VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

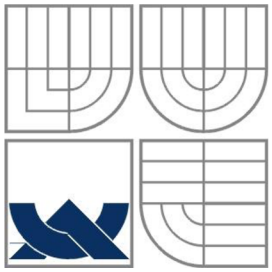
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

VYHODNOCOVÁNÍ ROZPOZNÁVÁNÍ OBRAZU

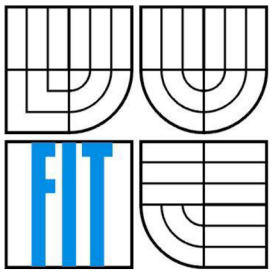
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE PAVLA KŮRKOVÁ
AUTHOR

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

VYHODNOCOVÁNÍ ROZPOZNÁVÁNÍ OBRAZU

EVALUATION OF IMAGE RECOGNITION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — PAVLA KŮRKOVÁ
AUTHOR

VEDOUCÍ PRÁCE — DOC. DR. ING. PAVEL ZEMČÍK
SUPERVISOR

BRNO 2010

Abstrakt

Práce se zabývá způsoby rozpoznávání obrazu a možnostmi jejich vyhodnocování. Součástí je návrh a implementace metody porovnání vlivu předzpracování na vybrané algoritmy rozpoznávání obrazu. Metoda byla testována a posouzena pro algoritmy lineární klasifikace, k-nearest neighbors, AdaBoost a SVM.

Abstract

This work deals with pattern recognition methods and possibilities of their evaluation. It involves design and implementation of the method comparing the influence of preprocessing to pattern recognition algorithms. The method was tested and evaluated for linear classification algorithm, k-Nearest Neighbors, AdaBoost and SVM.

Klíčová slova

rozpoznávání obrazu, statistické klasifikátory, AdaBoost, SVM, lineární klasifikátor, k-nearest neighbors

Keywords

image recognition, statistical classifiers, AdaBoost, SVM, linear classifier, k-Nearest Neighbors.

Citace

Kůrková Pavla: Vyhodnocování rozpoznávání obrazu, bakalářská práce, Brno, FIT VUT v Brně, 2010

Vyhodnocování rozpoznávání obrazu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením pana docenta Zemčíka.
Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

.....
Pavla Kůrková
18.5.2010

Poděkování

Poděkování patří panu vedoucímu za cenné rady k vytváření odborné práce, a dále pak rodině za podporu během studií.

Obsah

1 Úvod

Počítače dnes člověka doprovází na každém kroku, realizují náročné vědecké výpočty i pomáhají usnadnit každodenní činnosti, poskytují vzdělání stejně jako zábavu. Cílem moderní počítačové vědy je vytvářet autonomní systémy, které ještě více člověku usnadní či zautomatizují práci, rozšíří možnosti získávání informací nebo člověka zcela nahradí u nebezpečných a fyzicky náročných činností. Takovéto systémy musí nevyhnutelně reagovat na své okolí. Protože jedním z nejdůležitějších smyslů pro vnímání okolí je pro lidi zrak, i u umělých systémů hraje významnou roli zpracování a rozpoznání obrazu, tedy jakési vidění počítače. Snahou je vytvořit umělý systém, který bude ideálně napodobovat systém biologický – lidské vidění.

Zatímco pro člověka je vidění triviální záležitostí, u počítačů se jedná o náročnou a složitou činnost. Stroje totiž mají zcela odlišné možnosti získání a zpracování obrazové informace. Zatímco oko vnímá svět prostorově a v souvislostech a mozek informace zpracovává na základě předchozích znalostí, pro počítač je svět většinou reprezentován jako dvourozměrný obraz skutečnosti, ne vždy má k dispozici informace o objektech na obraze, a především jsou veškerá data zpracovávána pouze diskrétně. Napodobení našeho vidění se tak stává rozsáhlým a složitým problémem.

Oblast počítačového vidění se však neustále vyvíjí, zlepšují se jak hardwarové možnosti záznamu, ukládání a výkonu zpracovávajících strojů, tak jejich softwarové vybavení. Dnes již existuje velké množství nejrůznějších algoritmů pro rozpoznávání a ty jsou neustále zdokonalovány nebo nahrazovány novými. Narůstá rychlost zpracování, velikost pracovní paměti, nabízí se možnost zpracovávat data paralelně na více strojích. Inovace v oboru zpracování obrazu mohou být významným přínosem pro další obory, kde se automatická detekce obrazu využívá. Ilustračním příkladem může být automatické zpracování textu, identifikaci osob, vyhledávání objektů, automatizaci kontroly kvality výroby nebo posuzování snímků tkání v lékařství.

Posouzení kvality algoritmu je závislé na konkrétních požadavcích na jeho aplikaci – co všechno je potřeba z obrazu zjistit. Tato práce si klade za cíl vytvořit smysluplné porovnání vybraných algoritmů a jeho vyhodnocení. Je potřeba zvolit skupinu algoritmů a vhodný způsob srovnání, sady testovacích obrazů, provést testy a zpracovat získané výsledky.

Detailnější popis a souhrn dnešních možností rozpoznávání uvádí následující kapitoly 2 Rozpoznávání a 3 Analýza současného stavu. Návrh experimentů a teoretický rozbor očekávaných výsledků je popsán taktéž v kapitole 3. Praktickou implementací včetně popisu změn oproti původnímu teoretickému návrhu se zabývá další část práce, kapitola 4. Jsou zde popsány také výsledky získané během experimentů. V závěrečné 5-té kapitole je uveden souhrn dosažených výsledků, posouzení celé práce a několik návrhů na možná pokračování.

2 Rozpoznávání

Samotné rozpoznávání obrazu sestává z několika fází. Podle [1] se postup rozpoznání dělí do pěti kroků: snímání a digitalizace, předzpracování (preprocessing), segmentace obrazu, popis nalezených objektů a závěrečné porozumění obsahu. Postupně budou detailněji diskutovány některé důležité části jednotlivých fází, další pak budou pouze okrajově zmíněny pro ucelení přehledu. Komplexní rozbor je nad rámec této práce, v případě potřeby lze využít odbornou literaturu, viz reference na zdroje na konci práce.

2.1 Snímání a digitalizace

Snímaní a digitalizace se týkají samotného vytvoření obrazu. Podle [2] je snímání definováno takto: *„Snímání obrazu je převod optické veličiny na elektrický signál, který je spojitý v čase i úrovni“*. Pokud je snímán jas, bývá nejčastějším vstupním zařízením fotoaparát nebo kamera, případně scanner. Dále je možné snímat také jiné veličiny – teplotu povrchu, ultrazvuk nebo rentgenové záření. V těchto případech se pro snímání využívají specializovaná zařízení, například termokamery. Kvalita snímání závisí jednak na možnostech snímacího zařízení, dále na podmínkách okolí a vlastnostech snímaného objektu. Snímací zařízení ovlivňuje vlastnosti obrazu jako rozlišení, barevnou hloubku či ostrost. Vlivem okolí a objektu mohou potom být světelné podmínky, prašnost nebo odrazivost povrchu objektu.

Pokud je získaný vzorek analogový, je potřeba jej ještě digitalizovat – převést do podoby srozumitelné a zpracovatelné pro počítač. Digitalizace obrazu zahrnuje vzorkování do matice *MxN* bodů a kvantizaci analogového vstupu na *K* úrovní. Při těchto úpravách se ztrácí část vstupních informací, je tedy důležité vhodně zvolit celočíselné konstanty *M*, *N* a *K* s ohledem na požadovanou úroveň zachování detailů. Čím vyšší čísla zvolíme, tím více informací bude uchováno, ovšem vzroste také velikost vzorku a náročnost jeho zpracování.

Důležitým krokem digitalizace je volba intervalu vzorkování, tedy vzdálenosti dvou po sobě jdoucích vzorků. V praxi tuto otázku řeší Shannonova vzorkovací věta [3]:

Signál spojitý v čase je plně určen posloupností vzorků odebíraných ve stejných intervalech Δx, je-li jejich frekvence $f_s = 1/\Delta x$ větší nežli dvojnásobek nejvyšší frekvence v signálu f_{max}, tj. je-li

$$f_s > 2f_{max}.$$

Zjednodušeně lze říci, že zvolený interval vzorkování musí být menší nebo roven polovině rozměru nejmenšího detailu v obraze.

Další nedílnou součástí digitalizace je volba vzorkovací mřížky. Nejčastěji jsou používány mřížky čtvercové a hexagonální. Výhodou čtvercové mřížky je snadná realizovatelnost, většina

snímacích prvků vychází právě z takovéto mřížky. Výhoda hexagonální mřížky spočívá ve zjednodušeném určování vzdáleností a spojitosti objektů oproti předchozímu modelu. Hexagonální mřížka ovšem neumožňuje použití některých operací, například Fourierovy transformace, není proto vždy vhodná.

2.2 Preprocessing

Předzpracování zahrnuje nejrůznější úpravy a transformace vstupního obrazu. V závislosti na vstupním vzorku a požadované formě výstupu se zvolí, které budou použity v daném konkrétním případě. Hlavní cíle preprocessingu jsou snížení zkreslení způsobeného nevhodným snímáním, potlačení šumu vzniklého digitalizací a usnadnění pozdější analýzy obsahu obrazu a identifikace objektů zvýrazněním či potlačením některých rysů v obraze. Souhrnně lze cíle rozdělit do dvou skupin, vylepšení a restauraci obrazu. Podle [4] jsou tyto pojmy definovány následovně (volně přeloženo):

Operátory vylepšení zlepšují detekovatelnost důležitých detailů obrazu nebo objektů člověkem nebo strojem. Příklady operací zahrnují redukci šumu, vyhlazování, zvýšení kontrastu a detekce hran.

Restaurování obrazu se pokouší restaurovat poškozený obraz do ideálního stavu. Toto je možné pouze do určité úrovně v závislosti na tom, nakolik je znám a může být modelován proces vytváření ideálního obrazu a proces jeho degradace. Proces restaurace zahrnuje invertování procesu degradace pro návrat k ideálnímu obrazu.

Předzpracováním se nezíská z obrazu žádnou novou informaci, pouze některé budou zvýrazněny a jiné potlačeny. Transformace lze rozdělit podle [1] následovně:

1. bodové jasové transformace,
2. geometrické transformace,
3. lokální předzpracování, filtrace šumu, detekce hran,
4. obnovení obrazu při známé degradaci a
5. matematická morfologie.

První skupina zahrnuje jasové korekce a modifikace jasové stupnice. Jasovými korekcemi se rozumí úpravy jasu jednotlivých bodů v závislosti na jejich poloze v obraze. Toho se využívá při odstraňování poruch vzniklých u snímání, kdy snímací prvky nemají stejnou citlivost ve všech bodech obrazu, mění se množství světla propuštěného optickými prvky v závislosti na vzdálenosti od optické osy a osvětlení objektu je nerovnoměrné. Modifikace jasové stupnice na rozdíl od jasových korekcí nejsou závislé na umístění bodu v obraze. Je dán vztah pro transformaci T výchozí stupnice p na novou stupnici q [1]:

$$q = T(p).$$

Transformační vztah T může být libovolný, pro zvýšení kontrastu se ale nejčastěji používá metoda vyrovnání (ekvalizace) histogramu. Požadavky na výsledný histogram mohou být podle [4] dvojí, prvním cílem je vyrovnání hodnot histogramu tak, aby jednotlivé jasové úrovně byly zastoupeny zhruba stejně četně. Druhý způsob vyžaduje pouze to, aby byly využity všechny dostupné úrovně jasu, a nestanovuje už v jakém poměru.

Při snímání dochází ještě kromě šumu a jasových nerovnoměrností ke vzniku dalších zkreslení. Tato jsou způsobena snímáním neplošných objektů pod jiným než pravým úhlem optické osy snímače a snímané plochy. Souhrnně se hovoří o takzvaném geometrickém zkreslení. Pro jeho odstranění se využívají geometrické transformace. Podle [3] je geometrická transformace pro dvourozměrný obraz složený z bodů $[x, y]$ definována jako funkce:

$$T(u, v) = [x(u, v), y(u, v)],$$

kde souřadnice u a v jsou vstupy funkcí $x(u,v)$ a $y(u,v)$ a tyto funkce vracejí novou polohu bodu. Rozlišení původního a nového výstupního obrazu se může lišit. Je důležité si také uvědomit, že nové souřadnice jsou spojité, nikoliv diskrétní, a je nutné určit hodnoty jasu na celočíselných souřadnicích výstupního obrazu vhodnou interpolací. Nejjednodušší je interpolace nejbližším sousedem, existují však také složitější interpolace – lineární, bilineární či kubické [3]. Vhodnost zvolené interpolace závisí na aplikaci a požadavcích vzhledem k vstupnímu obrazu.

Třetí uvedená skupina využívá při výpočtech pouze lokální okolí zpracovávaného bodu, proto je nazvána lokální předzpracování. Náleží sem dvě obrazové úpravy, a to vyhlazování obrazu a gradientní operace. Vyhlazování potlačuje vyšší frekvence, odstraňuje tak náhodný šum, ale také jiné náhlé změny jasu, například hrany, což může být nežádoucí. Gradientní operace mají opačný efekt, zvýrazňují hrany v obraze, bohužel zvýrazní také šumové body. Některé algoritmy umožňují kombinaci obou úprav, tedy vyhlazování a ostření současně, příkladem může být algoritmus rotující masky [1].

Nejjednodušší metodou filtrace šumu pomocí okolí je obyčejné průměrování, kdy se hodnota bodu určuje jako průměr jeho okolí a maska filtru má tvar matice se samými jedničkami. Pro odstranění Gaussova šumu se používá upravená maska (zde pro okolí 3x3):

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}.$$

Aby byla zachována světlost obrazu, musí být součet hodnot v masce roven jedné. Celou matici je proto nutné vynásobit vhodným koeficientem, pro výše uvedenou je to koeficient 1/16. Ve frekvenční oblasti se průměrování chová jako nízkofrekvenční filtr [3].

Dalším způsobem vyhlazování je využití statistiky okolí pixelu, nejčastěji pomocí mediánu (Algoritmus 2-1) [3]. Tato metoda je vhodná pro odstranění bodového šumu, má ale negativní vliv na tenké čáry, což lze částečně obejít úpravou okolí na tvar kříže nebo na písmeno „X“.

Filtrování pomocí mediánu
Pro všechny pixely [i,j] v obrazu A 1: Načti body z intervalu [i-k, j-k][i+k, j+k] do pole M délky d = $(2k+1)^2$ 2: Seřaď pole M 3: Výstupní obraz B[i,j] = M[(d-1)/2]

Algoritmus 2-1 Filtrování pomocí mediánu

Gradientní operace jsou podobné vyhlazování, nový obraz *g(i,j)* vznikne součtem původního obrazu *f(i,j)* a funkce velikosti gradientu v jednotlivých bodech *s(i,j)* vynásobené koeficientem *c*:

$$g(i,j) = f(i,j) + c.s(i,j).$$

Gradient je hodnota derivace jasové funkce, v praxi se používají metody, které výpočet derivace aproximují konvolucí s vhodným jádrem (operátorem). Zdroj [9] uvádí jako nejjednodušší jádra (-1,1) a $(-1,1)^T$ nebo (-1,0,1) a $(-1,0,1)^T$. Všechny uvedené operátory používají dvě matice, kdy druhá je transponovanou verzí matice první. Jedna slouží pro detekci hran ve směru osy *x*, druhá potom detekuje hrany ve směru osy *y*. Obecně lze říci, že čím vyšší řád matice jádra má, tím odolnější je vůči šumu v obraze.

Používaným je Robertsův operátor, který určuje gradient ze zpracovávaného pixelu a jeho tří sousedních bodů [3]:

$$|\nabla f(i,j)| = |f(i,j) - f(i+1,j+1)| + |f(i,j+1) - f(i+1,j)|.$$

Pokročilejší Sobelův operátor je složen ze dvou komplementárních masek h a $\bar{h}$. Ty jsou vzájemně otočeny o 90° kolem středu a aplikují se dvakrát nezávisle na sobě. Výsledný obraz *G* je potom součtem dvou vzniklých obrazů [3]:

$$h = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \bar{h} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

$$|G| = \sqrt{h^2 + \bar{h}^2}$$

Dalším operátorem je Laplaceův operátor. Ten reaguje na hranu dvakrát – na nástupné a sestupné straně. Obecně může také vracet záporné hodnoty, v praxi se proto výsledek buď ořízne, posune do kladných hodnot nebo se použije absolutní hodnota. Maska pro 4-okolí resp. 8-okolí bodu má u Laplaceova operátoru tvar [3]:

$$h_4 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}, h_8 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Na Wikipedii [9] lze nalézt matice pro další známé operátory, například Prewittové operátor, Robinsonův nebo Kirschův operátor.

Na rozdíl od vyhlazování u ostření musí mít konvoluční matice součet nikoliv jedničkový, ale nulový, aby i odezva v oblastech s konstantní hodnotou byla nulová. Ve frekvenčním pásmu se ostření projeví jako opak filtrace, tedy jako vysokofrekvenční filtr.

Pokročilou detekci hran umožňuje Cannyho hranový detektor. Ten provádí detekci v několika krocích společně s dalšími úpravami obrazu. Podle [10] je postup zpracování následující:

1. Eliminace šumu Gaussovým filtrem
2. Určení gradientu (detekce hran operátorem)
3. Nalezení lokálních maxim (ztenčení)
4. Eliminace nevýznamných hran (prahování s hysterezí).

Uvedený postup je možné vícekrát opakovat, viz [5] – algoritmus 5.4: Canny edge detektor.

Pokud je znám charakter poruchy obrazu, je možné potlačit jeho poruchy pomocí obnovení (restaurování). Takovéto degradace mohou být způsobeny například pohybem snímače nebo snímaného objektu, vadami optické soustavy, špatným zaostřením nebo atmosférickými vlivy. Postupy obnovování se dělí do dvou skupin, podle vhodnosti použití pro zašuměné obrazy. První skupina není pro obrazy obsahující šum vhodná. Jedná se o takzvané deterministické obnovení, které využívá při výpočtu transformaci inverzní k transformaci degradační. Druhé, stochastické obnovení lze použít i pro obrazy s šumem. V tomto případě se pomocí stochastických modelů hledá nejvhodnější filtr pro obnovení obrazu [1].

Čím lépe je známa degradační transformace, tím lepších výsledků lze při obnovování dosáhnout. Často však informace o degradaci nejsou dostačující, potom se využívá modelování poruch. Parametry poruch zde buď jsou známy – pohyb kamery, vlastnosti čidla – a modelování určí detaily nebo se kompletně vše zjistí až analýzou degradovaného obrazu.

Poslední skupina uvedená ve výčtu, matematická morfologie, tvoří relativně samostatnou část analýzy. Celá úloha vyhodnocení obrazu je geometrizována, metody berou v úvahu tvar objektů a vychází z vlastností bodových. Využití těchto metod nemusí být pouze při předzpracování, jsou aplikovatelné také pro zdůraznění struktury objektu nebo popis objektů pomocí číselných charakteristik, vše jak pro binární, tak pro obrazy s více úrovněmi jasu. Jako příklad jsou uvedeny nejběžnější z těchto metod: dilatace, eroze, otevření, uzavření, sekvenční zesilování a ztenčování, vytvoření skeletu [1].

2.3 Segmentace

Segmentace obrazu bývá nejnáročnějším, ale také nejdůležitějším krokem celého rozpoznávání. Jejím úkolem je rozdělení obrazu do částí, které souvisí s objekty zachycenými na obraze. Pokud takto vzniklé segmenty jednoznačně odpovídají objektům vstupního obrazu a vzájemně se nepřekrývají, hovoří se o kompletní segmentaci. Tato vyžaduje buď vyšší úroveň zpracování se znalostmi řešeného problému, nebo speciální třídu vstupních obrazů, například kontrastní objekty na pozadí s neměnným

jasem. Jestliže vytvořené oblasti nekorespondují s objekty, ale jsou pouze homogenní vzhledem k určitým vlastnostem, jedná se o segmentaci částečnou. Výsledkem této segmentace je seznam jednotlivých, obecně vzájemně se překrývajících částí, které jsou potom zpracovány postupy vyšší úrovně.

Pro segmentaci lze využít různé vlastnosti obrazu. Metody pak podle [1] mohou být rozděleny do tří větších skupin:

- segmentace podle globálních vlastností,
- metody využívající určování hranic a
- postupy přímo vytvářející oblasti.

Základním druhem segmentace je prahování. Tato metoda je velmi jednoduchá, výpočetně nenáročná a dá se výhodně využít pro objekty s konstantním jasem povrchu na kontrastním pozadí. Jedná se o transformaci vstupního obrazu na výstupní binární obraz podle vztahu [1]:

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

kde T je předem určená hodnota prahu, $g(i,j) = 1$ pro elementy objektů a $g(i,j) = 0$ pro elementy pozadí. Základní algoritmus počítá s jedním prahem pro celý obraz, existují však také modifikace pracující s několika různými prahy pro různé oblasti obrazu nebo verze s více prahy, jejímž výsledkem potom není binární obraz, ale obraz s velmi malým počtem jasových úrovní. Volba vhodného prahu nebo prahů je stěžejní pro použitelný výsledek. Prahy je možno určovat manuálně nebo automaticky – pomocí hledání minim a maxim v histogramu nebo pomocí předchozích znalostí o objektu (kolik procent plochy objekt pokrývá, průměrná šířka čar u písma).

Literatura [5] uvádí speciální algoritmus pro optimální prahování – iterativní (optimální) výběr prahu (algoritmus 6.2, volně přeloženo):

Iterativní (optimální) výběr prahu

1: Za předpokladu nulových znalostí o přesné poloze objektů, uvažuj první aproximaci takovou, kdy čtyři rohy obrazu obsahují jen pozadí a zbytek obsahuje pixely objektu.

2: V kroku t, vypočti μ_B^t a μ_O^t jako průměr úrovně šedi pozadí a objektu, kde segmentace na pozadí a objekt v kroku t je definována prahovou hodnotou T^t určenou v předchozím kroku (rovnice 6.9)

$$\mu_B^t = \frac{\sum_{(i,j)\in \text{pozadi}} f(i,j)}{pocet\_pixelu\_pozadi}, \mu_O^t = \frac{\sum_{(i,j)\in \text{objekt}} f(i,j)}{pocet\_pixelu\_objektu}.$$

3: Nastav

$$\mu_B^t = \frac{\sum_{(i,j)\in \text{pozadi}} f(i,j)}{pocet\_pixelu\_pozadi}, \mu_O^t = \frac{\sum_{(i,j)\in \text{objekt}} f(i,j)}{pocet\_pixelu\_objektu}.$$

```
4: T(i+1) nyní obsahuje aktualizovanou hranici objekt-pozadí.
5: Pokud T(i+1) = Ti, skonči; jinak pokračuj na krok 2.
```

Algoritmus 2-2 Algoritmus iterativního výběru prahu

Další možností je segmentace pomocí detekce hranic. Jednotlivé hrany se určují již během předzpracování, viz předchozí kapitola. V průběhu segmentace jsou dříve nalezené hrany spojovány do řetězů popisujících hranici oblasti a jsou určeny vnitřní oblastí hranic.

Ve snímaném obraze se může nacházet více objektů a úkolem segmentace je potom jednoznačně oddělit a identifikovat tyto objekty. Jednoduchou metodou, která umožní takovéto rozdělení obrazu, je metoda barvení. Při barvení je každé z oblastí v obraze přiřazeno jednoznačné přirozené číslo. Opět je možné v literatuře nalézt mnoho rozličných algoritmů pro barvení, od rekurzivních postupů zpracovávajících celý obraz najednou, přes algoritmy pracující pouze s několika řádky, až po algoritmy určené k paralelnímu zpracování obrazu na několika strojích zároveň.

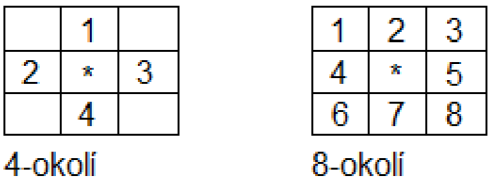
Zástupcem rekurzivních algoritmů je také algoritmus spojených komponent. Nechť je dán původní binární obraz **B** s **MaxRow** + 1 řádky a **MaxCol** + 1 sloupci. Průchodem algoritmu bude vytvořen nový obarvený obraz **LB**. Pixely obou obrazu mají souřadnice **L**, **P**. Algoritmus potom vypadá následovně [4]:

```
Algoritmus spojených komponent
1: procedure recursive_connected_components(B, LB);
2: {
3: LB := negate(B);
4: label := 0;
5: find_components(LB, label);
6: print(LB);
7: }
8:
9: procedure find_components(LB, label);
10:   {
11:   for L := 0 to MaxRow
12:         for P := 0 to MaxCol
13:               if LB[L,P] == -1 then
14:                     {
15:                     label := label + 1;
16:                     search(LB, label, L, P);
17:                     }
18:   }
19:
20:   procedure search(LB, label, L, P);
```

```
21:   {
22:   LB[L,P] := label;
23:   Nset := neighbors(L,P);
24:   for each [L', P'] in Nset
25:         {
26:         if LB[L', P'] == -1
27:         then search(LB, label, L', P');
28:         }
29:   }
```

Algoritmus 2-3 Algoritmus spojených komponent

Negace všech původních pixelů obrazu **B** (řádek 3) je nutná pro rozlišení bodů náležících oblasti s číslem 1 a dosud nezpracovaných bodů, které tak budou mít hodnotu -1. Procedura `neighbors()` vrací okolní body aktuálně zpracovávaného pixelu, podle potřeby se může jednat o 4-okolí nebo 8-okolí. Vraceny jsou pouze pixely patřící do obrazu v pořadí ilustrovaném na obrázku Obrázek 2-1.



Obrázek 2-1 4-okolí a 8-okolí bodu

V literatuře je uvedeno množství dalších pokročilejších způsobů segmentace, například pomocí aktivních kontur – snakes [5], mean-shift segmentace [5], [12] nebo detekce hran s využitím Houghovy transformace [4], [5].

2.4 Popis objektů

Objekty získané segmentací je nutné vhodným způsobem popsat, aby vzniklý exaktní popis mohl být předložen klasifikátoru. Samotná obrazová data bývají poměrně veliká a obsahují i velké množství nepotřebných informací. Část zbytečných dat je odstraněna během preprocessingu a segmentace, redukce na reálně použitelné informace se provádí při vytváření formálního popisu objektů v obraze. Formální popis je reprezentací znalostí o objektu, bez těchto znalostí není možné obrazu porozumět.

Termín znalosti úzce souvisí s umělou inteligencí, která je nedílnou součástí procesu rozpoznávání. Podle [5] lze znalosti rozdělit do dvou základních skupin, přičemž obě jsou potřebné pro klasifikaci objektu. První skupinou jsou informace týkající se samotného rozpoznávaného objektu, druhá skupina potom reprezentuje informace o třídách, do nichž mohou být objekty zařazovány.

Znalost musí být reprezentována libovolným, ale reprodukovatelným způsobem. Z tohoto důvodu jsou u reprezentace definovány dva základní pojmy: syntaxe a sémantika reprezentace. Slovně je lze popsat takto (volně přeloženo z [5]):

Syntaxe reprezentace specifikuje symboly, které mohou být použity, a způsoby, jak mohou být uspořádány.

Sémantika reprezentace specifikuje, jak je zakódován význam do symbolů a uspořádání symbolů povolených syntaxí.

Potom podle téhož zdroje reprezentace znalostí znamená následující:

Reprezentace je soubor syntaktických a sémantických konvencí, který umožňuje popisovat objekty.

Kniha [1] dělí popis znalostí na dvě základní skupiny – popis kvantitativní pomocí číselného vektoru příznaků a popis kvalitativní, využívající nečíselný syntaktický popis.

Kvantitativní popis

Prvním způsobem reprezentace znalostí je popis pomocí příznaků. Každý příznak $x_1, x_2, x_3, \ldots$ určuje jednu skalární vlastnost objektu, například velikost, kulatost nebo barvu. Jednotlivé příznaky potom mohou být sdruženy do vektoru příznaků $\boldsymbol{x}$. Jedná se o popis kvantitativní, jsou tedy reprezentovány pouze vlastnosti objektu, nikoliv jeho struktura. Příznaky jsou využívány u technik statistického rozpoznávání, viz kapitola 2.5.

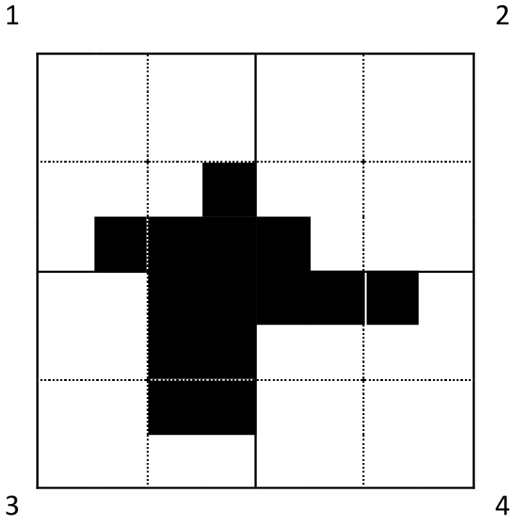
V [1] je zmíněna možnost popisu objektu pomocí tabulek vlastností. Jedná se o tabulku s počtem řádků odpovídajícím počtu rozpoznaných regionů a se sloupci odpovídajícími jednotlivým sledovaným vlastnostem. V podstatě se jedná také o příznakový popis, jeden řádek tabulky odpovídá jednomu vektoru příznaků, sloupce potom samotným příznakům. Takovouto tabulku je vhodné využít u obrazů, na kterých se vyskytuje více než jeden objekt.

Kvalitativní popis

Kvalitativních popisů existuje více, na rozdíl od kvantitativních reprezentací ale vždy kromě vlastností objektu zahrnují i jeho strukturu.

První, strukturální popis, obsahuje dvě části, jednu pro popisy objektů a druhou pro popisy tříd. Objekty jsou zde reprezentovány svými elementárními vlastnostmi, takzvanými primitivy, uspořádanými do určité struktury, která vyjadřuje relace mezi těmito primitivy. Příkladem mohou být segmenty hranice (primitiva) a jejich vzájemná návaznost (relace). Typické struktury pro vyjádření vztahů jsou stromy, grafy nebo řetězce. Třída je v strukturálním popise vyjádřena gramatikou nebo jazykem [5]. Ty definují, jakým způsobem lze z primitiv vytvářet struktury a jaké struktury a primitiva jsou přípustná.

Příkladem strukturálního popisu jsou kvadrantové stromy. Využívá se zde jednoduché stromové struktury, kde každý uzel reprezentuje čtverec o straně *a* bodů. Jednotlivé uzly mohou mít buď nula, nebo právě čtyři potomky odpovídající jednotlivým kvadrantům čtverce. Hodnoty uzlů jsou *full* (F) pro čtverec zcela pokrytý objektem, *empty* (E) pro čtverec obsahující pouze pozadí a *mixed* (M) pro smíšený kvadrant, který se potom rozvětví do další úrovně stromu [4]. Ukázka binárního obrazu a odpovídajícího kvadrantového stromu je na obrázcích Obrázek 2-2 a Obrázek 2-3.

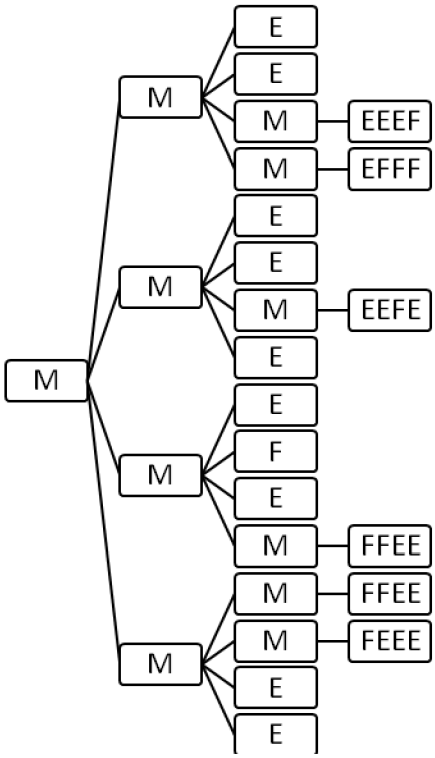


Obrázek 2-2 Ilustrační binární obraz

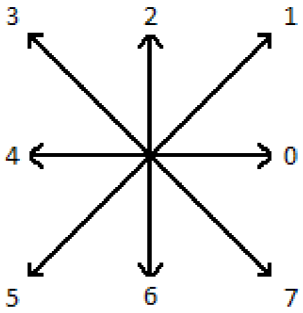
Dalším způsobem popisu pomocí datové struktury je popis hran. Nejjednodušší je prostý lineární seznam hraničních bodů segmentu. Variantou je popis Freemanovým řetězovým kódem [4], který nevyužívá souřadnice bodů, ale směr posunu po pravoúhlé mřížce. Tato reprezentace je paměťově úspornější než předcházející. Popis Freemanovým kódem ilustrují obrázky Obrázek 2-5 a Obrázek 2-3, počáteční bod hranice je označen *X*, kód odpovídající hranici na obrázku Obrázek 2-5 je potom *07710766554333222*. Další možností je aproximace hranice pomocí polygonu, která je méně náročná na velikost paměti a umožňuje následné jednodušší zpracování.

Podle [5] lze kvalitativní popis vyjádřit také predikátovou logikou. Jedná se matematický popis, na který lze použít matematickou dedukci a získat tak nové informace o objektu. Nevýhodou této metody je fakt, že nedokáže pracovat s neúplnými či nepřesnými daty.

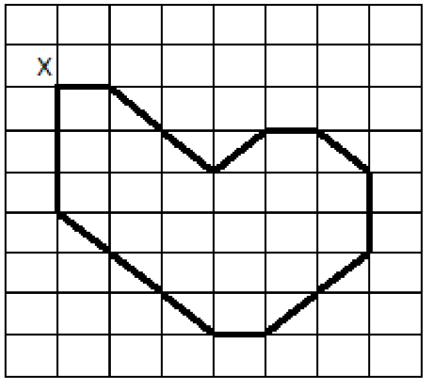
V literatuře [5] jsou zmíněny i další možnosti popisu objektů, pomocí odvozovacích pravidel ve tvaru „*IF podmínka THEN akce*“ a příslušné databáze znalostí, popisem pomocí rámů či s využitím fuzzy logiky. Dále zde jsou uvedeny sémantické sítě složené ze seznamu objektů, jejich popisu a popisu vztahů těchto objektů. Tyto metody zde nebudou detailněji rozebírány.



Obrázek 2-3 Kvadrantový strom



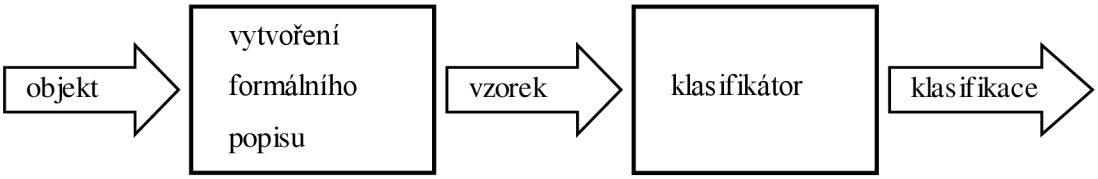
Obrázek 2-4 Freemanův řetězový kód



Obrázek 2-5 Ilustrační hranice segmentu

2.5 Porozumění obsahu

V závěrečné fázi rozpoznávání se pracuje s popisem objektů získaným v předchozím kroku. Objekty jsou na základě tohoto popisu klasifikovány do tříd – do skupin, které mají shodné vlastnosti. Podle jakých vlastností budou třídy vytvořeny, závisí na účelu aplikace. Algoritmus, který přiřazuje objektům třídu, je nazýván klasifikátorem. Typický postup rozpoznání objektu je na ilustraci Obrázek 2-6. Klasifikátory jsou typicky deterministické, jeden a týž objekt přiřadí vždy do téže třídy. Způsob rozdělování je určen ve fázi učení, kdy je klasifikátoru předloženo několik objektů s explicitně určenou třídou a tento podle nich vytvoří vzory jednotlivých tříd.



Obrázek 2-6 Hlavní kroky rozpoznávání

Statistické klasifikátory

První skupina klasifikátorů jsou klasifikátory statistické. Tyto využívají číselný popis objektů, příznaky, případně sdružené do příznakových vektorů $\boldsymbol{x} = (x_1, x_2, x_3, \ldots)$. Kombinace všech možných příznakových vektorů $\boldsymbol{x}$ tvoří příznakový prostor $\boldsymbol{X}$. Pokud jsou příznaky vhodně zvoleny, nachází se objekty jedné třídy v prostoru $\boldsymbol{X}$ blízko sebe a oblasti s objekty jednotlivých tříd lze oddělit diskriminační křivkou.

Pokud diskriminační křivka odděluje jednotlivé třídy dokonale, tedy neexistuje žádný objekt třídy *A*, který by se nacházel v oblasti třídy *B*, jedná se o separovatelné třídy. Klasifikace do separovatelných tříd je bezchybná, bohužel v praxi se takovéto případy vyskytují zřídka. Ukázka separovatelných tříd a diskriminačních funkcí je na obrázku Obrázek 2-7.

Statistický klasifikátor má typicky *n* vstupů, kde *n* odpovídá počtu příznaků, a jeden výstup, identifikátor třídy ω_r. Podle počtu výstupních tříd *R* se pak jedná o klasifikátor *R*-té třídy. Vztah mezi vstupem a výstupem klasifikátoru je dán rozhodovacím pravidlem [5]:

$$d(x) = \omega_r.$$

Vytvoření vhodného rozhodovacího pravidla je cílem celého návrhu klasifikátoru.

Jednotlivé podmnožiny K_r příznakového prostoru jsou odděleny pomocí *R* skalárních funkcí $g_1(\boldsymbol{x})$, $g_2(\boldsymbol{x})$, …, $g_r(\boldsymbol{x})$, zvaných diskriminační funkce. Dvě diskriminační funkce nesmí vracet stejnou hodnotu pro stejný vstup, musí splňovat podmínku:

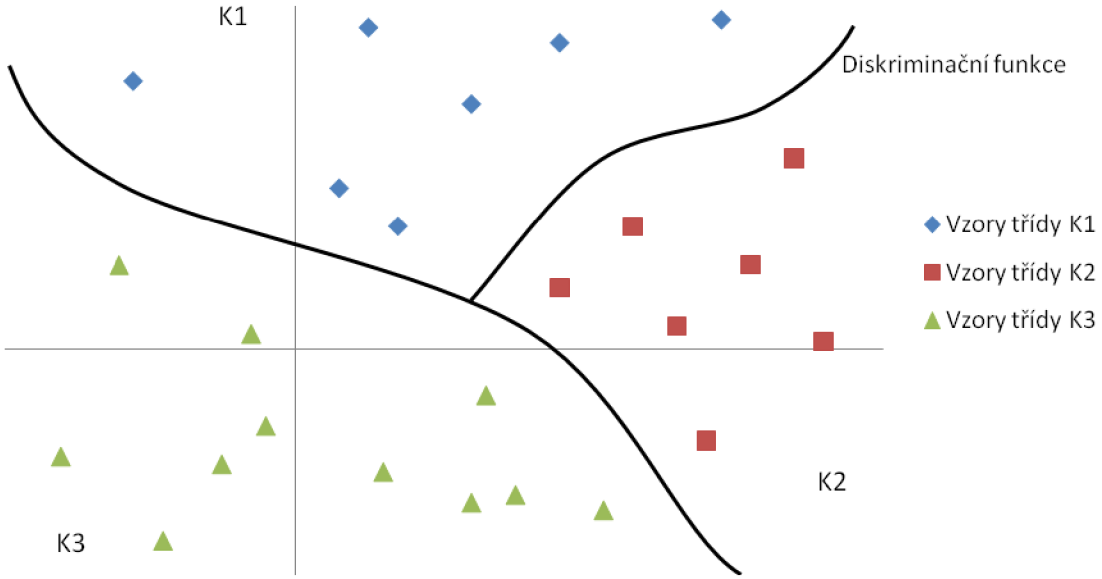
$$\forall \boldsymbol{x} \in K_r;\ s, r \in \{1, \ldots, R\};\ s \neq r \text{ platí } g_r(\boldsymbol{x}) \geq g_s(\boldsymbol{x})$$

Výjimku tvoří hranice oblastí, kde platí $g_a(\boldsymbol{x}) = g_b(\boldsymbol{x})$. Pokud jsou všechny distribuční funkce lineární, tedy ve tvaru:

$$g_r(x) = q_{r0} + q_{r1}x_1 + \ldots + q_{rn}x_n,$$

jedná se o lineární klasifikátor. Na základě diskriminačních funkcí lze vytvořit rozhodovací pravidlo, které objekt přiřadí do té třídy, jejíž diskriminační funkce vrátí nejvyšší hodnotu:

$$d(x) = \omega_r \Leftrightarrow g_r(x) = \max_{s=1,\ldots,R} g_s(x)$$



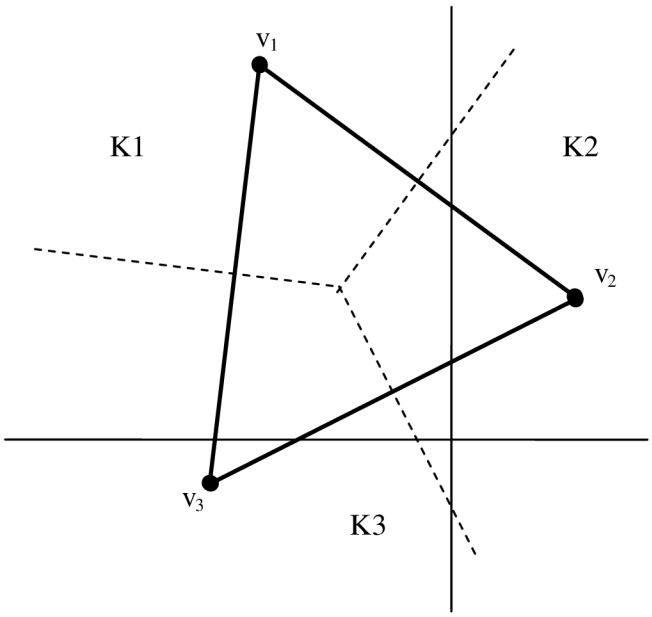
Obrázek 2-7 Příznakový prostor separovatelných tříd

Jiným způsobem vytvoření klasifikátoru je využití principu minimální vzdálenosti [5]. Každá třída má předem dané vzory v_1, v_2, …, v_R. Při klasifikaci se potom určí vzdálenost přiřazovaného objektu od všech těchto vzorů a vybere se nejmenší ze získaných hodnot. Rozhodovací pravidlo má tvar:

$$d(\boldsymbol{x}) = \omega_r \Leftrightarrow |v_r - \boldsymbol{x}| = \min_{s=1,\ldots,R} |v_s - \boldsymbol{x}|$$

Vzniklé diskriminační funkce jsou kolmé na spojnici příslušných vzorů a tuto spojnici půlí. Ukázku příznakového prostoru rozděleného klasifikátorem minimální vzdálenosti ilustruje Obrázek 2-8.

Pokud je každá třída reprezentována právě jedním vzorem, výsledný klasifikátor je lineární. V případě, že je dáno více vzorů na třídu, vznikne klasifikátor po částech lineární [5].



Obrázek 2-8 Klasifikátor minimální vzdálenosti

Klasifikátor s využitím minimální vzdálenosti je speciálním případem obecnějšího algoritmu *k* nejbližších sousedů – *k*-Nearest Neighbors, *k*-NN [4], [6]. Ten pracuje na stejném principu, jen bere v úvahu *k* nejbližších sousedů klasifikovaného objektu a objekt přiřadí do té třídy, do níž patří nejvíce z množiny sousedních objektů.

Jestliže diskriminační funkce nesplňují podmínku linearity, jedná se o nelineární klasifikátor. V takovém případě je běžným postupem transformace původního příznakového prostoru X^n na nový X^m, ve kterém již lze použít lineární klasifikátor. Transformační funkce je označována **Φ**, klasifikátor potom jako **Φ**-klasifikátor. Diskriminační funkce takovéhoto klasifikátoru má tvar:

$$(\boldsymbol{x}) = q_{r0} + q_{r1}\phi_1(\boldsymbol{x}) + \ldots + q_{rm}\phi_m(\boldsymbol{x}).$$

Příkladem použití jsou SVM (support vector machine) klasifikátory, kde transformační funkce **Φ** tvoří takzvanou jádrovou funkci. Transformační funkce převádí vzorky do nového prostoru, SVM klasifikátor potom hledá funkci, která vzorky v novém prostoru rozdělí s co největším okrajem [5].

Syntaktické klasifikátory

Druhou poměrně velkou skupinou jsou syntaktické klasifikátory. Na rozdíl od statistických využívají kvalitativní popis objektů. Typický postup vytváření takovéhoto klasifikátoru je podle [5]:

1. učení: podle problému se určí vhodná primitiva a možné vztahy
2. vytvoření popisující gramatiky pro každou z tříd

3. rozpoznávání: získání a rozpoznání primitiv objektu, popis jejich vztahů, vytvoření popisu objektu (popisného slova)
4. klasifikace předmětu do třídy, jejíž gramatikou lze vytvořit popisné slovo (pomocí syntaktické analýzy popisného slova).

Další klasifikátory

Oblast umělé inteligence nabízí ještě mnohé další způsoby vytváření klasifikátorů a klasifikace. Pro ilustraci se může jednat o neuronové sítě, metody srovnávání grafů nebo fuzzy systémy. Kromě toho je také možné využít nejrůznější způsoby optimalizace funkcí použitých u předchozích skupin klasifikátorů. Více informací lze nalézt v [5].

Boosting

Kniha [5] uvádí, že jen velmi zřídka jediný klasifikátor vyřeší problém kompletně. Z tohoto důvodu se používá metoda takzvaného boostingu, tedy kombinace několika slabých klasifikátorů do jednoho silného. Velmi rozšířeným algoritmem kombinování klasifikátorů je adaptive boosting algoritmus, zkráceně AdaBoost.

Při boostingu musí být jasně definováno, jak se rozdělí trénovací množina mezi jednotlivé slabé klasifikátory a jakým způsobem budou kombinována slabá pravidla pro získání výsledného silného [5]. Často se využívá postupné klasifikace. Trénovací množinu klasifikují postupně slabé klasifikátory a zvyšuje se váha vzorků, které předchozí klasifikátor označil chybně. Váhové koeficienty se objevují i při vytváření silného pravidla. Čím menší chybu produkuje pravidlo slabého klasifikátoru, tím větší váhu ve výsledku dostane. Tento princip je použit i u zmíněné metody AdaBoost,. Podle [8] zde váhové koeficienty nabývají hodnot od mínus nekonečna pro dokonalého lháře po plus nekonečno pro bezchybný klasifikátor.

3 Analýza současného stavu

Počítačové vidění je netriviální záležitostí z několika důvodů. Prvním je reprezentace 3D světa jako 2D obrazu. Při této transformaci dochází přirozeně ke ztrátě velkého množství informací, například o umístění a velikosti objektů. Malý objekt blízko snímače se tak počítači jeví jako stejný s velkým, ale vzdáleným objektem. Další komplikace se objevují u interpretace obrazu. K té jsou potřeba předchozí znalosti, stejný obraz může mít v jiné souvislosti zcela jiný význam, tenká linie může značit buněčnou membránu na mikrosnímku rostlinné tkáně stejně jako satelitní snímek řeky. Při interpretaci je proto potřeba brát souvislost v úvahu a systému ji explicitně sdělit. Problém s porozuměním nastává také z důvodu omezeného lokálního pohledu. Počítačový systém pracuje s obrazem po částech, nejčastěji s konkrétním bodem a jeho nejbližším okolím. Nemá tak žádné informace o globálním kontextu obrazu.

Komplikace s sebou přináší také zkreslení vznikající při snímání. Zašumění obrazu znesnadňuje nalezení důležitých objektů a geometrické zkreslení pozmění tvar, který potom neodpovídá modelu. Rozlišná světelnost obrazu způsobená různou intenzitou osvětlení, odrazivosti objektu a jeho pozice vzhledem ke snímači je problémová u vyhledávání totožných objektů na více různých snímcích.

V neposlední řadě komplikuje zpracování velké množství dat obsažených ve snímku. Jednak je potřeba více místa pro ukládání, dále není možné takový objem dat nahrát do paměti nebo vždy zpracovávat v reálném čase.

V průběhu zpracování se zvyšuje úroveň abstrakce obrazu, nejníže jsou nezpracovaná obrazová data, nejvýše potom stojí samotný interpret. Podle této úrovně lze samotné zpracování rozdělit na nízkoúrovňové a vysokoúrovňové. První neuvažuje žádné další informace o obraze kromě obrazové funkce samotné. Řadí se sem například ostření, filtrace a komprese obrazu, zvýrazňování hran a další postupy související se snímáním, preprocessingem a segmentací. Vstupem je digitalizovaný obraz, typicky reprezentovaný jako funkce jasu $f(x,y)$ se dvěma parametry – souřadnicemi bodu x a y. Oblast nízkoúrovňového zpracování je starší, dnes převládá snaha zefektivnit a zobecnit známé postupy, nikoliv vyvíjet nové. To je potřeba především kvůli nárůstu kvality a tím i velikosti zpracovávaných obrazů. Díky technickému pokroku je možné využít i další způsoby urychlení algoritmů, například paralelizaci. Stále nedořešeným problémem zůstává automatizace volby vhodného pořadí aplikace operátorů.

Vysokoúrovňové zpracování je oproti předchozímu založeno na znalostech problému a cílů řešení, snahou je vhodně popsat a následně porozumět obsahu obrazu. Velmi často se zde využívá umělá inteligence. Systém má k dispozici formální model světa a porovnává obraz s tímto modelem. Ne vždy je výsledkem úplná shoda obrazu a modelu, potom se použije shoda částečná, případně úpravy obrazu pomocí nižší úrovně zpracování, čímž lze získat nové informace a upravit model.

Vysokoúrovňové zpracování je poměrně mladou oblastí, stále se tak objevují nové přístupy a postupy. Bohužel zatím stále jediným zcela funkčním systémem rozpoznávání je samotný člověk.

Nízkoúrovňové zpracování ovlivňuje nezanedbatelně vyšší úroveň rozpoznávání, následující část práce se proto bude věnovat posuzování tohoto vlivu. Cílem bude navrhnout, provést a vyhodnotit sérii experimentů, kdy budou klasifikátorům předkládány skupiny obrazů po různém preprocessingu. Mělo by tak být odhaleno, nakolik se bude lišit chybovost téhož klasifikátoru při různých zpracováních vstupních obrazů, a dále porovnání chybovosti různých klasifikátorů při klasifikaci téže skupiny vstupů.

Velmi často se v praxi pouze slepě použije způsob vyhodnocování, který byl někde označen jako nejlepší. Může se však velmi snadno stát, že pro konkrétní úlohu příslušné řešení nebude vůbec vhodné. Bezmyšlenkovité použití některého z algoritmů se mi nejeví jako ideální, prozkoumat možnosti výběru nejvhodnějšího postupu je pro mne způsobem, jak se takovémuto slepému výběru v případě potřeby vyhnout. Dalším důvodem, proč považuji experimentování s preprocessingem za zajímavé, je fakt, že vyhodnocení vlivu předzpracování může mít překvapivé výsledky a úplně tak převrátit obecné vnímání kvality algoritmů.

Požadavky na metodiku jsou jednoduché, měla by klasifikovat nejrůzněji předzpracované obrazy a srovnávat výsledky pro různé klasifikátory. Velké množství implementací algoritmů rozpoznávání je dostupných na internetu zdarma pro výzkumné účely, nemá proto smysl vytvářet vlastní implementace. Práce s obrazem není v běžných programovacích jazycích nijak zvlášť podporována. To je důvodem, proč bude lepší využít některý z dostupných matematických programů, které obsahují mnoho zabudovaných funkcí pro zpracování obrazů.

3.1 Porovnání klasifikátorů nad reálnými daty

Srovnání algoritmů je často prováděno nad uměle vygenerovanou množinou příznaků a následně vyhodnoceno pomocí nejrůznějších výpočtů chyb. Navrhovaná metodika se zaměří na robustnost klasifikátorů vůči předzpracování vstupního obrazu, nebude tak pracovat s generovanými daty, ale se souborem příznaků získaných z reálných obrazů. Navrhovaný postup porovnání algoritmů sestává z kroků přibližně odpovídajícím běžnému zpracování obrazu: obecné předzpracování, porovnávací předzpracování, segmentace, extrakce příznaků a vytvoření datového setu a klasifikace. Nyní detailněji k jednotlivým fázím.

Jak bylo zmíněno, trénovací i testovací data budou vytvářena přímo z obrazů – fotografií. Z tohoto důvodu je nutné nejprve provést obecný preprocessing všech vstupů, v tomto případě se bude jednat o zmenšení na zadanou velikost a převod na šedotónový obraz. Obraz není automaticky ořezáván, změna velikosti je proto neproporcionální. Důvodem tohoto kroku je fakt, že objekt v obraze nemá pevně stanovenou polohu a při ořezu by mohlo dojít k odstranění části objektu. Vzniklá chyba by byla horší než zkreslení změnou proporcí objektu. Úpravy obecného

předzpracování zjednoduší a urychlí následné další zpracování. Tato fáze bude pro všechna srovnání stejná a bude zahrnuta vždy. Na výsledek by tak neměla mít žádný vliv.

Další kroky předzpracování jsou pro navrženou metodu porovnání stěžejní. Tato fáze se bude měnit a ovlivňovat tak výsledek. Celkově je navrženo sedm různých verzí předzpracování a to: žádné, přidání Gaussova šumu do obrazu, přidání Gaussova šumu a následné odstranění filtrací s maskou uvedenou v kapitole 2.2, detekce hran Sobelovým a Robertsovým operátorem, Cannyho detektor hran a ekvalizace histogramu.

Při segmentaci se využije fakt, že všechny vstupní obrazy obsahují pouze jediný objekt určený ke klasifikaci. Nebude tak potřeba provádět odlišení více objektů v jednom snímku, ale pouze oddělení objektu a pozadí. Segmentace bude dvojí, v závislosti na předzpracování. Pokud bude zkoumán vliv preprocessingu, segmentace bude obsahovat pouze jednu úpravu – iterativní prahování s výběrem optimálního prahu podle algoritmu Algoritmus 2-2. V druhém případě nebude použito žádné předzpracování, během segmentace pak se provede trojí prahování původního obrazu s různými hodnotami pevného prahu.

Nalezené objekty budou popsány pomocí momentů oblastí, viz [5], kapitola 8.3.2. Pro každý objekt tak vznikne popisný vektor momentů, který lze použít jako příznakový vektor pro statistický klasifikátor.

Klasifikace získaných dat proběhne vícekrát s rozdílnými klasifikátory. Vždy se budou detekovat vzorky dvou tříd, některé algoritmy totiž nelze použít pro klasifikaci více tříd. Pro srovnání byly vybrány tyto statistické klasifikátory: lineární klasifikátor, algoritmus *k* nejbližších sousedů, SVM a AdaBoost.

Na závěr budou jednotlivé klasifikátory posouzeny pomocí počtu korektně klasifikovaných objektů jednotlivých tříd z testovací množiny. Dále bude měřena doba trvání výpočtu klasifikace. Pro srovnání bude provedena klasifikace a stejný výpočet chyby i na množině generovaných dat s Riplyho rozložením.

3.2 Předpokládané výsledky

Jednotlivé klasifikátory se liší svou složitostí, nejjednodušší je klasifikátor lineární, nejpokročilejší potom AdaBoost. Lze tedy předpokládat, že pokročilejší algoritmy budou dosahovat lepších výsledků pro všechny testované množiny. Na druhou stranu doba výpočtu složitého algoritmu bude s vysokou pravděpodobností delší než pro výpočet jednoduchý.

Předzpracování bude mít také nezanedbatelný vliv na výsledek. Nezpracovaný obraz nemusí být pro segmentaci ideální, měl by však vykazovat lepší výsledky než vzorky se zašuměním. Určitá míra zkreslení je obsažena i v originálních obrazech, proti uměle přidávanému šumu ji však lze považovat za zanedbatelnou. U snímků s přidaným a neodstraněným šumem by výsledky měly být velmi špatné, odstranění šumu by mělo vykazovat lepší hodnoty.

Detekce hran by měla zlepšit dosažené výsledky oproti nezpracovanému obrazu pro všechny použité metody. Mezi jednotlivými operátory mohou vzniknout rozdíly, předpokladem je nejlepší výsledek pro nejsložitější Cannyho detektor a nejhorší pro Robertsův operátor, který je z uvedených postupů nejjednodušší.

Vliv ekvalizace histogramu závisí na druhu vstupních vzorků. Pokud je zpracováván obraz s vysokým kontrastem mezi objektem a pozadím, ekvalizace výsledky spíše zhorší, protože sníží tento rozdíl výhodný pro prahování. Pro obrazy s malým kontrastem bude efekt opačný, výsledky by tak měly být lepší.

Poslední zkoumaný vliv je hodnota prahu. Vektor příznaků je vytvářen z binárního obrazu, kde nula značí pozadí a jednička objekt, jako popis oblasti objektu. Je běžným jevem, že jako součást objektu je označena i část pozadí resp. naopak, některé oblasti objektu jsou označeny jako pozadí. Pokud bude obraz prahován s nevhodným prahem, mohou být nekorektně označené oblasti poměrně rozsáhlé a výrazně tak ovlivní oblast považovanou za objekt. Zkreslení výsledného popisu oblasti bude v tomto případě asi největší a lze tak očekávat i nejhorší výsledky klasifikace.

4 Implementace a testování

Implementace samotných algoritmů rozpoznávání nebyla součástí zadání, pro implementaci a experimenty proto bylo využito matematického programu MATLAB®, toolboxu Image Processing Toolbox (dále IPT) a několika toolkitů volně dostupných pro výzkumné účely. Konkrétně se jedná o následující moduly:

1. STPRTool; autoři Franc V. a Hlaváč V., zdrojové kódy a licenční podmínky dostupné online na http://cmp.felk.cvut.cz/cmp/software/stprtool/
2. CMPVia Toolkit (implementace funkcí ke knize [6]), autoři Svoboda T., Kybic J., Hlaváč V., dostupné online na http://visionbook.felk.cvut.cz/

Pro rozpoznávání bylo potřeba vytvořit sérii obrázků. Použité fotografie pochází z galerií Sunipix®, dostupné online na http://www.sunipix.com/. Licenční podmínky jsou k dispozici tamtéž. Seznam použitých snímků je uveden v příloze 2. Počet vzorků získaných přímo z uvedeného webu nebyl dostatečný, každá fotografie proto byla ještě upravena otočením, čímž se počet zdvojnásobil. Celkově bylo použito 17 vzorků pro trénování a 37 testovacích. Jak bylo uvedeno výše, bude se provádět binární klasifikace, vzorové obrazy tak obsahují vždy prvky dvou tříd, v tomto případě se jedná o jablka a květy růží.

Jednotlivé části byly implementovány v MATLABu, celkem se jedná o osm souborů s funkcemi a jeden vykonávací skript. Všechny uvedené soubory jsou k dispozici na DVD (Příloha 1).

4.1 Implementované funkce

První funkcí je implementace obecného preprocessingu v souboru `prep.m`. Využívají se zde základní funkce pro práci s maticemi. Zmenšení obrázku je neproporcionální pomocí funkce `imresize` z Image Processing Toolboxu. Převod na úrovně šedi odpovídá standardu NTSC, který uvažuje citlivost lidského oka na jednotlivé barvy RGB obrazu [13]:

$$I = .2989*rgb\_img(:,:,1) + .5870*rgb\_img(:,:,2) + .1140*rgb\_img(:,:,3).$$

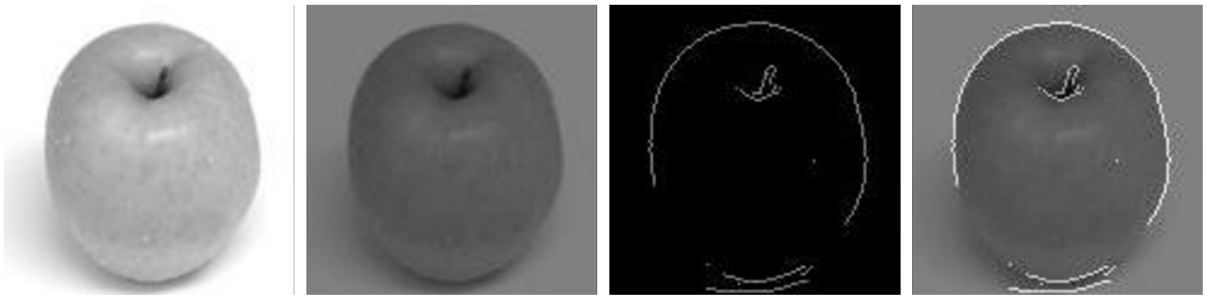
Pro ekvalizaci byl zvolen pokročilejší algoritmus `hist_equal` z knihy [5] (Algoritmus 5.1 Ekvalizace histogramu) implementovaný CMPVia Toolkitem. Původní záměr použít funkci `histeq` z Image Processing Toolboxu nebyl uskutečněn z důvodu nekorektního chování funkce při práci v konzoli bez grafického rozhraní. Funkci obsahuje soubor `histogram.m`.

Detekce hran využívá algoritmy implementované v Image Processing Toolbox, které vrací binární obraz hran (hodnoty 0) a pozadí (hodnoty 1). Protože se k popisu využívá popis oblastí nikoliv hranic, je potřeba získané hrany zpátky zkombinovat s původním obrazem tak, aby hrany byly zvýrazněny, ale obraz obsahoval celou oblast. Kombinace je prováděna v několika krocích, nejprve se

hodnoty původního obrazu zmenší z rozsahu <0, 255> na polovinu <0,128>. Binární obraz s detekovanými hranami se převede na obraz s hodnotami 0 a 127. Tyto dva obrazy se následně sečtou. Ukázka vstupních a výsledného obrazu pro Sobelův operátor je na obrázku Obrázek 4-1. Implementace detekcí hran je v souboru `edges.m`.

Zašumění a následné jednoduché odstranění šumu provádí funkce `noise`. Nejprve je do obrázku přidán Gaussův šum (funkce `imnoise` z IPT) a následně je odstraněn filtrováním (funkce `imfilter` z téhož toolboxu).

Prahování je rozděleno do dvou funkcí, `itThresh.m` a `fixThresh.m`. První provádí prahování s iterativním výběrem prahu – funkce `imthresh` z CMPVia Toolkitu. Druhá implementuje prahování s pevným prahem, hodnoty prahu se posouvají po čtvrtině rozsahu hodnot v obrázku. Nejprve se určí minimální a maximální hodnota obsažená ve vzorku, jejich rozdíl se vydělí čtyřmi a o toto číslo se postupně navyšuje použitý práh. Provede se tak vlastně rovnoměrné rozdělení rozsahu na čtvrtiny.



Obrázek 4-1 Originál, upravený obraz, nalezené hranice a výsledný obraz prahování

Funkce `segm` zpracovává vstupní binární obrazy na popisné vektory pomocí funkce `regiondescr` obsažené ve CMPVia Toolkitu. Algoritmus výpočtu je uveden v [6], odpovídající teoretický popis v [5]. Získaný vektor má sedm položek popisujících oblast objektu. Tento vektor je spojen s odpovídajícím vektorem rozdělení obrazů do tříd a je vytvořena datová struktura, která odpovídá požadavkům na vstupní data klasifikátorů implementovaných v STPRToolkitu [14].

Poslední fází je samotná klasifikace. Opět bylo využito funkcí implementovaných v toolkitu, tentokrát v STPRToolkitu. Protože se do těchto funkcí nijak nezasahuje ani není potřeba speciálních úprav před voláním, jsou klasifikátory obsaženy pouze v ukázkovém skriptu `main.m` bez další zaobalující funkce. Ve skriptu `main.m` jsou použity následující klasifikátory a jejich nastavení:

- lineární klasifikátor – Fisherův lineární diskriminant,
- *k*-nearest neighbors – výpočet pro 3 sousedy,
- AdaBoost – maximální počet pravidel 50, slabé klasifikátory z funkce `weaklearner` (součást STPRToolkitu) a
- SVM – sekvenční minimální optimizátor, funkce `smo` (STPRToolkit), jádro RBF (Gaussovské) [14].

V případě AdaBoostu musela být explicitně nastavena cesta, odpovídající algoritmus obsahují oba používané toolkity a nutně tak dojde k zastínění jedné z funkcí. Použitá verze patří do STPRToolkitu. Ostatní algoritmy se vyskytují vždy jen jednou a obdobné problémy tak nenastanou.

Pro snadnější ukládání obrázků přímo z třírozměrných matic, se kterými program pracuje, byla napsána pomocná funkce `writeIm`. Ta postupně zapisuje do souborů typu *jpeg* obrázky z matice, název obsahuje jméno zadané parametrem a automaticky generované pořadové číslo obrázku.

4.2 Skript s testy

Automatickou ukázku obsahuje skript `main.m`, který postupně volá funkce preprocessingu, prahování, segmentace a klasifikace. Součástí skriptu je i nastavení cest pro toolkity, předpokládá se jejich umístění ve stejném adresáři jako skript `main`, ve složkách pojmenovaných *stprtool* a *visionbook*. Vstupní obrázky z adresářů *samples/trn* a *samples/tst* jsou postupně zpracovávány výše uvedenými funkcemi. Velikost vzorků pro preprocessing byla zvolena 120*120 pixelů s ohledem na původní velikost a poměr stran obrazů. Výsledkem skriptu je osm matic, první čtyři obsahují chyby jednotlivých klasifikátorů pro srovnávané vstupy, druhé čtyři potom dobu běhu jednotlivých výpočtů.

Chyby jsou určovány funkcí `cerror` (STPRTool) ze známého vektoru tříd *ytrue* a vektoru tříd vráceného klasifikátorem *ypred*. Jedná se o tři hodnoty – chybu klasifikace pro první třídu, pro druhou třídu a celkovou chybu – určené podle vzorců uvedených v [11]. Matice pro jednotlivé klasifikátory mají názvy *linRes*, *knnRes*, *adaRes* a *svmRes* a pořadí úprav vstupu je následující: bez preprocesingu, Robertsův operátor, Sobelův operátor, Cannyho detektor, přidání šumu, odstranění šumu, ekvalizace histogramu, fixní prahování práh 1, práh 2, práh 3 a Riplyho dataset (součást STPRTool).

Doba běhu klasifikace obsahuje čas vytvoření, učení a následného použití klasifikátoru a je určena MATLABovskými funkcemi tic a toc. Výsledek je uložen v proměnných *linTime*, *knnTime*, *adaTime* a *svmTime*.

Matice s obrazy zabírají relativně velký prostor v paměti. Z důvodu šetření paměti a zajištění, že nedojde k jejímu vyčerpání v průběhu zpracování, obsahuje skript několik příkazů `clear`, které průběžně mažou nepotřebné proměnné. Po ukončení skriptu tak je k dispozici pouze 8 výše uvedených matic.

Během tetování skriptu `main.m` docházelo k pádu aplikace MATLAB z důvodu nedostatku prostředků. Tento pád způsoboval výpočet lineárního klasifikátoru pro Riplyho datový set. Důvodem bylo zřejmě spouštění MATLABu vzdáleně ze školního serveru Merlin v konzolovém okně, kde je omezen počet spustitelných vláken a alokovatelné paměti. Ve výsledném skriptu je proto tato část zakomentována jako "*% ----- nefunkcni pro konzoli --------*„. Další problémy způsoboval výpočet SVM klasifikace pro Riplyho set, který trval neúměrně dlouho a je možné, že docházelo k zacyklení. Proto byla i tato část z konečné verze vypuštěna a je uvedena pouze v komentáři.

4.3 Dosažené výsledky

Testování probíhalo na školním serveru Merlin, kde je naistalován MATLAB verze 7.6.0.324 (R2008a) včetně potřebného toolboxu IPT. Rozpoznávání bylo provedeno vzdáleným spuštěním MATLABu v konzolovém okně a následně skriptu `main.m` v adresáři obsahujícím potřebné složky s obrázky (*/samples/trn* a */samples/tst*), adresáře s toolkity (*/visionbook*, */stprtool*) a soubory s funkcemi. Výsledné hodnoty chyb vrácené v proměnných pro jednotlivé klasifikátory udávají tabulky v příloze Příloha 3 Výsledky klasifikace algoritmů.

Výpočet byl proveden několikrát pro tutéž množinu dat, klasifikátory jsou však deterministické a vždy vrátily stejné výsledky pro stejnou množinu dat. Odlišně se choval pouze SVM klasifikátor, ale toto může být dáno odchylkami vzniklými při výpočtu jádrové funkce.

Výsledky lineárního klasifikátoru mají mezi sebou relativně malou odchylku, rozdílný preprocessing má na klasifikátor pouze malý vliv. Největší rozdíly chybovosti se vyskytovaly u SVM. U nevhodného zpracování takřka není schopen klasifikace, což demonstruje klasifikace bez preprocessingu. S vhodnou úpravou naopak označil všechny objekty správně.

Poměrně překvapivé je, že pokročilé klasifikátory nedosáhly vždy výrazně lepších hodnot, v některých případech byly dokonce horší. Tento fakt potvrzuje domněnku, že nízkoúrovňové zpracování může značně ovlivnit výsledky rozpoznávání.

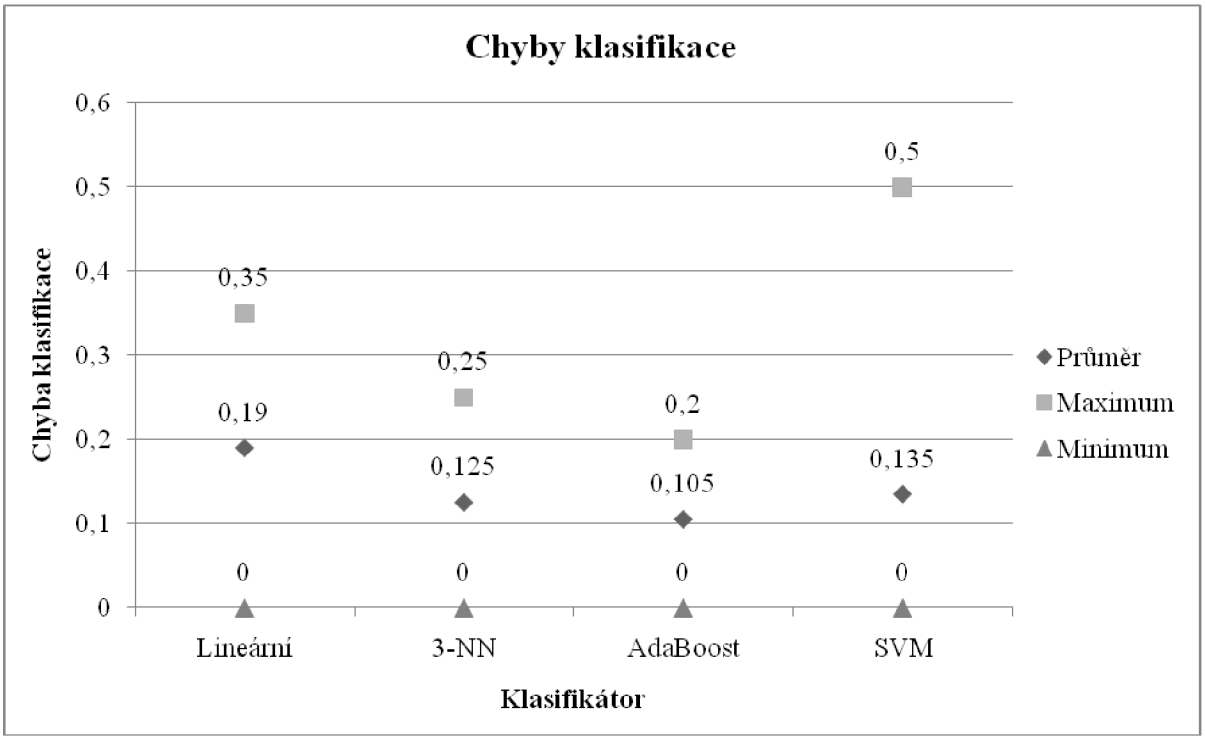
Jako nejvhodnější předzpracování lze označit ekvalizaci histogramu. Takto upravené vzorky byly klasifikovány bezchybně pro tři ze čtyř klasifikátorů, v posledním případě byla chyba jen velmi malá. Nejhorší výsledky již nelze tak generalizovat, liší se pro jednotlivé klasifikátory. Očekávaně však nedopadlo nejlépe prahování s pevným prahem.

Cannyho detektor je označován jako jeden z nejlepších detektorů hran. Výsledky provedené klasifikace ovšem ukazují spíše opak. Tento nesoulad bude pravděpodobně dán způsobem zapracování hran zpět do obrazu a využitím popisu pomocí momentů oblastí. Pokud by se pracovalo pouze se získanými hranami a objekty byly popisovány hranicemi, výsledky by se lišily.

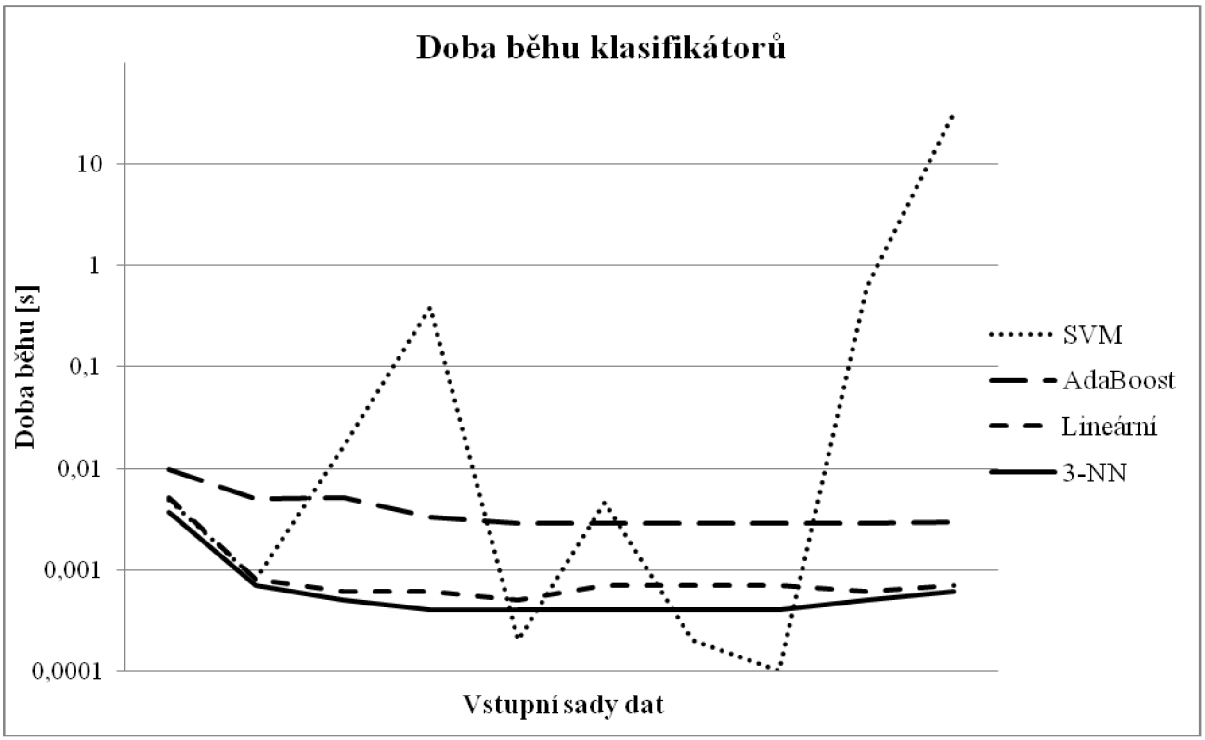
Statistiku hodnot dosažených chyb pro dané klasifikátory ilustruje Obrázek 4-2 Výsledné chyby klasifikace. U prvních tří klasifikátorů jsou výsledky předpokládané, nejlépe dopadl AdaBoost, nejhůře jednoduchý lineární klasifikátor. SVM dopadlo nečekaně špatně, silný vliv zde měla neschopnost přiřazení tříd u nezpracovaného vstupu.

Co se doby běhu algoritmů týče, dosažené výsledky (viz Příloha 4 Doba běhu algoritmů) odpovídají původnímu předpokladu, že výpočet složitých algoritmů bude pomalejší než výpočet jednoduchých. Ve zmíněné příloze je jasně vidět, že lineární klasifikátor a 3-NN algoritmus jsou rychlejší než AdaBoost nebo nejsložitější SVM. Získané výsledky však jsou ovlivněny dalšími souběžnými procesy, mezi jednotlivými pokusy se čísla lišila. U všech výsledků však byla patrná výše zmíněná tendence rychlejšího zpracování primitivnějších algoritmů. Dobu běhu algoritmu ilustruje Obrázek 4-3 Doba běhu klasifikátorů.

Fakt, že doba výpočtu pro Riplyho set je proti ostatním hodnotám stejného algoritmu mnohonásobná, je dán tím, že použitý Riplyho set obsahoval celkem 500 vzorů, zatímco druhá sada pouhých 54. Protože Riplyho set nebyl testován pro všechny algoritmy, není součástí obrázku doby běhu.



Obrázek 4-2 Výsledné chyby klasifikace



Obrázek 4-3 Doba běhu klasifikátorů

5 Závěr

Práce se zabývala tématem rozpoznávání vzorů v obraze. Hlavním cílem bylo zpracovat metodiku pro vyhodnocení vybraných známých algoritmů rozpoznání obrazu. Pro navrženou metodu se dále měly navrhnout a provést experimenty. Posledním krokem bylo vyhodnotit výsledky získané během provádění navržených experimentů. Všechny tyto požadované kroky byly splněny.

Po prostudování odborné literatury a analýze získaných obecných poznatků o rozpoznávání obrazu byla navržena metodika porovnání vlivu nízkoúrovňového zpracování obrazu na klasifikaci. Tato metoda zkoumá, jaké dopady má kvalita předzpracování na úspěšnost klasifikace objektů do tříd. S využitím dostupného softwaru byla metoda implementována a výsledný program byl testován na souboru reálných fotografií. Celkově proběhlo několik experimentů s různými postupy předzpracování. Vyhodnocení těchto pokusů se provádělo pro čtyři klasifikátory – lineární, algoritmus 3 nejbližších sousedů, AdaBoost a support vector machine.

Dosažené výsledky potvrzují, že předzpracování obrazu má nezanedbatelný dopad na následné rozpoznávání a nevhodný způsob úprav dokáže citelně snížit kvalitu výsledků i u pokročilých algoritmů. Jednodušší metody naopak vykazují větší robustnost vůči rozdílnému preprocessingu. Provést srovnání s uměle generovanými daty se nepovedlo, důvodem byly omezené možnosti použitého softwaru.

Navržená metoda porovnání nad reálnými daty by se dala využít v případech, kdy je potřeba zvolit nejvhodnější zpracování známé množiny obrazů pro určený klasifikátor. Taktéž by bylo možné její nasazení na opačné případy, kdy je hledán ideální algoritmus třídění pro již upravená data.

Nabízí se několik možností pokračování v tématu práce. Prvním je rozšíření testované množiny klasifikátorů, druhým přidání dalších způsobů předzpracování nebo nové kombinace stávajících. Další vhodnou činností je provedení obdobných testů pro jiné, rozsáhlejší množiny obrazů.

Literatura:

[1] Hlaváč, V., Šonka, M.: *Počítačové vidění*. Praha: Grada, 1992. Edice Nestůjte za dveřmi. 272 stran. ISBN 80-85424-67-3.

[2] Holota, R., Fiřt, J.: *Digitalizace a zpracování obrazu* [online]. Poslední aktualizace 2002-09-30 [cit. 2010-03-15]. Dostupné z: http://home.zcu.cz/~holota5/publ/DigZprO.pdf.

[3] Žára, J., Beneš, B., Sochor, J., Felkel, P.: *Moderní počítačová grafika*. 2. vydání. Brno: Computer Press, 2005. ISBN: 80-251-0454-0.

[4] Shapiro, L. G., Stockman G. C.: *Computer vision*. New Jersey: Prentice-Hall, 2001. 580 stran. ISBN 0-13-030796-3.

[5] Sonka, M., Hlavac, V., Boyle, R.: *Image processing, analysis, and machine vision*. Toronto: Thomson Learning, 2008. 829 stran. ISBN-13 978-0-495-08252-1

[6] Svoboda, T., Kybic, J., Hlavac, V.: *Image processing, analysis, and machine vision: A MATLAB companion*. Toronto: Thomson Learning, 2008. 255 stran. ISBN-13 978-0-495-29595-2

[7] Stork, D. G., Yom-Tov, E.: *Computer manual in MATLAB to accompany pattern classification*. Second edition. New Jersey: John Wiley & Sons, 2004. ISBN 0-471-42977-5.

[8] Raúl Rojas: *AdaBoost and the Super Bowl of Classifiers* [online]. Poslední aktualizace 2009-12 [cit. 2010-05-12]. Dostupné z: http://www.inf.fu-berlin.de/inst/ag-ki/adaboost4.pdf.

[9] *Detekce hran* [online]. Poslední aktualizace 2009-12-28 [cit. 2010-05-10], Wikipedie. Dostupné z: http://cs.wikipedia.org/wiki/Detekce_hran.

[10] *Cannyho hranový detektor* [online]. Poslední aktualizace 2009-10-06 [cit. 2010-05-10], Wikipedie. Dostupné z: http://cs.wikipedia.org/wiki/Cannyho_hranov%C3%BD_detektor.

[11] Franc, V., Hlaváč, V.: *Statistical Pattern Recognition Toolbox for Matlab: User's guide* [online]. Praha: Czech Technical University, 2004. ISSN 1213-2365. Dostupné z: http://cmp.felk.cvut.cz/cmp/software/stprtool/stprtool.pdf.

[12] Doubek, P.: *Mean-shift segmentace* [online]. Poslední aktualizace 2007-10-29 [cit. 2010-05-12] Doubek, online: http://cmp.felk.cvut.cz/cmp/courses/ZS1/Cviceni/cv4/meanshift.pdf.

[13] *MATLAB documentation* [online]. Poslední aktualizace 2010 [cit. 2010-05-16]. Dostupné z: http://www.mathworks.com/access/helpdesk/help/techdoc/.

[14] Franc, V., Hlavac, V.: *Statistical Pattern Recognition Toolbox Manual* [online]. Poslední aktualizace 2007-10-22 [cit. 2010-05-16]. Dostupné z: http://cmp.felk.cvut.cz/cmp/software/stprtool/manual/index.html.

Seznam příloh

Seznam obrázků

Seznam algoritmů

Příloha 2

Seznam obrázků použitých pro klasifikaci

Zdroj: http://www.sunipix.com/

Jablka:

1. http://www.sunipix.com/fruitsvegetables/Apple-1.jpg (Image ID: 44-012)
2. http://www.sunipix.com/fruitsvegetables/Apple-3.jpg (Image ID: 44-014)
3. http://www.sunipix.com/food/Granny%20Smith%20Apple-1.jpg (Image ID:44-007)
4. http://www.sunipix.com/fruitsvegetables/Granny%20Smith%20Apple-2.jpg (Image ID:44 034)
5. http://www.sunipix.com/fruitsveg/Gala%20Apple-10.jpg (Image ID:44-192)
6. http://www.sunipix.com/fruitsveg/Gala%20Apple-11.jpg (Image ID:44-193)
7. http://www.sunipix.com/fruitsveg/Gala%20Apple-13.jpg (Image ID:44-195)
8. http://www.sunipix.com/fruitsveg/Gala%20Apple-4.jpg (Image ID:44-187)
9. http://www.sunipix.com/fruitsveg/Golden%20Delicious%20Apple-14.jpg (Image ID:44-198)
10. http://www.sunipix.com/fruitsveg/Golden%20Delicious%20Apple-29.jpg (Image ID:44-201)
11. http://www.sunipix.com/fruitsveg/Golden%20Delicious%20Apple-27.jpg (Image ID:44-205)
12. http://www.sunipix.com/fruitsveg/Golden%20Delicious%20Apple-41.jpg (Image ID:44-210)

Růže:

1. http://www.sunipix.com/Rose/Rose-18.jpg (Image ID:47-004)
2. http://www.sunipix.com/Rose/Rose-20.jpg (Image ID:47-005)
3. http://www.sunipix.com/Rose/Rose-10.jpg (Image ID:47-006)
4. http://www.sunipix.com/Rose/Rose-27.jpg (Image ID:47-018)
5. http://www.sunipix.com/Rose/Rose-29.jpg (Image ID:47-020)
6. http://www.sunipix.com/Rose/Rose-5.jpg (Image ID:47-021)
7. http://www.sunipix.com/Rose/Rose-6.jpg (Image ID:47-022)
8. http://www.sunipix.com/Rose/Rose-8.jpg (Image ID:47-024)
9. http://www.sunipix.com/Rose/Rose-23.jpg (Image ID:47-030)
10. http://www.sunipix.com/Rose/Rose-106.jpg (Image ID:47-037)
11. http://www.sunipix.com/Rose/Rose-109.jpg (Image ID:47-040)
12. http://www.sunipix.com/Rose/Rose-102.jpg (Image ID:47-033)
13. http://www.sunipix.com/Rose/Rose-110.jpg (Image ID:47-046)
14. http://www.sunipix.com/Rose/Rose-115.jpg (Image ID:47-056)
15. http://www.sunipix.com/Rose/Rose-117.jpg (Image ID:47-058)

16. http://www.sunipix.com/Rose/Rose-169.jpg (Image ID:47-098)
17. http://www.sunipix.com/Rose/Rose-168.jpg (Image ID:47-097)
18. http://www.sunipix.com/Rose/Rose-40.jpg (Image ID:47-161)
19. http://www.sunipix.com/Rose/Rose-36.jpg (Image ID:47-162)
20. http://www.sunipix.com/Rose/Rose-38.jpg (Image ID:47-164)
21. http://www.sunipix.com/Rose/Rose-39.jpg (Image ID:47-165)
22. http://www.sunipix.com/Rose/Rose-1.jpg (Image ID:47-166)
23. http://www.sunipix.com/Rose/Rose-2.jpg (Image ID:47-167)
24. http://www.sunipix.com/Rose/Rose-48.jpg (Image ID:47-154)
25. http://www.sunipix.com/Rose/Rose-42.jpg (Image ID:47-157)
26. http://www.sunipix.com/Rose/Rose-43.jpg (Image ID:47-158)
27. http://www.sunipix.com/Rose/Rose-56.jpg (Image ID:47-144)
28. http://www.sunipix.com/Rose/Rose-57.jpg (Image ID:47-145)
29. http://www.sunipix.com/Rose/Rose-70.jpg (Image ID:47-135)
30. http://www.sunipix.com/Rose/Rose-89.jpg (Image ID:47-115)

Příloha 3

Výsledky klasifikace algoritmů

Výsledky lineárního klasifikátoru				
Bez preprocesingu	Robertsův operátor	Sobelův operátor	Cannyho detektor	Přidání šumu
0.1500	0.1500	0.2000	0.2500	0.1500
0	0.1176	0.1176	0.2353	0.1176
0.0811	0.1351	0.1622	0.2432	0.1351
Odstranění šumu	Ekvalizace histogramu	Prahování, práh 1	Prahování, práh 2	Prahování, práh 3
0.2000	0	0.3500	0.2000	0.2500
0.1176	0.2941	0	0	0
0.1622	0.1351	0.1892	0.1081	0.1351

Výsledky algoritmu 3-NN					
Bez preprocesingu	Robertsův operátor	Sobelův operátor	Cannyho detektor	Přidání šumu	
0	0.1000	0.1000	0.2500	0.1000	
0.0588	0	0	0.1176	0	
0.0270	0.0541	0.0541	0.1892	0.0541	
Odstranění šumu	Ekvalizace histogramu	Prahování, práh 1	Prahování, práh 2	Prahování, práh 3	Riplyho set
0.1000	0	0.1500	0.2000	0.2500	0.1020
0	0	0	0.1765	0	0.1660
0.0541	0	0.0811	0.1892	0.1351	0.1340

Výsledky algoritmu AdaBoost					
Bez preprocesingu	Robertsův operátor	Sobelův operátor	Cannyho detektor	Přidání šumu	
0.0500	0.1000	0.1500	0.1500	0.1000	
0	0	0.2353	0.2353	0	
0.0270	0.0541	0.1892	0.1892	0.0541	
Odstranění šumu	Ekvalizace histogramu	Prahování, práh 1	Prahování, práh 2	Prahování, práh 3	Riplyho set
0.1500	0	0.0500	0.1000	0.2000	0.0920
0.2353	0	0	0	0	0.1120
0.1892	0	0.0270	0.0541	0.1081	0.1020

Výsledky SVM klasifikátoru				
Bez preprocesingu	Robertsův operátor	Sobelův operátor	Cannyho detektor	Přidání šumu
0	0.1000	0	0.1000	0.1000
1.0000	0	0.1176	0.2353	0
0.4595	0.0541	0.0541	0.1622	0.0541
Odstranění šumu	Ekvalizace histogramu	Prahování, práh 1	Prahování, práh 2	Prahování, práh 3
0	0	0.5000	0.0500	0.5000
0.1176	0	0	0.4706	0.1176
0.0541	0	0.0270	0.2432	0.0811

Příloha 4

Doba běhu algoritmů

Klasifikátor	Bez preprocesingu	Robertsův operátor	Sobelův operátor	Cannyho detektor	Přidání šumu	Odstranění šumu
SVM	0.0049	0.0008	0.0163	0.3882	0.0002	0.0046
AdaBoost	0.0098	0.0051	0.0052	0.0033	0.0029	0.0029
Lineární	0.0052	0.0008	0.0006	0.0006	0.0005	0.0007
3-NN	0.0037	0.0007	0.0005	0.0004	0.0004	0.0004
Klasifikátor	Ekvalizace histogramu	Prahování, práh 1	Prahování, práh 2	Prahování, práh 3	Riplyho set	Průměr
SVM	0.0002	1.3855	0.6286	31.9869	-	3.3031
AdaBoost	0.0029	0.0029	0.0029	0.0030	0.3770	0.0041
Lineární	0.0007	0.0007	0.0006	0.0007	-	0.0011
3-NN	0.0004	0.0004	0.0005	0.0006	0.0096	0.0008