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

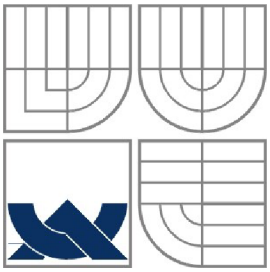
TEXTURNÍ PŘÍZNAKY

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

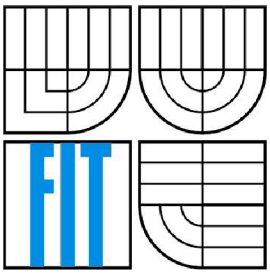
AUTOR PRÁCE
AUTHOR

ROMAN ZAHRADNIK

BRNO 2007



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

TEXTURNÍ PŘÍZNAKY
TEXTURE FEATURES

DIPLOMOVÁ PRÁCE
MASTER‘S THESIS

AUTOR PRÁCE
AUTHOR

ROMAN ZAHRADNIK

VEDOUCÍ PRÁCE
SUPERVISOR

ING. MIROSLAV ŠVUB

BRNO 2007

Abstrakt

Cílem tohoto projektu je zhodnotit účinnost využití texturních příznaků při rozpoznávání a klasifikaci textur v počítačovém zpracování obrazu. Stěžejním úkolem práce je porovnat a diskutovat experimentálně získané výsledky a efektivitu jejich dosažení použitím texturních příznaků implementovaných metodou lokálních binárních vzorů v konfrontaci s výsledky docílenými s využitím matic sousednosti při klasifikaci shlukovou analýzou.

Klíčová slova

Textury, texturní příznaky, klasifikace textur, klasifikátor, lokální binární vzory, matice sousednosti, histogram, pixel.

Abstract

Aim of this project is to evaluate effectivity of various texture features within the context of image processing, particulary the task of texture recognition and classification. My work focuses on comparing and discussion of usage and efficiency of texture features based on local binary patterns and co- ccurence matrices. As classification algorithm is concerned, cluster analysis was choosen.

Keywords

Textures, texture features, texture classification, classificator, local binary patterns, co-occurence matrices, cluster analysis, histogram, pixel.

Citace

Roman Zahradnik: Texturní příznaky, diplomová práce, Brno, FIT VUT v Brně, 2007

Texturní příznaky

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně a uvedl všechny literární prameny a publikace, ze kterých jsem čerpal. Rád bych poděkoval Ing. Miroslavu Švubovi za vedení práce, odborné konzultace a informace, které mi poskytl. Další poděkování náleží Ing. Michalu Španělovi, autoru systému Medical Data Segmentation Toolkit.

.....
Roman Zahradnik
22. 5. 2007

Obsah

1 Úvod

Zpočátku se počítačová grafika rozvíjela kvůli akademickým zájmům podporovaným vládou a armádou, později však začala pronikat do filmu a televize, kde se osvědčila jako konkurenceschopná náhrada za tradiční speciální efekty a animační techniky, takže i komerční firmy začaly přispívat k vývoji v tomto poli. V posledních letech patří k nejdynamičtěji rozvíjejícím se oborům vůbec a její možnosti v podstatě kopírují až neobvyklý růst výkonu a tím i možností počítačů. Je to i proto, že je přímo proslulá svými, někdy až neuvěřitelnými, nároky na hardware. Nutno podotknout, že se naštěstí rozvíjí tím správným směrem.

Mezi lidskými smysly má zvláštní postavení zrak, neboť představuje kanál s největší rychlostí přenosu informace, a proto především s jeho pomocí vnímáme svět. I ve vědě se výrazně uplatňuje obrazová informace, graf nebo fotografie jsou mnohem sdílnější než velký soubor dat ve formě tabulky, apod.

Počítačové vidění je moderní vědní obor, který se technickými prostředky snaží napodobit některé schopnosti lidského vidění. V širším smyslu je považováno za součást kybernetiky, či umělé inteligence. Je zde zřejmá snaha napodobit proces vnímání u člověka a jemu podobný způsob rozhodování na základě informace obsažené v obraze.

Na rozdíl od počítačové grafiky, která pracuje s výstupní informací, pojmy počítačové vidění a zpracování obrazu rozumíme postupy, kde obrazová informace je na vstupu do počítače. Jsou dva základní možné přístupy: vektorová a rastrová grafika. Rastrová grafika je pravidelná síť pixelů, organizovaná jako dvourozměrná matice bodů. Každý pixel nese specifické informace, například o jasu, barvě, průhlednosti bodu, nebo kombinaci těchto hodnot. Objekty v obraze jsou pokryty texturami. Tak lze z obdélníku vytvořit obraz, z válce plechovku od Coca-Coly, či z koule planetu.

A právě problematikou počítačového rozpoznávání textur v obraze se zabývá tato práce. K účelu klasifikace textur slouží speciální algoritmy, nazývané jako klasifikátory. Úkolem těchto prostředků je zařadit předložené textury podle určitých kritérií do předem daných tříd. K tomuto účelu je potřeba nalézt, zadefinovat a posléze extrahovat ideální množinu nejlépe číselných charakteristik, které danou texturu určují. Tyto charakteristiky se nazývají texturní příznaky.

Tento dokument v jednotlivých kapitolách seznamuje čtenáře s teoretickým základem týkajícím se počítačového zpracování obrazu, analýzy textur, problematiky klasifikátorů a shlukové analýzy. Nastiňuje také některé problémy při rozpoznávání textur. V neposlední řadě obsahuje návrh analyzátoru, který by měl ve spolupráci s klasifikátorem podat přesnější náhled na efektivitu různých texturních příznaků. Součástí práce je také vytvoření rozsáhlé texturní galerie a shrnutí velkého počtu testů a experimentů provedené na galerii s použitím implementovaného systému.

2 Teoretický úvod

2.1 Zpracování obrazu

Obraz může být modelován matematicky pomocí spojité skalární funkce *f* dvou nebo tří proměnných. Tato se nazývá obrazová funkce. Hodnota obrazové funkce může být jediné reálné číslo (například jas, intenzita červené barvy, teplota u termovizní kamery, atd.), v počítačové grafice je to ale obvykle více hodnot (například data z počítačového tomografu, či spektrální měření svitu hvězd), jindy je dokonce výhodné pracovat s komplexními čísly.

Při práci s obrazem v počítačové grafice máme zřídka k dispozici spojitý definiční obor funkce. Obyčejně častěji pracujeme v rastru, obrazová funkce je tedy představována maticí. Prvky matice jsou obrazové elementy, pixely (picture element), jejichž hodnota je úměrná množství světelné energie. Z hlediska zpracování digitálního obrazu je pixel dále nedělitelná nejmenší jednotka. V třírozměrném prostoru mluvíme o objemové funkci. Elementární prvek je označován jako voxel.

Zpracováním digitálního obrazu rozumíme transformaci jednoho obrazu na druhý pomocí rozličných metod za různými účely, například předzpracování. Příklady takovéto transformace jsou filtrace, detekce hran, vyhlazování, segmentace a jiné. Při filtraci obrazu se k výpočtu nové hodnoty pixelu využívá malé okolí právě zpracovávaného pixelu. Výsledná hodnota pixelu je pak dána výpočtem konvoluce okolí zpracovávaného pixelu s jádrem filtru. Matematická definice diskrétní konvoluce:

$$I(x,y)*h(x,y)=\sum_{i=-k}^{k}\sum_{i=-k}^{k} I(x-i,y-j)h(i,j) \qquad (1)$$

kde *I(x, y)* je vstupní obrazová funkce a *h(x, y)* je k-řádková, k-sloupcová matice nazývaná jádro filtru.

Dalším důležitým pojmem je korelace. Je to algoritmus umožňující jednoduše vzájemně porovnávat dva digitalizované signály (posloupnosti vzorků). Výsledkem porovnání je opět posloupnost čísel vyjadřující podobnost dvou porovnávaných signálů a některé původně skryté vlastnosti signálů.

$$I(x,y)*h(x,y)=\sum_{i=-k}^{k}\sum_{i=-k}^{k}I(x+i,y+j)h(i,j) \tag{2}$$

Speciální případ je pak tzv. autokorelace, kdy se porovnávají dva shodné signály.

$$I(x,y)*I(x,y)=\sum_{i=-k}^{k}\sum_{i=-k}^{k}I(x+i,y+j)I(i,j) \tag{3}$$

2.2 Textury

Předměty kolem nás mají dokonale hladký a jednobarevný povrch, jejich optické vlastnosti, či tvar se bod od bodu mírně liší. Je to způsobeno strukturou materiálu, ze kterého jsou zhotoveny. Technikou, kterou je možno toto postihnout, jsou textury. Aplikací textur lze výrazně zvýšit realitu scény. S texturou pracují různé vědní obory. Zpracování obrazu chápe texturu jako vlastnost (skupinu vlastností), které jsou konstantní na vymezené ploše, uvnitř vymezené oblasti. Syntéza obrazu používá textury jako prostředek pro dosažení přirozeného vzhledu objektů. Jako předmět vizuálního vnímání je textura definována následovně:

Vizuální textura je smyslový vjem, získaný prostřednictvím zrakových orgánů, je to zrakový vjem. Pomocí tohoto vjemu rozlišuje zraková soustava dvě sousední plochy zrakového pole, aniž by bylo nutné měnit pozici pozorovatele, nastavení zrakové (optické) soustavy.

Geometrickými prostředky můžeme popsat tvar těles, či ploch, definovat určitý objem v prostoru. Pomocí textur je potom možno stanovit vzhled povrchu těles. V kombinaci s vhodnými osvětlovacími modely můžeme texturami vyjádřit detailní vzhled geometricky určeného objektu, hrubost, či jemnou strukturu jeho povrchu.

Textura v počítačové grafice je obvykle definována jako funkce T přiřazující bodům v rovině hodnoty modulované veličiny:

$$T : D_T \rightarrow H_T, D_T \subset R^2 \tag{4}$$

kde H_T je množina hodnot, kterých mohou nabývat pixely v obraze, typicky stupně šedi, pr. 0-255, nebo vektor intenzit barev, atd.

Abychom mohli texturu aplikovat, musíme definovat ještě další funkci M, přiřazující každému bodu na povrchu tělesa bod z definičního oboru textury. Funkce je označována jako inverzní mapování a vyjadřuje proces nanášení textury.

$$M : D_M \rightarrow D_T \tag{5}$$

kde $D_M \subset R^3$ jsou povrchové body tělesa. Hledané zobrazení textury na těleso získáme složením MoT. Použití textury si lze představit jako polepení tělesa papírem, funkce T pak odpovídá tomu, co je na papír nakresleno, a funkce M způsobu, jakým je nalepen.

Textury mohou být zadefinovány dvěma způsoby, tabulkou nebo algoritmem. V prvním případě tabulka obsahuje hodnoty textury získané vzorkováním (obecně) spojitého obrazového prostoru ve zvoleném rastru. U procedurální textury nejsou její hodnoty nikde předem uloženy, ale počítají se vždy znovu pomocí vhodného vzorce, či algoritmu.

2.3 Prostorová textura

Každému bodu třírozměrného prostoru scény přiřadíme přímo hodnotu textury:

$$T : D_T \rightarrow H_T, D_T \subset R^3 \tag{6}$$

kde D_T jsou povrchové body tělesa a H_T je množina hodnot, kterých mohou nabývat pixely v obraze, typicky stupně šedi, pr. 0-255, nebo vektor intenzit barev, atd.

Tato textura se nazývá prostorová, oproti dříve zavedené rovinné textuře. Zatímco aplikaci rovinné textury můžeme přirovnat k polepení tělesa pomalovaným papírem, prostorová textura odpovídá jinému procesu – vyřezání z jednoho kusu materiálu, který má prostorovou strukturu popsanou texturou. Je tak možno modelovat např. tělesa vytesaná z mramoru, či vyřezaná ze dřeva. Nevýhodou prostorové textury je to, že v praktických příkladech v praxi nelze její navzorkované hodnoty uložit do 3D tabulky vzhledem k vysokým paměťovým nárokům.

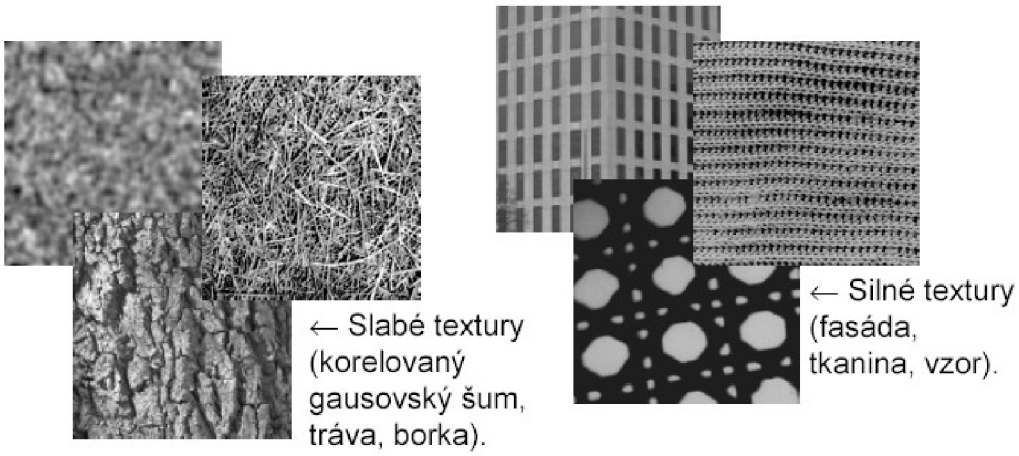
2.4 Analýza textur

Účel analýzy textur:

- vymezit oblasti textur jako základ segmentace (s max. Prostorovým rozlišením)
- klasifikovat (texturní) oblast obrazu do některé třídy (z co největší oblasti)

Síla textury

- **Slabé textury** jsou nedekomponovatelné, mají nepravidelnou strukturu.
- **Silné textury** jsou dekomponovatelné a mají výraznou strukturu.



Obrázek 1: Silné a slabé textury; převzato z [6]

Existují dvě základní metody texturní analýzy, syntaktická analýza a příznaková analýza.

Syntaktická analýza používá k popisu textury posloupností primitiv a jejich hierarchickou strukturou. Vytvořený popis se předkládá analyzátoru, který slouží k samotnému rozpoznávání. Analyzátor používá k rozpoznávání gramatiky. Gramatika musí být známa již před započetím rozpoznávání.

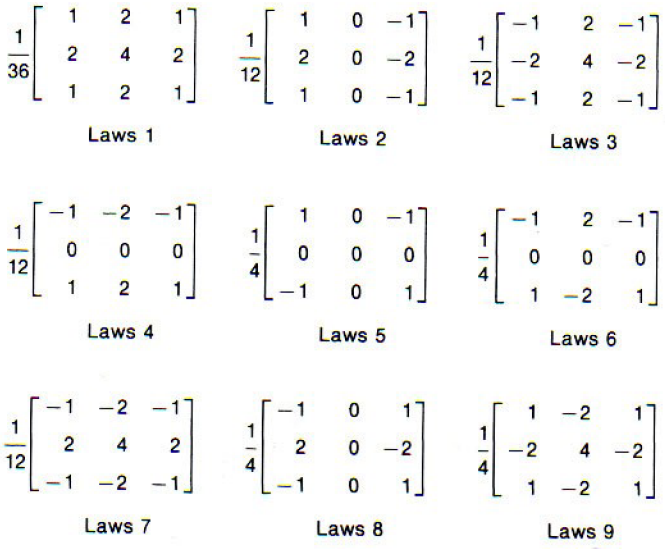
Příznaková analýza používá k popisu objektu hodnoty, které mají význam míry vlastnosti a nazývají se příznaky. Všechny příznaky, kterými popisujeme objekt, můžeme uspořádat do vektoru, který nazýváme vektor příznaků. Prostor všech těchto vektorů nazýváme příznakový prostor. Je to tedy kartézský součin oborů hodnot všech uvažovaných příznakových proměnných. Klasifikátor tedy zobrazuje příznakový prostor objektů na množinu indikátorů tříd. Je-li textura popsána modelem, který ji může generovat, parametry mohou být chápány jako příznaky. Protože každý obraz a objekty v něm obsažené mají jiné vlastnosti, velmi záleží na vhodné volbě množiny příznaků.

2.5 Příznakové přístupy

- **Lokální parametry** jsou číselné charakteristiky popisující texturu v kontextu blízkého okolí vybraných pixelů. Nevýhodou je skutečnost, že prostorové vztahy v textuře jsou ignorovány. Mezi takovéto příznaky patří
  - jas, barva
  - lokální jednorozměrné statistiky (lokální střední hodnota, lokální rozptyl, statistické momenty)
  - lokální spektra, parametry spekter (souvisí s autokorelačními příznaky)

- **Mikrostrukturní metody**

  Příznakem je v tomto případě vektor odezev lineárních operátorů (gaussovský vážený průměr, hranový detektor, atd...). Více lokálních lineárních operátorů → vektorový parametrický obraz



- **co-occurence matrices** („CoM - matice současného výskytu, plošná šedotónová závislost“)

  Význam matice sousednosti lze vysvětlit takto: Každý prvek matice C udává kolikrát spolu v celém obraze sousedí dva pixely o jasových hodnotách daných právě indexem tohoto prvku v C. Sousedství dvou pixelů je zde definováno směrem. Například pro pixel o souřadnicích [x; y] a směr [a; b] je určen jeho soused, který má souřadnice [x + a; y + b]. Mohou být chápány jako zobecněné 2D histogramy – matice q x q pro dvojice stejného okolí bodů (x,y) a $(x+\Delta x, y+\Delta y)$ v rámci obrazu o G úrovních jasu; matic se udává více pro různé dvojice $(\Delta x, \Delta y)$, obvykle spíše v polárních souřadnicích $(\Delta r, \varphi)$ - ovšem v dané vzorkovací mřížce › množina CoM.

  Prvky matice vyjadřují četnosti výskytu dané kombinace pixelů. Z těchto lze snadno odvodit pravděpodobnosti daných kombinací:

$$P(g,g') = \frac{C(g,g')}{\text{celkový počet pixelů}} \tag{7}$$

Marginální pravděpodobnosti pro dané intenzity jsou (G je počet úrovní šedi v obrázku) :

- Marginální řádková pravděpodobnost:

$$p_x(g) = \sum_{g'=1}^{G} p(g,g') \tag{8}$$

- Marginální sloupcová pravděpodobnost:

$$p_y(g') = \sum_{g=1}^{G} p(g,g') \tag{9}$$

$$p_{x+y}(g'') = \sum_{g=1}^{G}\sum_{g'=1}^{G} p(g,g'), \qquad g''=2,3,\ldots,2G \qquad \text{pro g+g'=g''} \tag{10}$$

$$p_{x-y}(g'') = \sum_{g=1}^{G}\sum_{g'=1}^{G} p(g,g'), \qquad g''=0,1,\ldots,G-1 \qquad \text{pro |g-g'|=g''} \tag{11}$$

Z marginálních pravděpodobností je možné odvodit výběrové průměry a výběrové rozptyly hodnot:

- výběrová střední hodnota

$$\mu_x = \sum_{g=1}^{G} g p_x(g) \tag{12}$$

$$\mu_y = \sum_{g=1}^{G} g p_y(g) \tag{13}$$

- výběrový rozptyl

$$\sigma_x = \sqrt{\sum_{g=1}^{G} p_x(g)(g-\mu_x)^2} \tag{14}$$

$$\sigma_y = \sqrt{\sum_{g=1}^{G} p_y(g)(g-\mu_y)^2} \tag{15}$$

- Z těchto jde poté získat řada texturních charakteristik. Mezi základní patří:

  - <u>energie</u>

$$\sum_{g=1}^{G} \sum_{g'=1}^{G} p^2(g, g') \tag{16}$$

  - <u>kontrast</u>

$$\sum_{g=0}^{G-1} g^2 \left(\sum_{g=1}^{G} \sum_{g'=1}^{G} p(g', g'') \right) \quad \text{pro } |g'-g''|=g \tag{17}$$

  - <u>entropie</u>

$$-\sum_{g=1}^{G} \sum_{g'=1}^{G} p(g, g') \log\{p(g, g')\} \tag{18}$$

- lokální homogenita

$$\sum_{g=1}^{G}\sum_{g'=1}^{G}\frac{1}{1+(g-g')^2}p(g,g') \tag{19}$$

- korelace

$$\frac{1}{\sigma_x\sigma_y}\left(\sum_{g=1}^{G}\sum_{g'=1}^{G}(g\cdot g')p(g,g')-\mu_x\mu_y\right) \tag{20}$$

- součtový průměr

$$\sum_{g=2}^{2G} gp_{x+y}(g) \tag{21}$$

- součtová entropie

$$-\sum_{g=2}^{2G} p_{x+y}(g)\log\left\{p_{x+y}(g)\right\} \tag{22}$$

- tyto parametry tvoří vektor příznaků
- nevýhodou je, že výpočet je časově náročný, což se odráží v rychlosti algoritmu.

- **run-length matrices** („RLm – matice délky běhů“)
  - „běh“ (run) – počet pixelů stejného jasu v určitém směru
  - 2D histogramy – jasová úroveň *versus* délka běhu, jeden pro každý ze 4 směrů (0˚,45˚, 90˚, 135˚)
  - z těchto matic se opět počítají odvozené heuristicky definované parametry
    - zdůraznění krátkých běhů
    - zdůraznění dlouhých běhů
    - nerovnoměrnost jasové úrovně
    - nerovnoměrnost délky běhů
    - rozdělení četnosti běhů ...

    které vytváří příznakový vektor pro každý bod (x,y)

- **lokální autokorelační funkce** (informačně ekvivalentní lokálnímu výkonovému spektru)

$$_{x,y}R_{ff}(m,n)=\sum_i \sum_k f_{i,k} f_{i-m,k-n} \tag{23}$$

  - opět se používají integrující deskriptory, odvozující z korelační funkce (matice) jednoduché příznaky
    - např. šířka hlavního laloku (úzký pro jemnou texturu,...)
    - stupeň a frekvence periodicit

- **Lokální binární vzory (Local Binary Patterns)**

  Operátor pracuje s hodnotami v osmi-okolí pixelu a vytváří tak lokální binární vzory. Výskyty rozdílných lokálních vzorů jsou promítnuté do histogramu, který je používaný jako texturní deskriptor. Lokálních binárních vzorů je možné efektivně využít pro popis textury. Nejprve definujeme v rámci textury určitou oblast. Pro každý pixel této oblasti vypočteme lokální binární vzor, tedy binární řetězec. Ten lze pomocí váhování interpretovat jako číslo z intervalu 0..255. Z těchto hodnot pro danou oblast sestavíme histogram, který bude představovat charakteristiku námi zvolené oblasti a tedy i celé textury.

  Matematická definice:

$$LBP_{P,R}=\sum_{p=0}^{P-1} s(g_p-g_c)2^P \tag{24}$$

  kde

$$s(x)=\begin{cases}1, x \geq 0\\ 0, x<0\end{cases}$$

- V rotačně invariantní klasifikaci je výhodné pracovat i s hodnotami v rozích této oblasti, získáme tak kruhové uspořádání kolem centra. LBP je samozřejmě invariantní vůči jakékoliv monotónní šedotónové transformaci. To znamená, že pokud se zvýší úroveň šedi v textuře, nemá to na výsledný LBP žádný vliv.
- V současné době se využívá LBP při segmentaci obrazu, v praxi při detekci obličeje, detekci pohybu, atd.

2.6 Klasifikátor

Z matematického hlediska je to algoritmus, který přiřazuje prvkům příznakového prostoru (množina všech možných hodnot příznakového vektoru) cílovou třídu. Dochází tedy k rozhodování o příslušnosti daného vektoru do jedné z předem daných tříd. Vektor, který má být klasifikován, musí být nejprve vhodně popsán. To znamená, že je potřeba vybrat jistou množinu elementárních vlastností. Tyto elementární vlastnosti se nazývají příznaky. Různé typy klasifikátorů se liší způsobem, kterým modelují podoblasti příznakového prostoru odpovídající daným třídám.

Teoreticky platí, že čím více příznaků, tím více informace bude klasifikátor mít a tím přesněji bude rozhodovat. Bohužel tento přístup je nereálný, protože s rostoucím počtem příznaků se značně komplikuje realizace klasifikátorů. Je proto potřeba nalézt silně diskriminující příznaky. Problém návrhu klasifikátoru spočívá v nalezení rozdělujících hranic mezi třídami a způsob kterým jsou hranice modelovány.

Klasifikační algoritmus se opírá o odlišení různých tříd vzorů, které budou posléze klasifikovány do skupin. Toto rozlišení se realizuje oddělením, pomocí vhodné, matematicky definované hranice. Způsob, kterým je hranice definována poté do značné míry určuje typ klasifikátoru. Možnosti reprezentace dělící (diskriminační) hranice jsou:

- Nadroviny v eukeidovském prostoru (lineární klasifikátor, neuronové sítě)
- Vícerozměrná gaussova plocha (GM modely)
- Vícerozměrná koule (K-Means, fuzzy C-Means)

V tomto projektu je používána klasifikační metoda, opírající-se o klasifikátor, jež využívá diskriminační oblasti s hranicí modelovanou vícerozměrnou koulí. Proto se tomuto klasifikátoru budeme věnovat poněkud blíže.

2.6.1 Klasifikátor s diskriminační hranicí tvaru vícerozměrné koule

Skutečnost, že je diskriminační oblast (nadplocha) modelována pomocí vícerozměrné koule lze interpretovat následujícím způsobem: každá třída vzorů odpovídá jednomu bodu v R^n rozměrném eukleidovském prostoru, přičemž n označuje délku příznakového vektoru, tedy rozměr příznakového prostoru. V prostoru máme umístěno m bodů, z nichž každý odpovídá jedné třídě. Vzor třídě odpovídá v tom smyslu, že představuje střed oné mnohorozměrné koule, které představuje diskriminační oblast. Klasifikace je pak prováděna tak, že pro vstupní (klasifikovaný) příznakový vektor je spočtena vzdálenost od jednotlivých vzorů a tento vstup je následně zařazen do té třídy, pro niž vzdálenost vyšla nejmenší.

Pro určování vzdálenosti mezi jednotlivými objekty (v tomto případě vektory) je zaveden pojem míra nepodobnosti. Obecně lze pro lineární prostor P definovat míru nepodobnosti definovat jako funkci $d: P \times P \rightarrow [0, \infty)$, jestliže pro všechna $X, Y \in P$ platí: $d(X, Y) = d(Y, X)$ a $d(X, X) = 0$. Mezi nejpoužívanější míry nepodobnosti mezi objekty patří:

- Eukleidovská míra
- Logaritmická míra
- Hammingova vzdálenost

Pro účely této práce je uvažováno s nejzásadnější, eukleidovskou mírou.

Matematická definice Eukleidovské míry:

$$\|x, y\| = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{25}$$

kde

$$x = (x_1, \ldots, x_n) \quad , \quad y = (y_1, \ldots, y_n)$$

2.6.2 Primitivní klasifikátor

Primitivní klasifikátor využívá pro reprezentaci jednotlivých tříd prototypové vzory. Nejprve je vybrána skupina vzorů, které samy představují středy diskriminačních ploch. Určení prototypových vzorů je nejčastěji prováděno průměrováním, přes určitou množinu podobných vzorů. Nevýhodu průměrování je, že dokáže postihnout pouze nejjednodušší lineární závislosti vstupu na jeho příznakovém vektoru, ale pro mnohé aplikace je tento přístup postačující.

2.6.3 Klasifikátor s učením

Klasifikátor s učením je vylepšená varianta primitivního klasifikátoru, která stanovuje centroidy jednotlivých tříd ne na základě průměru přes určitou množinu pozorování, ale pomocí trénovacího algoritmu. Trénovacích algoritmů je veliká spousta a každý odráží doménu svého použití.

Základní třídění trénovacích algoritmů tyto dělí na učení s učitelem a učení bez učitele.

Pod pojmem učení s učitelem se chápe takové učení (trénování) klasifikátoru, kdy mu předkládáme jak trénovací vektory tak i jejich správnou příslušnost ke třídě tzn., že trénovací množina je ve tvaru

$$T=\left\{(x_1,\omega_1),(x_2,\omega_2),\ldots,(x_R,\omega_R)\right\} \quad , \tag{26}$$

kde x_i představuje vzorový vstup a ω_i požadovaný výstup k danému vzoru. Tyto metody fungují tak, že algoritmicky upravují diskriminační oblasti tak, že vzory z trénovací množiny jsou zařazovány do jim předepsaných tříd.

U učení bez učitele správnou příslušnost vektorů neznáme a klasifikuje se do tříd bez této znalosti (viz shluková analýza). Vstupem pro trénovací algoritmus bez učitele je tedy pouze soubor vstupních příznakových vektorů. Trénovací algoritmus sám rozhoduje o tom, jakým způsobem budou namodelovány diskriminační oblasti, jelikož není možné provádět posouzení korektnosti na základě apriorních znalostí (předepsané třídy pro vzorové vektory - viz. učení s učitelem). Mechanizmů pro určování správnosti tvaru diskriminační oblasti je veliká spousta.

Nejrůznější klasifikátory nacházejí uplatnění v mnoha oborech, jako například počítačové vidění, lékařská diagnostika, technická diagnostika, rozpoznávání mluvené řeči, rozpoznávání psaných písem, nebo v dolování dat. Některé typy učících algoritmů pro klasifikátory:

- shluková analýza (modelování pomocí center oblastí)
- support vector machines (opět dělení pomocí nadrovin, ale dodatečná kritéria pro jejich polohu)
- gaussian mixture models (modelují danou oblast pomoci vícerozměrných gaussovských rozložení pravděpodobností)

Pro náš účel, tedy klasifikátor s diskriminačními oblastmi tvaru vícerozměrné koule jsou nejvhodnější trénovací metody založené ne shlukové analýze.

2.7 Shluková analýza

Shluková analýza je společný název pro celou řadu metod, jejichž cílem je využití informací z analýzy vícerozměrných dat k roztřídění množiny objektů do několika relativně homogenních podsouborů, označených jako shluky (*clustery*). Objekty uvnitř shluků mají být co nejvíce podobné a objekty patřících do různých shluků co nejvíce rozdílné. Roztříděním do několika podsouborů rozumíme klasifikaci, která vede k vytvoření systému tříd. Na závěr shlukovací analýzy se proto provádí charakterizace (popis) jednotlivých tříd (tj. shluků) a interpretace. Tímto způsobem lze i významně snížit dimenzionalitu úlohy tak, že původní sadu proměnných nahradíme příslušností k nové třídě. Shlukovací metody jsou úspěšné především v situacích, kdy objekty mají tendenci se seskupovat do přirozených tříd, než v případě náhodného rozmístění objektů v atributovém prostoru.

Shlukovací analýzu provádíme zpravidla na množině objektů, kde každý objekt je popsán řadou znaků (veličin). Takový postup označíme jako Q-techniku shlukování. Oproti tomu R-technika shlukování vychází z analýz množiny znaků, charakterizovaných prostřednictvím objektů.

Shlukovací analýzy je možné dělit podle různých kritérií, např. zda je na začátku určen počet shluků nebo se má v průběhu řešení nalézt optimální počet shluků, nebo podle výsledné struktury skupin objektů na hierarchické aglomerativní, kdy se postupným seskupováním vytváří stromová struktura od jednotlivých objektů až po jeden shluk a na divizivní, kdy se rozděluje počáteční celkový shluk do hierarchického systému dílčích skupin či objektů. Pro nehierarchické metody optimalizační a metody analýzy módů.

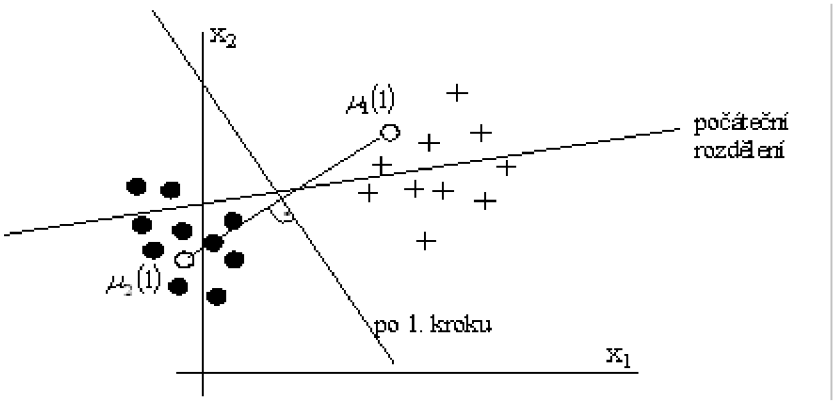
Míry nepodobnosti mezi shluky se definují pomocí míry nepodobnosti mezi objekty. Mezi nejpoužívanější principy patří:

- princip nejbližších sousedů
- princip nejvzdálenějších sousedů
- princip průměrné vzdálenosti
- vzdálenost průměrů shluků (centroidů)
- Kolmogorova zobecněná vzdálenost

2.7.1 Algoritmus K-Means

Jedná se o nejznámější algoritmus shlukové analýzy. Algoritmus K-means iterativně hledá hodnoty vektorů tak, že minimalizuje střední odchylku mezi zadanou množinou dat a vektory (vzdálenost bodu od etalonu shluku), které mají k těmto datům nejmenší euklidovskou vzdálenost a rozděluje je do předem daného počtu shluků (tříd) *K: C1,C2, … ,CK*. Jinými slovy algoritmus zařadí objekt do shluku, jehož střed (centroid) je mu nejblíže. Postup opakuje, dokud se zařazení nemění.

Předpokládáme, že známe počet shluků *r*. Vycházíme z počáteční volby vektorů středních hodnot $\mu_i(0)$. Kritérium *J* bude nabývat minima, jestliže každý vektor **x** náležící *T* je zařazen do shluku, jehož vektor střední hodnoty $\mu_i(k)$ dává nejmenší hodnotu $\|x-\mu_k\|^2$. Postup opakujeme dokud neplatí $\mu_i(k+1)=\mu_i(k)$.



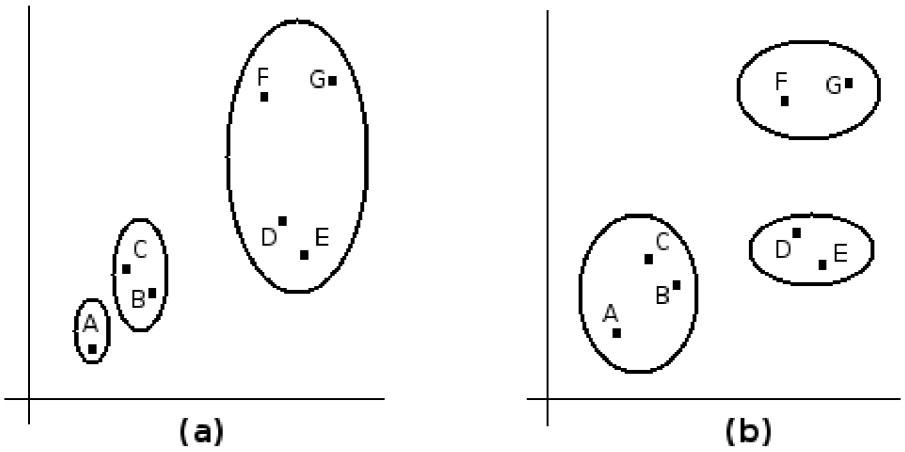
Obrázek 2: Hledání dvou kroků dvou shluků

Vlastnosti K-means:

1. jednoduchý
2. prvky se mohou přeskupovat mezi shluky
3. pouze pro metrická data
4. konverguje v konečném počtu kroků k nějakému řešení C*
5. může existovat více řešení C* v závislosti na počátečních podmínkách

Experimentální zkušenosti naznačují, že je tento algoritmus pro účely vektorové kvantizace velmi vhodný jak z hlediska kvality rozkladu trénovací množiny do shluků, tak i poměrně rozumnými výpočetními nároky.

Problém algoritmu K-Means je závislost na počátečním výběru reprezentantů. Z Obrázku 3. je patrné, že algoritmus pro K = 3 vytvoří shluky odlišně v případě, kdy bude počáteční volba reprezentantů za *a) A, B, C*, nebo za *b) A, D, F.*



Obrázek 3: Rozdílně vytvořené shluky

(a) počáteční volba reprezentantů A, B, D
(b) počáteční volba reprezentantů A, D, F

2.7.2 Algoritmus Fuzzy C-Means

Algoritmus Fuzzy C-Means vychází z algoritmu K-Means s tím rozdílem, že dovoluje, aby jeden objekt patřil do více shluků. Navíc určuje jakou měrou patří objekt do jednotlivých shluků. Algoritmus používá matici $U=(u_{ij})$, kde každý prvek určuje míru příslušnosti i-tého prvku do j-tého shluku. Cílem algoritmu je minimalizace objektové funkce

$$J_m=\sum_{i=1}^{N}\sum_{j=1}^{C} u_{ij}^{m}\|x_i-c_j\|^2 \qquad (27)$$

kde m je koeficient v oboru reálných čísel, větších než 1, x_i je vektor popisující jeden objekt, c_j je centroid.

Algoritmu sestává z následujících kroků

1. Inicializace $U=[u_{ij}]$ matice, $U(0)$
2. V k-tém kroku jsou vypočteny hodnoty centroidů:

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m \cdot x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{28}$$

3. Dojde k aktualizaci $U(k)$, $U(k+1)$

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|x_i - c_j\|}{\|x_i - c_k\|} \right)^{\frac{2}{m-1}}} \tag{29}$$

4. Pokud je $\|U(k+1) - U(k)\|$ menší než zadaná hodnota pak konec, jinak zpět k dobu 2.

Automatická volba počtu shluků

Předchozí algoritmus počítá optimální formaci při známém počtu shluků. Často však potřebujeme provést shlukovou analýzu bez apriorní znalosti počtu shluků. V takovémto případě můžeme postupovat následujícím způsobem. Předpokládejme, že máme k dispozici míru kvality pokrytí dané trénovací množiny pomocí jistého počtu shluků. Poté můžeme řešit shlukování jako minimalizační problém :

- začneme s minimálním počtem shluků
- spočítáme rozdělení současného počtu shluků
- zjistíme míru pokrytí
- zvýšíme počet shluků a přepočítáme kontrolní míru pokrytí
- je-li míra pro nové pokrytí menší než původní, opakujeme celý postup, nebo prohlásíme předešlé pokrytí shluky za optimální

Vhodnou mírou optimality pokrytí je například Dunnův index, který počítá poměr minimální vzdálenosti mezi dvěma shluky k velikosti největšího shluku. Jako metriku využívá většinou eukleidovskou míru. Více o Dunnově koeficientu je možné nalézt například v [7].

3 Klasifikace a rozpoznávání textur

Jedná se o třídění textur nalezených v obraze do skupin předem známých tříd na základě jejich charakteristik. Metody klasifikace se dělí do dvou základních skupin, které jsou úzce spjaty se způsobem popisu textur. Jedná se o příznakové rozpoznávání a strukturální rozpoznávání.

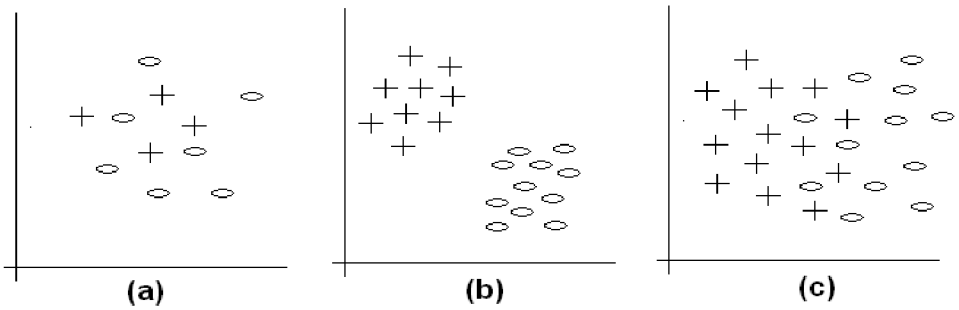
Příznakové metody jsou založeny na principu využití příznaků, což je skupina číselných charakteristik. Učení vlastního klasifikátoru zde může probíhat na principu učení s učitelem, nebo bez jako například u shlukové analýzy.

Strukturální rozpoznávání využívá jako vstupu kvantitativní popis. Objekty jsou zde popsány primitivy. Dále je definována abeceda, jazyk popisu a gramatiky jednotlivých tříd. Vlastní rozpoznávání je pak založeno na principu rozboru slova a kontroly správnosti syntaxe pro všechny třídy. Gramatika popisuje jak generovat vzor textury pomocí přepisovacích pravidel nad množinou symbolů.

Rozpoznávání textur se v praxi využívá při segmentaci vstupního obrazu, při detekcích objektů v obraze dále také například při analýze leteckých a družicových snímků, nebo při rozpoznávání tkání v medicínských aplikacích.

Problémy při klasifikaci

- zkreslení při pořízení vstupních dat (šum)
- nerovnoměrné osvětlení
- rotace
- škálování (otázka, zdali je orotovaná textura stále jedna a ta samá)
- porušené textury a artefakty v texturách



Obrázek 4: Příklady rozmístění trénovacích dat v příznakovém prostoru

(a) nevhodné pro shlukovou analýzu
(b) ideální pro lineární klasifikátor
(c) ideální pro gaussian mixture models

Příznakové metody se opírají o

1. volbu transformace z obrazového do příznakového prostoru, tedy nalezení souboru příznaků
2. volbou klasifikační metody, viz. Kapitola 2.6

V praxi se nejčastěji používají tyto varianty příznakových charakteristik

1. spektrální charakteristiky (fourierova transformace)
2. statistické charakteristiky (lokální momenty, matice sousednosti, lokální binární vzory)
3. statisticky-geometrické charakteristiky (konektivita, tvarové metriky)

4 Návrh testů úspěšnosti příznaků

Nejvhodnější způsob, jak určit který systém příznaků je nejefektivnější, je testovat tyto na problému klasifikace textur. Jedná se tedy o procentuální vyhodnocení úspěšnosti rozpoznávání textur na základě jejich příznakového vektoru.

V části s teoretickým základem je uvedeno pouze několik základních přístupů v extrakci , pro účely tohoto projektu však dostačujících. Hlavním účelem celého systému je porovnání úspěšnosti a efektivnosti v klasifikaci textur pomocí texturních příznaků získaných metodou lokálních binárních vzorů oproti metodě matic sousednosti. Životní cyklus analýzy bude přibližně následující:

1. výběr vzorku z galerie testovacích textur
2. definování oblasti v textuře
3. výpočet příznakového vektoru
   (a) algoritmem lineárních binárních vzorů
   (b) algoritmem matic sousednosti
4. klasifikace příznakového vektoru
5. rozhodnutí o typu textury
6. zaznamenání úspěšnosti rozpoznání

4.1 Galerie texturních vzorů

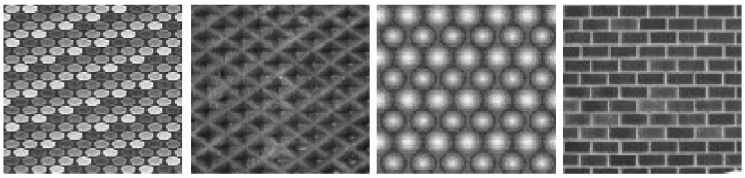
Prvním důležitým krokem je sestavení galerie texturních vzorů. Ta bude obsahovat přibližně 600 textur normalizovaných rozměrů. V galerii budou všechny textury čtvercové plochy o velikosti 100x100 pixelů. Při experimentování bude výhodnější nepoužívat barevné textury, ale pouze textury šedotónové, to znamená v různých odstínech šedi. Nutností v oboru počítačového vidění je, předzpracování obrazu. Nashromážděné textury je vhodné před zařazením do galerie upravit.

Aby bylo možné zkoumat různé aspekty efektivnosti použitých příznaků, je galerie dále dělena:

- Subgalerie slabých textur - 109 vzorů: Takovéto textury jsou většinou výrazně soběpodobné (Obrázek 5.). Typicky se jedná o trávu, písek, asfalt. Příklady takovýchto textur (viz. také kapitola 2.4.).

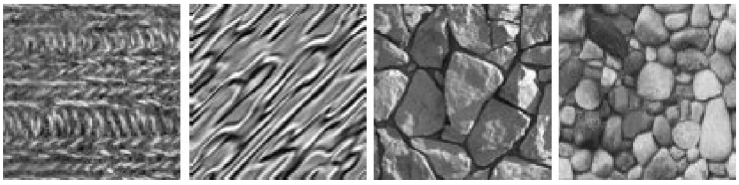
Obrázek 5: Příklady slabých textur v galerii

- Subgalerie silných textur - 63 vzorů: Jedná se o textury s výrazným primitivním, opakujícím se prvkem (Obrázek 6.). Typicky se jedná o tkaniny, cihlové a kachličkované vzory (viz. kapitola 2.4.).



Obrázek 6: Příklady silných textur v galerii

- Zbytek galerie obsahuje textury, které nelze jednoznačně zařadit ani do jedné z těchto tříd. Příklady takovýchto textur jsou na Obrázku 7.



Obrázek 7: Některé nejednoznačné textury v galerii

4.2 Systém příznaků

Pro účel porovnání vzájemné efektivnosti byly zvoleny dva systémy příznaků. Jeden představují příznaky vypočtené z matic sousednosti (viz. 2.5.), dále jen Haralickovy příznaky. Druhou skupinu tvoří histogramy lokálních binárních vzorů.

4.2.1 Matice sousednosti

Hodnota jasu pixelu zcela určitě není dostačující informace pro klasifikátor. Jas se často v rámci textury dramaticky mění, proto je cílem najít takové charakteristiky, které je možno získat z relativně blízkého okolí a které jsou stálé v rámci celé textury. Možností je tedy provádět klasifikaci na základě pixelu a charakter jeho blízkého okolí popsat pomocí vypočtených příznaků.

Prvky matice $P\Phi,d(i,j)$ o souřadnicích (i,j) představují pravděpodobnost, že dva pixely s jasovými hodnotami i a j je možné najít ve vzdálenosti d, a pod úhlem Φ. Takovéto matice je možné zkonstruovat pro libovolný tvar okolí pixelu. Obvyklý tvar okolí je čtvercové velikosti S. Je velmi důležité podotknout, že velikost okolí ovlivní maximální velikost primitivních texturních prvků, ze kterých se textury, tímto způsobem popsatelné, skládají.

Jestliže textura tvoří mozaiku, v ideálním případě se podobá např. šachovnici, pak má mnoho pixelů za souseda pixel s opačnou hodnotou jasu. V tomto mezním případě budou hodnoty v matici koncentrovány převážně v levém horním rohu, nebo pravém dolním rohu matice. V běžném kontrastu

ale mají sousední pixely sklon mít podobnou hodnotu jasu. V matici jsou potom takovéto kombinace charakterizovány vysokými hodnotami v úhlopříčkách.

Hodnoty jasu pixelu je potřeba kvantovat, aby se omezily rozměry matic. Nepříjemným důsledkem kvantování je vznik artefaktů v podobě dramatických změn na relativně jasově spojité ploše. To lze odstranit jemnějším, popřípadě žádným kvantováním, což by ale v důsledku vedlo k neúnosnému navýšení výpočetního času konstrukce matic.

Nemáme-li žádné informace o povaze analyzované textury, je obvyklé charakterizovat texturu pomocí čtyř matic pro $\Phi = 0^\circ, 45^\circ, 90^\circ, 135^\circ$ a $d = 1$.

Sestavíme příznakový vektor, který se skládá z 28 hodnot. Jsou to charakteristiky vypočtené z matic, někdy také nazývané Haralickovy texturní příznaky. Postupně za sebou energie, kontrast, entropie, lokální homogenita, korelace, součtový průměr a součtová entropie vypočtené pro všechny čtyři směry.

Maximální energii má spojitá plocha jedné úrovně šedi, neboli plocha ve stejné kvantifikační hladině.

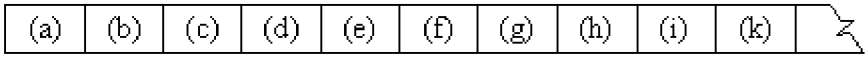
Kontrast vyjadřuje míru výskytu oblastí, které spolu sousedí a mají výrazné rozdíly v intenzitách šedi. Maximum dosahuje pro "šachovnicový vzor".

Entropie vyjadřuje míru neuspořádanosti a šumový charakter textury. Maximální hodnotu dosahuje pro šumovou texturu, jejíž intenzity pixelů mají rovnoměrné rozložení.

Lokální homogenita vyjadřuje míru souvislosti dílčích podoblastí textury. Maximální hodnoty dosahuje pro oblast jedné intenzity.

Korelace vyjadřuje míru lineární závislosti jednotlivých úrovní šedi v příslušné poloze. Maximální korelace dosahují oblasti spojitě vyplněné jednou barvou.

Všechny hodnoty texturních příznaků je nutné vhodně normalizovat, pro náš případ to odpovídá transformaci hodnot do intervalu <0..1>. Jedná se o prosté vydělení hodnoty příznaku jeho maximální možnou hodnotou.



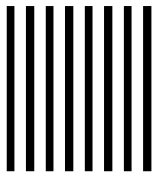
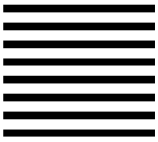
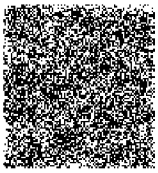
Obrázek 8: Příznakový vektor

(a) energie pro směr 0˚
(b) kontrast pro směr 0˚
(c) entropie pro směr 0˚
(d) lokální homogenita pro směr 0˚
(e) korelace pro směr 0˚
(f) součtový průměr pro směr 0˚
(g) součtová entropie pro směr 0˚
(a) energie pro směr 45˚
(b) kontrast pro směr 45˚
(c) entropie pro směr 45˚

Texturní příznaky získané z matic sousednosti dokáží texturu dobře popsat. Jejich nespornou výhodou je skutečnost, že jsou invariantní vůči velikosti stejnosměrné složky intenzity jasu v regionu. Problém těchto příznaků spočívá v kvantování hladin jasu. Jedná se o situaci, kdy je region vyplněn pixely velmi blízkých intenzit, které však po kvantování spadají do různých tříd. Tento jev se projeví artefakty v zasažených místech.

Úplné odstranění způsobených artefaktů je obtížné. Potlačení tohoto efektu lze dosáhnout stanovením minimálního prahu, kdy jsou hodnoty považovány za odlišné. V podstatě se jedná o výpočet střední hodnoty všech pixelů v daném regionu, a v případě, že se aktuální pixel liší od této střední hodnoty pouze o malou prahovou hodnotu, je namísto něj kvantována právě střední hodnota.

Dalším podstatným problémem texturních příznaků je časově velmi náročný výpočet matic sousednosti. V případech kdy je minimalizace výpočetního času nutností, může jisté informace o textuře poskytnout konvoluční sobelův operátor. Tento aproximuje směrové derivace a dává vyšší odezvy tam, kde v textuře dochází k rychlým změnám jasu. Vhodným průměrováním odezvy sobelova operátoru v okolí daného pixelu lze do jisté míry popsat texturní charakter.

Vzor	Entropie				Lokální homogenita			
	0°	45°	90°	135°	0°	45°	90°	135°
	A	A	0	A	0	0	MAX	0
	0	A	A	A	MAX	0	0	0
	MAX	MAX	MAX	MAX	0	0	0	0

Tabulka 1: Závislosti vybraných haralickových charakteristik na úhlu, pod kterým se nacházejí zkoumané pixely; $A \in \langle 0,1 \rangle$

4.2.2 Lokální binární vzory (LBP)

Hodnota každého bodu je konfrontována s hodnotami v jeho osmi-okolí a každému pixelu z osmi-okolí je přiřazena binární hodnota. Pokud je jeho hodnota vyšší nebo rovna hodnotě referenčního pixelu, je mu přidělena 1. V opačném případě 0. Tyto hodnoty prvků v osmi-okolí daného bodu jsou dále násobeny váhami danými postavením odpovídajícího prvku v sousedství centrálního bodu. Nakonec se osm získaných hodnot sečte a získáme LBP pro tuto texturní jednotku. Postup výpočtu je znázorněn na následujícím příkladu.

Příklad: **LBP** = 4 + 8 + 32 + 64 = **108**

1	4	9
6	5	3
5	8	2

(a)

0	0	1
1		0
1	1	0

(b)

1	2	4
8		16
32	64	128

(c)

0	0	4
8		0
32	64	0

(d)

(a) referenční pixel se svým osmi-okolím
(b) ohodnocení pixelů v osmi-okolí
(c) váhy pixelů v osmi-okolí
(d) výsledné hodnoty pro výpočet LBP

Pomocí LBP lze jednoduše detekovat některá primitiva:

Bod

1	1	1
1		1
1	1	1

Bod/světlý

0	0	0
0		0
0	0	0

Konec čáry

1	1	1
1		1
0	0	1

Hrana

0	0	1
0		1
0	1	1

Roh

1	1	1
0		1
0	0	1

Texturním příznakem v tomto případě není hodnota LBP, ale histogram všech hodnot v celé textuře, nebo v oblasti v textuře definované. Porovnáváním histogramu textury s histogramy známých textur se provádí klasifikace podle vzorů. Jelikož dokážeme rozlišit 256 lokálních binárních vzorů, bude mít histogram 256 oddílů. Nabízí se otázka, zdali všechny vzory přispívají stejnou měrou k celkovým diskriminačním vlastnostem příznaku (histogramu). Jendou z možností, jak omezit délku příznakového vektoru je použití tzv. uniformních lokálních binárních vzorů.

Definujeme míru neuniformity *U(LBP)*, která odpovídá počtu přechodů v kruhové bitové representaci LBP. Například, vzorky 0 (bitově 00000000) a 255 (11111111) mají hodnotu *U* nulovou, zatím co vzorky 1, 2, 4, 8, 16, 32, 64 a 128 (00000001, 00000010, atd ...) mají hodnotu *U* rovnu 2, protože v bitové representaci jsou dva přechody mezi 1/0 nebo 0/1. Podobně, vzorky 00000011, 00000111, 00001111, 00011111, 00111111, 01111111, a jejich kruhově orotované verze mají hodnotu *U* rovnu 2. Další vzorky mají *U* hodnotu minimálně 4. Uvažujeme, že menší množství přechodů 0/1 a 1/0 je ve vzorku, tedy menší míra neuniformity znamená, že je méně pravděpodobně, že vzorek

podstoupil nechtěné změny při geometrických transformacích jako například rotace. Na základě toho můžeme využít těchto devíti uniformních vzorků s U hodnotami nanejvýš 2 (00000000, 00000001, 00000011, 00000111, 00001111, 00011111, 00111111, 01111111, 11111111) a jejich kruhově orotovaných verzí pro transformačně invariantní analýzu. Těchto devět vzorků odpovídá 58 z 256 originálních nerotovaných vzorků, kterými můžeme popsat nejbližší okolí bodu. Zbývající jsou nahromaděné do jednoho sloupce, který se stává posledním sloupcem histogramu s číslem 59. Použití pouze částečné informace se může jevit jako nesmyslné, ale je to podporováno velmi důležitými poznatky, zdá se totiž, že vybraných devět uniformních vzorů nejvíce vystupuje v současných deterministických mikrotexturách.

Pro snadnější představu je uvedena Tabulka 2., která demonstruje na několika příkladech rozdělování reálných hodnot LBP do sloupců v uniformním histogramu.

reálná hodnota LBP	**index sloupce v uniformním histogramu**
00000000	1
00000001	3
00000010	4
00000101	59
00000110	7
00000111	8
00001001	59
00001010	59
00001100	10
11111111	2

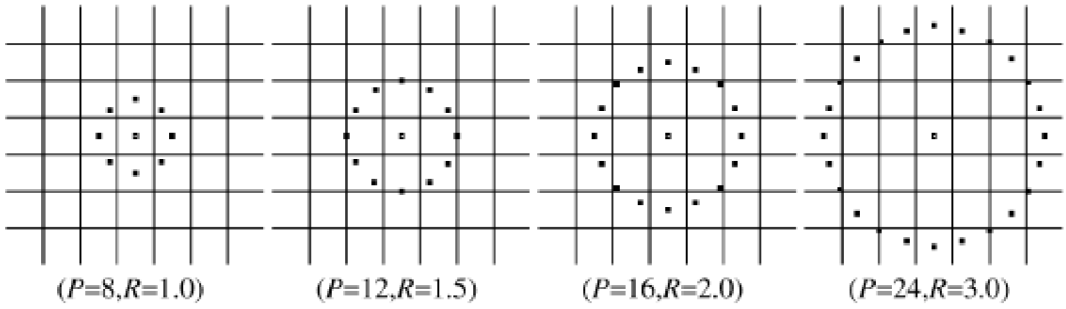
Tabulka 2: Příklady rozdělování LBP do uniformního histogramu

Výhodou LBP je snadná hardwarová realizace a s tím spojený rychlý výpočet. Také jsou invariantní vůči transformacím intenzity pixelu v obraze, tedy nezávislé na změně osvětlení. Jsou také rotačně invariantní. Toho se dosahuje tak, že se LBP rotuje, dokud nemá výsledný vzor minimální hodnotu.

Matematický zápis:

$$LBP^{ri}_{P,R} = min\{ROT(LBP_{P,R}, i) \mid i = 0,1,\ldots,P-1\} \quad (30)$$

Navržený algoritmus využívá pro určování LBP osmi-okolí pixelu. Při potřebě a v některých případech je ale možné uvažované okolí bodu rozšířit, jak je naznačeno na Obrázku 9. Pro tuto práci však bude dostačující použít základní variantu blízkého okolí bodu. V tomto případě tedy je P = 8 a R = 1.0.



Obrázek 9: Okolí bodu pro různá P, R; převzato z [3]

4.3 Natrénování klasifikátoru

Klasifikátor je potřeba natrénovat na rozpoznávání/rozřazování do tolika tříd, v závislosti na tom, kolik máme textur v galerii vzorů. Jelikož budeme rozpoznávat dva typy vstupních vektorů, je potřeba natrénovat dva klasifikátory, jeden pro LBP, druhý pro matice sousednosti. Pro natrénování na galerii vzorů využijeme, v teoretické části popsaný algoritmus Fuzzy C-Means. Trénovací množina bude obsahovat asi 600 vzorů (z každé třídy minimálně 30 příznakových vektorů).

4.4 Galerie testovacích textur

Dále je potřeba sestavit další galerii, tentokráte to bude množina testovacích textur, které se bude systém pokoušet zařadit do některé z tříd, tedy bude se pokoušet pro každou z testovaných textur nalézt odpovídající v galerii vzorů.

Inteligentní navržení testovací množiny bude mít zásadní vliv na kvalitu dosažených výsledků. V této galerii můžeme ponechat i některé textury beze změny oproti vzorům, ale aby měl pokus smysl bude potřeba použít textury, které jsou nějakým způsobem odlišné od vzorových. Na druhou stranu je potřeba aby si byly testované textury se vzorovými podobné. Jelikož je úkolem systému na základě příznaků správně přiřazovat vzorovým texturám ty, které jsou velmi podobné, popřípadě s nimi korespondují, bude nejlepším způsobem většinu vzorových textur nějakým způsobem upravit, pozměnit, nebo poškodit. Využijeme například zašumění gaussovským šumem. Bude zajímavé porovnat výsledky textur s různým druhem „poškození“. Vzhledem k navrženému algoritmu je mým teoretickým předpokladem, že například škálování nebo rotace změní texturu na jinou a výsledná textura by měla být zařazena do jiné třídy, než vzorová.

Gaussovský šum je takový druh šumu, u kterého četnost výskytu jeho amplitud je dána gaussovským (normálním) rozložením. Cílem bude v ideálním případě, takto vzniklé, pozměněné textury rozpoznat jako odpovídající vzorové textury ze kterých vzešly.

4.5 Test klasifikátorů

Úkolem klasifikátoru je na základě příznakového vektoru přiřadit testované textuře příslušnost do některé z tříd, tedy rozpoznat některou ze vzorových textur. To znamená, že v případě LBP se vybere v textuře určité procento bodů, které budou uvažovány pro výpočet. Výše popsaným způsobem se pro vybrané pixely a jejich nejbližší okolí spočítají lokální binární vzory. Pro tyto osmibitová čísla se dále sestaví histogram o 256 oddílech a ten se následně použije jako texturní příznak pro klasifikaci odpovídajícím klasifikátorem.

V případě matic sousednosti se také v textuře vybere určité procento bodů, pro které se následně spočítají příznakové vektory. Odpovídající, natrénovaný

klasifikátor poté provede klasifikaci příznakových vektorů, nejčastější shodu použije jako výsledný index do množiny texturních vzorů.

Očekávaným výsledkem klasifikace je rozhodnutí, zda testovaný prvek patří do množiny vzorů, v pozitivním případě přímo index ukazující na konkrétní rozpoznaný vzor.

4.6 Porovnání výsledků

Jelikož systém bude mít výchozí informace o příslušnosti testovaných prvků do tříd, bude poslední a nejdůležitější částí vyhodnocení dosažených výsledků. Dále stanovení procentuální úspěšnosti identifikace textur z testovací množiny v množině vzorů a to jak lokálních binárních vzorů, tak matic sousednosti. Porovnání dosažených výsledků obou přístupů by mělo potvrdit teoretické předpoklady o vyšší efektivnosti LBP.

5 Implementace

Je předem jasné, že samotná extrakce příznaků z textur a jejich zpracování bude tvořit jen dílčí část velkého celku činností, nutných k úspěšnému dovršení řešení tohoto projektu. Kompletní takovýto systém by bylo jistě velmi náročné vytvářet od nuly. Naštěstí se dnes na Ústavu počítačové grafiky a multimédií fakulty informačních technologií VUT v Brně vyvíjí velmi pokročilý systém na zpracování počítačového obrazu.

Tento programový nástroj, o nějž se velkou měrou opírá i má práce a stává se tak spíše jeho součástí, je systém MDSTk (Medical Data Segmentation Toolkit), jehož autorem je Ing. Michal Španěl. Jedná se o rozsáhlý a velmi propracovaný systém, koncipovaný jako soubor kooperujících modulů, z nichž každý má svou specifickou funkci. Lze zde nalézt rozsáhlý soubor implementovaných prvků realizujících nejrůznější funkce jako načítání či ukládání obrázků, segmentace, detekce hran, nejrůznější filtrace a mnoho dalších. Vhodným řazením tyto moduly vytvářejí efektivní řetězec pro zpracovávání a segmentaci dat, připravený i na použití v paralelním prostředí. Sám jsem se během prací s tímto systémem přesvědčil jak o jeho detailní propracovanosti, tak i velmi snadném a intuitivním užívání. Jedná se bezesporu o velký přínos na poli zpracování obrazových dat.

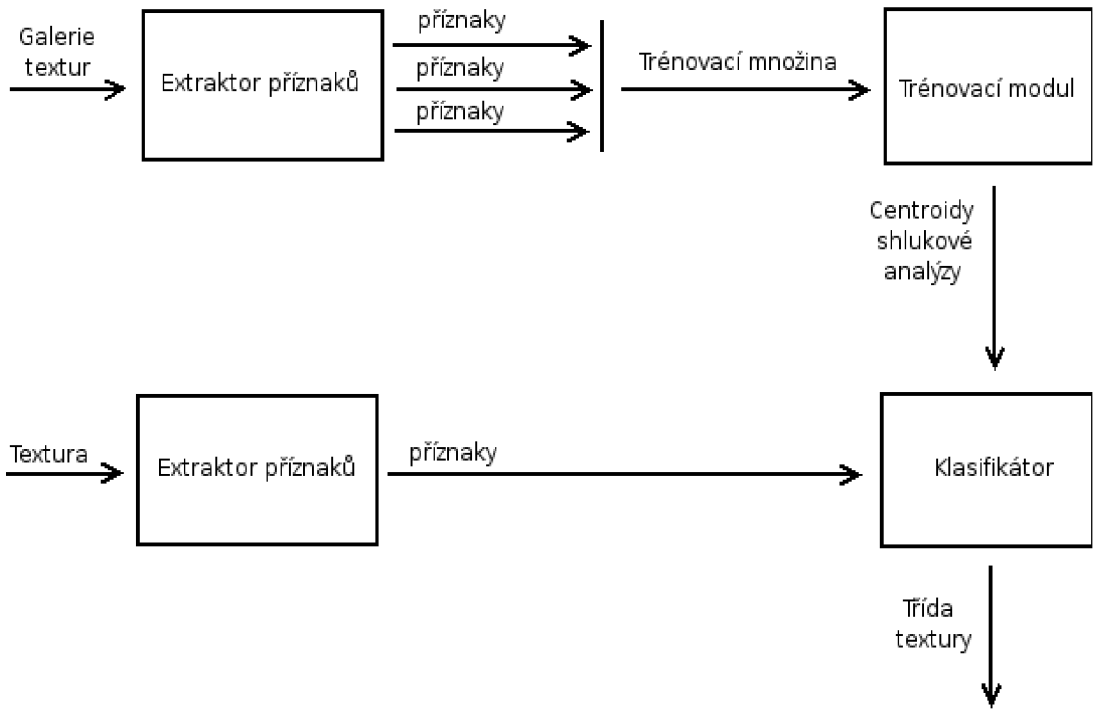
Jak už jsem naznačil, mnou implementovaný algoritmus vlastně představuje jeden ze spolupracujících modulů celého systému. Zároveň se tak stává na tomto systému zcela závislý a samostatně nevyužitelný. Pro implementaci mého analyzátoru z toho ale vyplývá nutnost dodržení některých v celkovém systému stanovených abstrakcí, nutných pro správnou funkčnost systému. V důsledku se to však jeví jako velká výhoda. Při implementaci jsem se snažil dodržovat také určitou štábní kulturu nastavenou tvůrcem MDSTk.

Celý systém je naprogramován v C++. Programy psané v tomto jazyce jsou při dodržení normy dobře přenositelné a při použití kvalitního kompilátoru i velmi rychlé. S ohledem na přenositelnost bylo vše implementováno pod operačním systémem Linux. Nejdůležitějším výstupem mé práce je zhodnocení výsledků testů a experimentů provedených s tímto systémem. Řízení a provádění veškerých testů není řízeno z aplikace, ale s výhodou je zde využito

skriptovacího jazyka bash. Skriptování v bash umožňuje zautomatizovat spoustu úloh, které by jinak vyžadovaly manuální zadávání spousty příkazů.

Pro spolupráci se svým projektem jsem vybral a použil následující moduly, jejichž činnost bych rád letmo objasnil. Galerie vzorů byla vytvořena jako soubor normalizovaných obrázků textur uchovávaných jako grafické soubory JPEG. Pro jejich načítání, resp. ukládání je tedy použit modul *mdsLoadJPEG*, resp. *mdsSaveJPEG*. Při předzpracování se využívá modul *mdsSliceRange*. Dále jsem ještě během prací použil několik pomocných modulů, jako například modul *mdsSliceView* pro zobrazení zpracovávaného obrázků apod.

Mým přínosem projektu byla především implementace modulů pro extrakci příznaků lokálních binárních vzorů *lbp_extraktor* a matic sousednosti *haralick_extraktor*. Dále jsem implementoval trénovací modul pro provedení shlukové analýzy využívající algoritmus fuzzy C-Means systému MDSTk a klasifikátor určený k rozhodování klasifikovaných textur na základě provedené shlukové analýzy. Dále také modul pro vytvoření gaussovského šumu v obrázku mnou využívaný pro „znehodnocování“ textur.



Obrázek 10: Schéma klasifikace

V celém řetězci první a zároveň hlavní částí je modul pro extrakci příznaků. Tento modul má za úkol vytvořit ze vstupních dat příznaky popisující danou texturu. Tomuto modulu se tedy předloží postupně všechny prvky z vzorové galerie textur a tím se vytvoří příznakový vektor. Tato rozsáhlá trénovací množina se posléze použije trénovacím modulem rozmístění v příznakovém prostoru a k sestavení centroidů jednotlivých shluků. Modul klasifikátoru je poté schopen, na základě promítnutí příznaků klasifikované textury do příznakového prostoru, určit nejbližší centroid a tím také třídu, do které bude textura klasifikována. Pokud se příznak klasifikované textury neshoduje přesně s centroidem některé z tříd, je klasifikátor schopen učit i míru příslušnosti dané textury do některé z tříd na základě absolutní vzdálenosti k jednotlivým centroidům. Schéma celého průběhu klasifikace je naznačeno na Obrázku 10.

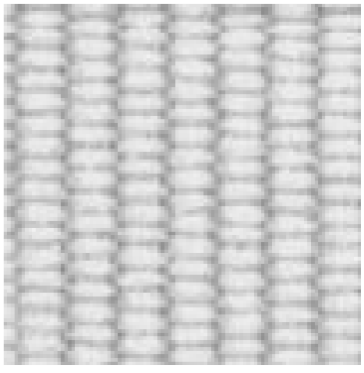
Aplikace neobsahuje žádné grafické uživatelské rozhraní, veškeré výsledky testů a experimentů jsou vypisovány do datových souborů s definovanou syntaxí. Vyhodnocování a případné další výpočty jsou prováděny pomocí skriptovacího jazyka. Během prací jsem také často využíval program *gnuplot* pro vykreslování získaných histogramů.

6 Experimentální výsledky

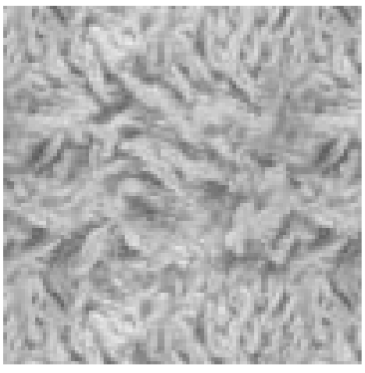
Tato kapitola popisuje výsledky některých provedených testů. V této práci jsem se zaměřil především na ověření efektivnosti metody lokálních binárních vzorů jako příznaků sloužících ke klasifikaci a popisu textur. Bylo tedy nutné implementovat jak algoritmus pro extrakci LBP příznaků, tak nějaký odlišný přístup, se kterým byly výsledky této implementace srovnávat. Jako referenční byl vybrán algoritmus pro získávání příznaků pomocí matic sousednosti, neboli Haralickovy texturní příznaky. Tento přístup patří mezi v texturní analýze zavedené a často používané, kdežto LBP patří mezi spíše nově se prosazující metody. Celá tato kapitola se tedy nese v duchu srovnání těchto dvou přístupů. Je zde také proveden důkladný rozbor experimentů, jelikož se při testování projevila značná závislost na typu vstupních dat.

6.1 Lokální binární vzory

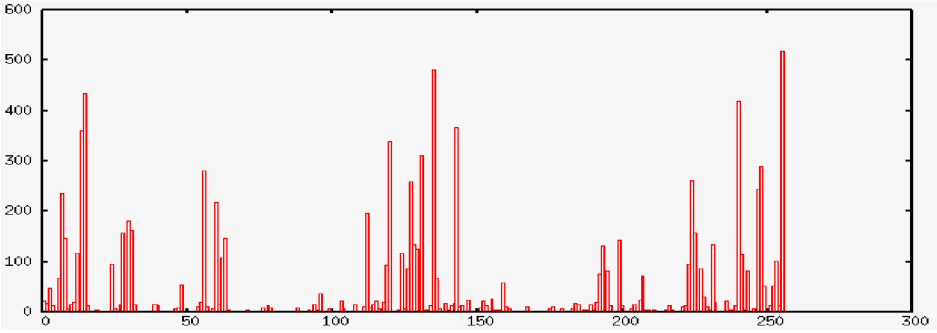
Nejprve pro lepší představu o příznakových vektorech získaných touto metodou uveďme dva příklady. V prvním případě je zvolené textura (Obrázek 11.) z množiny silných textur a v druhém případě z množiny slabých textur (Obrázek 12.).



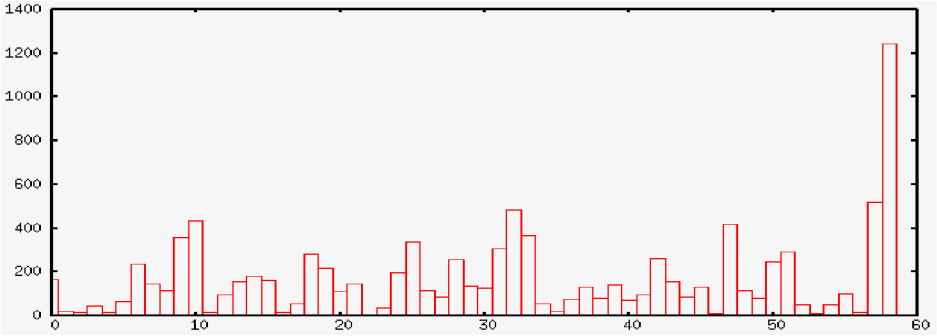
Obrázek 11: Silná textura



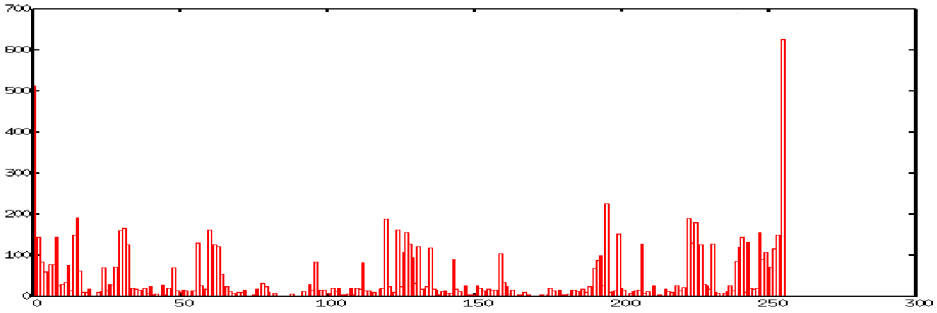
Obrázek 12: Slabá textura



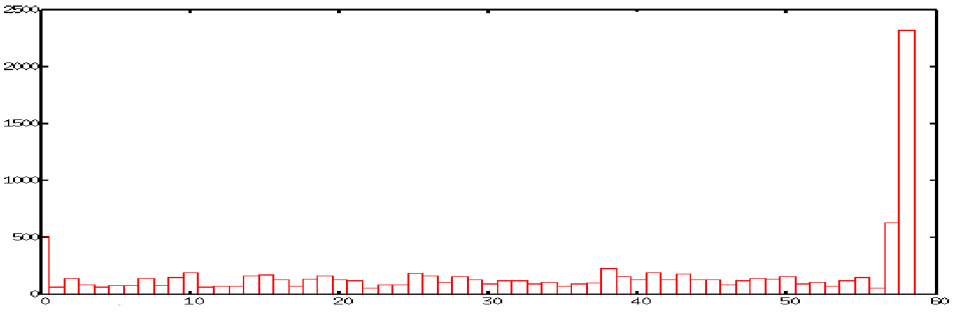
Obrázek 13: Histogram LBP, silná textura (Obrázek 11.)



Obrázek 14: Histogram uniformní LBP, silná textura (Obrázek 11.)



Obrázek 15: Histogram LBP, slabá textura (Obrázek 12.)



Obrázek 16: Histogram unifirmní LBP, slabá textura (Obrázek 12.)

6.2 Klasifikace na základě prototypových textur

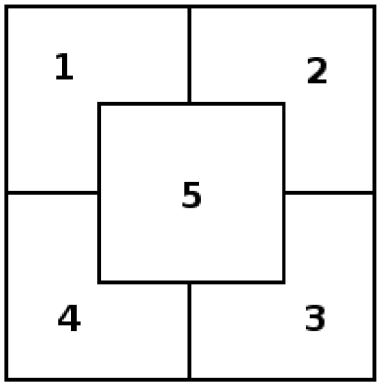
Jako výchozí test pro srovnání síly příznakového vektoru sestaveného na základě lokálních binárních vzorů a vektoru sestaveného na základě Haralickových příznaků byl zvolen test rozpoznávání textur. Jedná se v podstatě o variantu klasifikačního testu. Testovací metoda je popsána a s drobnými úpravami převzata z [3]. Výchozími daty pro srovnávání byla texturní galerie popsaná v kapitole 4.1. Vzorové textury z galerie mají rozměry 100x100 pixelů. Z každého vzoru byly vytvořeny dva příznakové vektory - trénovací a testovací.

Trénovací vektor představuje prototyp pro rozpoznávání primitivním klasifikátorem (viz. 2.6.2). Tento prototypový vektor byl nalezen následujícím způsobem

- Vstupní textura byla nejprve rozdělena na čtyři nepřekrývající se části o rozměrech 50x50 pixelů
- Z těchto čtyř částí byly vypočítány příznakové vektory (LBP histogram nebo soubor Haralickových příznaků)
- Výsledné příznakové vektory byly zprůměrovány aby vytvořily prototypový vzor.

Testovací vstup představoval příznakový vektor vypočtený na základě regionu 50x50 pixelů ze středu vstupní textury.

Celkově rozvržení zdrojových oblastí ukazuje Obrázek 17., přičemž trénovací data představuje průměr z oblastí 1 až 4 a testovací je oblast 5.



Obrázek 17: Regiony v textuře

Test byl proveden na třech množinách texturní galerie. První množinu označme ST1 a představuje slabé texturní vzory. Druhou množinu označme ST2 a tato představuje naopak silné texturní vzory. Poslední skupinou, označme ji OT, představují všechny textury nacházející se v galerii. Tato skupina obsahuje majoritní množství vzorků, které není možné zařadit ani do jedné ze skupin. Skupiny ST1 a ST2 jsou podmnožinami OP.

Velikosti subgalerií jsou následující :

- ST1 - 109 vzorů
- ST2 - 63 vzorů
- 0T - 574 vzorů

Klasifikovaná textura byla tedy porovnávána s každým prototypem na základě jejich vzájemné eukleidovské vzdálenosti. Srovnáváno bylo rozpoznávání na základě lokálních binárních vzorů v normální (plné) variantě a systém Haralickových příznaků. Jako Haralickovy příznaky byly použity všechny charakteristiky popsané v kapitole 4.2.1. Každá z charakteristik byla použita ve čtyřech variantách, pro čtyři různé úhly pod kterými se dva pixely mohou vyskytovat. Celková délka příznakového vektoru tedy byla:

- LBP normal - 256
- LBP uniform - 59
- Haralick - 28

Procentuální úspěšnost tohoto testu shrnuje Tabulka 3.

příznaky	**OT**	**ST1**	**ST2**
LBP normal	89,7%	98,4%	97,3%
LBP uniform	31,0%	55,0%	63,0%
Haralick	45,8%	69,7%	69,8%

Tabulka 3: Úspěšnost klasifikace textur

Z výsledků je vidět, že rozpoznávání na základě plného LBP histogramu bylo daleko nejúspěšnější. Tento přístup obstál velice dobře jak při rozpoznávání slabých textur, tak při rozpoznávání textur silných.

Pozoruhodnou se ukázala skutečnost, že uniformní LBP histogram naopak vykazoval ne příliš uspokojivé výsledky. Příčinou je skutečnost, že uniformní LBP histogram zařazuje všechny vzory s uniformitou větší než 2 do posledního oddílu histogramu. Dochází pak k tomu, že poslední pole histogramu je velice výrazné a zásadní měrou ovlivňuje vypočítanou vzdálenost. Pro odstranění tohoto jevu by bylo vhodné histogram normalizovat, nebo s tímto posledním jeho polem nepočítat vůbec.

Systém Haralickových příznaků se v testu ukázal jako méně efektivní, než normální LBP histogram. Normální LBP histogram se také ukázal výrazně lepší na náhodně zvolených texturách, což jasně ukazuje obecnost a robustnost klasifikace na základě tohoto příznaku.

6.2.1 Srovnání variant příznakového vektoru na základě Haralickových příznaků

V rámci předchozího testu bylo použito 7 Haralickových charakteristik. Nabízí se otázka, zdali je možné dosáhnout obdobných výsledků s použitím menšího počtu příznaků. Pro tento účel byly sestaveny dílčí podgalerie, každá o 16-ti texturách, označeny 16a až 16h. Každá z těchto skupin posloužila jako prototypy pro klasifikaci celé testovací galerie, a zjišťovány byly procentuální odchylky při použití omezeného systému Haralickových příznaků srovnávané s referenčním plným vektorem příznaků. Testované systémy příznaků byly:

- haralick6 - entropie, lokální homogenita, energie, kontrast, součtový průměr, součtová entropie
- haralick5 - entropie, lokální homogenita, energie, kontrast, součtový průměr
- haralick4a - entropie, lokální homogenita, energie, kontrast
- haralick4b - kontrast, součtový průměr, součtová entropie, korelace

Tabulka 4. shrnuje procentuálně chyby klasifikace:

odchylka [%]	16a	16b	16c	16d	16e	16f	16g	16h
haralick6	0,0	0,2	0,2	0,0	0,3	0,3	0,0	0,0
haralick5	0,7	0,2	0,0	0,2	0,9	1,4	0,7	0,0
haralick4a	16,1	22,6	14,2	24,4	29,3	15,6	21,7	16,0
haralick4b	0,2	0,2	0,0	0,0	0,0	0,2	0,2	0,0

Tabulka 4: Odchylky při použití modifikovaných haralickových příznaků

Z výsledků je vidět, že nejvýrazněji ovlivňuje klasifikaci příznak součtový průměr. Při vynechání součtového průměru z příznakového vektoru totiž začne docházet k znatelným odchylkám. Zajímavou skutečností je, že ostatní varianty příznakového vektoru na základě Haralickových příznaků se v důsledku nelišily.

6.3 LBP x Uniformní LBP

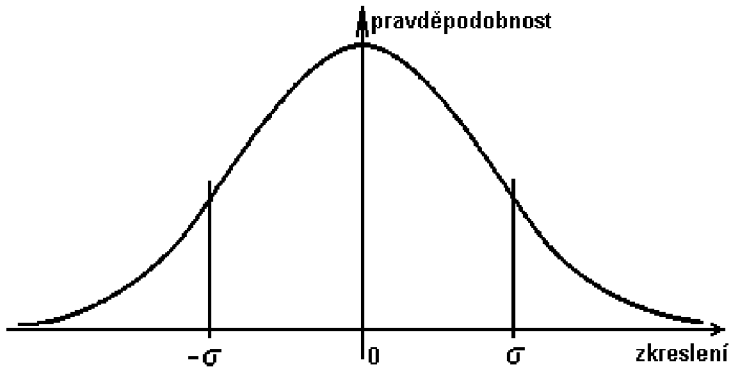
Obdobný test jako předchozí byl proveden i pro vzájemné srovnání klasifikace na základě plného LBP histogramu a uniformního LBP histogramu. Výsledky pouze potvrzují předchozí závěr, tedy skutečnost že uniformní LBP histogramy jsou výrazně slabším příznakem, než plný LBP histogram. Procentuální odchylky uniformního histogramu v porovnání s referenčním plným shrnuje Tabulka 5.

odchylka [%]	16a	16b	16c	16d	16e	16f	16g	16h
LBP uniform	41,6	46,0	44,4	43,6	51,0	45,6	58,7	55,1

Tabulka 5: Odchylka uniformního LBP

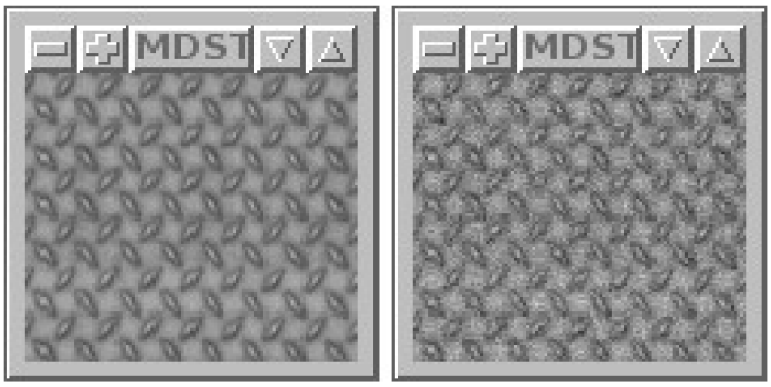
6.4 Odolnost vůči šumu

Jednou z možností prověření úspěšnosti klasifikace pomocí rozličných příznaků je test odolnosti vůči gaussovskému šumu. Gaussovský šum představuje změny v obraze, které ze statistického hlediska mají normální (gaussovské) rozložení pravděpodobnosti. To v podstatě znamená, že malé změny se v obraze vyskytují s větší pravděpodobností. Reálně 99% všech náhodných pokusů dopadá právě v intervalu $\mp 3\sigma$. Míru zkreslení tedy určuje parametr σ. Změnou hodnoty parametru sigma (v rozmezích 2 až 32) prog. modulu *sum* se nastavuje uroveň zašumění klasifikovaných textur pro potřebné experimenty.



Obrázek 18: Gaussovské rozložení výskytu šumu

Vliv gaussovského šumu na obrazová data ukazuje Obrázek 19. Na něm je vidět, že již malá úroveň šumu zkresluje a navíc viditelně mění charakter textur v obraze.



Obrázek 19: Vliv gaussovského šumu na obrazová data

Následující tabulky shrnují úspěšnost rozpoznávání různě zašuměných textur prováděných na galeriích textur. Pro testování byly požity zašuměné původní textury s rozptyly 2 až 16.

sigma	2	4	8	12	16
LBP normal	71,3%	55,2%	32,4%	19,9%	11,7%
LBP uniform	64,1%	39,4%	18,8%	9,2%	5,6%
Haralick	94,8%	88,3%	73,3%	56,4%	41,6%

Tabulka 6: Odolnost proti šumu, celá galerie

sigma	2	4	8	12	16
LBP normal	85%	65%	40%	20%	20%
LBP uniform	70%	50%	25%	15%	5%
Haralick	100%	95%	90%	65%	35%

Tabulka 7: Odolnost proti šumu, slabé textury

sigma	2	4	8	12	16
LBP normal	88%	81%	75%	50%	38%
LBP uniform	88%	75%	69%	38%	19%
Haralick	100%	100%	100%	88%	56%

Tabulka 8: Odolnost proti šumu, silné textury

Z výsledků je zcela jednoznačně vidět, že Haralickovy příznaky daleko lépe obstojí při rozpoznávání zašuměných textur. Důvodem je skutečnost, že statistickou povahu Haralickových příznaků gaussovský šum příliš nezkresluje a navíc vlivem kvantování je šum i do značné míry eliminován. Naproti tomu zašumění textury zásadním způsobem změní poměry primitivních prvků textury a tedy i celý LBP histogram.

6.5 Diskriminační síla příznakových vektorů

Mimo určení, jakým způsobem daná varianta příznakového vektoru ovlivňuje úspěšnost rozpoznávání, je také účelné porovnat diskriminační sílu jednotlivých příznaků. Diskriminační silou rozumíme obecně jaké množství textur (obvykle značně veliké) je ještě možné na základě hodnot daných příznaků rozlišit. Tuto je obecně obtížnější stanovit a obecně by se postupovalo metodami matematické statistiky. V této práci je diskriminační síla příznakových vektorů porovnána na základě jednoduchého pokusu :

- Nejprve je vytvořeno několik podmnožin textur o 100 texturách (konkrétně tři množiny A,B,C)
- Pomocí shlukové analýzy je natrénován klasifikátor a to tak, že trénovací algoritmus sám rozhoduje o optimálním počtu shluků. Využijeme algoritmus Fuzzy-C means s výpočtem Dunnova indexu (viz. 2.7.2).
- Systém příznaků, pro který vycházejí průměrné počty nalezených shluků nejvyšší prohlásíme za systém, mající z daných největší diskriminační sílu.

příznaky/ pokrytí	**Množina A**	**Množina B**	**Množina C**
LBP normal	8 shluků	6 shluků	6 shluků
LBP uniform	5 shluků	4 shluky	5 shluků
Haralick	6 shluků	4 shluky	5 shluků

Tabulka 9: Optimální počty shluků

Průměrné počty shluků při automatické shlukové analýze tedy vycházejí :

- 6,7 pro úplný LBP histogram
- 4,7 pro uniformní LBP histogram
- 5 pro Haralickovy příznaky

Lze tedy konstatovat, že největší diskriminační sílu budou mít LBP úplné histogramy jako příznaky pro klasifikaci. Samozřejmě je nutno podotknout, že navržená metoda dává pouze hrubou představu o diskriminační síle. Pro vzájemné porovnání systémů příznaků mezi sebou je však dostačující.

6.6 Náročnost výpočtu příznaků

Z hlediska nasazení systému příznaků v reálné aplikaci (detekční systémy, segmentační a klasifikační systémy) hraje zásadní význam časová složitost výpočtu příznaků. Proto je součástí této práce také diskuze tohoto problému.

Test rychlosti výpočtu jednotlivých příznakových vektorů byl proveden pro sestavení 10000 vektorů příznaků nad texturou o velikosti 100x100 pixelů. V tomto testu, jehož výsledek se zásadním způsobem promítne do stanovení výsledné efektivity metody, srovnáváme extrakci lokálních binárních vzorů a haralickových příznaků. Při testu nebyly používány vstup-výstupní operace, šlo pouze o opakované počítání příznakového vektoru.

Lokální binární vzory	**Haralickovy příznaky**		
	4	**8**	**16**
2m 0.902s	22m34.851s	3m58.788s	1m47.750s

Tabulka 10: Časová náročnost extrakce příznaků

Tabulka 10. ukazuje časovou složitost výpočtu jednotlivých příznakových vektorů. Pro Haralickovy příznaky je možné v tabulce najít tři hodnoty, které odpovídají různým úrovním kvantování hodnot šedi při výpočtu matic sousednosti (viz. kapitola 4.2.1). Číslo 16 znamená, že 16 po sobě jdoucích intenzit šedi je kvantováno do jedné úrovně. Při použití této hodnoty má sousednostní matice rozměry 16x16 a je vidět, že výpočet příznaků nad takovouto maticí probíhá rychleji, než výpočet úplného LBP histogramu. Nevýhodou je, že tímto Haralickovy příznaky výrazně tratí na síle, neboť nedokáží postihnout mnohé změny intenzity šedi v rámci textury. Pro 32 kvantovacích úrovní (rozlišení 8) mají sousednostní matice rozměry 32x32 a výpočet trvá přibližně dvakrát déle, než v případě úplného LBP histogramu.

Podíváme li se na způsob výpočtu, zjistíme že výpočet LBP histogramu využívá na rozdíl od výpočtu Haralickových příznaků operace ve fixní řádové čárce. LBP by byly tedy podstatně snadněji realizovány v hardwaru.

7 Závěr

V této práci byly rozebrány dva přístupy k příznakovému popisu textur. Jednalo se o statistické charakteristiky vypočtené z matic sousednosti (takzvané Haralickovy příznaky) a příznaky reprezentované histogramem lokálních binárních vzorů. Rozebrány byly teoretické principy i problematika realizace. Na obrázcích a tabulkách bylo ukázáno jaký charakter tyto příznaky mají, jaké konkrétní charakteristiky textur postihují a jak.

V rámci práce bylo provedeno množství testů s cílem porovnat tyto dva přístupy k popisu charakteristik textury. Zkoumána byla hlavně celková úspěšnost při rozpoznávání textur. Další skupina testů poté porovnávala kolísání úspěšnosti rozpoznávání při obměnách příznakového systému, ale již v rámci obou metod. Byla také otestována a porovnávána diskriminační síla příznaků při obou přístupech a to metodami shlukové analýzy. V praxi často vyskytující se zkreslení šumem a jeho vliv na úspěšnost rozpoznávání byl také zkoumán. V neposlední řadě bylo provedeno zhodnocení časové složitosti výpočtu obou variant příznaků.

Výsledky hovoří jednoznačně pro popis textury pomocí histogramu lokálních binárních vzorů. Tato relativně nová, jednoduchá metoda se ukázala v testu rozpoznávání textur efektivnější. U této metodiky popisu textury byla zaznamenána cca. 95% úspěšnost, zatímco u příznaků odvozených z matic sousednosti jen cca. 65%. Z testu diskriminační síly příznaků vzešel histogram lokálních binárních vzorů také vítězně. Hlavní nevýhodu tohoto přístupu však odhalil test na šumem zkreslených texturách, kdy podstatně lépe dopadl druhý systém příznaků. Pro použití histogramu lokálních binárních vzorů ale nakonec mluví i kratší doba zpracování.

Vliv na zjištěné výsledky měla také galerie použitých textur, která byla sestavena pro účely této práce. Jelikož je obecně obtížné takovéto galerie připravovat, nelze vyloučit, že použitá galerie částečně zkreslila dosažené výsledky. Sestavení galerie byla ale věnována značná pozornost a výsledky lze tedy považovat za objektivní.

Do budoucna je vhodné rozšířit práci následujícími směry :

- Rozsáhlejší a dle možností kvalitnější texturní galerie
- Pokročilejší metody normalizace příznaků zohledňující nelinearity.
- Zahrnutí dalších systémů texturních příznaků

Literatura

[1] Žára, J., Beneš, B., Sochor, J., Ferkel, P.: Moderní Počítačová Grafika, Computer Press, 2005

[2] Hlaváč, V., Sedláček, M.: Zpracování Signálů a Obrazů, ČVUT, 2002

[3] Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns, 2002

[4] Suri, S. J., Setaredan, S. K., Singh, S.: Advanced Algorithmic Approaches to Medical Image Segmentatiom, Springer, 2002

[5] Švub, M.: Segmentace obrazových a objemových dat pomocí neuronových sítí - Diplomová práce, VUT, 2006

[6] Španěl, M.: Podklady k přednáškám pro Počítačové vidění: Analýza a extrakce příznaků z textur, dokument dostupný na url: www.fit.vutbr.cz/study/courses/POV/private/lectures/pov_03_textury.pdf, 2006

[7] Brun, M., Sima, Ch., Hua, J., Lowey, J., Carroll, B., Suh, E., Doughetry, E.: Model-based evaluation of clustering validation measures, 2005

Seznam příloh

Příloha 1. Manuál k programu

Systém byl celý navržen, vyvíjen a testován na OS Linux. Všechny experimenty jsou prováděny dávkovým spouštěním skriptů v shellu, také proto bude postup instalace a ovládání programu popsáno pro použití v OS Linux.

Instalace programu:

1. Nejprve je nutné si do připraveného adresáře na disku zkopírovat z přiloženého CD archivy *tf.tar.gz* a *galerie_textur.tar.gz* a *MDSTk.tar.gz*
2. Rozbalit archivy

```
# tar xfz tf.tar.gz
# tar xfz galerie_textur.tar.gz
# tar xfz MDSTk.tar.gz
```

3. Zkompilovat systém MDSTk

```
# cd MDSTk
# make -f Makefile.linux
```

4. Zkompilovat vlastní systém pro extrakci příznaků a klasifikaci textur

```
# cd ../tf
# make
```

5. Nyní je systém plně nainstalován, v adresáři *galerie_textur* jsou normalizované obrázky textur.

Extrakce příznaků:

1. LBP, modul *lbp_extraktor*

Za pomoci systému MDSTk se extrahuje příznaky LBP z textury a sestaví jejich histogram. Systém modulů funguje tak, že si jednotlivé moduly předávají svoje výstupy přes rouru. Před samotným procesem extrakce příznaků je nutné vstupní texturu uloženou v souboru formátu JPEG nejprve načíst a předzpracovat. Pro načtení souboru použijeme modul *mdsLoadJPEG z MDSTk,* pro předzpracování obrázku modul *mdsSliceRange* rovněž z tohoto toolkitu. Celý postup extrakce příznaků bude vypadat následovně:

```
# mdsLoadJPEG < JPEG | mdsSliceRange | lbp_extraktor > FILE
```

Argumenty modulu *lbp_extraktor:*

```
# lbp_extraktor -x1 int -x2 int -y1 int -y2 int [-vse -prefix string]
-typ normal/uniform
```

-x1 *int*	x-ová souřadnice levého horního rohu regionu v textuře, ze kterého se budou příznaky počítat
-x2 *int*	x-ová souřadnice pravého dolního rohu regionu v textuře, ze kterého se budou příznaky počítat
-y1 *int*	*y*-ová souřadnice levého horního rohu regionu v textuře, ze kterého se budou příznaky počítat
-y2 *int*	*y*-ová souřadnice levého horního rohu regionu v textuře, ze kterého se budou příznaky počítat
-typ *normal/uniform*	typ extrahovaného histogramu
-vse	vypočítává histogramy čtyř maximálních nepřekrývajících se regionů mezi zadanými souřadnicemi, jejich průměrný histogram a histogram středového regionu
-prefix *string*	prefix jmen výstupních souborů, argument „-vse“

Příklad použití:

```
# MDSTk/bin/mdsLoadJPEG < pic.jpg | MDSTk/bin/mdsSliceRange |
lbp_extraktor -x1 1 -x2 100 -y1 1 -y2 100 -typ normal -vse -prefix
pic
```

2. Haralick, modul *haralick_extraktor*

Za pomoci systému MDSTk extrahuje Haralickovy příznaky z textury. Způsob použití je obdobný jako u 1., jen s jinými argumenty. S pomocí argumentů se nastaví, které Haralickovy příznaky mají být z textury vypočítávány.

Argumenty modulu *haralick_extraktor:*

```
# haralick_extraktor -x1 int -x2 int -y1 int -y2 int [-ent -lh -eng
-kon -sp -se -corr] -vystup file [-vse -prefix string]
```

-x1 *int*	x-ová souřadnice levého horního rohu regionu v textuře, ze kterého se budou příznaky počítat
-x2 *int*	x-ová souřadnice pravého dolního rohu regionu v textuře, ze kterého se budou příznaky počítat
-*y*1 *int*	*y*-ová souřadnice levého horního rohu regionu v textuře, ze kterého se budou příznaky počítat
-y2 *int*	*y*-ová souřadnice levého horního rohu regionu v textuře, ze kterého se budou příznaky počítat
-ent	pro sestavení Haralickových příznaků se použije entropie
-lh	pro sestavení Haralickových příznaků se použije lokální homogenita
-eng	pro sestavení Haralickových příznaků se použije energie
-kon	pro sestavení Haralickových příznaků se použije kontrast
-corr	pro sestavení Haralickových příznaků se použije korelace
-sp	pro sestavení Haralickových příznaků se použije součtový průměr
-se	pro sestavení Haralickových příznaků se použije součtová entropie
-vystup *file*	výstupní soubor
-vse	vypočítává Haralickovy příznaky čtyř maximálních nepřekrývajících se regionů mezi zadanými souřadnicemi, jejich průměrnou hodnotu a Haralickovy příznaky středového regionu
-prefix *string*	prefix jmen výstupních souborů, argument „-vse“

Příklad použití:

```
# MDSTk/bin/mdsLoadJPEG < pic.jpg | MDSTk/bin/mdsSliceRange |
haralick_extraktor -x1 1 -x2 100 -y1 1 -y2 100 -ent -lh -eng -kon -sp
-se -corr -vystup pic.h7
```

Trénování klasifikátoru

1. Normalní LBP, modul *lbp_trener_normal*

Modul určený pro natrénování klasifikátoru algoritmem Fuzzy C-Means s využitím histogramů s normálním rozdělením LBP jako příznakových vektorů. Jeho výstupem jsou třídy sestavené z vstupních příznakových vektorů a také centroidy jednotlivých tříd.

Argumenty modulu *lbp_trener_normal:*

```
# lbp_trener_normal -priznaky file -rozdeleni file -shluky file
[-tridy int -komplex]
```

-priznaky *file*	vstupní soubor s trénovacími příznakovými vektory
-rozdeleni *file*	soubor příznakových vektorů rozdělených do tříd
-shluky *file*	výstupní soubor s centroidy jednotlivých tříd
-tridy *int*	počet tříd použitých k rozdělení trénovací množiny, pokud není argument zadán využije se optimální počet shluků
-komplex	komplexní trénování hledá vhodnou podmnožinu trénovacích příznakových vektorů k natrénování

Příklad použití:

```
# lbp_trener_normal -priznaky priznaky.dat -rozdeleni rozdeleni.dat
-shluky shluky.dat -tridy 8
```

2. Uniformní LBP, modul *lbp_trener_uniform*

Modul určený pro natrénování klasifikátoru algoritmem Fuzzy C-Means s využitím histogramů s uniformním rozdělením LBP jako příznakových vektorů. Jeho výstupem jsou třídy sestavené z vstupních příznakových vektorů a také centroidy jednotlivých tříd. Modul má shodnou syntaxi atributů jako 1.

Argumenty modulu *lbp_trener_uniform:*

```
# lbp_trener_uniform -priznaky file -rozdeleni file -shluky file
[-tridy int -komplex]
```

Příklad použití:

```
# lbp_trener_uniform -priznaky priznaky.dat -rozdeleni rozdeleni.dat
-shluky shluky.dat
```

3. Haralick (7 příznaků), modul *haralick7_trener*

Modul určený pro natrénování klasifikátoru algoritmem Fuzzy C-Means s využitím Haralickových příznaků jako příznakových vektorů. Jeho výstupem jsou třídy sestavené z vstupních příznakových vektorů a také centroidy jednotlivých tříd. Modul má shodnou syntaxi atributů jako 1. Pro natrénování na příznakovém vektoru, který je tvořen z menšího počtu příznaků, se použijí odpovídající moduly *haralick6_trener, haralick5_trener* a *haralick4_trener.* Jejich určení je zřejmé.

Argumenty modulu *haralick7_trener:*

```
# haralick7_trener -priznaky file -rozdeleni file -shluky file
[-tridy int -komplex]
```

Příklad použití:

```
# haralick7_trener -priznaky priznaky.dat  -tridy 4 -rozdeleni
rozdeleni.dat -shluky shluky.dat
```

Klasifikace

Modul *klasifikator* slouží k zařazení vstupní textury do jedné z tříd. Na výstup podává jednotlivé příznakové vektory společně se zařazením do třídy.

Argumenty modulu *klasifikator:*

 # klasifikator –centroidy file vstup file –vystup file –velikost int –mira euk/log [–multi –html –norm]

-centroidy *file*	soubor, obsahující centroidy tříd
-vstup *file*	vstupní soubor s příznakovým vektorem ke klasifikaci
-vystup *file*	jméno výstupního souboru, ve kterém jsou k jednotlivým klasifikovaným texturám přiřazeny třídy
-velikost *int*	velikost příznakového vektoru, pro: LBP normal = 256 LBP uniform = 59 haralick7 = 28 haralick6 = 24 haralick5 = 20 haralick4 = 16
-multi	klasifikace více vstupů; v tomto případě je pro vstup použit soubor se seznamem příznakových vektorů ke klasifikaci
-html	pro výstup bude použit html soubor
-mira *euk/log*	použitá míra
-norm	normalizace vstupů

Příklad použití:

```
# klasifikator –centroidy shluky.dat –vstup vektory.dat –multi
–vystup rozdeleni.html –velikost 256 –mira euk –html
```

Zašumění obrazu gaussovským šumem

Pro ovlivnění vstupních obrazových dat gaussovským bílým šumem přeženeme vřadíme implementovaný modul *sum* dle schéma:

```
# mdsLoadJPEG < JPEG | mdsSliceRange | sum | mdsSaveJPEG > JPEG
```

Argumenty modulu *sum:*

```
# sum -sigma float
```

-sigma *float* — rozptyl gaussovy pravděpodobnosti, neboli úroveň zašumění

Příklad použití:

```
# MDSTk/bin/mdsLoadJPEG < 1.jpg | MDSTk/bin/mdsSliceRange |
sum -sigma 16.0 | MDSTk/bin/mdsSliceView
```

Příloha 2. Obsah přiloženého CD

CD přiložené k této knize obsahuje:

1. tf.tar.gz - zdrojové kódy modulů pro extrakci texturních příznaků implementovaných jako součást této práce
2. galerie_textur.tar.gz – galerie, čítající 574 textur normalizovaných dle požadavků této práce
3. MDSTk.tar.gz - Medical Data Segmentation Toolkit
4. dokumentace.pdf – technická zpráva
5. Dále obsahuje několik předpřipravených skriptů, pomocí kterých lze ověřit funkčnost systému a také vyzkoušet dávkové zpracování několika dílčích kroků prováděných v rámci jednodušších experimentů s klasifikacemi textur popisovaných v této práci.