 53	41 z 53

Tabulka 5.1: Výsledky genetického algoritmu

Pro určení detekovaného znaku vyhodnocováním průsečíků se šablonou by bylo vhodné využít neuronových sítí. To však již v návrhu systému obsaženo není a jedná se tak o možnost, jak tento systém výrazně vylepšit.

Detekce mezer

Systém rovněž detekuje mezery mezi slovy. Mezery jsou určovány z velikosti rozestupů mezi oblastmi znaků. Pokud je tento rozestup větší, než stanovená hranice, předpokládá se, že mezi danou dvojicí znaků se nachází mezera. Zmíněná hranice je potom stanovena dle následujícího vzorce:

$$\frac{V \cdot 10}{23}$$

Kde V je výškou oblasti znaku, koeficienty 10 a 23 jsou výsledkem odhadu a úpravy po provedení experimentů tak, aby byla dosažena vysoká kvalita výsledku.

5.7 Shrnutí

Na závěr této kapitoly je uvedeno shrnutí navrženého systému a popis postupu rozpoznání obrazového spamu.

Po načtení je obrázek převeden do stupňů šedi a je provedeno adaptivní prahování s čtvercovým okolím o velikosti 7 x 7 pixelů. Tím je obraz převeden na pozadí a na objekty popředí, kterými jsou znaky textu a případné části obrázků. Pokud jsou v textu přítomny obrázky, systém se je pokusí odstranit výše popsanou, pro tento systém přímo navrženou, metodou pro odstranění obrazů z textu. Následně je provedeno odstranění šumu velikostním filtrem, kdy jsou odstraněny všechny spojité oblasti menší než 6 pixelů. Poté je provedeno zužování pomocí algoritmu Zhang – Suen. Po rozpoznání oblastí řádků a jednotlivých znaků využitím histogramu řádků, resp. sloupců je na každé rozpoznané oblasti znaku provedena detekce průsečíků se šablonou. Z detekovaných průsečíků a pole hodnot pro jednotlivé znaky

je určeno, o který znak se s největší pravděpodobností jedná. Zároveň se ověřuje vzdálenost oblastí po sobě následujících znaků pro detekci mezer v textu. Systém také umožňuje upravit šablonu úseček pomocí genetického algoritmu.

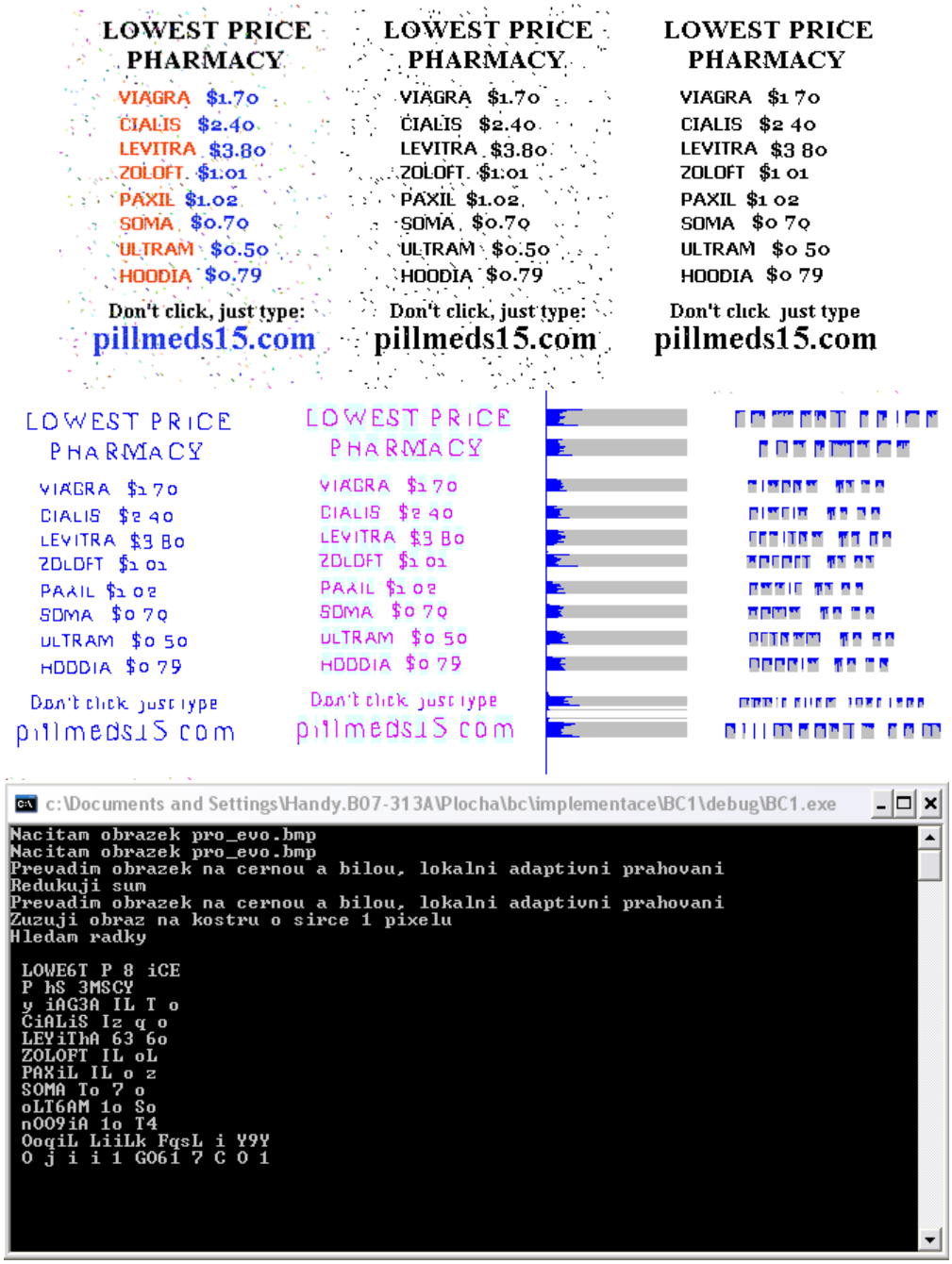
Příklad rozpoznání obrazového spamu

Na obrázku 5.5 je znázorněno zpracování obrazového spamu pomocí zmiňovaných metod. Postupně je zachycen obrazový spam v následujících fázích zpracování:

1. obrazový spam v původní podobě,
2. po převodu na dvě barvy pomocí lokálního prahování,
3. po redukci šumu velikostním filtrem,
4. po zúžení,
5. po rozpoznání oblastí řádků a znaků s využitím histogramů,
6. výstup programu – rozpoznaný text.

Pro větší přehlednost je také doplněn původní a rozpoznaný text:

LOWEST PRICE	LOWE6T P 8 iCE
PHARMACY	P hS 3MSCY
VIAGRA $1.7o	y iAG3A IL T o
CIALIS $2.4o	CiALiS Iz q o
LEVITRA $3.8o	LEYIThA 63 6o
ZOLOFT $1.o1	ZOLOFT IL oL
PAXIL $1.o2	PAXiL IL o z
SOMA $o.7o	SOMA To 7 o
ULTRAM $o.5o	oLT6AM 1o So
HOODIA $o.79	nOO9iA 1o T4
Don't click, just type:	OoqiL LiiLk FqsL i Y9Y
pillmeds15.com	O j i i 1 GO61 7 C O 1



Obrázek 5.5: Fáze zpracování obrazového spamu

Kapitola 6

Implementace

Program pro rozpoznání textu a zpracování obrazu s charakteristickými rysy obrazového spamu, který byl vytvořen jako součást této bakalářské práce, byl implementován v jazyce C++. Vývoj probíhal pod operačním systémem Microsoft Windows XP Professional v prostředí Microsof Visual Studio 2005, později Microsoft Visual Studio 2008.

Program využívá grafické knihovny OpenGL (Open Graphics Library) a doplňku GLUT (The OpenGL Utility Toolkit) k této grafické knihovně.

Jako základ programu byla zvolena kostra poskytnutá k projektům do předmětu IZG (základy počítačové grafiky). Autory jsou pan Ing. Igor Potůček, Ph.D. a pan Ing. Michal Seeman.

Program je spouštěn s jediným parametrem, kterým je název souboru, jenž má být zpracován. Jediným podporovaným formátem obrazu je formát BMP (Windows Bitmap), podpora více formátů nemá na samotné zpracování obrazu a detekci znaků vliv, proto nebyla implementována. Následně je možno najednou provést všechny operace vedoucí k úpravě obrazu a rozpoznání textu, nebo provádět tyto operace jednotlivě (ovládání viz tabulka 6.1).

klávesa	akce
e	spuštění genetického algoritmu[1]
a	provedení všech operací pro rozpoznání znaků
b	lokální prahování obrazu
p	lokalizace obrázků a jejich odstranění
n	redukce šumu velikostním filtrem
t	zúžení obrazu
r	nalezení oblastí znaků a jejich rozpoznání
Esc, x, q	ukončení programu

Tabulka 6.1: Ovládání programu

[1] Genetický algoritmus pracuje s obrazem evolution.bmp umístěným v adresáři programu. Obraz musí pro správnou funkčnost algoritmu obsahovat oddělený text uvedený níže a musí být upraven provedením operacemi prahování a zužování.
Text: AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz1234567890

Kapitola 7

Závěr

Cílem této práce bylo seznámit se s problematikou obrazového spamu, s metodami zpracování obrazu a s metodami rozpoznávání textu v obraze a následně navrhnout program zpracovávající obrazový spam.

Byly prostudovány rozmanité metody zpracování obrazu a následně při návrhu systému vybrány, případně přizpůsobeny pro lepší výsledky takové z nich, které se zdály být vhodnými.

Při testování zpracování obrazu a rozpoznávání znaků pomocí implementace návrhu byly prakticky získány důležité poznatky o prostudovaných metodách, které byly následně použity ke zlepšení výsledků poskytovaných systémem. Jako jeden příklad z mnoha je možné jmenovat situaci, kdy se v obrazovém spamu nalézal text i obrázek a metoda rozpoznání řádků selhala. Reakcí na tuto situaci bylo přidání metody pro detekci a odstranění obrázků.

Lokální prahování je vhodným způsobem převodu obrazového spamu do binární podoby, protože nedocházelo příliš často k chybnému určení bodů popředí či pozadí ani v obrazech s různou intenzitou v různých částech. Odstraňování šumu velikostním filtrem se osvědčilo jako vhodná metoda pro vyčištění záměrně poškozeného obrazu. Algoritmus Zhang – Suen pro zúžení obrazu i následná detekce průsečíků s použitím šablony modifikované genetickým algoritmem se ukázaly být dobrým zdrojem dat pro následné vyhodnocení znaků, ačkoliv by bylo třeba vyřešit detekci úzkých znaků a zachování vyšší míry detailů o podobě znaku po zužování. Způsob pro rozpoznání znaků byl pro potřeby práce dostačující, ale pro reálné použití by byla vhodnější detailnější modifikace například s využitím neuronových sítí.

Cíle práce byly tedy splněny, zároveň je dostatek možností dalšího studia zpracování obrazu a rozpoznávání znaků i vývoje systému tyto funkce implementujícího. Jedná se především o nalezení dalších způsobů detekce znaků. Ačkoliv se využití šablony ukázalo být dobrým zdrojem informací o podobě znaku, nutnost jejího přizpůsobování konkrétní situaci pomocí genetického algoritmu limituje všestrannost použití tohoto způsobu. Rovněž zmiňované neuronové sítě představují způsob, který by mohl vést k výraznému zlepšení dosažených výsledků.

Literatura

[1] Chemagazín: Umělé neuronové sítě – základy teorie a aplikace. [online], [cit. 2008-04-26].
URL http://www.chemagazin.cz/Texty/CHXVI_2_cl9.pdf

[2] Evett, D.: Spam Statistics 2006. [cit. 2007-12-16].
URL http://spam-filter-review.toptenreviews.com/spam-statistics.html

[3] Fikker, J.: Šum v digitální fotografii. [online], [cit. 2008-04-20].
URL http://www.fotoaparat.cz/article/7193/1

[4] Fisher, B.; Perkins, S.; Walker, A.; aj.: Mean Filter. [online], [cit. 2007-12-16].
URL http://www.cee.hw.ac.uk/hipr/html/mean.html

[5] Hlaváč, V.: Umělé neuronové sítě z pohledu rozpoznávání. [online], [cit. 2008-04-26].
URL http://cmp.felk.cvut.cz/~hlavac/TeachPresCz/31Rozp/72UmeleNN.ppt

[6] Jain, R.; Kasturi, R.; Schunck, B. G.: *Machine vision*. McGraw-Hill, 1995, iSBN 0-07-032018-7.

[7] Kelly, L.; Worboys, M.; Duckham, M.: *GIS: A Computing Perspective*. CRC Press, 2004, iSBN 0 415 28375 2.

[8] Lowe, F.: History Of Spam. [online], [cit. 2008-04-23].
URL http://www.mailmsg.com/SPAM_history.htm

[9] Luner, P.: Jemný úvod do genetických algoritmu. [online], [cit. 2008-04-25].
URL %http://cgg.ms.mff.cuni.cz/~pepca/prg022/luner.html

[10] Marshal: Spam Statistics. [online], [cit. 2008-04-26].
URL http://www.marshal.com/trace/spam_statistics.asp

[11] Martin, A.; Tosunoglu, S.: IMAGE PROCESSING TECHNIQUES FOR MACHINE VISION. [online], [cit. 2008-04-24].
URL http://www.eng.fiu.edu/mme/robotics/elib/am_st_fiu_ppr_2000.pdf

[12] Symnatec: The State of Spam, A Monthly Report – March 2008. [online], [cit. 2008-04-26].
URL http://eval.symantec.com/mktginfo/enterprise/other_resources/SpamReport_March08.pdf

[13] Veselovský, M.: Neuronové sítě. [online], [cit. 2008-04-26].
URL http://www.avari.cz/uir/index.php

[14] Wikipedia: Detekce hran. [online], [cit. 2008-05-02].
URL http://cs.wikipedia.org/wiki/Detekce_hran

[15] Wikipedia: Genetický algoritmus. [online], [cit. 2008-04-25].
URL http://cs.wikipedia.org/wiki/Genetick%C3%BD_algoritmus

[16] Wikipedia: Konvoluce. [online], [cit. 2008-05-02].
URL http://cs.wikipedia.org/wiki/Konvoluce

[17] Wikipedia: Medián. [cit. 2007-12-16].
URL http://cs.wikipedia.org/wiki/Medi%C3%A1n

[18] Wikipedia: Odstranění šumu. [online], [cit. 2008-04-20].
URL http://cs.wikipedia.org/wiki/Odstran%C4%9Bn%C3%AD_%C5%A1umu

[19] Wikipedia: Spam. [online], [cit. 2008-04-15].
URL http://cs.wikipedia.org/wiki/Spam