TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových studií

Studijní program: N2612 - Elektrotechnika a informatika
Studijní obor: 1802T007 - Informační technologie

Rozpoznávání vzorů v obraze

Image pattern recognition

Diplomová práce

Autor: **Bc. Karel Paleček**
Vedoucí práce: Ing. Josef Chaloupka, Ph.D
Konzultant: Ing. Zbyněk Koldovský, Ph.D.

V Liberci 29.5.2009

Prohlášení

Byl(a) jsem seznámen(a) s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé diplomové práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé diplomové práce (prodej, zapůjčení apod.).

Jsem si vědom(a) toho, že užít své diplomové práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Diplomovou práci jsem vypracoval(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum: .....

Podpis: .....

Děkuji svému vedoucímu diplomové práce Ing. Josefovi Chaloupkovi, Ph.D za užitečné konzultace, rady a přípomínky k této práci a především za jeho trpělivost.

Abstrakt

Tato práce se zabývá problematikou vyhledávání a rozpoznávání vzorů v obraze. V její první části byla vytvořena rešerže metod, která popisuje současný stav této problematiky. Algoritmy jsou zde rozděleny do tří skupin na metody založené na postupném prohledávání, geometrické metody a metody založené na lokálních deskriptorech. Každá z těchto skupin je poté zkoumána a jsou posuzovány výhody a nevýhody konkrétních metod.

Na základě této rešerže jsou poté vybrány a detailně popsány algoritmy vhodné pro řešení automatického vyhledávání a rozpoznávání vzorů v obraze. Je zde pospána metoda Speeded Up Robust Features (SURF), sloužící k vyhledávání zájmových bodů, jsou posuzovány a popsány metody náhodnostních k-d stromů a hierarchických k-means stromů pro efektivní vyhledávání nejbližšího souseda ve velké databázi, jsou zde vysvětleny pojmy afinní a projektivní transformace a jejich odhad z nalezených dvojic párů pomocí metod Random Sample Consensus (RANSAC) a nejmenšího mediánu čtverců (Least Median of Squares, LMS).

V praktické části jsou implementované algoritmy otestovány na databázi několika desítek objetků a je vyhodnocena celková úspěšnost systému. Dále je demonstrováno využití algoritmů pro autmatické vyhledávání reklamních log. V rámci této práce je také vytvořen počítačový program s grafickým uživatelským rozhraním, který umožňuje efektivní vyhledávání zadaných vzorů na testovacích obrázcích, přičemž je určena pozice, orientace a měřítko hledaného vzoru a zároveň je odhadnuta geometrická transformace aproximující případné otočení vzoru v prostoru a to až do přibližného úhlu 30°.

Klíčová slova: vyhledávání a rozpoznávání vzorů, lokální příznaky, SURF

Abstract

This work deals with the problem of automatic object detection and recognition. It reviews the current state-of-the-art algorithms, which can be divided into three main categories. These categories include appearance based methods, geometrically based methods and local features approach. Each of these categories is then examined in detail.

The appropriate methods for automatic object detection and recognition are chosen based on the evaluation of advantages and disadvantages of the stated methods. The selected algorithms are then explained in detail. These include the Speeded-Up Robust Features interest point detector and descriptor, fast approximative nearest neighbor searching methods such as multiple randomized k-d trees and hierarchical k-means trees and defition and estimation of affine and projective transformations using Random Sample Consensus (RANSAC) or Least Median of Squares (LMS).

The applied part of this work consists of implementing and testing the selected methods. These methods are evaluated on the database of 40 objects. Also the use of the methods in the task of automatic detection of advertisements and commercial logos is demonstrated. An application with a graphical user interface has been created, which allows for a simple parameter setting. This application is capable of efficient recognizing of a set of chosen flat objects based on provided training images. It robustly estimates the position, orientation and scale of the obejct up to roughly 30 degrees of affine rotation.

Key words: object detection and recognition, local features, SURF

Obsah

Seznam obrázků

Úvod

Problém automatického vyhledávání a rozpoznávání vzorů v obraze je jednou z nejstarších a nejčastějších úloh počítačového vidění. Tuto úlohu lze definovat mnoha různými způsoby v závislosti na požadavcích konkrétní aplikace. V zásadě je však vždy úkolem vyhledat zadaný objekt (vzor) reprezentovaný vzorovým obrázkem na jiném zadaném obrázku. Může se tedy jednat např. o automatické vyhledávání a rozpoznávání lidských tváří, optické rozpoznávání znaků (OCR), vyhledávání reklamních log v televizních záběrech či periodikách atp. Metody automatického vyhledávání a rozpoznávání vzorů lze využít také v různých oblastech medicíny nebo průmyslu. Základními požadavky na rozpoznávací systém obvykle bývá obecnost, tj. schopnost systému rozpoznávat libovolný zadaný objekt určitého typu, robustnost, tj. necitlivost systému na různé typy geometrických a jasových transformací, šumu, kompresi obrázků a jiným transformacím, a snadné učení, tj. schopnost systému naučit se libovolný vzor s minimální účastí uživatele.

Problematika automatického vyhledávání a rozpoznávání vzorů a objektů v obraze zahrnuje širokou škálu algoritmů z různých oblastí matematiky a umělé inteligence. Mezi jedny z prvních a nejjednodušších patří metody založené na postupném prohledávání a statistickém rozpoznávání. Tyto metody obvykle pracují na principu posuvného okna, tzn. že v každé lokaci na vstupním obrázku vyhodnotí podobnost vzoru a aktuálního výřezu obrázku určitou ohodnocovací funkcí (např. odchylka čtverců, korelace apod.). Tyto metody byly postupným vývojem optimalizovány, kdy např. oblasti jsou rozpoznávány hierarchicky. To znamená, že nesprávné oblasti jsou efektivně odstraněny již v počátcích prohledávání a více času tak může být věnováno verifikaci. Druhou skupinou metod, která se vyvíjela paralělně, jsou geometricky založené metody. V těchto metodách jsou obvykle zadané vzory rozpoznávány na základě tvaru. Při učení systému je pro každý objekt uživatelem vytvořen určitý model objektu a na testovacím obrázku poté hledány tvarové příznaky a jejich projekce na databázi. Relativně novou skupinou algoritmů, inspirovanou biologickými výzkumy, jsou metody založené na podobnosti lokálních příznaků. Tyto metody extrahují zájmové body či oblasti a jejich deskriptory ze vzoru i testovacího obrázku shodným způsobem. Následně je pak vyhledávána podobnost mezi těmito lokálními obrazovými příznaky a ověřena jejich geometrická konzistence. Uvedené rozdělení není jediné možné, metody by mohly být rozděleny i např. podle typu použitých příznaků na globální a lokální.

Tato práce se věnuje především první a třetí skupině algoritmů. V její první části je vytvořena rešerže současného stavu algoritmů. V druhé části jsou detailně popsány vybrané metody. V třetí části jsou pak tyto metody otestovány na databázi několika desítek vzorů a vyhodnocena dosažená úspěšnost.

Kapitola 1

Přehled problematiky

V následujících sekcích jsou nastíněny principy metod jednotlivých kategorií, zhodnoceny jejich výhody a nevýhody a na základě tohoto zhodnocení vybrány metody, které jsou poté implementovány v softwarové realizaci.

1.1 Metody založené na úplném prohledávání

Tradiční metody vycházejí z myšlenky úplného prohledávání, kdy jsou postupně procházeny všechny lokace vstupního obrazu a současně ohodnoceny určitou funkcí. Ta nejčastěji měří podobnost či vzdálenost natrénovaného vzoru a aktuálního posuvného okna vstupního obrazu, přičemž vzor je obvykle reprezentován příznakovým vektorem, redukujícím původní rozměr dat. Dalším krokem tohoto schématu pak je nalezení lokáních maxim této funkce a v každém takovémto bodě je nakonec třída hledaného vzoru označena určitým klasifikátorem.

1.1.1 Normalizovaná křížová korelace

Mezi nejjednodušší metody patřící do skupiny algoritmů založených na úplném prohledávání patří např. normalizovaná křížová korelace NCC, kde příznakový vektor tvoří ve skutečnosti přímo hodnoty intenzity vzoru. Nedochází tedy k redukci dimenze původních dat. Normalizovaná křížová korelace mezi vzorem t o výšce h a šířce w a obrazem f o výšce H a šířce W je definována vztahem

$$NCC(u,v) = \frac{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\left(f(u+i,v+j)-\overline{f_{u,v}}\right)\left(t(i,j)-\overline{t}\right)}{\sqrt{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\left(f(u+i,v+j)-\overline{f_{u,v}}\right)^2}\sqrt{\sum_{i=0}^{h-1}\sum_{j=0}^{w-1}\left(t(i,j)-\overline{t}\right)^2}} \tag{1.1}$$

kde

$$\overline{f_{u,v}} = \frac{1}{hw}\sum_{i=0}^{h-1}\sum_{j=0}^{w-1} f(u+i,v+j) \tag{1.2}$$

$$\overline{t} = \frac{1}{hw}\sum_{i=0}^{h-1}\sum_{j=0}^{w-1} t(i,j) \tag{1.3}$$

$$0 \le u \le H-h,\ 0 \le v \le W-w$$

Vzor t i posuvné okno $f_{u,v}$ v obraze o stejných rozměrech jsou tedy před násobením vycentrovány tak, aby měly nulový průměr a jejich následný skalární součin je normalizován. Rozsah hodnot NCC leží v intervalu $\langle -1, 1\rangle$ nezávisle na velikosti hledaného vzoru a oblasti, přičemž krajní body intervalu značí největší podobnost vzoru a oblasti a 0 naopak největší odlišnost. Pro lokace na obrázku, kde je rozptyl jasových hodnot nulový, je NCC nedefinovaná, jelikož první člen ve jmenovateli (1.1) je nulový. To odpovídá oblastem, kde se vyskytují větší jednobarevné plochy, např. nevyužitá místa na stránkách časopisů při hledání vzorů v oskenovaných dokumentech.

Po ohodnocení všech bodů v obraze funkcí NCC je dalším krokem nalezení jejích lokálních maxim, obvykle se používá okolí 3×3. Konečným krokem tohoto algoritmu je pak klasifikace nalezené oblasti do jedné ze dvou tříd ANO/NE, jedním z možných způsobů může být porovnání s předem stanovenou prahovou hodnotou T.

Výhodou korelačního koeficientu NCC je jeho jednoduchost a invariance vůči konstantním jasovým změnám (přičtení či vynásobení konstantou), čehož je docíleno odečtením průměrných hodnot $\overline{f_{u,v}}$, resp. $\bar{t}$ a normalizací. Naopak nevýhodou může být např. závislost na zvoleném měřítku a orientaci vzoru. To lze obejít tak, že se celý algoritmus provede opakovaně s tím, že v každé nové iteraci jsou zvoleny jiné měřítko (velikost) a orientace posuvného okna. Při použití tohoto postupu se však nevyhnutelně projeví významný problém, kterým je velká výpočetní náročnost $O\left(n^2\right)$ vztahu (1.1), především pak při použití velkého rozsahu měřítek nebo malých úhlových rozdílů mezi jednotlivými orientacemi. Algoritmus lze však urychlit několika způsoby. Jedním z nich je aplikace korelačního teorému (1.4) na čitatel vztahu (1.1).

$$f \otimes t = \mathcal{F}^{-1}\left\{\mathcal{F}\left\{f\right\} \cdot \mathcal{F}\left\{t\right\}^{\dagger}\right\} \tag{1.4}$$

Symboly $\otimes$, $\mathcal{F}$, $\cdot$ a $\dagger$ v (1.4) označují křížovou korelaci, Fourierovu transformaci, prvkové násobení, resp. komplexní sdružení. K urychlení dochází díky možnosti použití rychlé Fourierovy transformace (FFT), která má složitost pouze $O\left(n \log n\right)$. Další významné urychlení pak spočívá v použití tzv. integrálních obrazů [16, 36].

1.1.2 Integrální obrazy

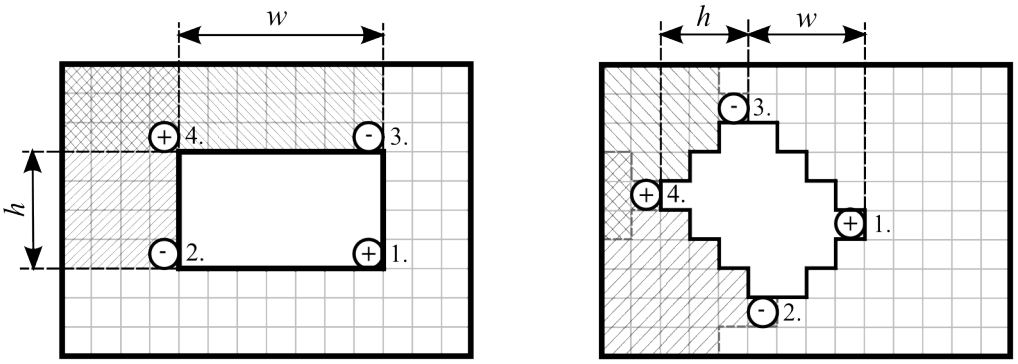
Integrální obrazy (Integral Image, Summed Area Table) jsou velmi často využívanou jednoduchou datovou strukturou, definovanou vztahem

$$I_f\left(u, v\right) = \sum_{i=0}^{u} \sum_{j=0}^{v} f\left(i, j\right) \tag{1.5}$$

Každý prvek $I_f\left(u, v\right)$ tedy obsahuje součet předešlých jasových hodnot obrazu f. Tato vlastnost umožňuje velice efektivně vypočítat součet jasových hodnot libovolně velké obdélníkové oblasti s konstantní složitostí $O(1)$ pomocí součtu čtyř hodnot, viz obr. 1.1a. Konstrukce integrálního obrazu spočívá v jednom průchodu vstupního obrazu zleva doprava a od shora dolů, přičemž jednotlivé hodnoty se získávají z rekurzivního vztahu (1.6).

$$I_f\left(u, v\right) = f\left(u, v\right) - I_f\left(u, v-1\right) - I_f\left(u-1, v\right) + I_f\left(u-1, v-1\right) \tag{1.6}$$

Výpočet součtu jasových hodnot obdélníkové oblasti je pak znázorněn na obr. 1.1. Algoritmus lze poměrně jednoduše rozšířit [17] i o možnost efektivního výpočtu



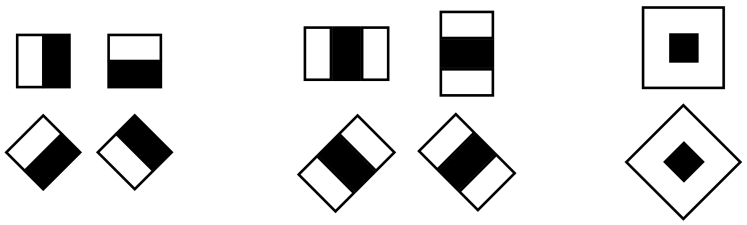
1. $I_f\,(u,v)$
2. $I_f\,(u,v-w)$
3. $I_f\,(u-h,v)$
4. $I_f\,(u-h,v-w)$

(a)

1. $I_f^R\,(u,v)$
2. $I_f^R\,(u+h,v-h)$
3. $I_f^R\,(u-w,v-w)$
4. $I_f^R\,(u-w+h,v-w-h)$

(b)

Obrázek 1.1: Výpočet součtu jasových hodnot obdélníkových oblastí $w \times h$ pomocí integrálního obrazu



(a) hranové (b) čárové (c) uzavřené

Obrázek 1.2: Příklady různých typů binárních příznaků

součtu obdélníkových oblastí, které jsou otočené o 45° (obr. 1.1). Integrální obraz je pak definován vztahem

$$I_f^R\,(u,v) = \sum_{i=\max(0,u-v)}^{\min(H,u+v)} \sum_{j=0}^{v-|u-i|} f\,(i,j) \tag{1.7}$$

Konstrukce je dvou-průchodová, první průchod (1.8) probíhá zleva doprava a shora dolů, druhý průchod (1.9) probíhá zprava doleva a zdola nahoru.

$$I_f^R\,(u,v) = f\,(u,v) + I_f^R\,(u,v-1) + I_f^R\,(u-1,v-1) - I_f^R\,(u-1,v-2) \tag{1.8}$$

$$I_f^R\,(u,v) = I_f^R\,(u,v) + I_f^R\,(u+1,v-1) - I_f^R\,(u,v-2) \tag{1.9}$$

Integrální obrazy lze využít pro urychlení řady algoritmů, např. zmiňované normalizované křížové korelace, kde po úpravě jeden z členů čitatele (1.1) obsahuje součet jasových hodnot posuvného okna a první člen jmenovatele obsahuje vždy součet čtverců jasových hodnot. Dalším častým využitím integrálních obrazů je velmi efektivní výpočet hodnoty konvoluce obrazu s různými typy binárních filtrů, viz obr. 1.2 (bílé plochy se přičítají, černé odečítají).

1.1.3 Reprezentace vzoru pomocí binárních příznaků

Ačkoliv výpočet NCC lze uvedenými způsoby značným způsobem urychlit, může být výsledná časová náročnost např. pro úlohy běžící v reálném čase stále příliš vysoká, především je-li požadavkem detekce vzoru v libovolném měřítku, případně natočení. Aby mohl výpočet probíhat v reálném čase, je nutné algoritmus dále zjednodušit. Jedním z možných způsobů je aproximace vzoru lineární kombinací nějakých bázových funkcí (vztah (1.10)), které umožňují efektivní výpočet konvoluce.

$$t \approx \sum_{k=1}^{K} c_k b_k \tag{1.10}$$

Pro tento účel je výhodné zvolit binární příznaky [33], např. obyčejné obdélníkové funkce nebo různé typy Haarových vlnek, které lze efektivně konvolvovat s obrazem použitím integrálního obrazu.

Optimální rozklad vzoru je však NP-úplný problém, v praxi tudíž neřešitelný. K rozkladu lze tedy využít pouze suboptimální metody, např. algoritmy Matching Pursuit (MP) [39] či Optimized Orthogonal Matching Pursuit (OOMP) [30]. Tyto metody se řadí mezi tzv. hladové algoritmy, tzn. že pracují iterativně, kdy v každém kroku vybírají lokální minimum nějaké ohodnocovací funkce.

Nechť $D = \{b_1, b_{2,} \ldots, b_N\}$ je množina (slovník) obsahující všechny možné binární obdélníkové funkce v obrazu o šířce w a výšce h. Pak platí $N = |D| = w\,(w+1)\,h\,(h+1)$. Dále nechť pro k-tou iteraci je $B_k = \left[b_{I(1)}, b_{I(2)}, \ldots, b_{I(k)}\right]$ do matice uspořádaná množina již vybraných bázových funkcí (převedených na sloupcové vektory), kde $I = \{I\,(1)\,, I\,(2)\,, \ldots, I\,(k)\}$ je množina indexů do D, $C_k = [c_1, c_2, \ldots, c_k]$ je vektor váhových koeficientů c_i a $R_k\,(t) = \sum_{i=1}^{k} c_i b_{I(i)}$ je aproximace vzoru.

Algoritmus MP vybírá v následující $(k+1)$-té iteraci takovou funkci, jejíž projekce na rekonstrukční odchylku má největší energii, tj. podle vztahu

$$b_{k+1} = \underset{b \in D - B_k}{\arg\max} \left|\langle t - R_k\,(t)\,, b\rangle\right| \tag{1.11}$$

kde operátor $\langle \cdot, \cdot \rangle$ značí vnitřní (skalární) součin. Pro koeficienty c_i pak platí

$$c_i = \langle t - R_k\,(t)\,, b_i\rangle \tag{1.12}$$

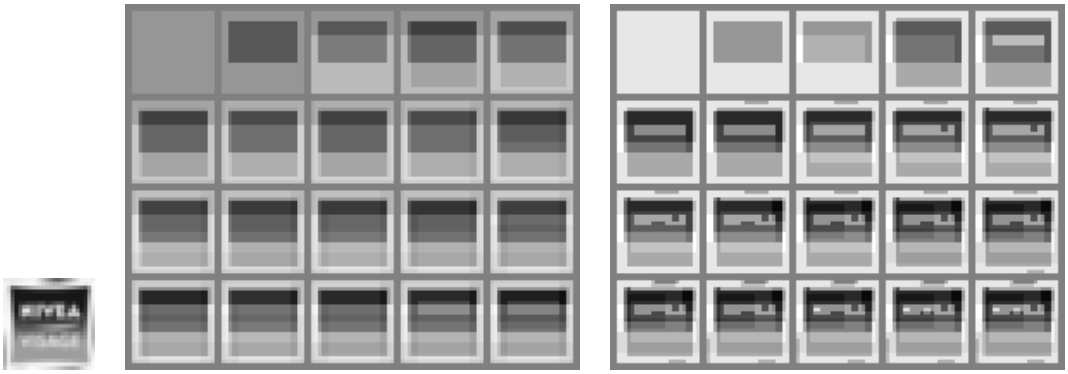
Algoritmus lze ukončit buď dosažením předem daného počtu iterací nebo požadované přesnosti $\epsilon = \|t - R_k\,(t)\|$, kde rekonstrukce $R_k\,(t)$ se vypočítá ze vztahu

$$R_k\,(t) = B_k \left(B_k^T B_k\right)^{-1} B_k^T t \tag{1.13}$$

Algoritmus OOMP pracuje podobně jako MP, pouze s odlišným způsobem výběru bázových funkcí, které jsou vybírány dle vztahu

$$b_{k+1} = \underset{b \in D - B_k}{\arg\max} \frac{\left|\langle b - R_k\,(t)\,, t - R_k\,(t)\rangle\right|}{\|b - R_k\,(b)\|} \tag{1.14}$$

Hlavní výhodou algoritmu OOMP oproti MP je rychlejší konvergence k původním datům, která je daná optimálním výběrem bázových funkcí, zajištěným kritériem jejich vzájemné ortogonality. Nevýhodou je pak podstatně vyšší výpočetní složitost. Příklad postupné aproximace vzoru je znázorněn na obr. 1.3.



(a) vzor (b) aproximace metodou MP (c) aproximace metodou OOMP

Obrázek 1.3: Postupná aproximace vzoru (48×48 pixelů) algoritmy MP a OOMP

Algoritmus NCC lze za použití aproximace (1.10) výrazně urychlit. Jmenovatele (1.1) lze totiž rozložit a zjednodušit na rozdíl $\sum_{i=0}^{h}\sum_{j=0}^{w} f\left(u+i,v+j\right)t\left(i,j\right) - \overline{f_{u,v}}\sum_{i=0}^{h}\sum_{j=0}^{w} t\left(i,j\right)$, kde první člen, značící křížovou korelaci, lze aproximovat použitím (1.15).

$$\sum_{i=0}^{h}\sum_{j=0}^{w} f\left(u+i,v+j\right)t\left(i,j\right) \approx \sum_{k=1}^{K} c_k \left\langle f_{u,v}, b_k \right\rangle \tag{1.15}$$

Díky tomuto zjednodušení lze prvkové násobení mezi vzorem a posuvným oknem obrazu počítat v konstantním čase, což je výhodné především pro prohledávání v různých měřítkách. Aproximace normalizované křížové korelace podle uvedeného způsobu je velice efektivní metoda vyhledávání vzorů, která při dodatečné optimalizaci pyramidovým schématem je schopná běžet téměř v reálném čase. Je však vhodná spíše pro specifické aplikace, kde není požadovaná invariance vůči rovinnému nebo prostorovému otočení. Velkou nevýhodou navíc může být enormní časová náročnost především algoritmu OOMP použitého pro rozklad a trénování příznaků, kdy rozklad vzoru o rozměrech 30×40 pixelů trvá přibližně 30 minut. Algoritmus MP je sice efektivnější, avšak kvalita aproximace není příliš dobrá, viz obr. 1.3.

Jiný způsob rozkladu je použitý v [12]. Vzor je zde rozložen na obdélníková políčka, která se mohou i překrývat. Každé z těchto polí je zpracováno odděleně, čímž se jednak redukuje čas potřebný na rozklad vzoru a zároveň lze tímto způsobem zvýšit robustnost vůči částečnému zakrytí hledaného vzoru. Jednotlivým políčkům je navíc možné přiřadit různé váhy, čímž lze zvýšit i robustnost vůči vnitřní variabilitě vzoru. Vzor je reprezentován množinou K políček $t \approx \{p_1^t, p_2^t, \ldots, p_K^t\}$, kde každé políčko p_i obsahuje několik (zpravidla méně než 30) binárních příznaků různých typů. Ty jsou vybírány heuristicky, v "zajímavých" oblastech vzoru, kterými mohou být např. hrany nebo oblasti s výrazně jiným jasem než okolí. Tyto oblasti jsou identifikovány na základě samotné absolutní hodnoty konvoluce jednotlivých binárních příznaků a vzoru, jelikož odpovídající binární filtry mají nejsilnější odezvu právě v těchto oblastech. Pro každé políčko je pak zvoleno 5% příznaků s nejvyšší absolutní hodnotou konvoluce se vzorem. Při rozpoznávání je hledáno minimum funkce

$$D\left(t,f,m,n\right) = \sum_{k=1}^{K}\sum_{l=1}^{L} w_k \left\| b_l * p_k^t - b_l * p_k^f\left(m,n\right)\right\| \tag{1.16}$$

kde b_l jsou binární příznaky a symbol $*$ značí konvoluci. Při řešení rozpoznávání

částečného zakrytí vzoru v obraze se postupuje tak, že vzor je nejprve rozdělen na čtyři obdélníkové nepřekrývající se oblasti a následně se hodnota funkce $D(t, f, m, n)$ bere v úvahu pouze na oblasti s nejnižší hodnotou této funkce. Není zde ovšem řešen případ, kdy je hledaný vzor zakrytý v prostřední části a tudíž ani jedna ze čtyř oblastí není viditelná úplně.

1.1.4 Metody strojového učení

Uvedené metody rozkladu a vyhledávání vzorů pracují do jisté míry heuristicky. Vzor je rozložen na lineární kombinaci určitých binárních příznaků, u kterých se předpokládá, že se v jednotlivých zobrazeních vzoru budou vždy opakovat, tj. jsou co nejvíce robustní vůči vnitřní variabilitě vzoru. Jedna realizace vzoru ovšem často nenese dostatek informace o této variabilitě a je nutné použít více trénovacích dat. K vhodnému výběru příznaků je pak výhodné použít nějakou z metod strojového učení.

Jednou z nejznámějších a nejrozšířenějších je metoda kaskádní klasifikace [36], založená na výběru binárních příznaků a trénování klasifikátorů pomocí algoritmu AdaBoost [11]. Ten patří do kategorie tzv. boostingových metod, jenž slouží k zvýšení klasifikační přesnosti libovolného algoritmu strojového učení. Trénovacími daty klasifikátoru je sada $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, kde $y_i \in \{0, 1\}$ označuje třídu (negativní/pozitivní) trénovacího obrázku x_i. Algoritmus AdaBoost pracuje iterativně, kdy v každé iteraci je vybrán nejlepší (ve smyslu nejmenší klasifikační odchylky) slabý klasifikátor

$$h(x, f, p, \theta) = \begin{cases} 1 & \text{pokud } pf(x) < p\theta \\ 0 & \text{jinak} \end{cases}$$

kde x je pod-okno vstupního obrázku, f je binární příznak, θ je prahová hodnota a p určuje směr nerovnosti. Pro následující iteraci jsou poté upraveny váhové koeficienty jednotlivých trénovacích obrázků, přičemž vyšší hodnoty jsou přiřazeny nesprávně klasifikovaným datům. Tímto způsobem je docíleno exponenciálního poklesu klasifikační chyby, jelikož v dalších iteracích je pozornost věnována i "obtížnějším" případům. Výběr je zastaven při dosažení uživatelem stanovených parametrů spolehlivosti detekce. Výsledný silný klasifikátor je tvaru

$$C(x) = \begin{cases} 1 & \text{pokud } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{jinak} \end{cases}$$

kde váhové koeficienty α_t závisí na klasifikační odchylce odpovídajícího klasifikátoru h_t. Jednotlivé silné klasifikátory, sestávající z několika slabých, pak tvoří kaskádové zapojení. Každé pod-okno vstupního obrazu je pak postupným procházením jednotlivých vrstev tohoto zapojení klasifikováno do jedné ze dvou tříd 0/1, přičemž při zařazení do třídy 0 se již v další klasifikaci nepokračuje. Tím se velmi rychle odstraní snadno odlišitelné lokace a algoritmus detekce se tak výrazně urychlí.

Algoritmus kaskádní klasifikace byl s úspěchem implementován při vyhledávání lidských tváří, je však možné ho aplikovat na vyhledávání libovolných objektů. Jeho výhodami jsou rychlost a schopnost běžet v reálném čase, detekce v libovolném rozsahu měřítek a robustnost vůči vnitřní variabilitě vzoru. Neřeší však případy, kdy může být vzor v obraze libovolně natočen. Algoritmus navíc není příliš vhodný pro

typické úlohy vyhledávání zadaného konkrétního vzoru, protože k dosažení dobré přesnosti je potřebný velký vzorek trénovacích dat, jenž v praxi nemusí být k dispozici. Díky své robustnosti je tedy vhodný spíše pro vyhledávání objektů určité třídy s velkou vnitřní variabilitou.

Z dalších metod robustního rozpoznávání vzorů založených na strojovém učení lze zmínit např. analýzu hlavních komponent (PCA - Principal Component Analysis, KLT - Karhunen-Loève Transform) [34] a její varianty, analýzu nezávislých komponent (Independent Component Analysis - ICA) [3], skryté Markovovy modely (Hidden Markov Models - HMM), algoritmy podpůrných vektorů (Support Vector Machines - SVM) nebo umělé neuronové sítě. Tyto algoritmy se osvědčily především v úlohách automatického rozpoznávání lidských tváří nebo např. optického rozpoznávání znaků (Optical Character Recognition - OCR). Jejich použití pro vyhledávání požadovaných objektů je ovšem vzhledem k výpočetní náročnosti problematické a obvykle se tedy kombinují s jinými metodami, které nejprve zredukují vstupní obrázek na co nejméně zájmových oblastí.

1.1.5 Metody invariantní vůči otočení

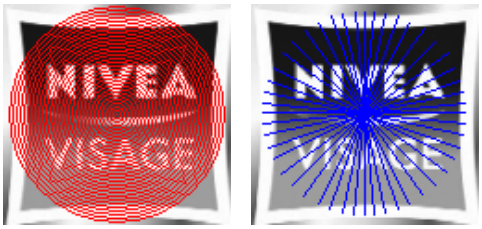
Častým nedostatkem uvedených metod je velká citlivost na otočení hledaného vzoru ve vstupním obrázku. Tento nedostatek lze sice obvykle odstranit opakovaným prohledáváním obrazu pro jednotlivá natočení, případně do určité míry i zvýšením robustnosti aplikací metod strojového učení, avšak tento způsob často příliš zvyšuje výpočetní náročnost celého postupu. Proto je výhodnější použití příznakového popisu, který je invariantní vůči otočení. Nejčastějším způsobem, jak tohoto dosáhnout, je transformace vzoru do polárních souřadnic. Příznakový vektor je pak obvykle vytvářen z jasových hodnot pixelů nacházejících se na soustředných kružnicích se středem uprostřed vzoru.

Jednou z metod řešící na posuvu, měřítku a rotaci invariantní vyhledávání vzoru je algoritmus CiRaTeFi [15]. V principu je tvořen kaskádovým zapojením tří klasifikátorů Cifi (Circular sampling filter), Rafi (Radial sampling filter) a Tefi (Template matching filter).

1. V prvním stupni zapojení jsou ze vzoru i každého pod-okna obrázku vytvořeny příznakové matice, které na n-tém řádku obsahují informaci o průměrných hodnotách intenzity podél l soustředných kružnic s předem zvolenými poloměry pro n-té měřítko. Pro výpočet pozic bodů náležících jednotlivým kružnicím jsou použity efektivní algoritmy počítačové grafiky. Následně je pro každé měřítko vyhodnocena podobnost příznakového vektoru vzoru a obrázku korelací. Posledním krokem tohoto stupně je pak porovnání maximální hodnoty korelace s nějakým prahem a případné označení lokace jako oblast zájmu, přičemž je zároveň odhadnuto pravděpodobné měřítko.

2. V druhém stupni jsou všechny oblasti zájmu zobrazeny na množinu m přímek se společným průsečíkem uprostřed oblasti. Následně je opět vytvořen vektor příznaků sestávající z průměrných hodnot jasu podél těchto přímek, podobnost vyhodnocena korelací a lokace případně označena jako oblast dalšího zájmu s odhadem pravděpodobného otočení.

3. V posledním stupni kaskádového zapojení je pak vypočtena podobnost všech doposud neodstraněných oblastí zájmu normalizovanou korelací s využitím



(a) Cifi (b) Rafi

Obrázek 1.4: Projekce vzoru v algoritmu CiRaTeFi

odhadů měřítka a orientace a výsledná hodnota opět porovnána s nějakým prahem.

Kroky 1 a 2 jsou ilustrovány na obr. 1.4. Autory [15] je uváděna přesnost detekce přes 98%, větší než u algoritmu SIFT [20] na stejných datech, ovšem při přibližně dvojnásobné časové náročnosti. Uvedené výsledky jsou však silně závislé na použitých datech, která nejsou dostupná, navíc algoritmus je limitován vlastnostmi normalizované korelace, kterými jsou poměrně značná citlivost na jiné než eukleidovské transformace (např. afinní nebo projektivní) nebo částečné zakrytí.

Jiné metody invariantní vůči rotaci jsou založeny na podobném principu, kdy složitější výpočty jsou prováděny jen v bodech zájmu. V [38] je pro vyhledávání zájmových bodů použitý stejný způsob, pro verifikaci jsou však zvoleny složitější metody. V [8] je použita rychlá Fourierova transformace pro urychlení vyhledávání zájmových bodů a Zernikeovy momenty jako rotačně invariantní deskriptor.

1.2 Geometricky založené metody

Druhou skupinou algoritmů jsou metody založené na vyhledávání geometrických prvků v obraze. Úkolem pak je rozhodnout, zda nalezená primitiva (např. přímky tvořené hranami), tvoří zobrazení nějakého objektu z databáze. Rozpoznávání pomocí geometrických metod spočívá v několika krocích [29]. Prvním krokem je vytvoření modelů hledaných objektů. Ten může mít podobu např. drátového modelu nebo jinou reprezentaci pomocí jednoduchých geometrických tvarů. Dalším krokem je poté nalezení obrazových příznaků na testovacím obrázku, kterými mohou být např. přímky, kružnice, elipsy apod. Každý příznak je klasifikován do určité skupiny základních tvarů. V obraze jsou poté identifikovány stabilní skupiny příznaků, které tvoří index do tabulky, ve které jednotlivé záznamy obsahují informaci o možné přítomnosti objektů na základě nalezené skupiny příznaků. Dalším krokem je výběr objektu, který nejlépe odpovídá nalezeným příznakům a odhad pozice a orientace, ve kterých se na obraze nachází např. pomocí Houghovy transformace. Posledním krokem je pak verifikace nalezených výsledků, založená např. na podílu nalezených příznaků z celkového počtu příznaků v modelu nebo jejich prostorovému rozložení.

Hlavními nevýhodami geometrických metod je velká závislost na spolehlivé detekci příznaků (přímky, kružnice, ...), která je závislá na světelných podmínkách, častá nejednoznačnost nalezených tvarů (nalezení na rozmezí dvou kategorií nebo neznámá kategorie), omezení na objekty složené z jednoduchých a snadno detekovatelných tvarů a především pak nutnost vytvářet databázové modely manuálně. Tyto

metody jsou vhodné spíše pro specifické aplikace, kde jsou hledány objekty jednoduchých a předem definovaných tvarů a klasifikátory nelze natrénovat jednoduchým a dostatečně obecným způsobem. Z těchto důvodů se jimi tato práce dále nezabývá.

1.3 Metody založené na lokálních deskriptorech

Třetí způsob vyhledávání a rozpoznávání objektů je založen na podobě, vzájemné geometrii a statistice lokálních příznaků. Do určité míry se jedná o kombinaci metod založených na popisu objektu podle celkového vzezření a geometrických metod. Na zadaném vzoru i vstupním obraze jsou vyhledány zájmové body či oblasti a každému takovémuto bodu je obvykle přiřazen určitý deskriptor popisují jejich okolí. Na základě podobnosti či vzdálenosti těchto deskriptorů je pak vytvořena množina dvojic pravděpodobně odpovídajících bodů ze vzoru a vstupního obrázku a mezi těmito skupinami následně hledána nějaká geometrická transformace. Posledním krokem je pak verifikace výsledku nejčastěji velice jednoduchým prahovým klasifikátorem.

1.3.1 Prostor měřítek

V úlohách automatického rozpoznávání vzorů a objektů obvykle není apriori známá jejich velikost. Metody založené na úplném prohledávání tento problém obvykle řeší iterativním procházením a klasifikací jednotlivých pod-oken ve všech měřítkách z určitého zvoleného rozsahu. Zobecněním toho postupu je pak reprezentace obrázku nezávislá na měřítku. Tu lze realizovat různými způsoby, např. pyramidovým schématem, nebo čtyřstromem (quadtree). Spojitou obdobou na měřítku nezávislé reprezentace obrázku s neměnnou vzorkovací frekvencí pak představuje pojem prostoru měřítek (scale-space) [37]. Jedná se o biologickými výzkumy inspirovanou, rozšířenou, kvalitativní reprezentaci obrázku, která obsahuje informace o obrazových strukturách odpovídajících jednotlivým měřítkům. Obrázek f reprezentovaný prostorem měřítek má tvar

$$F(x, y, \sigma) = f(x, y) * g(x, y, \sigma) \tag{1.17}$$

kde $\sigma \in R^+$ je koeficient měřítka a $g(x, y, \sigma)$ je konvoluční jádro. Z obrázku f jsou konvolucí postupně odstraňovány jemné detaily a se zvyšujícím se měřítkem σ získávány informace o nižších frekvencích. Jádro $g(x, y, \sigma)$ je tedy dolní propustí a platí $F(x, y, 0) = f(x, y)$. V principu je možné toto jádro zvolit libovolně, pro zachování požadavků na reprezentaci v prostoru měřítek lze však zvolit pouze Gaussovu funkci.

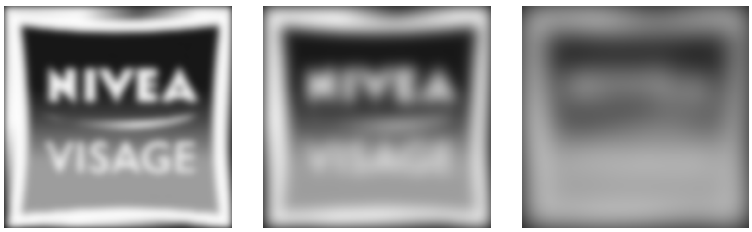
$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \cdot e^{\frac{-\left(x^2+y^2\right)}{2\sigma^2}} \tag{1.18}$$

Formální důkaz jedinečnosti Gaussovy funkce je předložen v [18].

Vztah (1.17) je definován jako spojitá funkce, pro potřebu počítačového zpracování je však nutné prostor měřítek i výpočet konvoluce diskretizovat. Jednotlivé vrstvy prostoru měřítek lze získat buď konvolucí s filtrem o odpovídajícím měřítku nebo efektivněji opakovanou konvolucí s filtrem o konstantních σ a velikosti. Pro výsledné měřítko σ gaussovského filtru získaného konvolucí jiných dvou gaussovských filtrů s měřítky σ_1 a σ_2 platí

$$\sigma = \sqrt{\sigma_1^2 + \sigma_2^2} \tag{1.19}$$

(a) $\sigma = 0$ (b) $\sigma = 1$ (c) $\sigma = 2$



(d) $\sigma = 4$ (e) $\sigma = 8$ (f) $\sigma = 16$

Obrázek 1.5: Reprezentace obrázku v prostoru měřítek

Při konstantním σ je tedy v každém kroku měřítko vynásobeno koeficientem $\sqrt{2}$. Pro další urychlení výpočtu pak je možné využít jeho separability a rozdělení prostoru měřítek na tzv. oktávy. Hranice těchto oktáv odpovídají mocninám dvou koeficientu měřítka σ, tj. s každou oktávou je měřítko zdvojnásobeno. Jelikož minimální poloměr struktur obsažených v obrázku odpovídá měřítku σ, je možné s každou oktávou snížit vzorkovací frekvenci na polovinu bez ztráty informace. Efektivně je tedy obrázek interpolován na poloviční rozměry metodou nejbližšího souseda. Výsledkem tohoto postupu je pole filtrovaných obrázků, kde každý nese informaci o jiných frekvencích, se vzrůstajícím σ od nejvyšších po nejnižší. Příklad takového pole (bez zmenšování obrázku) je ilustrován na obr. 1.5.

1.3.2 Detekce lokálních příznaků

Lokálním příznakem nějakého obrázku může být např. bod, hrana, či oblast, tj. lokace nějakého typického charakteru. Pro úlohu automatického rozpoznávání objektů by měl splňovat několik nejdůležitějších vlastností [35]:

- Opakovatelnost. Na dvou různých obrázcích téhož objektu by mělo být nalezeno co nejvíce shodných příznaků. Měl by tedy být buď invariantní nebo robustní vůči různým typům jasových a geometrických transformací.
- Rozlišitelnost. Oblasti jednotlivých příznaků a jejich deskriptory by měly být dobře rozlišitelné tak, aby bylo možné je snadno rozpoznávat.
- Lokálnost. Oblasti příznaků by měly zachycovat informaci pouze z malého okolí, tzn. aby nebylo zahrnuto i nežádoucí pozadí.
- Kvantita. Lokálních příznaků by se v obrázku mělo nacházet dostatečné množství.

Je zřejmé, že rozlišitelnost a lokálnost jsou protichůdné požadavky na vlastnosti lokálního příznaku, jelikož čím menší je okolní zájmová oblast, tím méně informace

poskytuje a je tedy hůře identifikovatelná. Podobně protichůdné jsou pak i požadavky na rozlišitelnost a invarianci (příp. robustnost). Pro výběr vhodných lokálních příznaků je tak vždy nutné volit kompromis z uvedených požadavků. Kromě jmenovaných vlastností by extrakce lokálních příznaků navíc neměla být z hlediska výpočetní složitosti příliš náročná, což je důležité především pro aplikace pracující v reálném čase.

Problematika stabilní extrakce lokálních obrazových příznaků je velice rozsáhlá a existuje nepřeberné množství algoritmů. V zásadě lze metody hrubě rozdělit podle typu lokálních příznaků, které extrahují. Těmi mohou být např. rohy, hrany, údolí, tzv. bloby, nebo uzavřené oblasti. Toto rozdělení však není úplně přesné, jelikož mnoho metod extrahuje příznaky více typů.

Jedním z prvních detektorů lokálních příznaků vyhledávající zájmové body je např. Harrisův rohový detektor [13]. Tento algoritmus je založený na Moravcově rohovém detektoru [26], kde je definován roh jako taková oblast v obrázku, která při posunu libovolným směrem výrazně změní svoji podobu. V každém bodě se obrázek extrapoluje využitím Taylorova rozvoje prvního řádu a měří se podobnost posunutého a neposunutého pod-okna pomocí součtu nejmenších čtverců váženého Gaussovou funkcí. Výhodami Harrisova detektoru jsou invariance vůči rotaci, dobrá opakovatelnost bodů či robustnost.

Rozsáhlou skupinu algoritmů tvoří metody založené na výpočtu druhých derivací. Jednou z nich je např. operátor LoG (Laplacian of Gaussian), definovaný vztahem

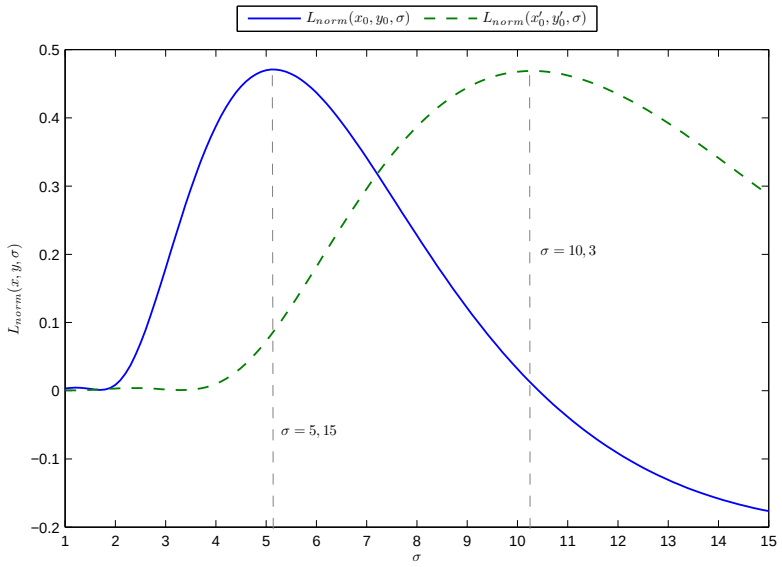
$$L\left(x,y,\sigma\right)=\nabla^2 F\left(x,y,\sigma\right)=F_{xx}+F_{yy} \tag{1.20}$$

kde $F\left(x,y,\sigma\right)$ je dáno vztahem (1.17), $F_{xx}=\frac{\partial^2 F}{\partial x^2}$ a $F_{yy}=\frac{\partial^2 F}{\partial y^2}$ jsou druhé derivace ve směrech x a y. Konvoluce s filtrem $\nabla^2 F\left(x,y,\sigma\right)$ pak má největší absolutní hodnoty v oblastech, které jsou jasově odlišné od svého okolí a mají přibližný poloměr σ. Tyto oblasti se označují jako tzv. bloby a jejich středy (lokální maxima) pak tvoří zájmové body, které lze robustním způsobem detekovat. Nevýhodou uvedeného operátoru, podobně jako Harrisova rohového detektoru, je však závislost na zvoleném měřítku σ. Hodnota derivací obvykle s rostoucím měřítkem klesá vlivem potlačování vysokých frekvencí a není tedy z hodnoty možné určit charakteristické měřítko, kterému nalezený bod nejlépe odpovídá. Tento problém lze vyřešit normalizací, která automatickou detekci měřítka lokálního příznaku umožňuje [19]. Vůči měřítku normalizovaný LoG operátor je definovaný vztahem

$$L_{norm}\left(x,y,\sigma\right)=\sigma^2 L\left(x,y,\sigma\right)=\sigma^2\left(F_{xx}+F_{yy}\right) \tag{1.21}$$

Automatická detekce měřítka v bodě (x_0,y_0) pak spočívá v nalezení lokálního maxima (minima) jednorozměrné funkce $L_{norm}\left(x_0,y_0,\sigma\right)$. Příklad průběhu této funkce je ilustrován na obr. 1.6, kde průběhy funkcí $L\left(x_0,y_0,\sigma\right)$ a $L\left(x_0',y_0',\sigma\right)$ jsou vypočteny z odpovídajících bodů na dvou obrázcích, přičemž průběh $L\left(x_0',y_0',\sigma\right)$ odpovídá obrázku, na kterém se blob nachází ve dvojnásobné velikosti. Tomu pak odpovídá i měřítko nalezeného maxima, které je rovněž dvojnásobné.

Obdobným způsobem pak lze automaticky detekovat měřítko u Harrisova rohového detektoru, výsledný detektor je pak označovaný jako Harris-Laplace [23]. K urychlení algoritmu LoG lze využít linearity konvoluce a není tedy nutné počítat nejprve konvoluci s gaussovským filtrem a poté laplaceovským, ale pouze jednou filtrem složeným. Dalším urychlením výpočtu pak může být aproximace LoG obyčejnými diferencemi. Tento způsob, označovaný jako DoG (Difference of Gaussians),



Obrázek 1.6: Automatická detekce měřítka (σ je měřítko, $L_{norm}\,(x, y, \sigma)$ normalizovaný Laplacián)

je použitý pro detekci zájmových bodů metody SIFT [20], kde je porovnána i kvalita jeho aproximace. Operátor DoG je definován jako

$$D\,(x, y, \sigma) = F\,(x, y, k\sigma) - F\,(x, y, \sigma) \tag{1.22}$$

Výhodou tohoto operátoru je, že není nutná měřítková normalizace. Automatická detekce bodů i měřítka pak probíhá shodným způsobem jako u metody LoG. Dalším možným způsobem detekce blobů je metoda DoH (Determinant of Hessian), která je založena na determinantu Hessovy matice (Hessiánu).

$$H\,(x, y, \sigma) = \det \begin{pmatrix} F_{xx} & F_{xy} \\ F_{yx} & F_{yy} \end{pmatrix} = F_{xx}F_{yy} - F_{xy}^2 \tag{1.23}$$

Tato metoda má mírně lepší vlastnosti [19] při afinních transformacích než LoG (stopa Hessovy matice) a její aproximace pomocí Haarových vlnek a integrálních obrazů je použita v algoritmu SURF [4].

Další metody využívající derivace jsou obvykle založené na výše uvedených operátorech, např. zavádějí invarianci vůči afinním transformacím. Toho je docíleno iterativním přizpůsobováním filtru okolí blobu tak, aby bylo dosaženo rotační symetrie. Druhou skupinou detekce jasově odlišných oblastí pak jsou metody založené na sledování strmosti změny intenzity. Do této kategorie patří např. algoritmus MSER (Maximally Stable Extremal Regions) [21]. Tato metoda je založená na segmentaci rozvodím, kdy jsou nejprve všechny pixely z obrázku seřazeny podle hodnoty jasu algoritmem s lineární výpočetní složitostí. Následně jsou iterativně pro každý práh (0-255) zpětně ukládány do obrázku a sledují se změny ve vznikajících oblastech. Jako maximálně stabilní extremální oblasti jsou pak vybrány takové oblasti, které vykazují minimální relativní změnu počtu pixelů při zvýšení prahu o 1. Metoda má několik výhod, mezi něž patří téměř lineární složitost $O\,(n \log \log n)$ v závislosti na

počtu pixelů, vlivem absence vyhlazování detekuje oblasti s libovolným tvarem a velikostí a je invariantní vůči afinním a monotonním jasovým transformacím. Nevýhodou je pak menší robustnost daná citlivostí na rozmazání obrázku. Detailní zhodnocení a porovnání jednotlivých afinně invariantních detektorů lokálních příznaků je v [22].

1.3.3 Deskriptory pro lokální příznaky

V úloze automatického rozpoznávání objektů se obvykle hledá nějaká geometrická transformace mezi skupinami zájmových bodů nalezených jednou z výše uvedených metod. Aby však byla úloha řešitelná v rozumném čase, je nutné znát mezi těmito skupinami korespondence, tj. dvojice odpovídajících bodů z obou skupin. K identifikaci jednotlivých bodů je tedy potřebný určitý deskriptor, na jehož základě lze usuzovat o jejich podobnosti. Nejjednodušším deskriptorem jsou samotné hodnoty pixelů okolí zájmového bodu a následné rozpoznávání pomocí normalizované křížové korelace (1.1). V praxi je ale tento způsob vlivem velké dimenze dat výpočetně příliš náročný, navíc ani neposkytuje dostatečnou robustnost[1]. Byla proto vyvinuta řada metod, mezi něž patří např. deskriptory zachycující rozložení jasových hodnot či hran, popis ve frekvenční oblasti, algebraické momenty nebo jiné.

Nejpočetnější skupinu tvoří deskriptory založené na histogramech. Zřejmě nejznámějším deskriptorem v této skupině je histogram orientací gradientů [20], použitý ve schématu SIFT (Scale Invariant Feature Transform). V této metodě se čtvercové okolí bodu o straně 16σ (kde σ je detekované měřítko zájmového bodu) nejprve rozdělí na 4×4 oblastí a v každé oblasti vypočte gradientní obraz. Hodnoty orientací gradientů vážené jejich velikostí jsou potom ukládány do histogramu rozděleného na 8 košů a výsledný deskriptor má tedy 128 hodnot. Vylepšenou variantou tohoto algoritmu je pak deskriptor GLOH (Gradient Location and Orientation Histogram) [25], který rozděluje okolí na 17 oblastí v polárních souřadnicích podle úhlu a poloměru a histogramy jsou rozděleny na 16 košů. Výsledný 272prvkový deskriptor je pak redukován na 128 hodnot metodou PCA. Redukce metodou PCA je použitá i v metodě PCA-SIFT [14], kde je zájmová oblast navzorkovaná v 39×39 bodech, vypočteny gradienty v x-ové a y-ové souřadnici a výsledný 3072prvkový vektor zredukován na 36 hodnot. Dalšími metodami založenými na podobném principu jako deskriptor SIFT pak jsou deskriptory Shape Context [6], geometrický histogram [2] nebo SURF (Speeded-Up Robust Features). Detailní srovnání vlastností jednotlivých deskriptorů je k dispozici v [25], kde z provedených testů vyplynulo, že nejlepší vlastnosti dosahují deskriptory této kategorie.

1.3.4 Indexování a vyhledávání podobných deskriptorů

Po natrénování vzorů, tj. vyhledání zájmových bodů a vytvoření jejich deskriptorů, jsou lokální příznaky uloženy do databáze. Rozpoznávání pak probíhá tak, že se nejprve extrahují lokální příznaky shodným způsobem jako v procesu trénování i na vstupním obrázku a mezi vzniklými deskriptory je hledána podobnost. Ta je založena na nějakém kritériu, kterým může být např. korelace (kovariance), nebo Mahalanobisova vzdálenost, nejčastěji se však používá eukleidovská vzdálenost.

[1] Algoritmy pro urychlení korelace uvedené v sekcích 1.1.3 a 1.1.4 není vhodné pro tento účel použít z důvodu dlouhého času potřebného pro natrénování (rozklad) vzoru.

Pro každý zájmový bod nalezený na vstupním obrázku je tedy nalezen jemu nejpodobnější uložený v databázi. Mnoho bodů je však detekováno na pozadí obrázku obsahujícím hledaný vzor, tzn. že takovým bodům žádný z natrénovaných neodpovídá. K identifikaci nesprávně přiřazených dvojic lze využít různé metody. Jednou z nich může být např. porovnání vzdálenosti (podobnosti) nalezené dvojice bodů s určitou prahovou hodnotou. Tento způsob však není příliš vhodný, protože je obtížné globálně nastavit práh na vhodnou hodnotu, jelikož deskriptory jednotlivých bodů nejsou stejně dobře rozlišitelné. Výhodnějším způsobem je porovnání poměru vzdálenosti nejbližšího d_{min1} a druhého nejbližšího souseda d_{min2} vůči nějaké prahové hodnotě [20]. Jako odpovídající jsou tedy identifikovány body, pokud

$$d_{min1} < T d_{min2} \tag{1.24}$$

kde práh může být např. $T = 0,8$.

Problémem uvedeného způsobu vyhledávání nejbližšího souseda je však značná časová náročnost, způsobená velkým počtem lokálních příznaků a jejich dimenzí. V závislosti na použitých lokálních příznacích se počet nalezených bodů pohybuje obvykle v rozmezí 10^1 až 10^3, přičemž dimenze deskriptoru je běžně 64. Vyhledávání v databázi čítající řádově např. 10^4 deskriptorů je pak nejnáročnější částí celého schématu automatického rozpoznávání objektů pomocí lokálních deskriptorů. Z tohoto důvodu je výhodné použít efektivnější algoritmy, než je obyčejné lineární prohledávání databáze. Jednou z možností jsou stromové datové struktury, např. k-d strom. Ten rekurzivně rozděluje k-rozměrný prostor (k je délka deskriptoru) na podprostory, přičemž dělící nadroviny se volí kolmo vůči osám prostoru. Pozice dělících nadrovin v základní verzi k-d stromu odpovídá mediánu dat v ose vykazující největší rozptyl. Při vyhledávání nejbližšího souseda zadaného bodu je pak vytvořený strom rekurzivně procházen od kořenu k listům a po počátečním odhadu zpětně procházeny všechny podprostory (větve grafu), které mají menší vzdálenost od zadaného bodu než aktuální odhad (protínají nadkouli se středem v nalezeném bodě o poloměru aktuální nejmenší vzdálenosti). Takto je zaručeno nalezení nejbližšího souseda v nejhorším případě se složitostí $O\left(kM^{1-1/k}\right)$, kde M je počet uzlů stromu. V algoritmu se však projevuje tzv. prokletí dimenzionality, což znamená, že pro velké dimenze k nepřináší k-d strom žádné urychlení oproti lineárnímu prohledávání. Jelikož neexistuje žádný algoritmus (významně) urychlující vyhledávání nejbližšího souseda pro dimenze větší než přibližně 10, je pro efektivní rozpoznávání nutné využít aproximativní algoritmy, vyhledávající nejpodobnější body pouze s určitou pravděpodobností. Jedním z těchto algoritmů je např. Best Bin First (BBF) [5], který je založen na k-d stromu. Zpětné procházení stromu je zde upraveno tak, že všechny podprostory protínající nadkouli jsou seřazeny od nejbližší po nejvzdálenější a procházeny prioritně. Prohledávání je pak ukončeno po předem stanoveném počtu kroků. Tímto způsobem lze pro deskriptory o délce 128 v závislosti na volbě maximálního počtu kroků dosáhnout přesnosti 95% při přibližně stonásobném zrychlení [20]. Dalšími aproximativními algoritmy pak jsou např. náhodnostní k-d stromy nebo hierarchické k-means stromy, které jsou založeny na podobných principech jako algoritmus BBF, avšak s jiným kritériem výběru dělících nadrovin.

1.3.5 Verifikace

Samotná podobnost jednotlivých dvojic lokálních příznaků však není dostatečně spolehlivým ukazatelem přítomnosti hledaného vzoru a to i přes odstranění falešných korespondencí poměrovým kritériem. Pokud se hledaný vzor nachází v obraze, je velmi pravděpodobné, že dvojic odpovídajících lokálních příznaků bude nalezených více než jedna. Pro robustní automatické rozpoznávání vzorů je tedy nutné ověřit, zda množina nalezených dvojic je konzistentní a odpovídá nějaké globální geometrické transformaci, která popisuje zobrazení mezi vzorem a vstupním obrázkem. Všechny dvojice, které této transformaci neodpovídají, pak jsou odstraněny.

Transformace mezi skupinami bodů má pro automatické vyhledávání plochých objektů obvykle tvar projektivní transformace (homografie, kolinearita). Pro výpočet se zavádějí tzv. homogenní souřadnice, kde každý bod n-rozměrném projektivním prostoru $\mathbb{P}^n$ je dán $(n+1)$-rozměrným vektorem, přičemž poslední souřadnice udává měřítko. Dva body $\mathbf{x} = (x_1, x_2, \ldots, x_{n+1})$ a $\mathbf{y} = (y_1, y_2, \ldots, y_{n+1})$ v $\mathbb{P}^n$ jsou si pak rovny tehdy a jen tehdy, jestliže pro všechna $i = 1, 2, \ldots, n$ platí $x_i/x_{n+1} = y_i/y_{n+1}$. Pokud poslední souřadnice má nulovou hodnotu, pak se bod nachází v nekonečnu. V dvourozměrném projektivním prostoru je tedy každý bod popsán trojrozměrným vektorem a projektivní transformace je definovaná jako

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} \tag{1.25}$$

neboli $\mathbf{x}' = \mathbf{H}\mathbf{x}$. Matice $\mathbf{H}$ se nazývá homogenní a má 8 stupňů volnosti, jelikož všechny členy lze z principu rovnosti bodů v homogenních souřadnicích vydělit prvkem h_{33}. Projektivní zobrazení zachovává kolinearitu a dvojpoměr bodů na přímce a jelikož pro každý bod v obraze vzniknou dvě rovnice (pro x-ovou a y-ovou souřadnici), je možné jej odhadnout z minimálního počtu čtyř párů bodů. Variantou projektivního zobrazení je afinní transformace

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1.26}$$

neboli $\mathbf{x}' = \mathbf{H}_A\mathbf{x}$, která má 6 stupňů volnosti (lze odhadnout ze tří párů bodů) a oproti projektivnímu zobrazení zachovává navíc rovnoběžnost, poměr délek na rovnoběžkách a poměr ploch. Třetí variantou je pak podobnost

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1.27}$$

neboli $\mathbf{x}' = \mathbf{H}_S\mathbf{x}$, která má 4 stupně volnosti (lze odhadnout ze dvou párů bodů) a navíc zachovává úhly a obecný poměr délek.

Jelikož se hledaný vzor může na obrázku nacházet vícekrát, je často nutné nejprve identifikovat shluky dvojic, které přibližně odpovídají jednomu výskytu. Např. v [20] je k tomuto účelu použita upravená Houghova transformace, kde akumulátor je implementovaný pomocí hešové tabulky. Pro každou dvojici je tedy nejprve odhadnuta pozice vzoru např. na bázi podobnosti[2] (1.27) a odhad přičten do akumulátoru. Pro každé pole akumulátoru, které obsahuje tři nebo více záznamů, je pak

[2] Jeden pár tvoří nalezená dvojice a druhý pozice počátků souřadnic.

odhadnuta afinní transformace (1.26) mezi databázovými a nalezenými body. Její výpočet přitom probíhá iterativně, kdy body s projekční odchylkou větší než určitý práh, jsou postupně odstraňovány a výpočet koeficientů transformace upřesňován. Konečné rozhodnutí o přítomnosti hledaného objektu je pak učiněno na základě bayesovské analýzy, kde uvažovanými parametry jsou počet bodů, velikost oblasti a odchylka zobrazení. V [24] probíhá verifikace tak, že po přiřazení nejpodobnějších bodů na základě Mahalanobisovy vzdálenosti se mezi jednotlivými dvojicemi bodů vypočte křížová korelace a porovná s určitým prahem. Dalším krokem je pak nalezení buď projektivní transformace nebo fundamentální matice metodou RANSAC (Random Sample Consensus) [9].

1.4 Shrnutí

V problematice automatického vyhledávání a rozpoznávání vzorů v obraze existují tři základní skupiny metod: metody založené na postupném prohledávání, geometricky založené metody a metody založené na lokálních deskriptorech. Algoritmy by přitom měly splňovat především čtyři základní požadavky, kterými jsou:

1. *Obecnost.* Metoda by měla být dostatečně obecná, neměla by být příliš závislá na konkrétních hledaných vzorech.

2. *Robustnost.* Zvolená metoda by měla být dostatečně robustní, tzn. správně vyhledávat a rozpoznávat požadované vzory i při přítomnosti obrazového šumu, rozmazání nebo různých světelných a geometrických transformací.

3. *Jednoduchost učení.* Zvolená metoda by měla mít možnost natrénovat hledaný vzor z co nejmenšího počtu obrázků. Navíc by tento proces měl být co nejvíce automatický s minimální účastí uživatele.

4. *Efektivita.* Algoritmus by měl být co nejefektivnější z hlediska paměťové a časové náročnosti.

První skupinu algoritmů podle rozdělení uvedeného na začátku kapitoly tvoří metody založené na úplném prohledávání. Tyto metody vyhodnocují podobnost vzoru a testovacích dat v každém bodě vstupního obrázku, nejčastěji na základě určitého příznakového popisu. Výhodou těchto metod je především obecnost a efektivita v případě využití určitých omezení na rozsah měřítek či orientací, ve kterých se hledaný vzor na obrázku může nacházet. Nevýhodou pak je malá robustnost, kterou lze sice odstranit použitím metod strojového učení nebo procházením velkého prostoru všech přípustných transformací vzoru, tím však vzniká požadavek na dostatečný počet trénovacích dat, které nemusí být v praxi k dispozici, resp. přílišná časová náročnost.

Druhou skupinou jsou geometricky založené metody. Tyto algoritmy jsou sice dostatečně robustní, avšak přinášejí mnoho nevýhod, kterými jsou nutnost reprezentovat vzor modelem, který musí být vytvořen manuálně, přílišná závislost algoritmů na konkrétně zvolených datech (extrakce různých geometrických tvarů v závislosti na modelu) a nejsou tedy dostatečně obecné.

Třetí skupinou jsou metody založené na lokálních deskriptorech. Tyto metody kombinují vlastnosti dvou předešlých kategorií, kdy nejprve vyhodnocují podobnost

na základě lokálních příznaků a poté je zkoumána jejich geometrická závislost. Díky tomuto postupu tak splňují všechny čtyři požadavky na obecnost, robustnost, jednoduchost učení a efektivitu. Obecnost je dosažena nezávislostí extrakce deskriptorů na zvolených datech, všechna data jsou tedy reprezentována stejnými datovými strukturami. Robustnost vůči geometrickým a fotometrickým transformacím je dosažena vhodnou volbou detektoru a deskriptoru lokálních příznaků. Díky spolehlivému a robustnímu rozpoznávání příznaků není nutné vytvářet velkou databázi trénovacích dat, která by zachycovala všechny možné fotometrické transformace, ve kterých by se vzor mohl na hledaných obrázcích nacházet. Tyto metody navíc posuzují podobnost jen ve vybraných bodech, čímž je dosaženo i efektivity.

Na základě uvedených vlastností tedy byly do softwarové realizace zvoleny následující metody:

- metoda SURF [4], která slouží k vyhledávání a extrakci deskriptorů zájmových bodů,
- lineární vyhledávání, náhodnostní k-d stromy a hierarchické k-means stromy pro vyhledávání podobných lokálních příznaků [27],
- metody shlukové analýzy, RANSAC [9] a LMS pro odhad geometrické transformace.

Tyto metody jsou detailně popsány a vyhodnoceny v následujících kapitolách a implementovány v softwarové realizaci.

Kapitola 2

Popis použitých algoritmů

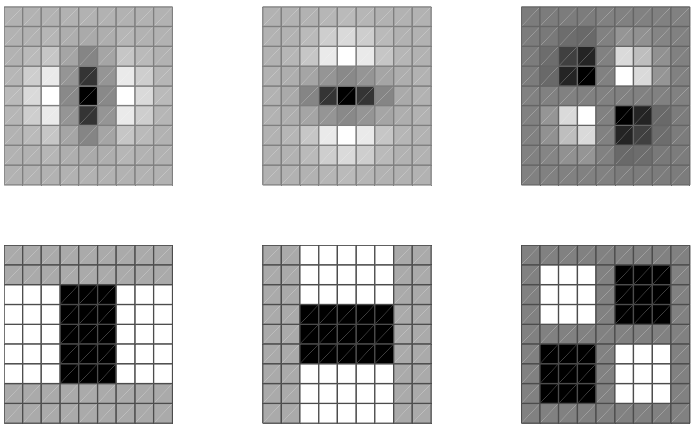
V následujících sekcích jsou detailně popsány metody použité pro řešení automatického vyhledávání a rozpoznávání vzorů v obraze. Systém je řešen obvyklým způsobem, kdy je nejprve hledaný vzor natrénován, příznakové vektory uloženy do databáze a poté vyhledán na vstupním obrázku. Trénování spočívá v nalezení zájmových bodů a jejich deskriptorů na vzorovém obrázku metodou SURF a jejich uložení do některé z datových struktur halda, náhodnostní k-d strom, k-means strom nebo kombinací uvedených. Vyhledávání a rozpoznávání poté probíhá v několika krocích, kdy nejprve jsou na vstupním obrázku vyhledány zájmové body a jejich deskriptory stejným způsobem jako při procesu trénování, pro každý z těchto bodů je nalezen nejbližší soused v natrénované databázi (využitím jedné z uvedených datových struktur) a u vzniklých dvojic odhadnuta geometrická transformace metodou RANSAC nebo nejmenšího mediánu čtverců (Least Median of Squares - LMS). Posledním krokem je pak verifikace nalezených transformací.

2.1 Algoritmus SURF

Algoritmus SURF (Speeded-Up Robust Features) slouží k efektivnímu a robustnímu vyhledávání zájmových bodů. Je založený na aproximaci Hessiánu pomocí Haarových vlnek a integrálního obrazu a deskriptoru typu SIFT. V následujících podsekcích jsou detaily jednotlivých kroků algoritmu.

2.1.1 Aproximace Hessiánu v prostoru měřítek

Prvním krokem algoritmu je vyhledávání zájmových bodů, které je založeno na determinantu Hessovy matice (1.23). Jelikož je obrázek reprezentován digitálně, musí být jednotlivé prvky Hessovy matice diskretizovány, čímž dochází k částečné ztrátě rotační invariance. Dochází tak k tomu, že nejstabilněji jsou detekovány body, jejichž orientace jsou násobkem $\frac{\pi}{2}$. Tato vlastnost je daná čtvercovým tvarem filtru, v praxi se však příliš výrazně neprojevuje. Jednotlivé prvky Hessovy matice, reprezentující druhé derivace obrázku vyhlazeného gaussovským filtrem o rozptylu σ^2, jsou aproximovány haarovskými filtry o odpovídající velikosti. Diskrétní konvoluční jádra a jejich aproximace jsou zobrazeny na obr. 2.1, kde v horní řadě jsou znázorněny diskrétní filtry F_{xx}, F_{yy} a F_{xy}, v dolní pak jejich aproximace D_{xx}, D_{yy} a D_{xy}, u kterých lze konvoluce s obrazem počítat efektivně užitím integrálních obrazů.



Obrázek 2.1: Příklad aproximačních filtrů o velikosti 9×9 ($\sigma = 1, 2$)

Nechť odezvy aproximačních filtrů normalizované vůči velikosti jsou označeny jako $D_{xx} \approx F_{xx}$, $D_{xy} \approx F_{xy}$ a $D_{yy} \approx F_{yy}$, pak aproximace (1.23) má tvar

$$H_a(x, y, \sigma) = \det \begin{bmatrix} D_{xx} & wD_{xy} \\ wD_{xy} & D_{yy} \end{bmatrix} = D_{xx}D_{yy} - w^2 D_{xy}^2 \tag{2.1}$$

kde w je koeficient zachovávající poměr energií filtrů D_{xx} (D_{yy}) a D_{xy}. Tento koeficient lze získat z poměru hodnot Frobeniových norem filtrů D_{xx} a D_{xy}

$$w(\sigma) = \frac{\|F_{xy}(\sigma)\|_F \|D_{xx}(\sigma)\|_F}{\|F_{xx}(\sigma)\|_F \|D_{xy}(\sigma)\|_F} \tag{2.2}$$

kde $F_{xx}(\sigma)$, $F_{xy}(\sigma)$, $D_{xx}(\sigma)$ a $D_{yy}(\sigma)$ jsou konvoluční jádra odpovídajících derivačních a aproximačních filtrů o měřítku σ a operátor $\|\cdot\|_F$ označuje Frobeniovu normu. V praxi tento parametr ale nemá příliš velký význam a jeho hodnota je nastavena nezávisle na měřítku na $w = 0, 9$.

Jedna z výhod tohoto postupu spočívá v tom, že není nutné konstruovat pyramidu převzorkovaných obrázků, konvoluci s filtry tvořené Haarovými vlnkami je totiž možné vypočítat pro libovolnou velikost v konstantním čase. Namísto úpravy velikosti obrázku a opakované konvoluce s filtrem o konstantní velikosti se tedy mění velikost filtru a obrázek zůstává nezměněn. Prostor měřítek je rozdělen na oktávy (viz sekce 1.3.1), přičemž každá oktáva má 4 vrstvy. Každé vrstvě oktávy odpovídá nějaká velikost filtru, přičemž nejmenší možná velikost je 9×9, odpovídající měřítku $\sigma = 1, 2$. S každou další vrstvou oktávy je nejmenší možný přírůstek velikosti strany filtru 6 bodů, jelikož musí být zachovaná lichá délka strany u každého ze tří (resp. čtyř) obdélníků tvořící filtry D_{xx} a D_{yy} (resp. D_{xy}). Pro vyšší oktávy je pak tento přírůstek vždy dvojnásobný oproti předchozí, tj. 12, 24 a 48 bodů pro 2., 3., resp. 4. oktávu. Koeficient měřítka se s rostoucí velikostí filtru zvyšuje lineárně. Velikosti filtrů a jejich odpovídající měřítka pro jednotlivé vrstvy oktáv jsou uvedeny v tabulce 2.1.

Jelikož vzdálenost mezi měřítky je poměrně velká, existuje druhá varianta rozdělení prostoru měřítek, kdy jsou rozměry vstupního obrázku zdvojnásobeny bilineární interpolací. Počáteční délka strany filtru je pak 15 a přírůstky v jednotlivých oktávách dvojnásobné. Tím lze dosáhnout s použitím trikvadratické interpolace většího

Tabulka 2.1: Délka stran filtrů metody SURF (v závorce jsou uvedena odpovídající měřítka)

oktáva/vrstva	1.	2.	3.	4.
I.	9 (1,2)	15 (2,0)	21 (2,8)	27 (3,6)
II.	15 (2,0)	27 (3,6)	39 (5,2)	51 (6,8)
III.	27 (3,6)	51 (6,8)	75 (10,0)	99 (13,2)
IV.	51 (6,8)	99 (13,2)	147 (19,6)	195 (26,0)

rozsahu detekovatelných měřítek, viz následující podsekce. Jako urychlení lze s každou oktávou zdvojnásobit vzorkovací periodu.

2.1.2 Přesná lokalizace zájmových bodů

Druhým krokem algoritmu je nalezení zájmových bodů v prostoru měřítek. Pro detekci bodů jak v prostoru (x, y), tak v měřítku σ je využita hodnota Hessiánu, na rozdíl od metody Harris-Laplace zmiňované v sekci 1.3.2, kde pro detekci měřítka je použitá hodnota LoG. Zájmové body jsou identifikovány jako lokální maxima trojrozměrného prostoru $H_a(x, y, \sigma)$. Každý bod je nejprve porovnán s určitou prahovou hodnotou t, čímž se jednak odstraní nestabilní lokální maxima s malou hodnotou Hessiánu a navíc se tak vlivem menšího počtu bodů urychluje následné zpracování. Pokud je hodnota Hessiánu v bodě vyšší než práh t, je porovnán s 26 sousedními hodnotami z okolí o rozměrech 3×3×3. V této práci k tomuto účelu není použitý algoritmus efektivního potlačování nemaximálních hodnot (Non-Maximal Suppresion - NMS) [28] implementovaný v [4]. V praxi totiž nepřinesl žádné urychlení, jelikož velká většina bodů lokálními maximy není a jsou tedy rychle odstraněny i postupným porovnáváním.

Pozice zájmového bodu v trojrozměrném prostoru $H_a(x, y, \sigma)$ je poté upřesněna kvadratickou interpolací [7]. Okolí bodu $\mathbf{x}_m = (x_m, y_m, \sigma_m)^\top$ je interpolováno Taylorovým rozvojem druhého řádu

$$T(\mathbf{x}) = H_a(\mathbf{x}_m) + \frac{\partial H_a(\mathbf{x}_m)}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^\top \frac{\partial^2 H_a(\mathbf{x}_m)}{\partial \mathbf{x}^2}\mathbf{x} \tag{2.3}$$

kde $\mathbf{x} = (x - x_m, y - y_m, \sigma - \sigma_m)^\top$ jsou souřadnice bodů okolí, přičemž derivace v Taylorově rozvoji jsou aproximovány diferencemi. Maximum funkce $T(\mathbf{x})$ pak leží v bodě, kde její derivace podle $\mathbf{x}$ je nulová, tj.

$$\hat{\mathbf{x}_m} = -\left(\frac{\partial^2 H_a(\mathbf{x}_m)}{\partial \mathbf{x}^2}\right)^{-1} \cdot \frac{\partial H_a(\mathbf{x}_m)}{\partial \mathbf{x}} \tag{2.4}$$

Pokud vyjde interpolované maximum $\hat{\mathbf{x}_m}$ v kterékoliv souřadnici od bodu $\mathbf{x}_m$ dále než 0,5, je interpolace provedena znovu v bodě, kterému je $\hat{\mathbf{x}_m}$ nejblíže. Takto se postupuje tak dlouho, dokud řešení nekonverguje k jednomu bodu, nebo není překročen maximální počet opakování. V této práci jsou povolena max. dvě opakování, jelikož u žádného bodu z testovací databáze nedošlo ke konvergenci po přesažení tohoto počtu. Nejčastěji docházelo k opakovanému výběru bodů, ve kterých interpolace již v předchozích krocích selhala. Interpolace polohy maxima v prostoru H_a v principu není nutná, přináší však několik výhod. První z nich je přesnější odhad měřítka zájmového bodu, který je důležitý především z důvodu velkých rozdílů mezi hranicemi

Obrázek 2.2: Haarovy vlnky použité pro výpočet orientace

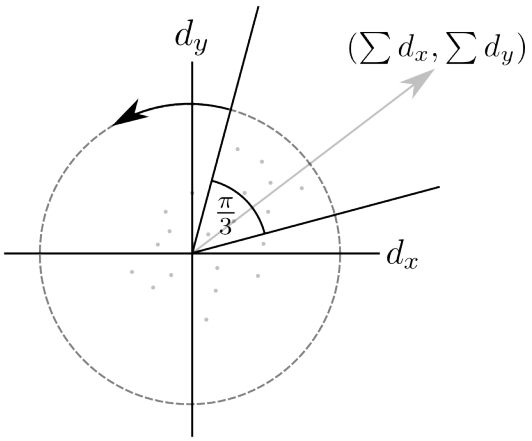
jednotlivých vrstev oktáv. Odhad měřítka je v dalším kroku použitý pro výpočet velikosti okolí bodu, ze kterého se extrahuje příznakový popis (deskriptor), proto interpolace usnadňuje pozdější identifikaci lokálních příznaků. Další výhodou je pak větší rozsah detekovatelných měřítek. Jelikož jsou lokální body vyhledávány jako lokální maxima, nemohou být nalezeny na okraji prostoru H_a, což v praxi znamená, že nejmenší a největší detekovatelné měřítko např. první oktávy odpovídá filtru o velikosti 15×15, tj. $\sigma_{min} = 1,2 \cdot {}^{15}\!/_{9} = 2,0$ a $\sigma_{max} = 1,2 \cdot {}^{21}\!/_{9} = 2,8$. S interpolací Taylorovým rozvojem jsou však při zachování požadavku maximální vzdálenosti 0,5 od původního bodu ve všech souřadnicích minimální a maximální detekovatelná měřítka $\sigma_{min} = 1,2 \cdot {}^{12}\!/_{9} = 1,6$ a $\sigma_{max} = 1,2 \cdot {}^{24}\!/_{9} = 3,2$. Je tedy zachovaný poměr měřítek ${}^{3,2}\!/_{1,6} = 2$ pro oktávu. Poslední výhodou je přesnější odhad souřadnic v obrázku, která zvyšuje stabilitu detekovaných příznaků.

2.1.3 Přiřazení orientace zájmovým bodům

K dosažení rotační invariance při identifikaci lokálních příznaků je možné použít dva způsoby. Prvním z nich může být invariance (příp. robustnost) deskriptoru vůči geometrickým transformacím, druhým pak výpočet deskriptoru relativně vůči nějakému směrovému vektoru udávajícímu orientaci zájmového bodu v obrázku, jež je stabilně a robustně detekována. Algoritmus SURF používá druhý z jmenovaných způsobů.

Při výpočtu orientace se vychází ze součtu odezev filtrů tvořených Haarovými vlnkami v kruhovém okolí o poloměru 6σ zájmového bodu, kde σ je detekované měřítko. Pro každý bod okolí jsou vypočteny dvě hodnoty konvoluce s Haarovými vlnkami aproximujícími první derivace ve směru x a y (viz obr. 2.2, kde vlevo je filtr d_x, vpravo d_y.), přičemž body jsou vzorkované s periodou σ, délka strany filtrů je nastavena na 4σ a odezvy jsou vážené Gaussovou funkcí se středem v bodě zájmu a rozptylem 2σ. Souřadnice bodů lze buď zaokrouhlit (nejbližší soused), nebo např. bilineárně interpolovat. Každá dvojice odezev (d_x, d_y) tvoří bod v dvojrozměrném prostoru, který je následně procházen okénkovací funkcí. Tato funkce vybírá body na základě úhlu, který svírají s horizontální osou x. V každém okénku, postupně se otáčejícím okolo středu s nějakým krokem a úhlovou šířkou $\frac{\pi}{3}$, jsou sečteny odezvy d_x a d_y, čímž vznikne vektor $(\sum d_x, \sum d_y)$. Nejdelší z těchto vektorů pak určuje výslednou orientaci zájmového bodu. V této práci byl jako úhlový rozdíl mezi jednotlivými okénky empiricky zvolen krok $\frac{\pi}{6}$, jenž nabízí optimální poměr mezi stabilitou přiřazování orientace a počtem iterací. Okénkovací funkce je znázorněna na obr. 2.3, kde jsou znázorněny body, které sestávají z dvojic odezev (d_x, d_y), jejichž součet tvoří směrový vektor $(\sum d_x, \sum d_y)$ přiřazený okénku. Nejdelší takový vektor pak představuje orientaci zájmového bodu.

Na rozdíl oproti originálnímu algoritmu [4] je zde navíc bodům přiřazována



Obrázek 2.3: Okénkovací funkce

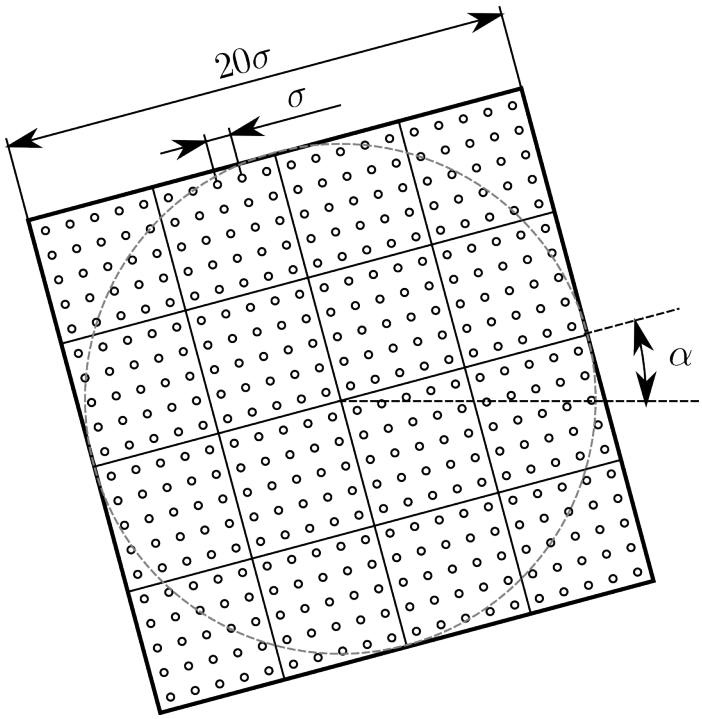
vícenásobná orientace, podobně jako je uvedeno v [20]. Pokud některý z vektorů $(\sum d_x, \sum d_y)$ je delší než 0,8násobek maxima, je vytvořen nový zájmový bod s touto orientací na shodné pozici. Tento postup je použitý především z důvodu zvýšení robustnosti. Mnoho bodů totiž vykazuje dva dominantní směry a vlivem šumu či geometrických a jasových transformací poté na různých obrázcích může mít maximální velikost kterákoliv z těchto dvou orientací. Přiřazením obou se pak zvyšuje šance na správnou identifikaci odpovídajících bodů z různých obrázků.

2.1.4 Výpočet deskriptoru

Posledním krokem algoritmu SURF je výpočet deskriptorů pro každý z nalezených zájmových bodů. Tyto deskriptory robustním způsobem popisují okolí zájmového bodu a na základě jejich podobnosti pak lze vyhledávat podobné body v různých obrázcích, které mají být porovnávány.

Výpočet deskriptoru sestává z několika kroků. Prvním je konstrukce čtvercového okolí bodu o straně 20σ, kde σ je detekované měřítko. Orientace tohoto okolí se volí relativně vůči detekovanému směrovému vektoru zájmového bodu, jak je popsáno v předchozí sekci 2.1.3. Následně je okolí rozděleno na 4×4 čtvercových oblastí o straně 5σ a v každé z těchto oblastí vypočteny součty odezev $\sum d_x$, $\sum d_y$, $\sum |d_x|$ a $\sum |d_y|$. Odezvy v každém z 5×5 bodů vzorkovaných s periodou σ, jsou vypočteny pomocí příznaků uvedených na obr. 2.2 a váženy Gaussovou funkcí se středem v zájmovém bodě a rozptylem $3,3\sigma$. Pro zachování efektivity a možnosti použití integrálního obrazu navíc nejsou filtry orientovány rovnoběžně se stranou čtvercového okolí bodu, nýbrž je jejich hodnota vypočtena na neotočeném obrázku a následně interpolována. Uspořádaná čtveřice $(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$ pak představuje deskriptor každé z 16 oblastí, výsledný deskriptor vytvořený seřazením těchto čtveřic za sebe má tedy 64 prvků. Konstrukce okolí zájmového bodu je znázorněna na obr. 2.4, kde kde σ je detekované měřítko a α orientace zájmového bodu.

V aplikacích, kde není příliš důležitá vysoká efektivita algoritmu je možné použít rozšířenou variantu deskriptoru SURF, která sestává z 128 prvků. Výpočet přitom probíhá pouze s tím rozdílem, že součty $\sum d_x$ a $\sum |d_x|$ jsou počítány odděleně pro $d_y < 0$ a $d_y \geq 0$, analogicky pak jsou odděleny i součty $\sum d_y$ a $\sum |d_y|$ podle znaménka d_x. Tím vznikne dvojnásobný počet prvků výsledného vektoru příznaků, čímž se zvyšuje časová náročnost především měření podobnosti deskriptorů. Naopak



Obrázek 2.4: Okolí zájmového bodu použité pro výpočet deskriptoru

v aplikacích, kde je možné slevit z požadavku na rozlišitelnost deskriptorů je možné využít redukovanou variantu deskriptoru SURF. Tato varianta obsahuje pouze 36 prvků, jejichž výpočet probíhá stejným způsobem jako ve verzi s 64 hodnotami, avšak z okolí o straně pouze 15σ rozděleného na 3×3 oblastí. Hlavní výhodou je pak rychlejší výpočet podobnosti mezi dvěma deskriptory.

Kromě deskriptoru je dalším příznakem rozlišujícím body znaménko LoG, neboli stopy Hessovy matice $D_{xx} + D_{yy}$, které určuje typ blobu, na kterém byl zájmový bod detekován. Tmavým blobům, tj. oblastem s nižším průměrným jasem než okolí, odpovídá kladné znaménko LoG a naopak. Díky tomuto příznaku, poskytujícímu dodatečnou informaci o typu blobu, je pak možné efektivně třídit body a dosáhnout tak větší efektivity při vyhledávání nejbližšího souseda, kdy před samotným porovnáváním deskriptorů je nejprve ověřena shoda znamének LoG. Při rovnoměrném rozložení typů zájmových bodů v obrázcích a lineárním vyhledávání je tedy průměrné urychlení přibližně dvojnásobné.

Rozdílem oproti originálnímu algoritmu SURF je v této práci přiřazení jednotlivým bodům informaci o tom, zda-li příliš velká část čtvercového okolí není mimo hranice obrázku. Především u bodů detekovaných ve vyšších měřítkách může totiž poměrně často dojít k tomu, že velká část okolí vychází mimo meze obrázku (kde se přiřazují nulové hodnoty odezev filtrů), deskriptor tak nemá dostatečně vypovídající hodnotu a při rozpoznávací fázi dochází k chybným přiřazením.

2.2 Vyhledávání nejbližšího souseda

Jedním z kroků schématu automatického vyhledávání a rozpoznávání vzorů v obraze pomocí lokálních deskriptorů je vyhledávání podobných bodů na trénovacích a testovacích obrázcích. Jak bylo popsáno v sekci 1.3.4, může být tento krok především

při velkém počtu zájmových bodů výpočetně velice náročný. V softwarové realizaci proto byly zvoleny různé algoritmy[1] vyhledávání nejbližšího souseda, za účelem otestování a výběru nejvhodnějšího. Všechny způsoby pro měření vzdálenosti mezi dvěma deskriptory užívají Euklidovu vzdálenost. Pro urychlení a redukci počtu porovnávání vyhledávání je navíc využito rozhodování podle znaménka LoG (viz sekce 2.1.4), což znamená, že pro oba typy blobů jsou vždy vytvořeny datové struktury samostatně. Při vyhledávání nejbližšího souseda se pak nejprve zjišťuje znaménko LoG (vypočtené během detekce zájmových bodů) a na jeho základě se pak pokračuje v hledání v odpovídající datové struktuře. Pro každý bod z testovacího obrázku jsou nalezeny dva nejpodobnější body z natrénované databáze, přičemž v dalším zpracování se uvažují pouze body, jejichž vzdálenost k nejbližšímu sousedovi z databáze je menší než 0,8násobek vzdálenosti druhého nejbližšího souseda, jak bylo popsáno v sekci (1.3.4).

2.2.1 Lineární prohledávání

První variantou vyhledávání nejbližšího souseda je sekvenční (lineární) prohledávání. Tento způsob nevyžaduje konstrukci stromových datových struktur, trénovací data (zájmové body a jejich deskriptory) jsou uložena v neuspořádané haldě. Vyhledávání nejbližšího souseda pak spočívá v postupném prohledávání celé haldy a v nalezení minimální vzdálenosti deskriptorů porovnáním se všemi deskriptory nacházejícími se v haldě.

2.2.2 Náhodnostní k-d stromy

Druhou variantou vyhledávání nejbližšího souseda implementovanou v softwarové realizaci jsou náhodnostní k-d stromy. Data jsou zde uložena v jednom nebo více k-d stromů. Konstrukce stromu probíhá podobně jako v případě základní varianty k-d stromu (viz sekce 1.3.4), kdy jsou data rekurzivně rozkládána na podmnožiny. V každé podmnožině bodů se v jednotlivých osách k-rozměrného prostoru (k je v tomto případě délka deskriptoru) vypočte rozptyl dat. Následně se vybere náhodně jedna z D os, které vykazují největší rozptyl a data jsou poté rozdělena na dvě podmnožiny podle hodnoty mediánu ve vybrané ose. V této práci je vybrána jedna z $D = 5$ os s největším rozptylem. Pro obě podmnožiny se rekurzivně zopakuje uvedený postup a algoritmus tímto způsobem pokračuje tak dlouho, dokud každý uzel stromu neobsahuje právě jeden bod.

Vyhledávání v náhodnostních stromech spočívá v postupném procházení stromů od kořene k listům, viz sekce 1.3.4. Při zpětném procházení všech uzlů, které jsou zadanému bodu blíže než aktuální nejbližší soused, se postupuje prioritně, přičemž přednost mají větve danou právě jejich vzdáleností od vstupního bodu. Prohledávání se ukončí v případě, kdy již neexistují žádné další bližší větve nebo bylo navštíveno více než E_{max} uzlů.

[1] Výběr a některé parametry algoritmů vycházejí z implementace knihovny FLANN, která byla použita v softwarové realizaci.

2.2.3 Hierarchické k-means stromy

Třetí variantou vyhledávání nejbližšího souseda jsou hierarchické k-means stromy. Tento způsob je podobný předchozímu, kdy během učící fáze je vytvořen strom. Jeho konstrukce opět spočívá v rekurzivním rozkládání dat na podmnožiny, kdy v každé úrovni stromu jsou data rozdělena na K shluků metodou k-means. Ta pracuje iterativně tak, že nejprve nějakým způsobem (např. náhodně) vybere několik bodů jako středy shluků a poté každý bod ze zadané množiny přiřadí do shluku, jehož střed je nejblíže podle nějaké metriky. V dalším kroku upraví středy těchto shluků jako těžiště bodů v něm obsažených a zopakuje celý postup znovu. Takto se pokračuje buď do dosažení konvergence, tj. body se přiřazují do stejných shluků, nebo do dosažení nějakého počtu iterací. V této práci je pro počáteční výběr možno využít náhodný výběr, Gonzalezův algoritmus nebo odhad podle algoritmu k-means++ [1]. Rozkládání dat na podmnožiny je ukončeno, pokud každá oblast obsahuje maximálně K bodů.

Prohledávání stromu probíhá podobně jako v případě k-d stromů, kdy pro zadaný bod je vždy vybrán shluk s nejbližším středem a po dosažení listu jsou opět prioritně procházeny shluky, jejichž některý bod by mohl být blíže než aktuální nejbližší soused. Prohledávání je pak opět ukončeno po navštívení nejvýše E_{max} uzlů stromu.

2.3 Verifikace

I přes odstranění velkého množství falešných korespondencí pomocí poměrového kritéria (1.24) korespondence jednotlivých samostatných dvojic bodů není dostatečně dobrým ukazatelem přítomnosti hledaného vzoru na testovacím obrázku. Je tedy nutné identifikovat shluky párů, které odpovídají jednomu výskytu vzoru v určité pozici, orientaci a měřítku a u těchto shluků ověřit, zda odpovídají nějaké konzistentní geometrické transformaci.

2.3.1 Shluková analýza

Shluky korespondujících párů bodů jsou nalezeny jednoduchým způsobem. Pro každou dvojici je na bázi podobnosti (1.27) odhadnuta pozice (u_i, v_i), orientace o_i a měřítko s_i vzoru, tedy $p_i = (u_i, v_i, o_i, s_i)$. Jeden pár tvoří pozice počátků $\mathbf{0}$ a $\mathbf{0'}$ na vzoru a testovacím obrázku a druhým párem je pak korespondence nalezená vyhledáváním nejbližšího souseda. Pro každou uspořádanou čtveřici p_i, která tvoří bod ve čtyřrozměrném prostoru, je nalezen shluk, pro jehož střed $m_i = (m_{ui}, m_{vi}, m_{oi}, m_{si})$ platí $|u_i - m_{ui}| \leq s_i a$, $|v_i - m_{vi}| \leq s_i a$, $|o_i - o_{ui}| \leq b$ a $\left|\frac{s_i - s_{ui}}{s_{ui}}\right| \leq c$, kde a je větší z rozměrů vzoru, b je maximální úhlový rozdíl (v této práci 30°) a c maximální rozdíl v měřítku (50%). Pozice nalezeného středu shluku je poté upravena na průměr všech bodů v něm obsažených. Pokud žádný shluk, jehož střed by splňoval uvedené podmínky, není nalezen, je vytvořen nový se středem v bodě p_i. Během přidávání bodů se navíc shluky průběžně třídí podle jejich velikosti algoritmem vkládání (merge sort), na výstupu jsou tedy seřazeny podle počtu bodů od největšího po nejmenší[2]. V dalším kroku je pak celé pole procházeno znovu a všechny shluky, jejichž středy

[2]V praxi tento způsob příliš neovlivňuje rychlost výpočtu, jelikož setříděním shluků podle velikosti se snižuje průměrná doba nalezení shluku, který splňuje uvedené podmínky (za předpokladu existence vzoru na obrázku).

vzájemně splňují uvedené podmínky, jsou sloučeny. Celý algoritmus má v nejhorším případě složitost $O(n^2)$, kde n je počet korespondencí, avšak v praxi se výpočet výrazně nezpomalí ani pro velká n a pohybuje se řádově max. v $\times 10^1$ ms (procesor Athlon XP 2200+ 1,8GHz, 1280MB RAM).

2.3.2 Odhad afinní a projektivní transformace

Po nalezení shluků takových páru bodů, které odpovídají jednomu výskytu vzoru ve stejné pozici, orientaci a měřítku, je dalším krokem odhad geometrické transformace. Ta může mít tvar afinního (1.26) nebo projektivního (1.25) zobrazení.

V případě afinní transformace se parametry odhadují následujícím způsobem. Rovnici (1.26) lze přepsat do nehomogenních souřadnic

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_i & y_i & 1 \end{bmatrix} \begin{bmatrix} \mathbf{a}^{1\top} \\ \mathbf{a}^{2\top} \end{bmatrix} = \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} \tag{2.5}$$

neboli $\mathbf{A}_i\mathbf{h}_A = \mathbf{x}'_i$, kde $\mathbf{a}^j$ označují řádky matice $\mathbf{H}_A$. Každé dvojici korespondujících bodů $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$ ze vzoru a testovacího obrázku pak odpovídá další dvouřádková matice $\mathbf{A}_i$, pro výpočet 6 prvků sloupcového vektoru $\mathbf{h}_A$ jsou tedy potřeba nejméně 3 dvojice bodů. Přidáním všech bodů tak vznikne soustava $\mathbf{A}\mathbf{h}_a = \mathbf{X}$, kde matice $\mathbf{A}$ a vektor $\mathbf{X}$ vzniknou zápisem jednotlivých matic $\mathbf{A}_i$ a vektorů $\mathbf{x}_i$ pod sebe. Při větším počtu než 3 korespondence se soustava řeší metodou nejmenších čtverců pomocí singulárního rozkladu matice (Singular Value Decomposition, SVD) [32].

V případě projektivního zobrazení se postupuje podobným způsobem. Rovnici (1.25) lze z principu ekvivalence bodů v homogenních souřadnicích přepsat do tvaru $\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \mathbf{0}$, kde symbol $\times$ označuje vektorový součin. Platí tedy

$$\begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{bmatrix} \mathbf{h}^{1\top} \\ \mathbf{h}^{2\top} \\ \mathbf{h}^{3\top} \end{bmatrix} = \mathbf{0} \tag{2.6}$$

neboli $\mathbf{W}_i\mathbf{h} = \mathbf{0}$, kde $\mathbf{h}^j$ označuje j-tý řádek matice $\mathbf{H}$. Podobně jako v předchozím případě jsou matice $\mathbf{H}_i$ pro každý bod zapsány pod sebe. Řešení ve tvaru $\mathbf{W}\mathbf{h} = \mathbf{0}$ lze opět řešit singulárním rozkladem matice $\mathbf{W}$, která je vytvořena zapsáním jednotlivých matic $\mathbf{W}_i$ pod sebe a má rozměr $3n \times 9$, kde n je počet dvojic bodů. Pro urychlení výpočtu je však 9prvkový sloupcový vektor $\mathbf{h}$ vypočten jako vlastní vektor matice $\mathbf{W}^\top\mathbf{W}$, který náleží jejímu nejmenšímu vlastnímu číslu [32]. Pro zvýšení přesnosti výpočtu se pak body a jejich měřítka před výpočtem nejprve normalizují, neboli $\widetilde{\mathbf{x}}_i = \mathbf{H}_{pre}\mathbf{x}_i$ a $\widetilde{\mathbf{x}}'_i = \mathbf{H}'_{pre}\mathbf{x}'_i$. Matice $\mathbf{H}_{pre}$ a $\mathbf{H}'_{pre}$ mají tvar

$$\mathbf{H}_{pre} = \begin{bmatrix} d_x^{-1} & 0 & -m_x \\ 0 & d_y^{-1} & -m_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

kde d_x a d_y jsou střední absolutní odchylky a m_x a m_y jsou průměrné hodnoty v souřadnicích x a y. Touto normalizací je dosaženo toho, že body mají střed v $\mathbf{0}$ a jednotkovou střední odchylku. Řeší se tedy soustava $\widetilde{\mathbf{x}}'_i = \widetilde{\mathbf{H}}\widetilde{\mathbf{x}}_i$ a původní matici $\mathbf{H}$ projektivního zobrazení (1.25) lze získat jako $\mathbf{H} = \mathbf{H}_{pre}^{-1\prime}\mathbf{H}\mathbf{H}_{pre}$.

2.3.3 Algoritmy RANSAC a LMS

Uvedené algoritmy však pracují správně pouze za předpokladu, že data neobsahují tzv. odlehlé hodnoty. Těmi mohou být nesprávně přiřazené korespondence bodů a mohou výrazně ovlivnit výsledek celého postupu. Proto je při odhadu nutné tyto odlehlé hodnoty identifikovat a v dalším výpočtu neuvažovat. K tomu lze využít robustní metody regresní analýzy, jakými jsou např. RANSAC (Random Sample Consensus) nebo nejmenší medián čtverců (Least Median of Squares, LMS).

Oba algoritmy pracují velice podobně. V každé iteraci je vybráno náhodně několik bodů (při výpočtu afinní transformace 3, při homografii 4). Pro tyto vzorky je vypočten model jednou z předchozích metod uvedených v sekci 2.3.2.

- V algoritmu LMS je poté zjištěn medián reprojekčních odchylek r_i^2 vypočtených jako Euklidova vzdálenost původních a transformovaných bodů podle odhadnutého modelu.
- V algoritmu RANSAC je zjištěn počet bodů t, které odpovídají vypočtenému modelu, přičemž body jsou takto označeny podle porovnání jejich reprojekční odchylky s nějakým prahem. Pokud je tento počet větší než aktuální maximální hodnota t_{max}, je model přepočítán pro všechny náležící body.

Tímto způsobem algoritmy pokračují do dosažení předem stanoveného počtu iterací a výsledným modelem (tj. maticí geometrické transformace) je pak ten, jehož medián, resp. počet náležících bodů je minimální,resp. maximální. Pokud data obsahují max. podíl ϵ odlehlých hodnot, pak pravděpodobnost P, že při m-násobném výběru p bodů bude alespoň při jednom výběru všech p bodů konzistentních s nějakým modelem je $P = 1 - (1 - (1 - \epsilon)^p)^m$. Počet potřebných opakování výběru pak lze odhadnout jako

$$m = \frac{\log(1 - P)}{\log(1 - (1 - \epsilon)^p)} \tag{2.8}$$

Počet iterací algoritmů RANSAC a LMS lze pak zvolit např. jako $3m$.

2.3.4 Potlačení falešných výskytů

Pro výpočet afinní a projektivní transformace není nutný příliš velký počet korespondencí. Někdy tak může docházet k tomu, že několik špatně přiřazených dvojic bodů vytvoří shluk, který je konzistentní pod nějakou geometrickou transformací a vznikají tak falešné výskyty vzoru. Tyto je nutné nějakým způsobem odstranit, k čemuž je v této práci použito několika velice jednoduchých způsobů.

- U všech výskytů je ověřen poměr délek stran, který se u původního a transformovaného vzoru nesmí lišit o více než 50%. Tento způsob je použitý jen při odhadu afinní transformace.
- Podobně je pak ověřen i jejich úhel, který se rovněž nesmí lišit o více než 50% (45° zkosení). U projektivní transformace se pak tímto způsobem ověřuje úhel u levého horního a pravého spodního vrcholu transformovaného obdélníka, přičemž povolená je 60% odchylka.

- U všech výskytů je vypočtena plocha s a zjištěn počet nalezených bodů n. Následně je výskyt verifikován, pouze pokud platí

$$Sn \geq tsN \tag{2.9}$$

  kde S a N jsou plocha a počet bodů na původním obrázku vzoru a $t \in \langle 0, 1 \rangle$ je nějaká prahová hodnota. Ve všech testech v praktické části bylo zvoleno $t = 0,1$.

Tyto způsoby jsou sice triviální, avšak pro odstranění falešných výskytů vzniklých náhodným přiřazením shluku bodů dostačující.

Kapitola 3

Praktická část

Jednotlivé algoritmy popsané v předchozí kapitole byly otestovány na různých datových sadách za účelem nastavení optimálních parametrů pro automatickou detekci a rozpoznávání vzorů v obraze.

3.1 Detekce bodů

Jednou ze stěžejních částí celého schématu automatického vyhledávání a rozpoznávání pomocí lokálních příznaků je detekce zájmových bodů. Nejdůležitější vlastností detektoru je přitom opakovatelnost bodů (viz sekce 1.3.2), tj. jaké relativní množství bodů je stabilně nalezeno na dvou obrázcích lišících se nějakou geometrickou a jasovou transformací. V této práci je použita shodná definice opakovatelnosti jako v [31]. Nechť $\{\widetilde{x_r}\} = \{x_r | H_{ri} x_r \in I_i\}$ a $\{\widetilde{x_i}\} = \left\{x_i | H_{ri}^{-1} x_i \in I_r\right\}$ jsou množiny bodů (nacházející se na společné části) z referenčního I_r, resp. i-tého obrázku I_i. Matice H_{ri} označuje projektivní transformaci z obrázku I_r na obrázek I_i. Pak opakovatelnost bodů z obrázku I_r na obrázku I_i je definovaná jako

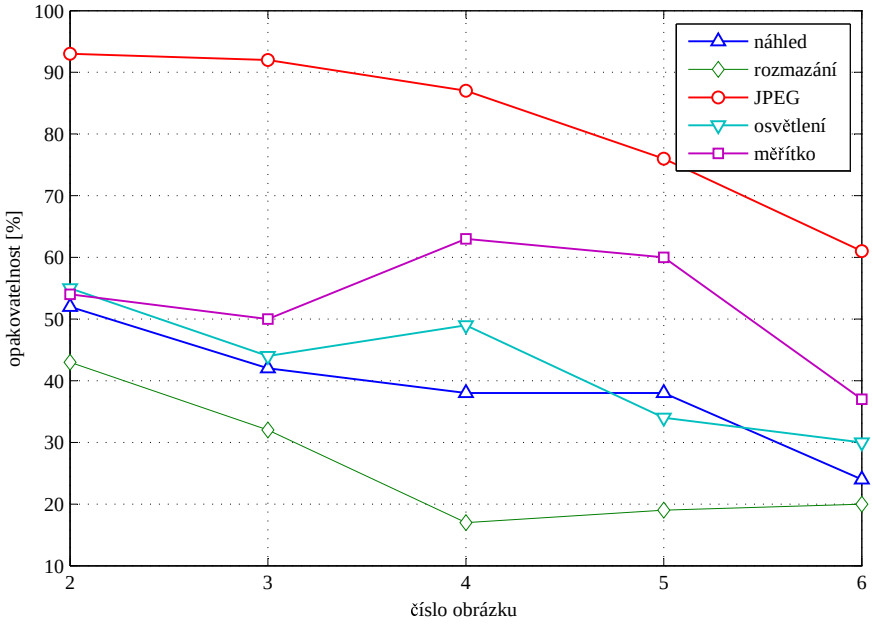
$$r_i(\epsilon) = \frac{|R_i(\epsilon)|}{\min\left(|\{\widetilde{x_r}\}|, |\{\widetilde{x_i}\}|\right)} \tag{3.1}$$

kde $R_i(\epsilon) = \{(\widetilde{x_r}, \widetilde{x_i}) \, | dist\left(H_{ri}\widetilde{x_r}, \widetilde{x_i}\right) < \epsilon\}$. Bod $\widetilde{x_r}$ je tedy považován za nalezený na obrázku I_i, pokud existuje nějaký bod $\widetilde{x_i}$, jehož vzdálenost od transformovaného bodu $H_{ri}\widetilde{x_r}$ je menší něž nějaký práh.

Algoritmus vyhledávání zájmových bodů, v [4] označovaný jako Fast Hessian (FH), byl otestován na datové sadě několika obrázků[1] různých druhů scén, mezi kterými existuje známá projektivní transformace. Testována byla opakovatelnost bodů při změně pohledu na scénu, zvětšení či zmenšení, rotaci, osvětlení, rozmazání a JPEG kompresi.

Výsledky jsou zobrazeny na obr. 3.1, kde jsou uvažovány body s hodnotou Hessiánu větší než 250. Z grafu je patrné, že s rostoucím úhlem pohledu na plochý objekt poměrně významně klesá opakovatelnost bodů, což má za následek nedostatečný počet bodů při rozpoznávání vzorů, které se v obraze vyskytují prostorově natočené. Při rozmazání je naopak bodů nalezeno dostatek, avšak jejich lokalizace i přes interpolaci (2.3) není příliš přesná, což se projevuje v tabulce jako pokles opakovatelnosti. Algoritmus FH je poměrně robustní vůči změnám měřítka, avšak přibližně

[1] http://www.robots.ox.ac.uk/~vgg/research/affine/



Obrázek 3.1: Opakovatelnost bodů

od dvojnásobného zmenšení dochází k výraznějšímu poklesu opakovatelnosti bodů. Algoritmus FH se jevil poměrně citlivý na na změny osvětlení, naopak ani velká komprese typu JPEG nezapříčinila významnější úbytek detekovaných bodů.

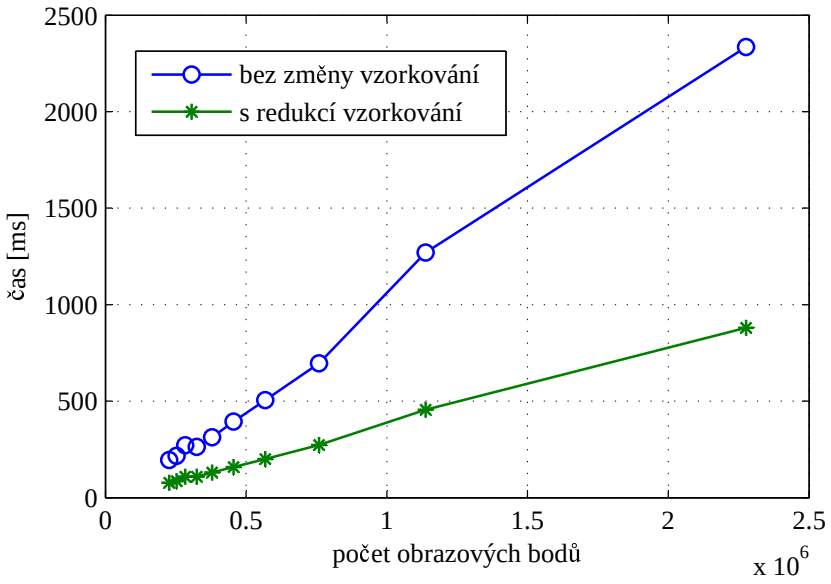
Detekce bodů je pro malé až středně velké obrázky (přibližně do 2 Mpx) časově nejnáročnější operací celého schématu detekce a rozpoznávání vzorů. Pro velké obrázky (za předpokladu velkého počtu detekovaných bodů) má větší náročnost pouze vyhledávání podobných deskriptorů, avšak to lze řešit efektivními aproximativními metodami (viz sekce 2.2), čímž náročnost významně klesá. Časová náročnost detekce bodů v závislosti na velikosti obrázků je znázorněna na obr. 3.2. Na grafu je patrná lineární závislost a vliv zdvojnásobování vzorkovacího kroku pro vyšší oktávy (viz sekce 2.1.1), kdy je algoritmus přibližně 2,5násobně rychlejší.

3.2 Deskriptory

Robustnost deskriptorů vůči geometrickým transformacím, zvětšení či zmenšení, rotaci, osvětlení, rozmazání a JPEG kompresi byla otestována na stejném datovém vzorku jako algoritmus FH. Pro každou dvojici obrázků byly nejprve vytvořeny množiny bodů $\{\widetilde{x_r}\} = \{x_r | H_{ri} x_r \in I_i\}$ a $\{\widetilde{x_i}\} = \left\{x_i | H_{ri}^{-1} x_i \in I_r\right\}$ stejným způsobem jako v předešlém případě, přičemž všem bodům byly přiřazeny deskriptory. Pro každý bod $x_r \in \{\widetilde{x_r}\}$ pak byl lineárním prohledáváním nalezen nejbližší soused na základě podobnosti deskriptorů a tyto páry tvořily množinu D_i. Úspěšnost pak byla vypočtena ze vztahu

$$s_i(\epsilon) = \frac{|R_i(\epsilon) \cap D_i|}{|R_i(\epsilon)|} \tag{3.2}$$

Výsledky jsou znázorněny na obrázku 3.3. Z grafů je patrné, že u všech tří typů

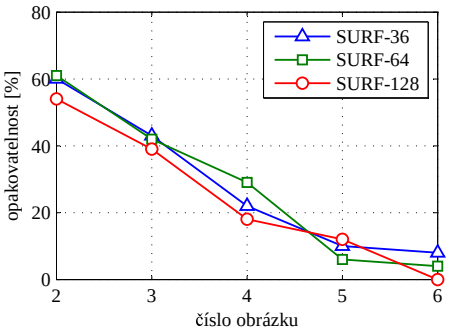


Obrázek 3.2: Časová náročnost detekce zájmových bodů

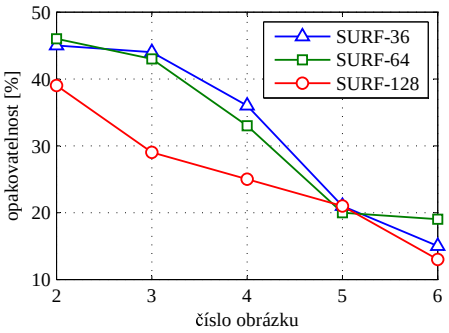
deskriptorů rozdělených podle jejich délky, byly dosaženy podobné výsledky. S rostoucí dimenzí sice roste rozlišitelnost deskriptorů, avšak se zároveň snižuje robustnost, což má za následek nepřesné rozpoznávání deskriptorů. Nejstabilnějších výsledků bylo dosaženo u dimenze 64 (SURF-64). Největším problémem pro správné rozpoznávání deskriptorů je změna pozorovacího úhlu. Tato vlastnost, společně s nízkou opakovatelností pro vyšší pozorovací úhly, pak způsobuje prudký pokles úspěšnosti rozpoznávacího skóre v závislosti na pozorovacím úhlu. Deskriptory jsou poměrně robustní vůči částečnému rozmazání obrazu. Při vyšších faktorech vyhlazení však dochází k přílišné ztrátě informací v obraze, tudíž i zde algoritmus nutně vykazoval nižší úspěšnost rozpoznávání. Naopak komprese JPEG či změna osvětlení nezpůsobily významné snížení úspěšnosti, deskriptory jsou vůči těmto typům transformací robustní. Jedním z problémů je také změna měřítka, kde podobně jako v případě algoritmu FH dochází při přibližně dvojnásobném zmenšení původního obrázku k výraznému poklesu úspěšnosti rozpoznávání. Tato vlastnost však podobně jako v případě vyhlazování souvisí se ztrátou obrazové informace.

3.3 Vyhledávání nejbližšího souseda

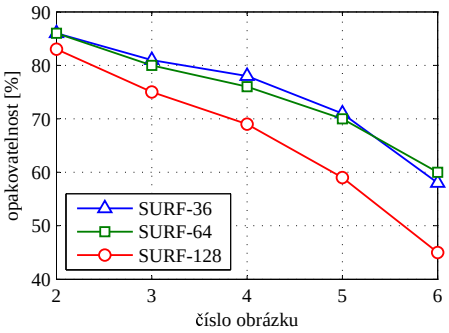
Další časově velice náročnou částí systému je vyhledávání a přiřazování podobných deskriptorů. V systému je možno k tomuto účelu využít tři různé algoritmy, viz sekce 2.2. Tyto algoritmy byly otestovány následujícím způsobem. Byly vybrány první dva obrázky ze sekvence graffiti v použité datové sadě. Na prvním obrázku bylo nalezeno $n_1 = 7217$ bodů bez prahování hodnoty Hessiánu. Deskriptory těchto bodů tvořily natrénovanou databázi. Na druhém obrázku bylo nalezeno $n_2 = 1964$ bodů prahováním Hessiánu nad hodnotu 200. Deskriptory těchto bodů tvořily testovací sadu. Následně pro každý bod z testovací sady byl nalezen nejbližší soused v natrénované databázi lineárním prohledáváním a takto vzniklé páry tvořily množinu dvojic X



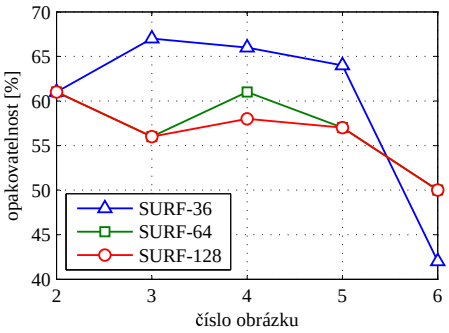
(a) náhled



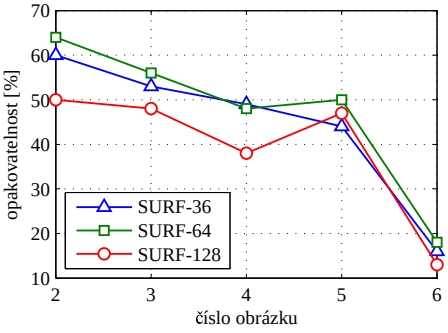
(b) rozmazání



(c) JPEG



(d) osvětlení



(e) změna měřítka a rotace

Obrázek 3.3: Úspěšnost rozpoznávání deskriptorů

Tabulka 3.1: Vyhledávání nejbližšího souseda - k-d strom, $d = 64$

n_t	E_{max}	t_b [ms]	t_s[ms]	s [%]	k
1	32	71	102,9	72,7	31,6
1	48	72,6	137,2	79,6	23,7
1	64	70,8	169,2	84,8	19,2
1	80	70,6	202,9	87,1	16,0
1	96	83,5	233,8	89,6	13,9
2	32	129,5	111,7	77,7	29,1
2	48	126,5	151,6	84,0	21,4
2	64	130,3	190,7	88,3	17,0
2	80	127,1	225,0	89,9	14,4
2	96	127,2	267,0	92,8	12,2
3	32	192,4	117,3	79,8	27,7
3	48	186,0	159,6	85,9	20,4
3	64	185,8	203,3	90,5	16,0
3	80	216,8	245,2	92,5	13,2
3	96	186,7	290,7	94,1	11,2
4	32	245,7	127,0	82,0	25,6
4	48	245,9	171,0	87,1	19,0
4	64	250,5	227,4	90,1	14,3
4	80	250,9	261,7	92,9	12,4
4	96	246,9	338,0	94,6	9,6

skutečných nejbližších sousedů. Pro každý bod z testovací databáze pak byla vytvořena množina nejbližších sousedů Y jednou z metod náhodnotstních k-d stromů a hierarchických k-means stromů. Úspěšnost algoritmů pak byla vyhodnocena podle vztahu

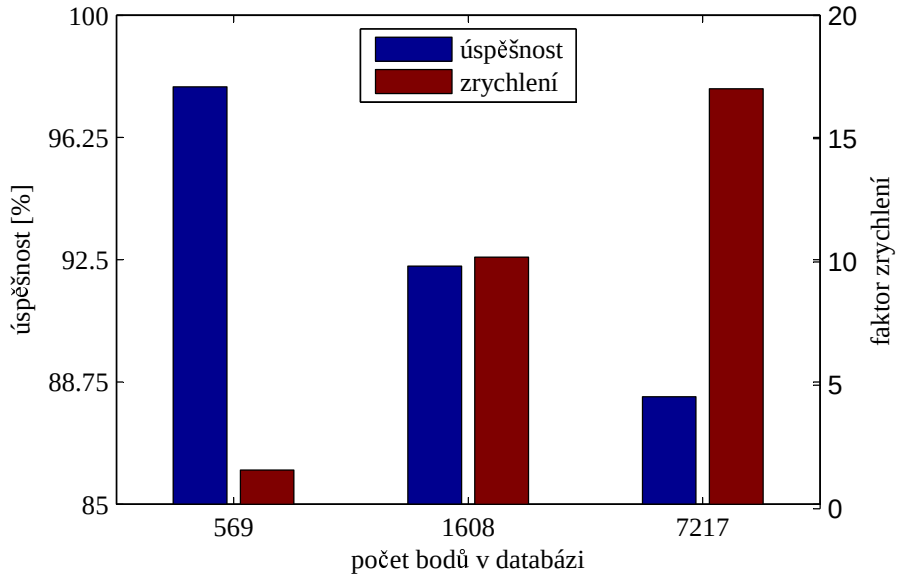
$$s = \frac{|X \cap Y|}{n_1} \tag{3.3}$$

Výsledky testování pro k-d strom a dimenzi $d = 64$ deskriptorů jsou uvedeny v tabulce 3.1, kde n_t je počet k-d stromů, E_{max} je max. počet navštívených uzlů (viz sekce 2.2.2), t_b je čas potřebný k vytvoření stromu, t_s čas potřebný k nalezení všech 1964 korespondencí a k faktor zrychlení oproti lineárnímu vyhledávání, které v tomto testu trvalo 3248 ms. Parametrem s největším dopadem na celkovou úspěšnost a rychlost vyhledávání je max. počet navštívitelných uzlů E_{max}. Pro takto velké databáze bodů zlepšuje vlastnosti větší počet k-d stromů než pouze jeden, avšak poměr úspěšnosti a potřebného času je pro jediný i mnohonásobné stromy přibližně stejný. Při použití např. 4 k-d stromů lze tedy vyhledávání nejbližšího souseda zrychlit při 90% úspěšnosti přibližně 14×.

Výsledky testování pro hierarchický k-means strom jsou uvedeny v tabulce 3.2, kde K je počet shluků algoritmu k-means, viz sekce 2.2.3. Ostatní parametry jsou shodné jako v případě k-d stromu. Pro hierarchický k-means strom jsou nejdůležitější parametry K a E_{max}. Počet dělení uzlů v jednotlivých vrstvách k-means stromu má velký vliv na časovou náročnost především při vytváření stromu. Největší úspěšnosti bylo dosaženo pro hodnoty $K \in \{8, 16, 32\}$. Poměr dosažené úspěšnosti a časové náročnosti vyhledávání pak byl přibližně stejný jako v případě k-d stromů, tj. při

Tabulka 3.2: Vyhledávání nejbližšího souseda - k-means strom, $d = 64$

K	E_{max}	t_b [ms]	t_s [ms]	s [%]	k
8	32	374,4	165,9	86,6	19,6
8	48	366,1	223,7	90,3	14,5
8	64	385,0	312,5	94,0	10,4
8	80	408,8	390,6	96,8	8,3
8	96	402,8	453,3	96,4	7,2
16	32	674,9	169,6	86,0	19,2
16	48	489,8	242,4	91,3	13,4
16	64	467,7	280,7	94,0	11,6
16	80	467,2	348,4	96,0	9,3
16	96	487,9	403,9	96,6	8,0
32	32	685,7	158,8	84,2	20,5
32	48	702,4	220,6	91,0	14,7
32	64	683,6	252,1	93,7	12,9
32	80	706,2	279,9	93,8	11,6
32	96	692,8	337,5	97,0	9,6
64	32	1062,6	184,2	72,7	17,6
64	48	1094,4	187,6	76,4	17,3
64	64	1065,7	236,2	85,7	13,8
64	80	1082,6	267,9	87,0	12,1
64	96	1063,7	289,4	87,7	11,2
128	32	1758,5	156,9	73,2	20,7
128	48	1794,7	177,6	77,9	18,3
128	64	1753,3	201,2	81,7	16,1
128	80	1719,5	221,0	85,7	14,7
128	96	1717,4	246,1	88,6	13,2



Obrázek 3.4: Vlastnosti efektivního vyhledávání nejbližšího souseda

přibližně 90% úspěšnosti dosaženo 14násobné urychlení.

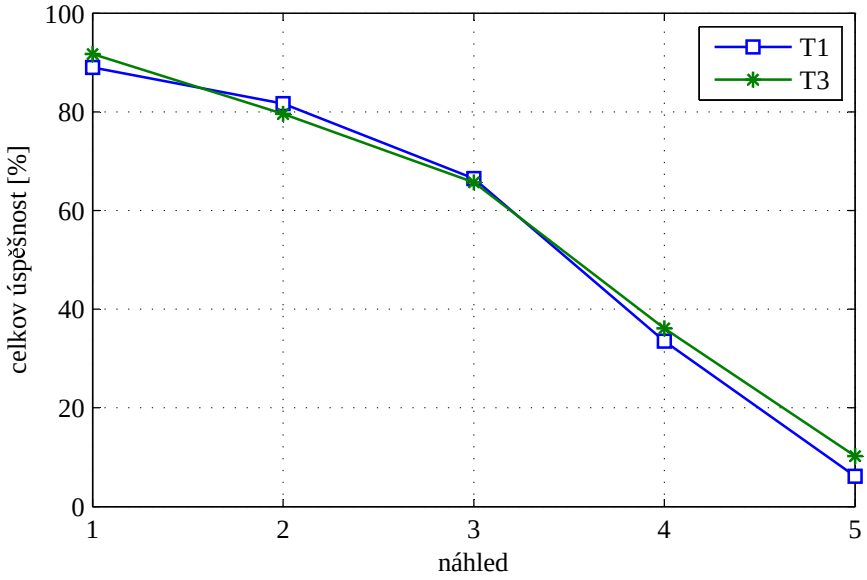
Pro doplnění jsou na obrázku 3.4 znázorněny úspěšnost a faktor zrychlení v závislosti na velikosti trénovací databáze při dimenzi deskriptoru $d = 64$ a použití 2 k-d stromů. Pro ostatní dimenze (36 a 128) byly výsledky velice podobné. Z grafu je patrné, že pro malé databáze nepřináší algoritmy aproximativního vyhledávání přílišné urychlení a lepší volbou tak je lineární prohledávání, které zaručuje vždy nalezení skutečného nejbližšího souseda. Pro středně velké a velké databáze deskriptorů je však urychlení značné a to i při zachování dostatečné přesnosti.

3.4 Úspěšnost rozpoznávání

Celý algoritmus automatického vyhledávání a rozpoznávání vzorů byl otestován na volně dostupné databázi SOIL[2]. Tato databáze obsahuje celkem 48 objektů nasnímaných z různých úhlů pohledu a pod různým osvětlením a byla původně vytvořena pro účely testování algoritmů založených na rozpoznávání pomocí barevných histogramů. Objekty jsou převážně ploché, lze tedy rozpoznávat pomocí výpočtu projektivních či afinních transformací. Každý objekt obsahuje jeden trénovací a 20 testovacích obrázků, přičemž na těchto je nasnímán s postupně rostoucím úhlem pohledu, od -90° do +90°. V databázi je navíc sada obrázků, na kterých se nachází více vzorů současně, přičemž u některých dochází k částečnému překrytí nebo rovinnému natočení. Tato sada obrázků je nasnímána shodných způsobem jako v předchozím případě, tedy s postupně se měnícím úhlem pohledu. Všechny obrázky mají rozlišení 720×576 pixelů (norma PAL).

Trénování databáze probíhalo následujícím způsobem. Bylo vybráno celkem 40 trénovacích objektů, přičemž z každého obrázku byly extrahovány zájmové body a

[2]http://www.ee.surrey.ac.uk/CVSSP/demos/colour/soil47/



Obrázek 3.5: Celková dosažená úspěšnost

vypočteny jejich deskriptory. Ve výsledné databázi byl poté navíc každý bod označen indexem označujícím pořadové číslo vzoru, na kterém byl detekován. Vyhledávání vzorů pak probíhalo použitím metod popsaných v předchozích sekcích, avšak s několika rozdíly. U shlukové analýzy bylo použito indexu zájmových bodů k identifikaci shluků, tzn. že shlukovány byly pouze body se shodným indexem. Tímto způsobem bylo dosaženo výrazného snížení celkového času (přibližně desetinásobného) potřebného k rozpoznávání většího množství vzorů (celkem 40), oproti opakovanému vyhledávání všech vzorů jednotlivě. Výpočetní složitost v závislosti na počtu vzorů v databázi pak odpovídala výpočetní složitosti náhodnostních k-d stromů (viz sekce 2.2.2).

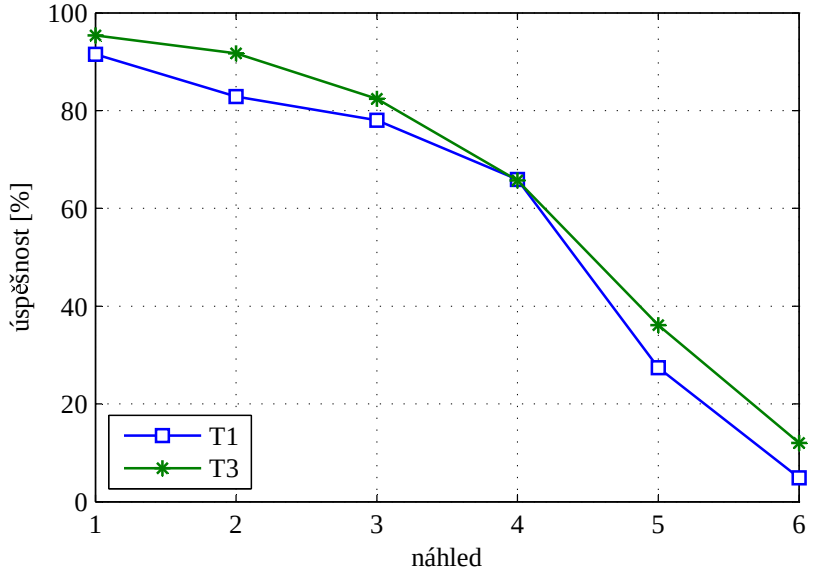
Výsledky celkové úspěšnosti jsou znázorněny na obrázku 3.5. Algoritmus byl testován několika sadách obrázků, kde byl postupně zvyšován úhel pohledu. Graf hodnot T1 odpovídá rozpoznávacímu skóre zjištěnému na obrázcích, kde se vyskytuje vždy právě jeden objekt. Graf T3 pak odpovídá rozpoznávacímu skóre zjištěnému na obrázcích, kde se vyskytují dva až tři objekty, které se na některých obrázcích navíc vzájemně překrývají. Náhled č. 1 odpovídá přibližně čelnímu pohledu, náhled č. 5 pak přibližně 30° pozorovacímu úhlu. Z dosažených výsledků je patrné, že největší problém při rozpoznávání způsobuje změna úhlu pohledu na scénu. Tato vlastnost vychází z opakovatelnosti bodů a robustnosti deskriptorů, které pro rostoucí pozorovací úhly klesají. Pro přibližně čelní pohled bylo dosaženo rozpoznávacího skóre 89% pro samostatné objekty a 91,7% pro obrázky s více objekty. Obecně větší skóre pro druhý případ lze připsat menšímu rozsahu této datové sady a navíc tyto obrázky obsahují trojrozměrné objekty, které poměrně výrazným způsobem komplikují rozpoznávání[3], v menší míře než v případě prvním.

Na Testovacích obrázcích se nacházejí rovněž objekty nepatřící do natrénované

[3]Trojrozměrné objekty je možné rozpoznávat pouze s nižší úspěšností, vlivem odhadu afinní nebo projektivní transformace, jelikož tyto transformace se uplatňují pouze u plochých objektů.

Tabulka 3.3: Tabulka hodnot TPR, FPR, TNR a FNR (vše v [%])

T1				T3			
TPR	FPR	TNR	FNR	TPR	FPR	TNR	FNR
89,0	0,0	100,0	11,0	93,5	0,0	100,0	6,5
81,7	5,0	95,0	18,3	81,5	0,0	100,0	18,5
66,5	5,0	95,0	33,5	67,6	0,0	100,0	32,4
33,5	0,0	100,0	66,5	38,0	0,0	100,0	62,0
6,1	5,0	95,0	93,9	10,2	0,0	100,0	89,8



Obrázek 3.6: Celková dosažená úspěšnost bez prahování počtu bodů

databáze. Schopnost systému správně klasifikovat známé a neznámé vzory je shrnuta v tabulce 3.3, ve které jsou uvedeny hodnoty procentuální míry správně pozitivních závěrů (True Positive Rate, TPR), nesprávně pozitivních závěrů (False Positive Rate, FPR), správně negativních závěrů (True Negative Rate, TNR) a nesprávně negativních závěrů (False Negative Rate, FNR).

Jelikož bylo při testech dosaženo velmi nízkých hodnot FPR, byl algoritmus otestován znovu shodným způsobem, avšak s vynecháním klasifikace výskytů na základě počtu bodů (viz sekce 2.3.4). Celková úspěšnost v závislosti na pozorovacím úhlu je znázorněna na obrázku 3.6, hodnoty TPR, FPR, TNR a FNR pak v tabulce 3.4.

Z výsledků je patrné, že při prahování na základě počtu nalezených bodů dochází k poměrně výraznému snížení rozpoznávacího skóre především pro vyšší pozorovací úhly. K tomu dochází díky vlastnosti detektoru FH zájmových bodů a obecně malé robustnosti deskriptorů vůči geometrickým transformacím. Pro vyšší pozorovací úhly tak algoritmus detekuje nedostatečné množství bodových korespondencí mezi vzorovým a testovacím obrázkem a nalezené výskyty nevyhovují třetímu kritériu verifikace (2.9). Při vynechání tohoto kritéria tedy dochází k nárůstu rozpoznávacího skóre pro vyšší pozorovací úhly, avšak pouze za cenu zvýšení pravdě-

Tabulka 3.4: Tabulka hodnot TPR, FPR, TNR a FNR (vše v [%])

T1				T3			
TPR	FPR	TNR	FNR	TPR	FPR	TNR	FNR
91,5	15,0	85,0	8,5	97,2	0,0	100,0	2,8
82,9	15,0	85,0	17,1	93,5	0,0	100,0	6,5
78,0	10,0	90,0	22,0	84,3	0,0	100,0	15,7
65,9	0,0	100,0	34,1	67,6	0,0	100,0	32,4
27,4	5,0	95,0	72,6	38,0	0,0	100,0	62,0
4,9	0,0	100,0	95,1	12,0	0,0	100,0	88,0

Tabulka 3.5: Časová náročnost rozpoznávání

učení jednoho vzoru	53,3 ms
detekce zájmových bodů	148,8 ms
vytváření deskriptorů	43,7 ms
vyhledávání korespondencí	50,0 ms
shluková analýza	0,2 ms
odhad transformace, verifikace	7,6 ms
celkem	303,6 ms

podobnosti nesprávně pozitivních závěrů, tzn. že poměrně často může docházet k falešným výskytům vzoru. Tato vlastnost se pak projevila především při testování na reálných datech.

Pro doplnění jsou v tabulce 3.5 uvedeny průměrné hodnoty časů jednotlivých kroků algoritmu, potřebných pro nalezení vzorů na testovacích obrázcích. Uvedených hodnot bylo dosaženo na PC s procesorem Athlon XP 2200+ (1,8GHz) a 1280MB RAM.

3.5 Ukázka použití algoritmů na reálných datech

Pro ověření funkčnosti systému byly algoritmy otestovány v aplikaci automatického vyhledávání reklamních značek. Byla vytvořena malá databáze 4 log, viz obr. 3.7. Dále byla vytvořena malá testovací sada 41 obrázků, na kterých se uvedená loga nacházela. Na těchto obrázcích jsou loga různě deformovaná, prostorově natočená, nebo částečně zakrytá. Parametry systému byly nastaveny shodným stejným způso-

(a) ING (b) Kenvelo (c) Nivea (d) Škoda

Obrázek 3.7: Vyhledávaná loga

Tabulka 3.6: Rozpoznávací skóre na reálných datech

logo	celkem	rozpozn.	FP	úsp. [%]
ING	11	10	0	90,9
Kenvelo	6	5	0	83,3
Nivea	39	30	0	76,9
Škoda	15	12	0	80,0
celkem	71	57	0	80,3

Tabulka 3.7: Rozpoznávací skóre na reálných datech při vyhledávání vzorů jednotlivě

vzor	celkem	rozpozn.	FP	úsp. [%]
ING	11	10	1	90,9
Kenvelo	6	5	1	83,3
Nivea	39	35	5	89,7
Škoda	15	13	0	86,7
celkem	71	63	7	88,7

bem jako při testování na databázi SOIL. Dosažené výsledky jsou uvedeny v tabulce 3.6, kde FP značí počet falešných výskytů (False Positive).

Jedním z problémů algoritmu je, že neřeší případy, kdy jednotlivé vzory obsahují sobě podobné části. Pokud jsou si nějaké hledané vzory částečně podobné, vznikají poté v databázi deskriptorů podobné příznakové vektory. Při vyhledávání nejbližšího souseda je pak vždy vybrán pouze jeden, který vykazuje největší podobnost. Tento výběr však díky podobnosti nejbližšího a druhého nejbližšího souseda často nemusí splňovat poměrové kritérium (1.24), což má za následek nedostatečný počet korespondencí a snížení rozpoznávacího skóre. Tato vlastnost komplikuje také shlukovou analýzu, kdy ze dvou podobných deskriptorů z různých vzorů je vybrán nejpodobnější víceméně náhodně v závislosti na konkrétních geometrických transformacích v okolí detekovaného zájmového bodu a výsledné shluky jsou tak menší. Algoritmus byl tedy otestován na testovací sadě znovu, přičemž vzory byly vyhledávány jednotlivě, tj. opakovanou aplikací algoritmů pro každý vzor samostatně. Tento způsob však není vhodný pro velké databáze, jelikož má vzhledem k počtu vzorů lineární časovou náročnost. Dosažené výsledky jsou uvedeny v tabulce 3.7.

Oproti předchozímu případu bylo dosaženo přibližně o 9% vyššího celkového rozpoznávacího skóre. Při testování se však projevil problém falešných výskytů, způsobený částečnou podobností log, konkrétně Kenvelo a Nivea. Možným řešením v případě samostatného vyhledávání vzorů je zvýšení prahu kritéria (2.9), avšak pouze za cenu snížení celkového rozpoznávacího skóre. Příklad správně rozpoznaného vzoru (ING) je znázorněn na obr. 3.8.

3.6 Poznámky k softwarové realizaci

V rámci této práce byl vytvořen počítačový program s grafickým uživatelským rozhraním (GUI), umožňující testování implementovaných algoritmů. Program byl vytvořen v programovacím jazyce C++. Při jeho vytváření byla použita knihovna

Obrázek 3.8: Správně rozpoznané logo

Tabulka 3.8: Podporované formáty obrázků

Formát	Přípona souboru
Windows bitmap	BMP, DIB
Obrázky JPEG	JPEG, JPG, JPE
Portable Network Graphics	PNG
Portable image format	PBM, PGM, PPM, PXM, PNM
Sun rasters	SR, RAS
Obrázky TIFF	TIFF, TIF
Obrázky OpenEXR HDR	EXR
Obrázky JPEG 2000	JP2

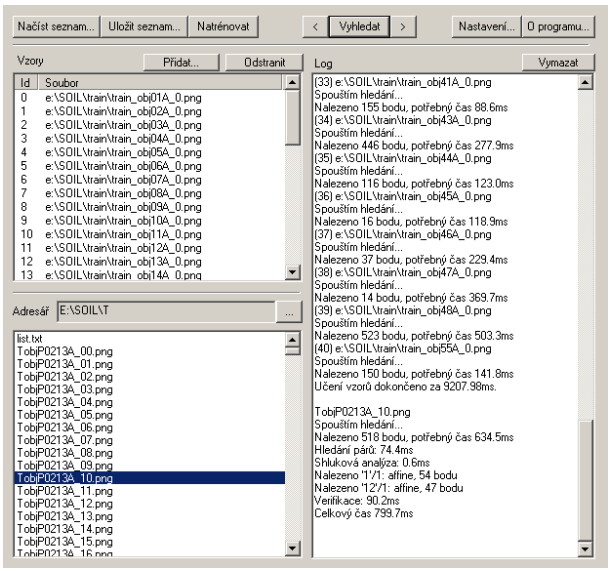
OpenCV[4] verze 1.1pre. Tato knihovna je nezávislá na platformě, má otevřený kód a je vydána pod licencí GNU/GPL. Byla využita pro načítání obrázků různých formátů a vytváření integrálních obrazů. Podporované formáty jsou uvedeny v tab. 3.8. Dále byla upravena část jejího kódu pro odhad projektivní transformace a doplněna o možnost robustního odhadu afinní transformace ze zadané množiny korespondencí bodů.

Další použitým softwarem třetí strany je knihovna FLANN[5], která je rovněž vydána pod licencí GNU/GPL, tedy s otevřeným kódem. V této knihovně jsou implementovány algoritmy efektivního aproximativního vyhledávání nejbližšího souseda, viz sekce 2.2. Navíc pak umožňuje automatickou konfiguraci parametrů vyhledávání, viz [27].

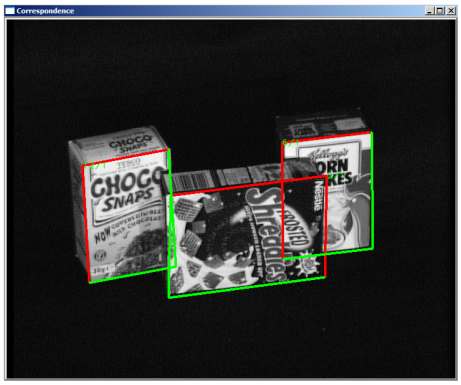
Pro usnadnění vytváření grafického uživatelského prostředí testovacího softwaru byla využita knihovna WTL[6]. Podobně jako v předchozích případech má i tato knihovna otevřený kód, přičemž vydána je pod licencí CPL 1.0. Tato knihovna je úzce spojená s aplikačním rozhraním WinAPI a objektovým modelem COM, díky čemuž je dostupná pouze pro operační systém MS Windows.

Klientská část hlavního okna aplikace je zobrazena na obr. 3.9. Aplikace umožňuje načítat a ukládat seznamy vzorů pomocí textových souborů, případně přidávat či odebírat vzory jednotlivě. Dále pak zobrazuje a prochází soubory v zadaném adresáři. Umožňuje rovněž nastavit parametry jednotlivých částí rozpoznávacího schématu, kterými jsou vyhledávání bodů, vytváření deskriptorů, hledání nejbliž-

[4]http://sourceforge.net/projects/opencvlibrary/
[5]http://www.cs.ubc.ca/~mariusm/index.php/FLANN/FLANN
[6]http://sourceforge.net/projects/wtl



Obrázek 3.9: Hlavní okno aplikace



Obrázek 3.10: Ukázka rozpoznávání (databáze SOIL)

šího souseda a verifikace. Výsledek rozpoznávání (obr. 3.10) pak zobrazuje graficky v samostatném okně a na textovém výstupu.

Závěr

V rámci této práce byl vytvořen systém pro automatické rozpoznávání vzorů v obraze. Tento systém umožňuje detekovat libovolné množství zadaných vzorů jak v lineárním, tak v sublineárním čase, což znamená, že nedochází k významnému zpomalení systému ani pro větší množství zadaných vzorů. Na obrázcích s rozlišením PAL (720×576) bylo pro 40 zadaných vzorů dosaženo průměrného času detekce a rozpoznání přibližně 0,3s, systém tedy pracuje téměř v reálném čase. Algoritmus je přitom invariantní vůči rovinné rotaci a zvětšení měřítka vzoru, vyhledává tedy objekty v libovolné orientaci a velikosti větší než původní vzor. Robustní je pak vůči zmenšení měřítka a změně pozorovacího úhlu projevujícího se jako afinní či projektivní transformace. Pro čelní pohledy s přibližnou odchylkou pozorovacího úhlu $\pm 10°$ byla na testovací databázi dosažena celková úspěšnost 90,1%, resp. 92,9% v závislosti na zvolených parametrech. Rovněž byla ověřena funkčnost systému v reálné aplikaci, konkrétně v automatickém vyhledávání reklamních log v obraze. Byla vyhledávána celkem 4 loga, přičemž bylo dosaženo celkové úspěšnosti 80,3%, resp. 88,7%, opět v závislosti na zvolených parametrech a toleranci falešných výskytů.

Jako největší problém algoritmů se jevila změna pozorovacího úhlu. Na testovací databázi bylo pro obrázky vzorů nasnímané z přibližného úhlu 25-30° dosaženo celkového rozpoznávacího skóre pouze 7,5%, resp. 30,6%, v závislosti na zvolených parametrech a toleranci falešných výskytů. Algoritmus se naopak jevil poměrně robustní vůči ostatním typům transformací, jako jsou změna osvětlení, komprese obrázku nebo zašumění. Možným řešením problému změny pozorovacího úhlu by mohlo být použití více pohledů pro jeden vzor při trénování databáze. Podobně však jako při vyhledávání vzájemně podobných vzorů by však při tomto způsobu nevyhnutelně docházelo k vytváření podobných deskriptorů v natrénované databázi, což má za následek obtížné vyhledávání podobnosti mezi vzorem a testovacím obrázkem. Dalším vylepšením systému by tak mohlo být vyhledávání podobnosti mezi zadanými vzory a shlukování podobných deskriptorů. Jiným možným řešením problému změny pozorovacího úhlu by pak mohlo být použití afinně invariantních detektorů zájmových bodů s následnou normalizací nalezené oblasti.

Literatura

[1] David Arthur, Sergei Vassilvitskii. *k-means++: The advantages of careful seeding*. Technical Report 2006-13, Stanford InfoLab, 2006. ISBN 978-0-898716-24-5.

[2] A. P. Ashbrook, N. A. Thacker, P. I. Rockett, and C. I. Brown. *Robust recognition of scaled shapes using pairwise geometric histograms*. In BMVC '95: Proceedings of the 6th British conference on Machine vision (Vol. 2), p. 503–512, Surrey, UK, UK, 1995. ISBN 0-9521898-2-8.

[3] Marian Stewart Bartlett, Javier R. Movellan, and Terrence J. Sejnowski. *Face recognition by independent component analysis*. IEEE Transactions on Neural Networks, 13, p. 1450–1464, 2002. ISSN 1045-9227.

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. *Surf: Speeded up robust features*. In Proceedings of European Conference on Computer Vision, p. 404–417, 2006. ISSN:1077-3142.

[5] Jeffrey S. Beis and David G. Lowe. *Shape indexing using approximate nearest-neighbour search in high-dimensional spaces*. In CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97), 1997. ISBN 0-8186-7822-4.

[6] S. Belongie, J. Malik, and J. Puzicha. *Shape matching and object recognition using shape contexts*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 24(4), p. 509–522, 2002. ISSN 0162-8828.

[7] Matthew Brown and David Lowe. *Invariant features from interest point groups*. In In British Machine Vision Conference, p. 656–665, 2002.

[8] Min-Seok Choi and Whoi-Yul Kim. *A novel two stage template matching method for rotation and illumination invariance*. Pattern Recognition, 35(1), p. 119–129, 2002.

[9] Martin A. Fischler and Robert C. Bolles. *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*. Commun. ACM, 24(6), p. 381–395, 1981. ISBN 0-934613-33-8.

[10] W. T. Freeman and E. H. Adelson. *The design and use of steerable filters*. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 13(9), p. 891–906, 1991. ISSN 0162-8828.

[11] Yoav Freund and Robert E. Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. 1995. ISSN 0022-0000.

[12] Guodong Guo and Charles R. Dyer. *Patch-based image correlation with rapid filtering*. Computer Vision and Pattern Recognition, IEEE Computer Society Conference, 2007. ISBN 1-4244-1180-7.

[13] Chris Harris and Mike Stephens. *A combined corner and edge detector*. Proceedings of 4th Alvey Conference, Cambridge, pp.147-51, 1988.

[14] Yan Ke and R. Sukthankar. *Pca-sift: a more distinctive representation for local image descriptors*. In 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, p. 506–513, 2004. ISBN 0-7695-2158-4.

[15] Hae Yong Kim and Sidnei Alves de Araújo. *Grayscale template-matching invariant to rotation, scale, translation, brightness and contrast*. 2007.

[16] J. P. Lewis. *Fast normalized cross-correlation*. 1995. URL: <http://www.idiom.com/~zilla/Work/nvisionInterface/nip.pdf>

[17] R. Lienhart and J. Maydt. *An extended set of haar-like features for rapid object detection*. 2002. ISBN 0- 7803-7622-6.

[18] T. Lindeberg. *Scale-space theory: A basic tool for analysing structures at different scales*. J. of Applied Statistics, 21(2), p. 224–270, 1994.

[19] Tony Lindeberg. *Feature detection with automatic scale selection*. International Journal of Computer Vision, 30:79–116, 1998.

[20] David G. Lowe. *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, 60, p. 91–110, 2004. ISSN 0920-5691.

[21] J. Matas, O. Chum, M. Urban, and T. Pajdla. *Robust wide baseline stereo from maximally stable extremal regions*. In In British Machine Vision Conference, volume 1, p. 384–393, 2002. ISSN 0262-8856.

[22] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. *A comparison of affine region detectors*. Int. J. Comput. Vision, 65(1-2), p. 43–72, 2005. ISSN 0920-5691.

[23] Krystian Mikolajczyk and Cordelia Schmid. *Indexing based on scale invariant interest points*. In In Proc. ICCV, p. 525–531, 2001. ISBN 0-7695-1143-0.

[24] Krystian Mikolajczyk and Cordelia Schmid. *An affine invariant interest point detector*. Int. J. Comput. Vision, 2002. ISSN 0920-5691.

[25] Krystian Mikolajczyk and Cordelia Schmid. *A performance evaluation of local descriptors*. IEEE Trans. Pattern Anal. Mach. Intell., 27(10), p. 1615–1630, 2005. ISSN 0162-8828.

[26] Hans Peter Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover*. PhD thesis, Stanford, CA, USA, 1980.

[27] Marius Muja. *Fast approximate nearest neighbors with automatic algorithm configuration*. In International Conference on Computer Vision Theory and Applications (VISAPP), 2009. ISBN 978-989-8111-69-2.

[28] A. Neubeck and L. van Gool. *Efficient non-maximum suppression.* Proceedings of the 18th International Conference on Pattern Recognition - Vol. 03, 2006. ISSN1051-4651.

[29] Arthur R. Pope. *Model-based object recognition - a survey or recent research.* Technical Report TR-94-04, 1994.

[30] Laura Rebollo-Neira and David Lowe. *Optimised Orthogonal Matching Pursuit Approach.* 2002. ISSN 1070-9908.

[31] Cordelia Schmid, Roger Mohr, and Christian Bauckhage. *Evaluation of Interest Point Detectors.* Int. J. Comput. Vision, 37(2), p. 151–172, 2000. ISSN 0920-5691.

[32] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image Processing, Analysis, and Machine Vision.* Thomson-Engineering, 2007. ISBN-13 9780534953935.

[33] Feng Tang and Hai Tao. *Fast Multi-scale Template Matching Using Binary Features.* Applications of Computer Vision, IEEE Workshop, 2007. ISSN 1550-5790.

[34] Matthew Turk and Alex Pentland. *Eigenfaces for recognition.* Neuroscience, 3(1), p. 71–86, 1991. ISSN 0898-929X.

[35] Tinne Tuytelaars and Krystian Mikolajczyk. *Local invariant feature detectors: a survey.* Found. Trends. Comput. Graph. Vis., 3(3), p. 177–280, 2008. ISSN 1572-2740.

[36] Paul Viola and Michael J. Jones. *Robust real-time face detection.* Journal of Computer Vision, 57(2), p. 137–154, 2004. ISSN 0920-5691.

[37] Andrew P. Witkin. *Scale-space filtering.* In 8th Int. Joint Intelligence, vol. 2, p. 1019–1022, 1983.

[38] Chih-Cheng Wei Yi-Hsien Lin, Chin-Hsing Chen. *Translation, rotation, and scale-invariant template matching.* 2006.

[39] Zhifeng Zhang. *Matching pursuits with time-frequency dictionaries.* Transactions on Signal Processing, 41, p. 3397–3415, 1993. ISSN 1053-587X.

Obsah přiloženého DVD

- Text diplomové práce ve formátu PDF
- Použitá literatura ve formátu PDF
- Testovací databáze SOIL
- Sada testovacích obrázků reklamních log
- Software pro automatickou detekci a rozpoznávání vzorů v obraze včetně zdrojových kódů