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

CHARAKTERISTIKY 2D TEXTUR

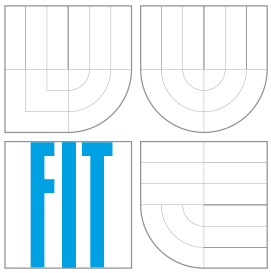
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VÁCLAV PASÁČEK

BRNO 2009

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

CHARAKTERISTIKY 2D TEXTUR

2D TEXTURE FEATURES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VÁCLAV PASÁČEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MIROSLAV ŠVUB,

BRNO 2009

Abstrakt

Protože je textura objektu velice cenná informace pro počítačové vidění, je důležité ji nějakým způsobem popsat. K tomu slouží texturní příznaky. Optimální výběr příznaků je důležitý pro rozpoznávání textur. V této práci byla pro získání příznaků vybrána metoda lokálních binárních vzorů (LBP). Texturním příznakem u této metody není její hodnota, ale histogram četnosti hodnot v celé textuře. Pro porovnání těchto histogramů se zde užívá Euklidovská vzdálenost, Bhattacharyyova vzdálenost nebo Mahalanobisova vzdálenost. Hlavním účelem této práce je vzájemné porovnání klasifikací textur několika variantami metody LBP a vyhodnocení jejich výsledků Euklidovskou, Bhattacharyyovou nebo Mahalanobisovou vzdáleností.

Abstract

Because texture of object is very valuable information in computer vision, it is important to describe it somehow. And for this serve texture features. Optimal selection of features is very important for recognizing texture. In this bachelor thesis were used local binary patterns (LBP) as a method of gaining texture feature. In this method is not its value the texture feature, but histogram of percent occurrence values in the entire texture. To compare histograms there is used Euclidean distance, Bhattacharyya distance or Mahalanobis distance. Main purpose of this thesis is mutually comparing of texture clasification by several variants of LBP and evaluation of their outcomes by Euclidean distance, Bhattacharyya distance or Mahalanobis distance.

Klíčová slova

Textura, statická klasifikace textur, lokální binární vzory, invariance vůči rotaci, euklidovská vzdálenost, Bhattacharyyova vzdálenost, Mahalonobisova vzdálenost.

Keywords

Texture, static texture analysis, localy binary patterns, rotation invariant, euclidean distance, Bhattacharyya distance, Mahalanobis distance.

Citace

Václav Pasáček: Charakteristiky 2D textur, bakalářská práce, Brno, FIT VUT v Brně, 2009

Charakteristiky 2D textur

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Miroslava Švub

.....
Václav Pasáček
18. května 2009

Poděkování

Velmi rád bych poděkoval vedoucímu mé bakalářské práce Ing. Miroslavu Švubovi, bez jehož rad a pomoci by tato práce nemohla vzniknout. Také bych chtěl poděkovat své rodině a přátelům za jejich vytvalou morální podporu.

Obsah

Kapitola 1

Úvod

V počítačovém vidění dnes souvisí s texturou mnoho problémů.

Prvním z nich je segmentace obrazu. Protože region obrazu má stejnou texturu, lze na základě klasifikace textur, určit hranice jednotlivých regionů obrazu (neboli provést segmentaci obrazu). Segmentace obrazu se využívá v mnoha oborech lidské činnosti. Jedním z nich je například automatická kontrola kvality povrchu, kdy povrch bez kazu má konstantní texturu a kaz by se projevil změnou textury. Segmentace obrazu může být také využita při zpracování leteckých či družicových snímků, sloužících pro metereologické, hospodářské, geografické, zemědělské nebo jiné účely.

Druhým využitím textury v počítačovém vidění je klasifikace obrazu. Na základě klasifikace textury obrazů(snímků), pořízených medicínskými přístroji lze provést například diagnozů(nebo ji podpořit) srdce, ledvin a mnoha dalších orgánů.

Třetí využití textury spočívá v určení geometrie(tvaru) povrchu, na kterém je textura nanesena. Jako příklad uvedu plech pokrytí rovnoběžnými čárami nebo jinou pravidelnou texturou. Pokud by byl zdeformován, dala by se z deformace textury určit orientace povrchu.

Dalším využitím textury je syntéza povrchu objektu na základě texturního vzoru. To znamená pomocí malého vzorku textury vytvořit dostatečně velkou texturu k pokrytí požadovaného objektu.

Tato práce se zabývá rozpoznáváním(klasifikací) textur za výše uvedeným účelem. Pro rozpoznávání textur je důležité optimálně zvolit texturní příznaky. Pro získání texturních příznaků bylo zvoleno několik variant metody lokálních binárních vzorů (dále jen LBP) a několik druhů porovnání jejich výsledků, aby bylo možno na závěr práce srovnat jejich výsledky a úpěšnost.

Po úvodní kapitole následuje druhá část, která popisuje teorii klasifikace textur, lokálních binárních vzorů metody pro vyhodnocení jejich výsledků. Třetí část přináší návrh výpočtu texturních příznaků i galerie texturních vzorů. Čtvrtá část se zabývá kontkrétní implementací programu pro klasifikaci textur. Pátá část pojednává o experimentálních výsledcích jednotlivých variant LBP a přináší vyhodnocení jejich výsledků pomocí Euklidovské, Bhattacharyyho nebo Mahalanobisovy vzdálenosti. Poslední šestá kapitola obsahuje závěrečné zhodnocení.

Kapitola 2

Teorie

Tato část práce si klade za cíl seznámit čtenáře s teorií nutnou k pochopení daného problému.

2.1 Obrazová data

Jsou to údaje reprezentující obraz. Ukládáním a zpracováním obrazových dat se zabývá počítačová grafika. Tato práce se zabývá pouze 2D obrazovými daty.
Obrazová data lze získat dvěma způsoby:

- Nasnímáním reálného světa - fotoaparátem, skenerem nebo jiným vstupním zařízením. Takováto obrazová data reprezentují konkrétní objekt reálného světa.
- Umělým vytvořením v počítači - grafickými programy lze vytvořit uplně nová nebo upravit již existující obrazová data.

Existují dva různé přístupy k ukládání a reprezentaci obrazových dat:

- Vektorová grafika - obraz se v ní popisuje pomocí množiny grafických primitiv (úseček, křivek, polygonů). U takto reprezentovaného obrazu lze snadno měnit velikost, navíc nezabírá oproti rasterovému obrazu tolik místa v paměti. Vektorový obraz se však velmi špatně pořizuje. Většina grafických aplikací pracuje s touto reprezentací obrazu.
- Rasterová grafika - obraz je popsán pomocí jednotlivých barevných bodu. Tyto body se nazývají **pixely**. Každý pixel v obrazu má svou pevně určenou polohu a barvu. Pixel se skládá ze subpixelů. Každý subpixel nese hodnotu pouze jedné barevné složky. Smíchaním těchto subpixelů získáme barvu samotného pixelu. U takto reprezentovaného obrazu nelze měnit velikost bez ztráty kvality obrazu. Ten navíc zabírá oproti vektorovému obrazu víc místa v paměti. Rasterový obraz lze však velmi jednoduše pořídit - například fotoaparátem. S touto reprezentací obrazu pracuje například monitor, obrazovka, projektor nebo fotoaparát.

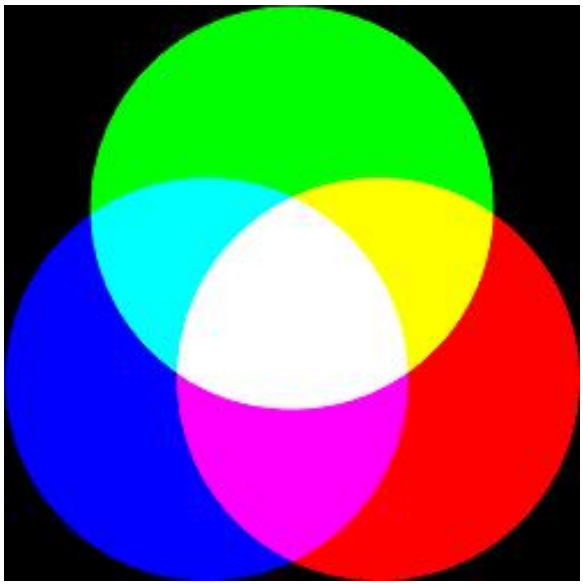
Protože výstupní zařízení pracují pouze s rasterovou grafikou, je nutné před zobrazením převést vektorovou grafiku na rasterovou. Tento proces se nazývá **rasterizace**.

2.1.1 Barevné modely

Barevné modely umožňují pracovat s barvami, a to tak, že realizují jejich míchání. Přestože jich je mnoho, v praxi se používají hlavně ty, které poskytují co nejpřesnější barevný dojem při co nejmenší složitosti modelu. Uvedu zde pouze dva nejpoužívanější.

RGB

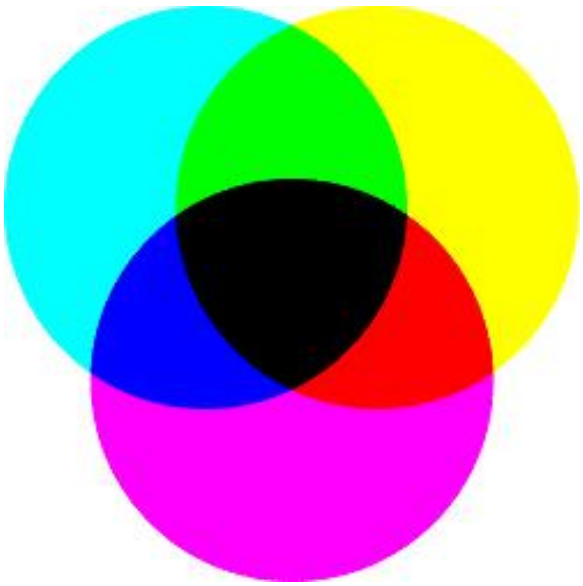
Pracuje s aditivním mícháním barev. Základními barvami jsou červená, zelená a modrá. Smícháním všech těchto barev vznikne bílá barva. Využívá se například v monitorech, obrazovkách a projektorech.



Obrázek 2.1: Aditivní míchání barev

CMY

Pracuje se substraktivním míchání barev. Základní barvy jsou žlutá, azurová a purpurová. Smícháním všech těchto barev vznikne černá barva. Využívá se například v tiskárnách nebo plotrech.



Obrázek 2.2: Substraktivní míchání barev

2.1.2 Obraz ve stupních šedi

Protože je barevný obraz v praxi většinou kódován na 24 bitů(každý barevný kanál má 8 bitů, tzn. 3x8 bitů), dá se snadno převést do stupňů šedi kódovaných na osmi bitech. Proto, i když oko v praxi nerozpozná víc jak 64 stupňů šedi, se pro obrázek ve stupních šedi používá 256 stupňů. Převod obrazu do stupňů šedi se nazývá „Grayscaling“. Tento převod se provádí podle následujícího vzorce, kde I je nová intenzita jasu pixelu, R je intenzita červeného subpixelu, G je intenzita zeleného subpixelu a B je intenzita modrého subpixelu.

$$I = 0.299 * R + 0.587 * G + 0.114 * B$$

Obrázek 2.3: Původní barevný obrázek

Obrázek 2.4: Obrázek převedený do stupňů šedi

2.2 Textura

Povrch každého reálného objektu má svou specifickou povrchovou strukturu, barvu a optické vlastnosti. Texturou se všeobecně myslí popis těchto vlastností. Texturu vytváří opakující se struktura primitiv. Tato primitiva se nazývají texely. To znamená že textura je reprezentována texely(stejně jako je obrázek reprezentován pixely). Pokud počet primitiv mnohem větší než jejich variabilita, tak mají textury výrazné statické vlastnosti. Textury se dají dělit podle:
rozměrů na:

- 2D - plošné struktury
- 3D - prostorové textury
- 3D - prostorové textury, které se mění v čase

Tato práce se zabývá pouze 2D texturami.
reprezentace na:

- Datově - Mapou(obrázkem) uloženou v paměti
- Procedurálně - Matematickou funkcí

V této práci jsou textury reprezentovány datově.
Dále se textury dají dělit podle složky na:

- Vlastní - Vlastní zbarvení povrchu
- Nevlastní - Závislá na osvětlení, vzniká různým osvětlením povrchu

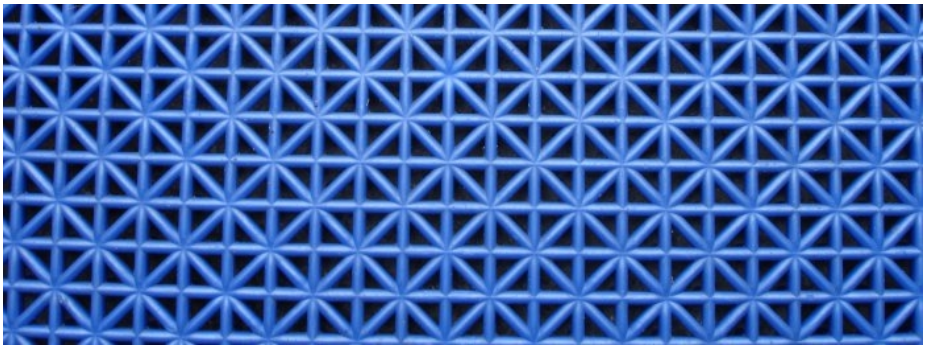
A nakonec je lze rozdělit podle síly na:

- Slabé - Jsou nedekomponovatelné nebo mají nepravidelnou strukturu.

Obrázek 2.5: Slabá textura

- Silné - Jsou dekomponovatelné nebo mají pravidelnou strukturu.

[7].



Obrázek 2.6: Silná textura

2.3 Klasifikace textur

Abychom mohli textury klasifikovat, musíme si položit otázku „Jak popsat vlastnosti textury?“. A právě k tomuto účelu slouží charakteristika textur (neboli také texturní příznaky).

Klasifikace textur je porovnání texturních příznaků se vzory známých textur.

Existují dva základní přístupy, jak klasifikovat texturu:

- Strukturní - Tento přístup se používá, pokud jsou texely(grafická primitiva) dost velké nato, aby je šlo odlišit od pozadí. Jak již název napovídá, tento přístup popisuje strukturu(tj. typy texelů a jejich vzájemné geometrické uspořádání). Struktura lze popsat polygonální sítí nebo gramatikou.
- Statická - Používá se, pokud je počet primitiv mnohem větší než jejich variabilita, protože pak mají textury výrazné statické vlastnosti. Popisuje rozložení intenzit v textuře. Textura je popsána množinou čísel zvanou příznakový vektor.

Tato práce se zabývá statickou klasifikací textur.

Existuje mnoho způsobů získání texturních příznaků, pro přehled jich několi uvedu:

- Histogram tónu a kontrastu textury - Tento příznak (histogram) však bohužel nepostihuje prostorovou strukturu
- Pomocí výkonového spektra:
  Kde P je výkonové spektrum, I je obrazová funkce, F je fourierova transformace

$$P(I) = |F(I)|$$

- Autokorelační funkce - Popisuje velikost texelů a prostorové závislosti v určitých vzdálenostech. I je obrazová funkce, F je fourierova transformace

$$C = F^{-1}(|F(I)|^2)$$

- Heralickovy příznaky:
  Jedna sada těchto příznaků se počítá pro každou konfiguraci pixelů.
  C(i,j) je pravděpodobnost výskytu pixelů s jasovými hodnotami i a j v dané konfiguraci. Konfigurací se v tomto případě myslí vzdálenost a úhel mezi pixely.

  - H1 Energie - je největší pokud pixely pro které se počítá mají totožnou hodnotu jasu.
    $$\sum_{i,j} C^2(i,j)$$
  - H2 Entropie - určuje míru náhodnosti. To znamená, jak moc závisí hodnota následujícího pixelu na hodnotě současného pixelu.
    $$-\sum_{i,j} C(i,j) log_2 C(i,j)$$
  - H3 Kontrast - čím více se střídají body s rozdílnou hodnotou (a čím větší rozdíl mezi nimi je), tím je její hodnota větší.
    $$\sum_{i,j} |i-j| C(i,j)$$
  - H4 Homogenita - je největší, pokud pixely, pro které se počítá, mají totožnou hodnotu jasu.
    $$\sum_{i,j} \frac{C(i,j)}{1+|i-j|}$$
  - H5 Korelace -
    $$\frac{cov(i,j|C)}{std(i|C)std(j|C)}$$

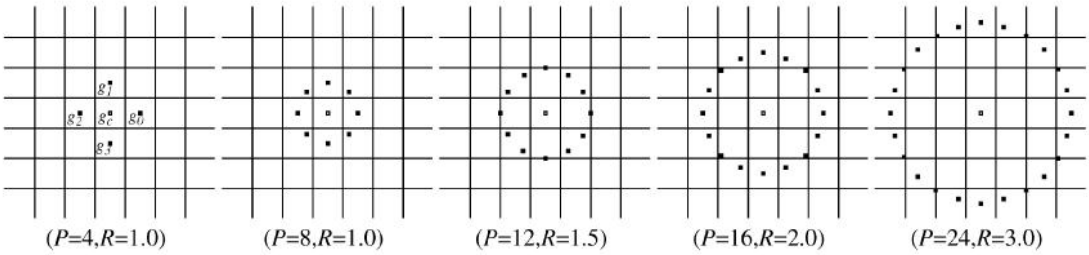
- Lokální binární vzory - tuto metodu získávání texturních příznaků jsem vybral a implementoval v této práci. Proto jí bude věnována celá další podkapitola.

Existuje ještě mnoho složitější metod získávání texturních příznaků (např. Gáborovská analýza, Markovská pole), ale domnívám se, že pro tuto práci postačuje vysvětlení těchto základních. [6]

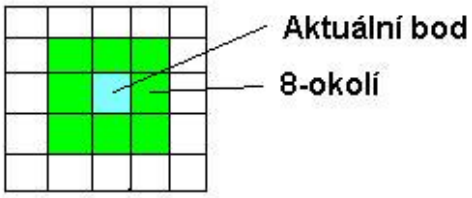
2.4 Lokální binární vzory

Základní myšlenkou metody lokálních binárních vzorů je projití celého obrázku po jednom bodu a ohodnocení jeho okolí nějakým způsobem. Z hodnot okolí (různých druhů okolí) se potom vytvoří histogram četnosti jejich výskytu. Počet bodů v okolí a jejich vzdálenost od aktuálního bodu může být rozdílná. Pokud **P** je počet bodů a **R** vzdálenost od aktuálního bodu, vypadají různá okolí jako na obrázku 2.7. [1]

Hodnota bodů, které přesně nezapadají do mříže pixelů, se interpoluje z okolních bodů. Nejčastěji se počítá LBP z tzv. „8-okolím“ (tj. druhá možnost na předcházejícím obrázku



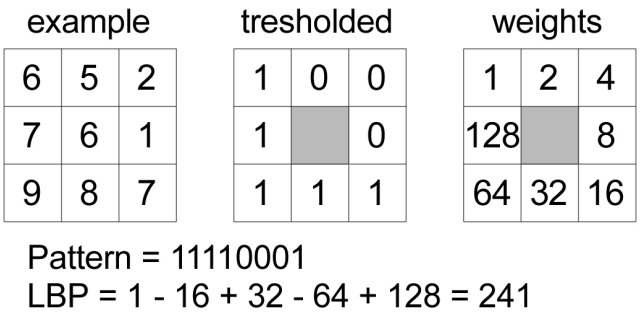
Obrázek 2.7: Různé parametry P a R



Obrázek 2.8: 8-okolí

pro P = 8 a R = 1) a právě takovými variantami LBP se bude tato práce zabývat. Vlastní výpočet LBP vypadá následovně:

Nejdříve se obrázek převede do stupňů šedi(tzv. Grayscalling). Poté se projde po jednom pixelu celý obrázek(kromě krajních pixelů) a 8-okolí každého se prahuje (tzv. Tresholding) hodnotou aktuálního pixelu. Hodnota pixelů prahovaného 8-okolí je znásobena váhou, která je mu přidělena. To znamená že okolí každého pixelu je ohodnoceno číslem od 0 do 255. Z těchto hodnot se potom vytvoří histogram četnosti jejich výskytu.
Nejlépe to ilustruje následující obrázek:



Obrázek 2.9: Výpočet LBP

Texturním příznakem je potom tento 256 položkový histogram, spočítaný pro celou texturu. Toto je **základní varianta LBP**, která je sice výpočetně nenáročná, ale není rotačně invariantní. O tom, jak dosáhnout rotační invariance bude pojednávat následující podkapitola.

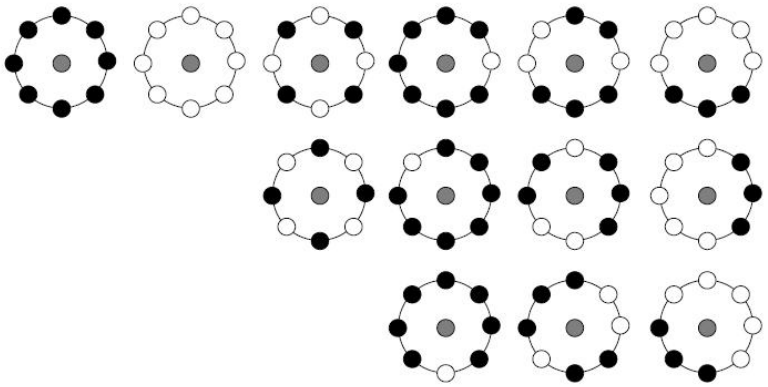
Vylepšení základní metody

Vzhledem k tomu [2] že není všech 256 druhů okolí opravdu nezbytných, je možné použít pouze jistou jejich podmnožinu. Vyvstává však otázka, jak optimálně zvolit tuto podmnožinu. Existuje metoda zvaná „Beam search“, která ji dokáže určit, je ale časově velmi náročná.

Její alternativou je metoda „Uniform patterns“, kde se jako podmnožina vzorů, zvolí pouze vzory s počtem přechodů 0/1 nebo 1/0 rovná 2 (což jsou 00000001, 00000011, 00000111, 00001111, 00011111, 00111111, 01111111) jejich rotované varianty a vzory 00000000, 11111111. Všechny ostatní vzory se sečtou a dají dohromady. Vznikne tedy pouze 59 možných vzorů.

2.5 Rotační invariance

Protože ve skutečnosti nebudem mít vždy příležitost pracovat s texturami, které jsou natočené tak, jako vzorky textur s kterými je porovnáváme, vzniká potřeba vylepšit metodu LBP tak, aby byla rotačně invariantní. Musíme si uvědomit, že pokud bychom metodu LBP použili na pootočený snímek, okolí každého bodu se pouze pootočí. Nejlépe to ilustruje obrázek 2.4 . Na obrázku vidíte vedle sebe 6 různých vzorů a pod nimi některé jejich možné rotované varianty.



Obrázek 2.10: 6 různých vzorků a jejich možné rotace

Rotační invariance LBP se tedy dosahuje tím, že z původní hodnoty a jejich sedmi bitových rotací se vybere nejmenší hodnota. Příklad cyklické bitové rotace:
šipku vpravo jsem použil jako značku cyklické bitové rovace vpravo

00100100 -> 00010010 -> 00001001 -> 10000100 -> 01000010 -> 00100001 -> 10010000 -> 01001000

Obrázek 2.11: Cyklické bitové rotace

V tomto případě by z původního 00100100 vzniklo 00001001. Tato metoda tedy redukuje počet možných vzorů(a tedy i šířku histogramu)z původních 256 vzorů na 36.

Existuje také uniformní verze LBP invariantních vůči rotaci. Ta funguje stejně jako předcházející ale s tou změnou, že počítá pouze vzory s maximálně dvěma přechody 0/1 nebo 1/0. Protože vzory s velkým počtem přechodů se příliš často nevyskytují, je ztráta dat zanedbatelná.

2.6 Metody porovnání texturních příznaků

Protože se k aktuálně spočtenému texturnímu příznaku hledá v databázy „nejpodobnější“, musí existovat metoda, jak určit míru „podobnosti“. Pokud by bylo texturním příznakem pouze jedno číslo, byla by míra „podobnosti“ dvou příznaků rozdíl těchto čísel(čím menší rozdíl, tím podobnější příznaky). Jenže u metody LBP je texturním příznakem histogram četnosti výskytu, a proto není určení míry „podobnosti“ triviální záležitostí. V následujících podkapitolách budou popsána tři možná řešení určení míry „podobnosti“ pro histogramy četnosti.

2.6.1 Euklidovská vzdálenost

V Euklidovském prostoru, který může být definován jako konečněrozměrný unitární prostor nad množinou reálných čísel, je vzdálenost mezi dvěma body $P(p_1, p_2, ..., p_n)$ a $Q(q_1, q_2, ..., q_n)$ definována jako:

$$\sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

A pokud Euklidovskou vzdálenost označíme d , tak bude definována jako:

$$d = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2}$$

Pokud si představíme že histogram šířky N je souřadnice bodu v Euklidovském N rozměrném prostoru, lze zjednodušeně říci, že Euklidovská vzdálenost v tomto případě vyjadřuje hledanou míru „podobnosti“.

Přitom platí, že čím je Euklidovská vzdálenost menší, tím jsou si histogramy „podobnější“. [5]

2.6.2 Bhattacharyyova vzdálenost

Tato veličina se používá ve statistice, kde vyjadřuje „míru podobnosti“ mezi dvěma rozděleními pravděpodobnosti.

Pokud označíme Bhattacharyyovu vzdálenost $BC(p, q)$, kde p a q jsou dvě porovnávaná rozdělení pravděpodobnosti s diskrétní rozložením, bude mít potom vzorec:

$$BC(p, q) = \sum_{x} \sqrt{p(x)q(x)}$$

Pokud budou mít rozdělení pravděpodobnosti stejný počet N položek, lze tento vzorec také zapsat jako:

$$BC(p,q) = \sum_{x=1}^{N} \sqrt{p_x q_x)}$$

Abychom však mohli použít Bhattacharyyovu vzdálenost, musíme mít k dispozici rozložení pravděpodobnosti. To se dá vypočítat následovně:
Z histogramu četnosti $h(n)$ o n položkách se spočítá celkový počet bodů S, pro který byl vytvořen :

$$S = \sum_{x=1}^{n} h(x)$$

Pravděpodobnost P (X = x) se spočítá pro každý bod:

$$P(x) = h(x)/S$$

Přitom platí, že čím je Bhattacharyyova vzdálenost větší, tím jsou si histogramy „podobnější“. [3]

2.6.3 Mahalanobisova vzdálenost

Používá se k určení míry „podobnosti“ mezi neznámou množinou veličin(v našem případě histogramem) a známými množinami veličin(histogramy). Od Euklidovské vzdálenosti se liší tím že počítá i s korelací jednotlivých veličin. Pojmy pochopení následujícího textu jsou důležité tyto pojmy:

- Diagonální matice - matice, která má nenulové prvky pouze na diagonále.
- Korelace - vzájemný vztah mezi veličinami.
- Kovariance - střední hodnota součinu odchylek obou náhodných veličin od jejich středních hodnot.
- Kovarianční matice - veličina charakterizující vícerozměrnou náhodnou veličinu. Je to symetrická matice mající na diagonále rozptyly jednotlivých veličin a na ostatních pozicích jsou kovariance jednotlivých veličin.

Korelace je u vícerozměrných náhodných veličin vyjádřena kovarianční maticí. Pokud je navíc kovarianční matice diagonální(to znamená, že v ní jsou pouze rozptyly a neobsahuje kovariance), vypadá vzorec pro Mahalanobisovu vzdálenost D , kde p a q jsou porovnávané množiny veličin o N prvcích a o je směrodatná odchylka množiny p, následovně:

$$D(p,q) = \sqrt{\sum_{i=1}^{N} \frac{(p_i - q_i)^2}{(o_i)^2}}$$

Abychom mohli použít Mahalanobisovu vzdálenost, musíme mít k dispozici rozložení pravděpodobnosti. Jak se k němu dostat, bylo již popsáno v předchozí podkapitole.

Přitom platí, že čím je Mahalanobisova vzdálenost menší, tím jsou si histogramy „podobnější“. [4]

Kapitola 3

Návrh

Tato kapitola se zabývá návrhem galerie textur a výpočtem texturních příznaků.

3.1 Cíle

Cílem této práce je vybrat vhodný soubor příznaků pro statistický popis textur a implementovat metody pro jejich výpočet. V této práci bylo k získávání texturních příznaků vybráno několik variant metody Lokálních binárních vzorů, která bude dále označována pouze zkratkou LBP. První z variant bude **základní metoda LBP**, která není rotačně invariantní a v následujícím textu bude označována jako **Classic**. Další variantou bude základní rotačně invariantí varianta, která bude v následujícím textu označována jako **Shifted** a její uniformní varianta označená slovem **Invariant**. Všechny tyto varianty jsou popsány v předcházející kapitole v podkapitole pojednávající o LBP. Cílem této práce také je vytvořit program, který bude na základě trénovací galerie textur pomocí některé z variant LBP rozpoznávat textury z testovací sady.

3.2 Galerie textur

Aby bylo možné spočítat úspěšnost různých variant metody LBP, je nutné vytvořit galerii textur. Tato galerie musí obsahovat nejen trénovací sadu textur, ale i druhou testovací sadu. Testovací sada však ještě musí být rozdělena na další dvě části - a to na ty, které nejsou oproti trénovací sadě textur pootočeny a na ty, které jsou oproti trénovací sadě textur pootočeny. Všechny vzorky textur(soubory s obrázky) jsou uloženy v samostatné složce nazvané **img**. Tyto vzorky jsou pojmenovány podle následující konvence:
Nx.jpg N - pořadové číslo textury v galerii.
x - na tomto místě může být _ to by znamenalo, že je to vzorek z testovací sady textur, který není otočený oproti trénovacím vzorku. Pokud za číslem následuje r, je to otočený vzorek z testovací sady textur. Pokud už za číslem nenásleduje _ nebo r je to vzorek z trénovací sady textur.

Jak ale získat dva vzorky textury, které nejsou proti sobě pootočené a třetí, který je otočený proti původnímu o 90 stupňů?

Tento problém jsem vyřešil tak, že jsem stáhnul 40 různých vzorků textur ze stránky: http://www.texturewarehouse.com/gallery/ . Každý vzorek jsem poté horizontálně rozdělil na dva přibližně stejně velké části. Vrchní část jsem použil jako trénovací vzorek textury a

Obrázek 3.1: 22.jpg

Obrázek 3.2: 22_.jpg

spodní část jako testovací vzorek textury. Třetí pootočený vzorek jsem vytvořil otočením původního testovacího vzorku o 90 stupňů.

3.3 Návrh řešení

Existují dva základní přístupy, kterými lze dosáhnout požadovaného cíle:

- Vytvořit pouze program, který porovná 2 vzorky textur a jehož výstupem bude číslo, které udává míru „podobnosti“ obou textur. Tento program poté skriptem spustit tolikrát, kolik máme známých vzorků textur, vždy pro rozpoznávaný vzorek textury a pro jednu ze známých vzorků textur. Na základě míry „podobnosti“ poté rozhodnout, který ze známých vzorků je nejpodobnější.
- Vytvořit program, který spočítá LBP(histogram četnosti výskytu) vždy pouze pro jeden soubor a ten poté podle parametrů, s kterými je spuštěn, buď uloží do souboru a nebo ho porovná se všemi LBP, které už jsou v souboru uloženy. Tento způsob umožňuje vytvořit v jednom souboru jakousi knihovnu, ve které jsou uloženy LBP všech vzorků z trénovací sady.

Zvolil jsem druhý způsob, který je efektivnější, protože nemusím při každém spuštění načítat všechny texturní vzory z trénovací sady a počítat jejich LBP (histogram četnosti výskytu),

Obrázek 3.3: 22r.jpg

neboť ty už jsou spočítány a uloženy v souboru. S tím souvisí skutečnost že po vytvoření souboru s LBP již nepotřebujeme trénovací sadu textur.

3.3.1 Knihovna vzorů LBP

Je nutné upozornit že knihovnu vzorů ale kvůli kompatibilitě a správnosti načítání nelze ukládat do obyčejného binárního souboru. Proto tuto knihovnu vzorů ukládáme do souboru typu XML.

XML je obecný značkovací jazyk, který umožňuje popsat strukturu dokumentu pomocí značek. XML dokument má stromovou strukturu. Každá implemetovaná varianta LBP (tzn. Classic, Shifted, Invariant) bude mít vlastní knihovnu histogramů četnosti výskytu, i přestože by stačila jedna a to Classic a zbývající 2 by se z ní dali spočítat. Je to kvůli tomu, abychom potom nemuseli ve všech variantách pracovat se stejnou testovací sadou textur. Tyto tři knihovny jsou pojmenováné *classic_samples.xml*, *shifted_samples.xml* a *invariant_samples.xml*. V XML souboru je nejdříve uložen typ vzorků textur, poté už následují jednotlivé vzorky textur. U každého vzorku je uvedeno jméno, pod kterým byl uložen do knihovny, jeho rozměry(rozměry obrázku v němž byl uložen) a poté histogram četnosti výskytu.

I když funkce pro porovnání texturních vzorů pracují s rozložením pravděpodobnosti místo histogramu četnosti, ať už z toho důvodu, že Bhattacharyova a Mahalonobisova

```
<?xml version="1.0" encoding="iso-8859-2" ?>
- <Classic_samples>
  - <sample nazev="1" width="700" height="285">
      <histogram hodnota="56394" poradi="0" />
      <histogram hodnota="2179" poradi="1" />
      <histogram hodnota="1376" poradi="2" />
      <histogram hodnota="1366" poradi="3" />
      <histogram hodnota="1842" poradi="4" />
      <histogram hodnota="126" poradi="5" />
      <histogram hodnota="1216" poradi="6" />
      <histogram hodnota="8492" poradi="7" />
      <histogram hodnota="1354" poradi="8" />
      <histogram hodnota="42" poradi="9" />
                    :
                    :
      <histogram hodnota="12" poradi="250" />
      <histogram hodnota="133" poradi="251" />
      <histogram hodnota="356" poradi="252" />
      <histogram hodnota="1185" poradi="253" />
      <histogram hodnota="93" poradi="254" />
      <histogram hodnota="1322" poradi="255" />
    </sample>
  - <sample nazev="2" width="571" height="258">
      <histogram hodnota="7344" poradi="0" />
      <histogram hodnota="767" poradi="1" />
      <histogram hodnota="1120" poradi="2" />
      <histogram hodnota="1532" poradi="3" />
      <histogram hodnota="740" poradi="4" />
```

Obrázek 3.4: XML knihovna

vzdálenost je vyžadují, nebo jenom proto, aby porovnávané vzorky nemusely mít stejnou velikost, do XML se ukládají histogramy četnosti výskytu. To je z toho důvodu možnosti nejednoznačné interpretaci datových typů s plovoucí desetinnou čárkou v XML. Ne všechny XML knihovny mají totiž funkce na uložení a načtení datových typů s plovoucí desetinnou čárkou, a tak by muselo být toto číslo uloženo jako textový řetězec, což není optimální řešení.

3.4 Postup výpočtu texturních příznaků

Vlastní výpočet texturních příznaků probíhá následovně: Po načtení obrázku a jeho převedení do stupňů šedi(to se samozřejmě provádí pouze pokud je obrázek barevný, když už je ve stupních šedi , tak se nepřevádí), se celý obrázek kromě krajních pixelů projde a pro každý pixel se spočítá podle již dříve uvedeného postupu hodnota jeho okolí. Pokud se jedná **Classic** variantu LBP, tak se přímo tato hodnota započítá do histogramu četnosti. Pokud je to **Shifted** varianta, rotuje se bitově tato hodnota vpravo na nejmenší hodnotu. Když je to varianta **Invariant**, tak se u rotované hodnoty spočítá počet přechodů 0/1 a 1/0 a do histogamu četnosti výskytu se započítá pouze, pokud je počet těchto přechodu dva a méně. Texturním příznakem u LBP je právě tento histogram četnosti.

Kapitola 4

Implementace

Tato kapitola se zabývá implementací programu pro rozpoznávání textur na základě charakteristiky textur získaných metodou LBP. Program je implementován v jazyce C++. Pro uložení a nahrávání galerie textur do souboru typu XML byl vybrán parser **TinyXML** z toho důvodu, že jeho „instalace“ je minimální (v tomto případě stačí pouze připojit několik hlavičkových souborů) a navíc je jednoduchá na obsluhu. Tyto hlavičkové soubory se nacházejí v adresáři *tinyxml*. Dokumentace i odkaz k jeho stáhnutí jsou na stránce http://www.grinninglizard.com/tinyxml/. Pro práci s obrázky byla vybrána knihovna **libjpeg** , neboť plně postačuje našemu účelu, kterým je v tomto případě pouze načtení obrázku do paměti, protože další práce s ním je už náplní této práce. Navíc je její instalace a práce s ní jednoduchá. OpenGL by bylo pro tento účel zbytečně robustní. Dokumentace i odkaz k jeho stáhnutí jsou na stránch http://www.ijg.org/.

4.1 Ovládání programu

Program byl vytvořen jako konzolová aplikace, to znamená, že nemá uživatelské rozhraní. To je proto aby mohl být spouštěn dalšími programy nebo ve skriptech. Reaguje na čtyři parametry(nultým je jméno spuštěného souboru)nebo při mazání pouze na tři. Pokud mu je předán jiný počet parametrů, nebo je nějaký neplatný, program to oznámí chybovou hláškou a skončí.

První parametr určuje druh prováděné operace, jakýkoliv jiný řetězec je na tomto místě neplatný:

- -add Přidání texturního vzoru do galerie textur.
- -cmp Porovnání texturního vzoru se všemi vzory v galerii textur.
- -del Smazání texturního vzoru z galerie textur.

Druhý parametr určuje variantu LBP, jakýkoliv jiný řetězec je na tomto místě neplatný:

- -classic Základní metoda LBP, která není rotačně invariantní.
- -shifted Základní rotačně invariantní metoda LBP.
- -invariant Uniformní rotačně invariantní metoda LBP.

Třetím parametrem je jméno souboru se vzorkem textury. Typicky je to nějaký soubor ze složky **img**. Jenom pokud je první druh prováděné operace mazání vzorku textury, je třetím parametrem jméno mazaného vzorku.

Čtvrtý parametr je závislý na prvním, tzn. na druhu prováděné operace. Pokud je druh operace přidání, znamená čtvrtý parametr jméno, pod kterým se tento texturní vzorek uloží do galerie. Je-li druhem operace porovnání, tak tento parametr má význam druh porovnání.

Druhy porovnání jsou následující, jakýkoliv jiný řetezec je na tomto místě neplatný:

- -n Porovnání na základě Euklidovské vzdálenosti.
- -b Porovnání na základě Bhattacharyovy vzdálenosti.
- -m Porovnání na základě Mahalanobisovy vzdálenosti.

Smazání čtvrtý parametr nemá.
Příklad spuštění, ve kterém se **přidává** do galerie textur **základní metody LBP**, vzorek textury načtený ze souboru **22.jpg** nacházející se v adresáři **img** pod jménem 22:

- *./lbp -add -classic img/22.jpg 22*

Smazání čtvrtý parametr nemá.
Příklad spuštění, ve kterém se **porovnávává** načtený vzorek ze souboru **22.jpg** se vzorky v galerii textur pomocí **invariantní unifomní metody LBP**. Porovnávání se provádí na podle **Bhattacharyovy vzdálensoti**:

- *./lbp -cmp -invariant img/22.jpg -b*

4.2 Pomocné skripty

Součástí implementace je také sada pomocných skriptů, bez kterých by sice implementace fungovala, ale práce s ní by byla nepohodlná. Například vytvoření knihovny texturních vzorů by postupným manuálním spouštěním programu pro všechny vzorky textur by bylo neefektivní a zbytečně časově náročné. Skript pro vytvoření knihovny texturních vzorů vykoná tuto práci místo uživatele tak, že postupně spouští program pro všechny vzorky textur. Jméno toho skriptu pro základní variantu LBP se nazývá *AddClasSamples.sh*, pro ostatní varianty to jsou *AddShiftSamples.sh* a *AddInvariantSamples.sh*.

Druhý typ skriptů slouží k porovnání všech vzorků z testovací sady se vzorky v knihovně texturních vzorů. Důvodem vytvoření těchto skriptů je opět zefektivnění práce s programem. Tyto skripty jsou pojmenovány:

$$CmpDRUHLBPSamplesDRUHPOROVNANIR.sh$$

Kde DRUHLBP může být jeden z trojice *Clas, Shift, Invariant*
DRUHPOROVNANI je jeden z výše uvedené trojice v-*n, -b, -m*
Pokud navíc za řetězcem DRUHPOROVNANI následuje písmeno R, znamená to, že porovnáná testovací sadu, která je otočená o 90 stupňů oproti trénovací sadě. Pokud zde písmeno R není, pracuje s neotočenou testovací sadou.
Takže jméno skriptu je potom například *CmpShiftSamplesBhatR.sh*.

4.3 Volba datových typů

Pro reprezentaci obrázku byla vytvořena třída readed_img, která obsahuje histogram četnosti výskytu okolí v LBP. Tento histogram je implementován jako pole hodnot typu int. Dále obsahuje pole typu double, ve kterém je uloženo rozložení pravděpodobnosti výskytu okolí v LBP. Také jsou v ní uloženy šířka a výška a jméno načteného vzorku. Samozřejmě obsahuje i ukazatel na obrázek, který je už ve stupňích šedi.

Další implementovaná třída se nazývá vector_readed_img. Tato třída obsahuje vektor readed_img a iterátor do tohoto vektoru. K načtení obrázku se používají datové struktury jpeg_decompress_struct jpeg_error_mgr definované v knihovně **libjpeg** . Z této struktury se poté obrazová data uloží do jednorozměrného pole typu unsigned char.

Při ukládání a načítaní histogramů z XML jsou použity datové typy TiXmlDocument, pro XML dokument a TiXmlElement pro XML element, definované v knihovně **TinyXML**.

4.4 Vlastní implementace

Program byl vyvíjen a testován na OS **Linux Mandriva 2008** za použití již dříve zmiňovaných knihoven **TinyXML** a **libjpeg** , ale díky přenositelnosti implementace a po nainstalovaní příslušných knihoven bude funkční i na OS Windows. Celá vlastní implementace je ve zdrojovém souboru *lbp.cpp*.

Program po spuštění načte a zpracuje parametry, s kterými byl spuštěn. Podrobným popisem těchto parametrů se zabývá podkapitola Ovládání programu, proto se zde jimi už nebudu zabývat. Po vyhodnocení toho, s jakými parametry byl program spuštěn, se na základě druhu požadované operace dělí implementace na tři části - přidání, porovnání nebo smazání vzorku.

4.4.1 Přidání texturního vzoru do galerie

Přidání texturního vzoru do galerie textur probíhá následovně:

Vytvoříme proměnné typu jpeg_decompress_struct a jpeg_error_mgr, to jsou struktury definované v knihovně **libjpeg** . Do nich následně načtem obrazová data ze souboru. Poté vytvoříme instanci třídy readed_img nazvanou *obrazek* a do jejích prvků *width* a *height* uložíme rozměry načtených obrazových dat. Poté alokujeme v místo paměti, do něhož načtem pomocí funkce jpeg_read_scanlines,která je součástí knihovny **libjpeg** , obrazová data. Tato funkce načítá obrazová data ze struktury typu jpeg_decompress_struct po jednotlivých řádcích. Ty potom ukládá do paměti, kterou jsme si předtím alokovali, jako jednorozměrné pole a dají se tedy lépe procházet po jednotlivých pixelech, než kdyby byly ve struktuře typu jpeg_decompress_struct. Podle toho, jestli jsou obrazová data barevná nebo ve stupních šedi, se zavolá jedna z metod třídy readed_img. Tou je *GreyScale* pro barevný obrázek nebo *ReadGrey* pro obrázek ve stupních šedi. V těchto metodách se do prvku *picture* instance *obrazek* uloží ukazatel na obrazová data ve stupních šedi reprezentována jako jednorozměrné pole. V metodě *ReadGrey* se tedy pouze zkopírují, zatímco v metodě *GreyScale* se musí převést do stupňů šedi.

Následně se podle zvoleného druhu LBP spustí jedna z metod pro vytvoření histogramu třídy readed_img. Tou je Make_Norm_Histogram pro základní rotačně neinvariantní metodu LBP, Make_Shift_Histogram pro rotačně invariantní metodu LBP a Make_Invariant_Histogram pro uniformní rotačně invariantní metodu LBP. To, jak přesně se spočítá LBP a histogram četnosti je popsáno v kapitole Teorie - Lokální binární vzory. Vytvořený histogram četnosti

se uloží do prvku *histogram* instance třídy readed_img. Poté se vytvoří instance třídy vector_readed_img, nazvaná *histogramy*, a pomocí její metody Nacti_Hist se načte z příslušného XML souboru (jehož jméno záleží na druhu LBP) knihovna histogramů texturních vzorů. Následně se zavolá metoda Vloz_Dalsi v které se tato knihovna projde, abychom zjistili, zda se v ní již nenachází vzorek se stejným jménem jako aktuálně ukládaný vzorek. Pokud by se v ní takový vzorek nacházel, aktuální vzorek se nepřidá, vzniklá situace bude ohlášena uživateli a program skončí. V opačném případě se však aktuální histogram přidá do instance třídy vector_readed_img. Takto doplňěná knihovna histogramu se poté metodou Uloz_Hist uloží do příslušného XML souboru.

Pokud se přidání texturního vzoru nepovede, například protože v knihovně už existuje vzor se stejným jménem, je výstupem programu oznámení, že se přidání nezdařilo. Pokud se přidání povede, program nic nevypíše.

4.4.2 Porovnání texturního vzoru se všemi vzory v galerii

Začátek porovnání texturního vzoru probíhá velmi obdobně jako přidání texturního vzoru. To znamená, že zde také, jako v předchozím případě, probíhá načtení obrazových dat do sturktur knihovny **libjpeg** , z kterých se následně pomocí funkce pro čtení obrazových dat po řádcích uloží do paměti reprezentovány jako jednorozměrném pole.

Následuje převod obrázku z barevného do stupňů šedi a vytvoření histogramu četnosti LBP, podle zvoleného druhu LBP. Poté se z příslušného XML souboru načte knihovna histogramu četnosti. Až k tomuto kroku byl postup stejný jako u přidávání texturního vzoru. Dále se však liší.

Pro aktuální vzorek, k němuž hledáme v knihovně „nejpodobnější“, a jehož histogram četnosti je uložen v instanci třídy readed_img, se spustí metoda Make_Perc_Histogram, která do prvku perc_histogram uloží rozložení pravděpodobnosti výskytu LBP. Toto rozložení pravděpodobnosti se samozřejmě vypočítá z histogramu četnosti výskytu. Pro knihovnu texturních vzorů načtenou v instanci třídy vector_readed_img, se spustí metoda Make_Perc_Histogram, která pro všechny vzorky v knihovně, obdobně jako v předchozím případě, vypočíta rozložení pravděpodobnosti a uloží ho do prvku perc_histogram.

Následně se podle zvoleného druhu porovnání spustí jedna z metod třídy vector_readed_img sloužících pro porovnávání histogramů. Tou je porovnej_norm_histogramy pro porovnání na základě Euklidovská vzdálenosti, porovnej_bhat_histogramy pro porovnání na základě Bhattacharyyovy vzdálenosti a porovnej_maha_histogramy pro porovnání na základě Mahalanobisovy vzdálenosti. Tyto metody pracují tak, že projdou vektor, do kterého byly načteny, rozložení pravděpodobnosti všech texturních vzorů z knihovny a pro každý spočítají míru „podobnosti“ s aktuálně načteným vzorem. Tato míra „podobnosti“ je právě určena v jednotlivých metodách Euklidovskou, Bhattacharyyovou nebo Mahalanobisovou vzdáleností. Přičemž u Euklidovské a Mahalanobisovy vzdálenosti platí, že čím **menší** je, tím „podobnější“ jsou si rozložení pravděpodobnosti(a tím pádem i vzory textur které reprezentují). Zatímco u Bhattacharyyovy vzdálenosti platí, že čím **větší** je, tím „podobnější“ jsou si rozložení pravděpodobnosti. Všechny tyto metody vracejí jako svou návratovou hodnotu, číslo vyjadřující právě onu míru „podobnosti“ a jméno vzoru z knihovny, který je s porovnávaným vzorem „nejpodobnější“. Výstupem programu je v tomto případě jedna řádka, která vypadá následovně:

- rozdíl: *velikost rozdílu* nejpodobnejsi: *jméno nejpodobnějšího*

Kde *velikost rozdílu* je číslo představující míru „podobnosti“ a *jméno nejpodobnějšího* je jméno, pod kterým je je v texturní galerii uložen „nejpodobnější“ vzorek.

4.4.3 Smazání texturního vzoru

Smazání texturního vzoru z galerie textur probíhá tak, že zavoláním metody Nacti_Hist instance třídy vector_readed_img nazvané *histogramy* se načte knihovna histogramů texturních vzorů z příslušného XML souboru (jehož jméno záleží na druhu LBP). Následně se zavolá její další metoda, pojmenovaná *Smaz*, která projde načtenou knihovnu histogramů texturních vzorů a pokud ní nalezne i histogram vzoru, který chceme smazat, provede jeho vymazání. Pokud ale takovýto vzorek nenalezne, bude o tom program uživatele informovat výpisem na konzoly a poté skončí - v tomto případě se nic neukládá. Jinak je ale pozměněná knihovna histogramů uložena metodou Uloz_Hist do příslušného XML souboru.

Pokud je požadovaný vzorek v knihovně, je výstupem programu oznámení o úspěšném smazání vzorku z knihovny. Jinak je výstupem oznámení o tom, že požadovaný vzorek nebyl nalezen.

Kapitola 5

Experimentální výsledky

Experimentem se zde myslí spuštění programu s cílem nalezení v galerii vzorů „nejpodobnější“ vzor k testovanému vzoru, pro který se program spouští.

Úspěšným experimentem se zde myslí experiment, kdy byl k testovanému vzorku vyhodnocen jako „nejpodobnější“ vzorek, s kterým byl vytvořen z jednoho původního obrázku. Takovéto vzorky mají stejné číslo. Více o tom naleznete v podkapitole Návrh - Galerie textur.

Kombinací se v této kapitole myslí jeden druh LBP, jeden druh porovnání a jedna část testovací sady textur. Protože jsou implementovány tři druhy LBP (základní rotačně neinvariantní, základní rotačně invariantní a uniformní rotačně invariantní), tři druhy porovnání (podle Euklidovské vzdálenosti, Bhattacharyovy vzdálenosti nebo Mahalanobisovy vzdálenosti) a dvě části testovací sady textur (první je neotočená oproti trénovací sadě textur a druhá je otočená o 90 stupňů), je výsledný počet kombonací následující:

$$3 * 3 * 2 = 18$$

Tři druhy LBP, tři druhy porovnání a dvě části sady textur.

První věc, kterou je nutné udělat před začátkem experimentování, je naplnit knihovnu texturních vzorů trénovací sadou textur. Poté se postupně provádí experimenty pro všechny vzorky textur z testovací sady (pro její otočenou i neotočenou část), pro ruzné druhy LBP a pro různé druhy porovnávání. Pro usnadnění experimentů byly použity skripty popsané v podkapitole Implementace - Pomocné skripty.

5.1 Tabulka výsledků

Za účelem experimentování byla vytvořena trénovací galerie čítající 40 vzorků textur a testovací galerie, jejíž obě části mají po 40 vzorcích textur. Dohromady má tedy testovací galerie 80 vzorků textur. Pro každou kombinaci je tedy provedeno 40 experimentů - pro každý vzorek jeden. Úspěšnost každé kombinace je v tabulce vyjádřena v procentech a spočítá se podle následujícího vzorce:

$$P = M/N * 100$$

Kde M je počet úspěšných experimentů, N je celkový počet experimentů a P je úspěšnost v procentech.

Úspěšnosti všech kombinací jsou znázorněny v následujících tabulkách. Pro přehlednost jsou výsledky rozděleny do dvou tabulek - jedna pro neotočenou část testovací sady vzorků

a druhá pro otočenou část testovací sady vzorků.
V prvním sloupci jsou druhy LBP:

- Classic - základní metoda LBP, která není rotačně invariantní.
- Shifted - základní rotačně invariantní metoda LBP.
- Invariant - uniformní rotačně invariantní metoda LBP.

V prvním řádku jsou druhy porovnání:

- Euclidean - porovnání na základě Euklidovské vzdálenosti.
- Bhattacharya - porovnání na základě Bhattacharyovy vzdálenosti.
- Mahalonobis - porovnání na základě Mahalanobisovy vzdálenosti.

Výsledky pro neotočenou část testovací sady vzorků

********	**Euclidean**	**Bhattacharya**	**Mahalonobis**
Classic	95%	97,5%	97,5%
Shifted	87,5%	90%	87,5%
Invariant	82,5%	85%	82,5%

Výsledky pro otočenou část testovací sady vzorků

********	**Euclidean**	**Bhattacharya**	**Mahalonobis**
Classic	30%	37,5%	40%
Shifted	87,5%	90%	87,5%
Invariant	82,5%	85%	82,5%

5.2 Podrobný rozbor výsledků pro neotočené vzorky textur

Výsledky těchto experimentů obsahuje první tabulka. Z této tabulky také jasně vyplývá několik zásadních věcí:

- První z nich je, že pro neotočené vzorky textur je nejúspěšnějším z testovaným druhů LBP základní rotačně neinvariantní metoda LBP. Tato metoda byla při mých experimentech o 7,5% až 10% lepší než druhá - základní rotačně invariantní metoda LBP. Také byla o 12,5% až 15% lepší než třetí - uniformní rotačně invariantní metoda LBP.
- Další je, že výsledky nezáleží pouze na druhu LBP, ale také na druhu porovnání. Rozdíly mezi různými druhy porovnání jsou do 2,5%. To znamená že konečné výsledky ovlivní druh porovnání méně, než druh LBP. Pro základní rotačně neinvariantní metoda LBP je nejlepší druh porovnání pomocí Bhattacharyovy vzdálenosti nebo Mahalanobisovy vzdálenosti, které mají shodnou úspěšnost 97,5%, což je vůbec nejlepší

dosažená úspěšnost. Porovnání na základě Euklidovské vzdálenosti má o úspěšnost 95% což je o 2,5% horší než u obou předchozích druhů porovnání. U obou rotačně invariantních druhů LBP (základní i uniformní) bylo nejúspěšnější porovnání na základě Bhattacharyovy vzdálenosti s úspěšností o 2,5% větší, nežli porovnávání na základě Mahalanobisovy nebo Euklidovské vzdálenosti, které měly stejnou úspěšnost.

- Další věcí, která je ale patrná pouze při pohledu na podrobné výsledky je, že u klasické rotačně neinvariantní metody LBP mají porovnání pomocí Bhattacharyovy i Mahalanobisovy vzdálenosti stejnou procentuální úspěšnost 97,5% - což je při 40ti experimentech jeden neúspěšný - ale každý druh porovnání chybně rozpozná jiný vzorek. Pro připomenutí uvádím, že vzorky v galeriích jsou očíslovány od 1 do 40, přičemž by program měl jako „nejpodobnější“ k vzorku s číslem 1 z testovací sady přiřadit vzorek s číslem 1 z trénovací sady a tak dále až do 40. Při porovnání pomocí Bhattacharyovy vzdálenosti se chybně rozpozná vzorek číslo 15 a jako „nejpodobnější“ se mu přiřadí vzorek číslo 40. Zatímco při porovnání pomocí Mahalanobisovy metody se chybně rozpozná vzorek číslo 12 a jako „nejpodobnější“ se mu přiřadí vzorek číslo 37. Porovnání na základě Euklidovské vzdálenosti vyhodnotí oba tyto vzorky chybně. To znamená že vzorku číslo 15 přiřadí jako „nejpodobnější“ vzorek s číslem 40 a vzorku číslo 12 přiřadí vzorek číslo 37. Proto má také porovnání na základě Euklidovské vzdálenosti o 2,5% nižší úspěšnost.

  Chybně rozpoznané texturní vzory v pořadí:
  testovaný vzor, vzor, ke kterému měl být přiřazen a vzor, ke kterému byl programem přiřazen:

Obrázek 5.1: Vzorek z testovací sady číslo 15

- Další věcí je, že u základní rotačně invariantní metody LBP se nestala stejná věc jako v předchozím případě, to znamená že se chybně rozpoznávají stejné vzory. To znamená že při všech druzích porovnání se chybně rozpozná vzorek číslo 13 jako číslo 28, 22 jako 40, 32 jako 26 a 36 jako 13. K tomu se navíc u Euklidovské a Mahalanobisovy vzdálenosti chybně rozpozná i vzorek číslo 10 jako číslo 4.

- Poslední věc se týká uniformní rotačně invariantní metody LBP, kde se opět chybně rozpoznávají různé vzory(tak jako u základní rotačně neinvariantní metody), ale ne všechny a u každého druhu porovnání. Liší se jich *přibližně* polovina(přibližně protože každá metoda má jinou úspěšnost a nelze to tedy říct pro všechny dohromady přesně). Protože má tato metoda výrazně nižší úspěšnost - 82,5% až 85%, nebudu zde uvádět

Obrázek 5.2: Vzorek z trénovací sady číslo 15

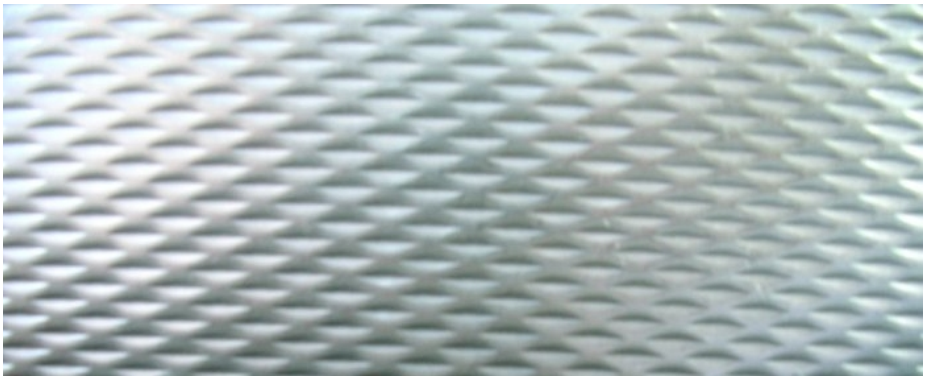
Obrázek 5.3: Vzorek z trénovací sady číslo 40

které konkrétní vzory byly u jakého druhu porovnání chybně vyhodnoceny. Bylo by to nepřehledné a nemělo by to velký praktický užitek.

Za zmínku stojí pouze prohození rozpoznání vzorků 4 a 10. To znamená že vzorek z testovací sady s číslem 4 byl chybně rozpoznán jako vzorek číslo 10, zatímco vzorek z testovací sady s číslem 10 byl rozpoznán jako vzorek číslo 4.

Obrázek 5.4: Vzorek z testovací sady číslo 12

Obrázek 5.5: Vzorek z trénovací sady číslo 12



Obrázek 5.6: Vzorek z trénovací sady číslo 37

5.3 Podrobný rozbor výsledků pro otočené vzorky textur

Výsledky těchto experimentů obsahuje druhá tabulka. Z ní je jasně patrné, že ačkoliv základní rotačně neinvariantní metoda LBP se svou úspěšností 30% až 40% selhává, jsou i zde rozdíly mezi jednotlivými druhy porovnání. Avšak zatímco jinde se rozdíly mezi různými druhy porovnání pohybují maximalně do 2,5%, zde je tento rozdíl až 10%. Nejlépe dopadla porovnání na základě Mahalanobisovy vzdálenosti, o 2,5% hůře dopadlo porovnání na základě Bhattacharyovy vzdálenosti a o dalších 7,5% huře dopadlo porovnání na základě Euklidovské vzdálenosti.

Výsledky obou dvou rotačně invariantních metod (základní i uniformní) LBP dopadly u otočených vzorků stejně jako s neotočenými vzorky, což samozřejmě dokazuje rotační invarianci obou dvou metod LBP. Jinak ale nemá smysl zde jejich výsledky podrobně rozebírat, protože to již bylo učiněno v předchozí podkapitole.

Obrázek 5.7: Vzorek z trénovací sady číslo 4

Obrázek 5.8: Vzorek z testovací sady číslo 4

Obrázek 5.9: Vzorek z trénovací sady číslo 10

Obrázek 5.10: Vzorek z trénovací sady číslo 10

Kapitola 6

Závěr

Cílem práce bylo získání charakteristiky 2D textur a následné rozpoznávání(klasifikace) na základě této charakteristiky. Pro statický popis textur byla jako způsob jejich získání zvolena metoda Lokálních binárních vzorů v šedotonovém obraze. Protože má tato metoda několik variant (některé jsou rotačně invariantní a některé ne), kladla si tato práce za cíl také vzájemné porovnání těchto variant LBP.

Protože „klasifikací“ se zde myslí porovnávání testovaného vzoru se všemi vzory v galerii textur za účelem nalezení „nejpodobnějšího“, je nutné zvolit optimální způsob nalezení „nejpodobnějšího“ vzoru. Nalezením „nejpodobnějšího“ se v tomto kontextu myslí způsob porovnání dvou histogramů respektive dvou rozložení pravděpodobnosti. Protože nikde v odborné literatuře jsem nenalezl vyhodnocení toho, která metoda je pro porovnávání histogramů „nejlepší“, bylo dalším cílem práce srovnání několika metod vhodných pro porovnání histogramů.

Pro klasifikaci textur, které nejsou natočeny oproti trénovací sadě, se podle předpokladů stala nejlepší *základní rotačně neinvariantní metoda LBP* s úspěšností 95% až 97,5%. Obě další implementované metody sice také lze použít, ale jejich úspěšnost je o 7,5% (u základní rotačně invariantní metody) respektive 12,5% (u uniformní rotačně invariantní metody) horší.

Při použití základní rotačně neinvariantní metody LBP je nejlepším druhem porovnání na základě Mahalonobisovy vzdálenosti, těsně následované porovnáním na základě Bhattacharyovy vzdálenosti a nejhorším je porovnání na základě Euklidovské vzdálenosti.

Z výsledků experimentů je také jasné, že základní rotačně neinvariantní metoda LBP je pro pootočené vzorky nepoužitelná. Pro pootočené vzorky je nejlepší *základní rotačně invariantní metoda LBP* s úspěšností 90% až 87,5%, která je o 5% lepší než *uniformní rotačně invariantní metoda LBP*. Pro obě metody LBP je nejlepším druhem porovnání na základě Bhattacharyovy vzdálenosti, které je v těchto případech o 2,5% úspěšnější než porovnání na základě Mahalonobisovy nebo Euklidovské vzdálenosti.

Při použití rotačně neinvariantní metody je tedy nejlepší porovnání na základě Mahalonobisovy vzdálenosti, zatímco pro rotačně invariantní metody je nejlepší porovnání na základě Bhattacharyovy vzdálenosti. Porovnání na základě Euklidovské vzdálenosti vychází z těchto experimentů s nejhorším výsledkem.

Práce jasně ukázala že úspěšnost klasifikace textur nezáleží pouze na zvoleném druhu LBP, ale také na způsobu jejich porovnání.

Možným vylepšením by bylo implementovat jinou metodu LBP nebo další druh porovnání. Dalším vylepšením by také mohlo být provést porovnání na základě dvou a více kritérií

(například Mahalonobisovy a současně i Bhattacharyovy vzdálenosti). Program by se také dal vylepšit tak, že bychom měli dva trénovací vzorky jedné textury a míra „podobnosti“ by v tomto případě byla součet míry „podobnosti“ s oběma trénovacími vzorky. Experimentálně by se také dala určit jakási „manimální potřebná míra podobnosti“ (pro každý druh porovnání by byla samozřejmě jiná) a pokud by pod ní klesla „míra podobnosti“ s „nejpodobnější“ texturou, program by ji označil jako *nerozpoznanou*. Experimenty byly prováděny s galerií 40 textur, pokud by se ale prováděli s rozsáhlejší galerií, je pravděpodobné, že by se výsledky ještě zpřesnily. Lze však předpokládat obecnější platnost vyvozených závěrů.

Literatura

[1] Mäenpää, T.: *The local binary pattern approach to texture analysis extensions and applications*. Faculty of Technology University of Oulu, 2003, iSBN 951-42-7075-4.

[2] Mäenpää, T.; Ojala, T.; Pietikäinen, M.; aj.: *Robust Texture Classification by Subsets of Local Binary Patterns*. 2002.

[3] WWW stránky: Stránky v angličtině o Bhattacharyově vzdálenosti. `http://en.wikipedia.org/wiki/Bhattacharyya_coefficient`.

[4] WWW stránky: Stránky v angličtině o Mahalanobisově vzdálenosti. `http://en.wikipedia.org/wiki/Mahalanobis_distance`.

[5] WWW stránky: Stránky v češtině o Bhattacharyově vzdálenosti. `http://cs.wikipedia.org/wiki/Euklidovský_prostor`.

[6] Španěl, M.: *Počítačové vidění - Analýza a extrakce příznaků z textur*.

[7] Šára, R.: *Analýza textury*. FEL ČVUT, 1999.