VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

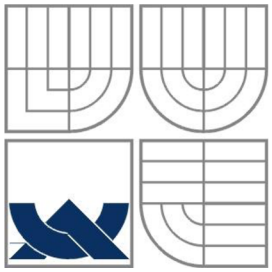
DETEKCE OBJEKTŮ V OBRAZE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

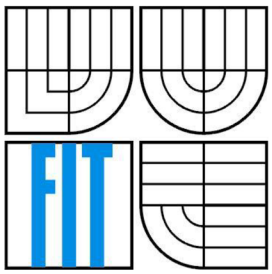
AUTOR PRÁCE
AUTHOR

Bc. JIŘÍ KUBÍNEK

BRNO 2009



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE OBJEKTŮ V OBRAZE

DETECTING OBJECTS IN IMAGES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. JIŘÍ KUBÍNEK

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL HRADIŠ

BRNO 2009

Abstrakt

Práce je věnovaná metodám detekce objektů v obraze. Seznamuje čtenáře se základními přístupy a algoritmy užívanými v této problematice, zejména pak s algoritmem AdaBoost, jeho rozšířením WaldBoost a s některými příznaky užívanými pro detekci objektů. Významnou část práce tvoří rozšíření datových sad pro trénování klasifikátoru a implementace histogramu gradientů pro rozšíření stávajícího systému pro detekci objektů. Nedílnou součástí práce je zhodnocení dosažených výsledků v podobě provedených experimentů.

Abstract

This work is dedicated to methods used for object detection in images. There is a summary of several approaches and algorithms to solve this matter, especially AdaBoost algorithm with its improvement, WaldBoost and several features used for object detection. Vital part of this work is dedicated to extending training datasets for classifier training and extending the current object detection framework with histogram of gradients features implementation. Integral part of this work is analysis of results by experiments evaluation.

Klíčová slova

Detekce objektů, Top-down, Bottom-up, Template matching, Appearance-based metody, Viola-Jones detektor, AdaBoost, Haarovy vlnky, Integrální obraz, WaldBoost, Local Binary Patterns, Histogram gradientů, SVM, Náhodné transformace

Keywords

Object detection, Top-down, Bottom-up, Template matching, Appearance-based methods, Viola-Jones detector, AdaBoost, Haar waves, Integral image, WaldBoost, Local Binary Patterns, Histogram of Gradients, SVM, Random transformations

Citace

Jiří Kubínek: Detekce objektů v obraze, diplomová práce, Brno, FIT VUT v Brně, 2009

Detekce objektů v obraze

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Michala Hradiše. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jiří Kubínek
1.6.2009

Poděkování

Tímto bych rád poděkoval vedoucímu mé práce, panu Ing. Michalovi Hradišovi za vedení mé práce, cenné rady a podnětné návrhy, které mi vždy pomohly k pochopení problematiky a navedly mne na správnou cestu.

Obsah

Úvod

Problematika detekce objektů v obraze se v poslední době těší nebývalému zájmu. Zasahuje do mnoha oblastí lidské činnosti, počínaje průmyslovým nasazením pro automatickou kontrolu výstupní kvality a konče např. u detekce úsměvu v digitálních fotoaparátech. Detekce objektů spadá do poměrně mladé oblasti informačních technologií nazývané počítačové vidění (computer vision). Vývoj v této oblasti se začal rozmáhat až koncem sedmdesátých let dvacátého století, kdy začaly být počítače dostatečně výkonné pro tyto účely.

Ve své práci se budu věnovat zejména detekci obličejů, která je klíčová pro celou řadu dalších úloh. Zjednodušeně lze tento proces vysvětlit jako oddělení částí obrazu obsahující obličej od ostatních částí obrazu. Pokud by tato metoda nalezla právě všechny obličeje v obraze, tedy by žádný nevynechala a zároveň neoznačila žádnou oblast bez obličeje za obličej, jednalo by se o dokonalou metodu. Jak se však přesvědčíme v následujících kapitolách, v dnešní době existující postupy se tomuto ideálu pouze přibližují. Kromě obličejů je však v mé práci věnována pozornost také detekci chodců a dopravních značek.

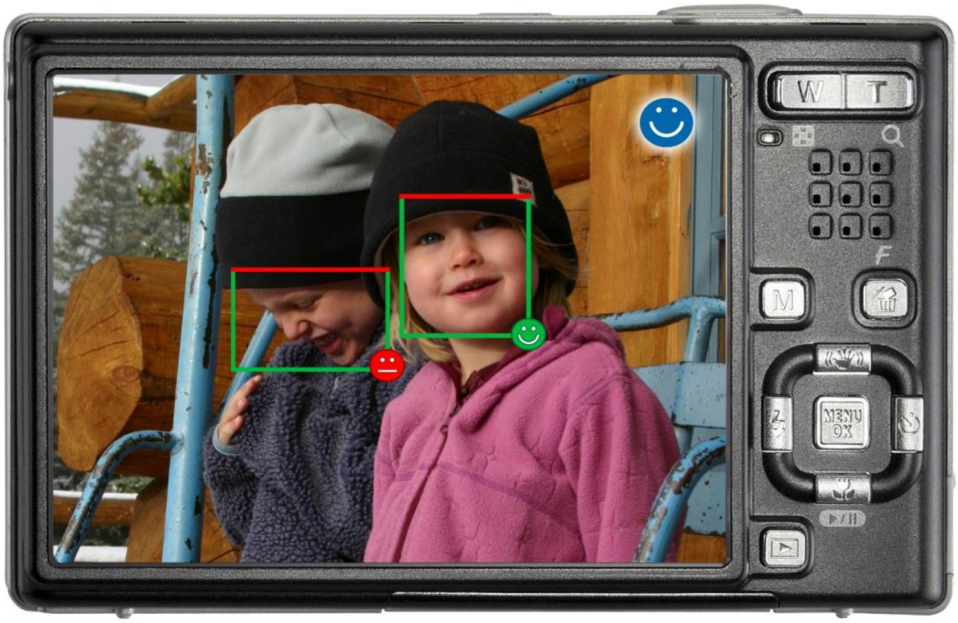
První kapitolu jsem zasvětil přehledu existujících přístupů k diskutované problematice. V téže kapitole je také popsán Viola-Jones detektor a WaldBoost, učící se algoritmus, který užívám v další práci. Uvedeny jsou zde také vybrané příznaky, které jsou pro tuto práci podstatné. Druhá kapitola pak shrnuje navržená rozšíření stávající implementace systému pro detekci objektů. Výsledky těchto rozšíření prostřednictvím provedených experimentů prezentuje kapitola třetí. Obsahem jsou grafy porovnávající různá nastavení příznaků, ale také samotné příznaky mezi sebou na třech různých datových sadách. Součástí je také obecné shrnutí poznatků z těchto experimentů.

Text této diplomové práce navazuje na semestrální projekt, ze kterého vycházejí zejména kapitoly 1.1, 1.2 a 1.3 obsahující přehled existujících přístupů k detekci objektů. Semestrální projekt již také obsahoval kapitoly 2.2 a 2.3 věnující se metodám vedoucím k rozšíření trénovacích datových sad. Součástí bylo také zhodnocení jejich přínosů, uvedené v kapitolách 3.1 a 3.2.

1 Detekce objektů v obraze

Jak jsem již zmínil v úvodu, motivací pro hledání objektů v obraze existuje mnoho. Mezi nejvýznamnější aplikace lze jistě zařadit detekci dopravních značek, SPZ, hledaných vozidel, navigaci robotů či různé pomůcky pro nevidomé. Pokud se zaměříme na detekci obličejů, lze nalézt uplatnění např. u kamer a fotoaparátů (zaostření na obličej, detekce úsměvu) nebo pro komunikaci člověka s počítačem (odhad směru pohledu nebo třeba rozpoznání nálady a emocí). Důležitou oblastí je také bezpečnost, kde se detekce obličejů uplatní např. při sledování osob a jejich identifikaci. V neposlední řadě má tato technika využití u digitálních záznamů, kde pomocí detekce obličejů lze vyhledávat dle osob, nebo např. provádět automatický střih video sekvencí.

Cílem této snahy je, zjednodušeně řečeno, přiblížit se lidskému vnímání světa. Přesněji řečeno, detekovat všechny smysluplné objekty v obraze a klasifikovat je dle jejich příslušnosti do tříd. Současný stav má však k této vizi poměrně daleko. Existují komerční aplikace, např. již zmíněná detekce obličeje ve fotoaparátech (obrázek 1.1), která pracuje v reálném čase s přijatelnou úspěšností. Pomocí dosud objevených metod však nelze detekovat více různých objektů v reálném čase. Rovněž v případě sledování a identifikace osob je současné využití poměrně limitované. Překážek k dosažení či alespoň významnému přiblížení k dokonalému stavu je více. Vývoj je náročný nejen časově, ale také znalostně a v neposlední řadě i finančně. Přesto je mu věnována obrovská pozornost a lze tak doufat, že se již brzy dočkáme významných pokroků přibližující nás k vizi počítačů s lidským vnímáním.



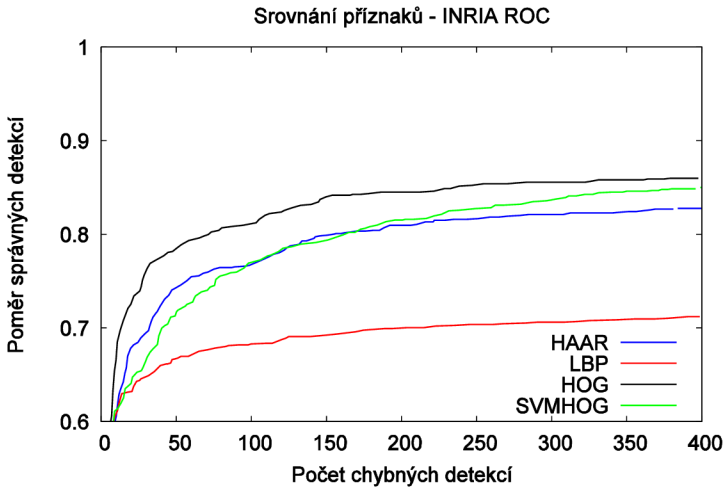
Obrázek 1.1: Příklad detekce obličejů ve fotoaparátu Canon PowerShot G6

Prvním krokem každého detekčního systému je nalezení pozic v obraze, na kterých se nacházejí hledané objekty, v našem případě obličeje. Tento krok je však poměrně náročný z důvodu značné variability ve velikosti, pozici, orientaci (pootočení) a úhlu pohledu (zepředu, zboku). Nemalý význam má také výraz obličeje, zakrytí jeho částí a také osvětlení.

Několikrát jsem již použil termín detekce, aniž bych jej definoval. Nyní to tedy napravím. V závorkách uvádím zažitý anglický název. Máme-li libovolný obraz, cílem detekce obličeje (face

detection) je určit, zda se v daném obraze obličeje nacházejí, či nikoliv a pokud ano, určit pozici a velikost každého z nich. S tímto problémem je blízce spjata řada dalších. Pokud hledáme pozici obličeje v obraze a víme, že se v něm právě jeden nachází, jedná se o lokalizaci obličeje (face localization). Jde tedy o zjednodušenou detekci. Dalším problémem je identifikace obličeje (face identification, face recognition), kdy porovnáváme vstupní obličej s databází a hledáme případnou shodu. Smyslem tohoto procesu je ověřit identitu dané osoby. Význam má také sledování obličeje (face tracking), jehož cílem je sledování pozice a případně orientace obličeje ve video sekvenci v reálném čase. Rozpoznání výrazu obličeje (facial expression recognition) si klade za cíl rozpoznání citového rozpoložení (veselý, smutný, znechucený apod.) sledované osoby. Je zjevné, že pro řešení všech výše uvedených problému je prvním krokem detekce obličeje (případně lokalizace, tyto termíny je vhodné odlišovat).

Než přistoupím k rozdělení přístupů k detekci objektů, zmíním ještě několik používaných pojmů. Poměr mezi počtem správně detekovaných objektů a počtem všech objektů se označuje jako poměr správných detekcí (hit rate). Oblast definována detektorem jako objekt se považuje za správnou detekci, pokud překryv detekované oblasti se skutečnou je vyšší, než určitá hranice. Obecně mohou detektory produkovat dva typy chyb: false negatives, kdy jsou vynechány některé objekty a false positives, kdy jsou naopak detektorem označena místa, která ve skutečnosti nejsou hledaným objektem. V praxi se pak tyto chyby navzájem ovlivňují. Lze nastavit detektor tak, aby našel co nejvíce objektů, pak ale bude produkovat více falešných detekcí. Nebo lze minimalizovat počet falešných detekcí, pak ale detektor pravděpodobně některé objekty nenajde. Pro zobrazení výkonnosti detektoru budu užívat ROC (Receiver–Operating Charakteristic) křivku. Příklad takové křivky je na obrázku 1.2. Vodorovná osa znázorňuje počet chybných detekcí (false positives), svislá osa pak poměr správných detekcí (hit rate). Jak je patrné, detektor označený červenou barvou vykazuje nejhorší výsledky, zatímco černě znázorněný detektor si vede nejlépe.



Obrázek 1.2: Ukázka ROC křivky

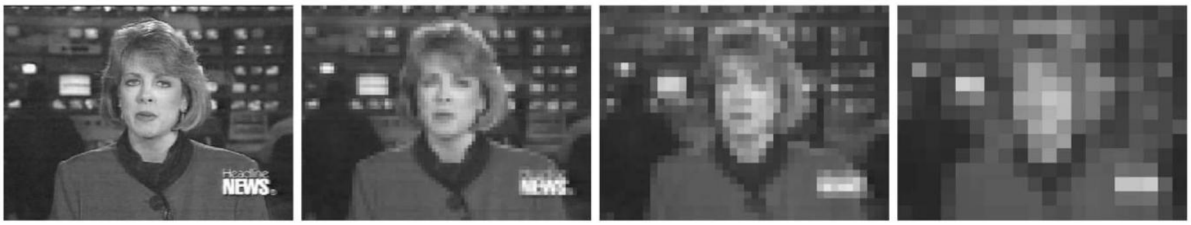
1.1 Existující přístupy

V této kapitole se budu věnovat existujícím technikám používaných k detekci obličejů v obraze [1]. Metody pracující nad jedním obrazem rozdělujeme do 4 kategorií. Samozřejmě existují metody přesahující hranice jednotlivých kategorií, o těch se zmíním na konci této kapitoly. Zvláštní pozornost

pak věnuji přístupu uvedenému v kapitole 1.1.4 (Appearance-based metody), který je zvláště důležitý pro následující kapitoly.

1.1.1 Top-down

První možný přístup je shora dolů (top-down). Jedná se o metodu založenou na vědomostech o hledaném objektu (knowledge-based). Principem je vyhledání vhodných kandidátů na nízké úrovni detailů a jejich následná verifikace na vyšší úrovni detailů. Jako příklad lze opět uvést případ detekce obličeje. Na základě našich znalostí víme, že většina obličejů v obraze obsahuje dvě symetrické oči, nos a ústa. Mezi těmito částmi obličeje pak definujeme vztahy na základě relativní vzdálenosti a pozice. V obraze pak hledáme jednotlivé části obličeje a na základě vytvořených pravidel určujeme kandidáty. Pokud se hledané objekty nacházejí na vhodném (jednoduchém) pozadí ve vhodné pozici (např. jen zepředu), vykazuje tato metoda velmi dobré výsledky. Nevýhodou je nutnost převádět znalosti o hledaném objektu do srozumitelných pravidel. S tím také souvisí určení síly pravidel. Pokud je nastavíme příliš striktně, mohou být některé objekty vynechány. A naopak, pokud budou pravidla příliš obecná, hrozí větší množství falešných detekcí. Náročnost tvorby pravidel dále stoupá, pokud chceme detekovat objekty z více úhlů pohledů. Vytvořit pravidla pro všechny možné situace je velmi komplikované.



(a) (b) (c) (d)

Obrázek 1.3: (a) n = 1, původní obraz. (b) n = 4. (c) n = 8. (d) n = 16. Ukázka hierarchie obrázků s podvzorkováním. Každá buňka se skládá z n * n pixelů s hodnotou průměru z těchto pixelů

Tohoto přístupu využili Yang a Huang [2]. Sestavili hierarchii tří úrovní pravidel. Na nejvyšší úrovni jsou vyhledáváni všichni kandidáti na obličej posunováním skenovacího okna přes všechny pozice v obraze a aplikováním pravidel na každé z nich. Pravidla na vyšší úrovni jsou obecným popisem obličeje, zatímco pravidla na nižší úrovni se zaměřují na detaily obličeje. Na základě podvzorkování (prováděném průměrováním) se vytvoří hierarchie obrázků podobně, jako na obrázku 1.3. V obrázku s nejnižším rozlišením se vyhledávají kandidáti na základě obecných pravidel pro obličej, kteří jsou následně zpracováni ve vyšším rozlišení. Následně je pro každého kandidáta z předchozí úrovně zpracován histogram a provedena detekce hran. Kandidáti, kteří postoupí dále, jsou podrobeni dalším pravidlům, která odpovídají částem obličeje, jako např. oči nebo ústa. Přestože tento přístup nevykazuje nejlepší výsledky (v testu s 60 obrázky bylo nalezeno 50 obličejů a zároveň v 28 obrázcích se objevila falešná detekce), myšlenka s hierarchií obrázků s podvzorkováním a sadou pravidel byla využita v řadě dalších studií.

1.1.2 Bottom-up

Dalším možným přístupem je zdola nahoru (bottom-up). Podobně jako předchozí metoda je i tato založená na vědomostech o hledaném objektu (knowledge-based). Základem je fakt, že lidé jsou

schopni snadno najít objekty v různých polohách za různého osvětlení a tak musí být možné nalézt nějaké společné vlastnosti či prvky, které jsou neměnné i přes uvedenou variabilitu. Bylo publikováno mnoho metod, které prvně detekují jednotlivé části objektů (u obličeje např. oči, ústa, nos) a na jejich základě pak sestavují statistický model popisující vztahy mezi jednotlivými detekovanými částmi. Podle něj se pak rozhodne, zda se na daném místě objekt vyskytuje, či nikoli. Problémem těchto metod je, že jednotlivé hledané části mohou být porušené z důvodů osvětlení, šumu a zakrytí. Komplikací jsou také stíny, které mohou vytvářet ostré hrany a ztížit či znemožnit tak správné oddělení jednotlivých částí objektu.

V dalším textu se již budu opět věnovat pouze detekci obličejů. Způsobů, jak oddělit oblast s obličejem od pozadí existuje samozřejmě více. Uvedu jeden z nich [3]. Na základě mapy hran (získaných Cannyho detektorem [4]) a heuristiky odstraňuje a shlukuje hrany tak, že zachová pouze hrany tvořící obrys obličeje. Tato hranice je následně aproximována elipsou oddělující oblast s hlavou od pozadí. Na databázi 48 obrázků dosahuje tento algoritmus 80% úspěšnosti.

Lidský obličej má jedinečnou strukturu, kterou lze také využít pro oddělení obličeje od ostatních objektů. Augusteijn a Skufca [5] vyvinuli metodu, která vyvozuje přítomnost obličeje z identifikace struktury podobné obličeji. Pracuje na oblastech 16x16 pixelů a předpokládá 3 typy struktur: pleť, vlasy a ostatní. Pro jejich klasifikaci využili neuronové sítě. Výsledky detekce ani lokalizace obličejů však nejsou zveřejněny.

Barva pleti se ukázala jako efektivní prvek pro detekci obličeje. Přestože se barva pleti mezi různými lidmi odlišuje, několik studií odhalilo, že hlavní odlišnost spočívá v různé intenzitě barvy a nikoli v chrominanci. V mnohých metodách byly navrženy různé modely barvy pleti. Ten nejjednodušší definuje oblast obsahující pleť užitím hodnot Cr a Cb (chrominanční složky barvy). S pečlivě vybranými prahy $[Cr_1, Cr_2]$ a $[Cb_1, Cb_2]$ je pixel považován za část pleti, pokud jeho složky (Cr, Cb) leží v daném rozmezí, tedy platí $Cr_1 \leq Cr \leq Cr_2$ a $Cb_1 \leq Cb \leq Cb_2$.

V poslední době se objevuje mnoho metod kombinující výše uvedené postupy. Mnoho z nich využívá obecných vlastností, jako jsou barva pleti, velikost a tvar, k nalezení kandidátů a ty následně verifikuje užitím lokálních prvků, jako jsou obočí, nos nebo vlasy. Typický přístup nejprve detekuje oblasti s barvou pleti a poté je spojuje např. shlukováním. Pokud je tvar takto vzniklé spojené oblasti eliptický, stává se kandidátem na obličej. Posledním krokem je verifikace v podobě detekce lokálních prvků.

1.1.3 Template matching

Třetí skupinou je vyhledávání vzorů (template matching). Přístup je založen na vytvoření modelu objektu obsahující tvar, barvu a texturu. Vytvořit jej je možné buď ručně, nebo na základě nějaké vhodné funkce. Následně se hledají v obraze jednotlivé prvky objektu samostatně a zjišťuje se míra podobnosti s vytvořeným modelem. Existence hledaného objektu je vyvozena z míry podobnosti jednotlivých prvků objektu. Jednoznačnou výhodou tohoto přístupu je snadná implementace, ukázalo se však, že pro detekci obličejů není vhodná z důvodů nízké odolnosti proti variabilitě. Změna velikosti, pozice nebo tvaru objektu významně ovlivňuje výsledky metody. Tento nedostatek řeší různá rozšíření pracující na větším počtu rozlišení, velikostí a s deformovatelnými vzory.

Jedna z prvních metod detekující obličeje [6] je založena na modelu tvořeného podvzory pro oči, nos, ústa a celkový tvar obličeje. Každý z těchto podvzorů je definován úsečkami. V zadaném obraze se vyhledávají úsečky na základě změny gradientu a následně se porovnávají s podvzory. Na základě míry podobnosti definovaného tvaru obličeje a nalezených úseček se nejprve určí kandidáti. Tito jsou následně verifikování dle zbylých podvzorů. Jinými slovy se nejprve hledají oblasti zájmu

(region of interest) a následně se v nich vyhodnocují detaily ke zjištění, zda se obličej na daném místě skutečně nachází. Tato myšlenka s hledáním oblastí zájmů a podvzory našla uplatnění v pozdějších studiích o detekci obličejů.

Další metoda, o které se zmíním, využívá deformovatelných vzorů [7]. Pro prvky obličeje jsou vytvořeny elastické modely na základě parametrizovatelných vzorů. Hrany, maxima a minima v obraze jsou na základě energetické funkce přiřazeny odpovídajícím parametrům vzoru. Nejlépe vyhovující tvar elastického modelu je nalezen minimalizací energetické funkce parametrů. Experimentální výsledky demonstrují dobrou schopnost vyhledávat elastické prvky, nevýhodou však je nutnost umístění deformovatelného vzoru do blízkosti objektu zájmu.

1.1.4 Appearance-based metody

V kontrastu s předešlou metodou vyhledávání vzorů, kdy jsou modely definovány ručně, se u tohoto přístupu vytvářejí modely automatickým učením z poskytnutých dat. Obecně se tyto metody spoléhají na techniky ze statistické analýzy a strojového učení k nalezení relevantní charakteristiky obličejových částí obrazu a pozadí (neobličejových částí). Naučená charakteristika ve formě modelů rozložení nebo rozlišujících funkcí se následně používá pro detekci objektů. Pro navýšení efektivity výpočtu a účinnosti detekce se často snižuje počet rozměrů. Důležitým zástupcem této kategorie metod je klasifikační algoritmus AdaBoost, kterému věnuji kapitolu 1.2.1.

Mnoho těchto metod lze chápat jako pravděpodobnostní systém. Vektor příznaků (feature vector) získaný z obrazu je zobrazen jako náhodná proměnná x, charakterizována pro obličeje a pozadí tzv. třídně podmíněnou funkcí hustoty pravděpodobnosti (class-conditional probability density function) $p(x|obličej)$ a $p(x|pozadí)$. Ke klasifikaci oblastí v obraze lze použít Bayesovské teorie rozhodování nebo metody maximum likelihood. Přímou implementaci Bayesovské klasifikace však nelze použít mimo jiné z důvodů vysokého počtu rozměrů proměnné x. Z tohoto důvodů se pozornost věnuje empiricky potvrzeným parametrickým a neparametrickým aproximacím funkcí $p(x|obličej)$ a $p(x|pozadí)$.

Jiným přístupem je najít vhodnou rozlišující funkci mezi třídami obličejů a pozadí. Konvenční postup spočívá v zobrazení obrazových vzorů do prostoru s nižším počtem rozměrů a následné aplikaci rozlišující funkce (většinou založené na vzájemné vzdálenosti) pro klasifikaci.

Ve své práci věnované identifikaci osob využívá Kohonen [8] tzv. charakteristických vektorů (eigenvectors) a jednoduché neuronové sítě. Ta odhaduje charakteristiku obličeje ze zarovnaného normalizovaného obrazu odhadnutím charakteristických vektorů. Pro tyto vektory se později zavedl termín Eigenface.

1.1.5 Shrnutí

Uvedl jsem čtyři základní kategorie, do kterých dělíme metody pro detekci objektů (zejména obličejů). Samozřejmě ale existují metody, které mohou být řazeny do více kategorií. Jako příklad mohu zmínit metody založené na vyhledávání vzorů, které obvykle užívají model obličeje a podvzory k nalezení prvků obličeje. Tyto prvky pak následně užívají k lokalizaci nebo detekci obličeje. Zmínit lze také poměrně křehkou hranici mezi metodami založenými na vědomostech a některými metodami založenými na vyhledávání vzorů, protože pro vytvoření vzorů využíváme obvykle lidských znalostí o obličeji.

Na druhé straně lze jistě definovat jiné rozdělení metod pro detekci objektů. Jako příklad může posloužit rozdělení metod podle toho, zda jsou založeny na lokálních vlastnostech objektu nebo

na objektu jako celku. Nicméně většina pramenů prezentuje toto rozdělení jako nejvhodnější pro většinu metod.

1.2 Viola-Jones detektor

První představení objektového detektoru Viola-Jones [9], [10] proběhlo v roce 2001. V originální podobě pracoval detektor pouze s šedo-tónovými obrazy a skládal se ze tří základních částí: klasifikačního algoritmu AdaBoost, Haarových příznaků a integrálního obrazu. Mezi jeho výhody se řadí rychlost, dostatečná spolehlivost a značná odolnost vůči variabilitě v osvětlení a velikosti sledovaného objektu. Pro tyto vlastnosti je detektor často využíván v praxi např. pro detekci obličejů a slouží jako základ pro mnoho pozdějších modifikací.

1.2.1 AdaBoost

Klasifikační algoritmus AdaBoost (Adaptive Boosting) [18] vychází ze skupiny metod strojového učení boosting. Cílem metody boosting je vylepšit přesnost klasifikace libovolného algoritmu strojového učení. Způsob, jak tohoto cíle dosáhnout, spočívá ve vytvoření více klasifikátorů označovaných jako slabé klasifikátory (weak classifiers) pomocí výběru vzorků ze základní trénovací množiny. Každý takovýto klasifikátor má relativně špatnou úspěšnost (jen o málo lepší než přesnost odhadu), avšak jejich vzájemným spojením vzniká silný klasifikátor (strong classifier), jehož celková klasifikační přesnost je zesílena (boosted).

AdaBoost (algoritmus 1.1) tedy využívá pro učení tzv. slabých klasifikátorů $h_t(x)$, které jsou vybírány z množiny klasifikátorů H. Na základě rovnic 1 vzniká jejich lineární kombinací nelineární, tzv. silný klasifikátor H(x):

$$f(x) = \sum_{t \in T} \alpha_t h_t(x) \qquad H(x) = sign\big(f(x)\big), X \to \{-1,1\},$$
$$H(x) = sign[\alpha_1 h_1(x) + \alpha_2 h_2(x) + \alpha_3 h_3(x) \dots \alpha_T h_T(x)], \tag{1}$$

kde $H(x)$ je výsledný nelineární silný klasifikátor, $h_t(x_i)$ je slabý klasifikátor odpovídající t-tému příznaku na vzorku x_i a α_t je váha daného slabého klasifikátoru.

Vstupem algoritmu je trénovací množina S složená z dvojic (x_i, y_i), kde x_i je vzorek a y_i je třída odpovídající vzorku i, $y_i \in \{-1,1\}$ pro i=1,…M, kde M je velikost trénovací množiny. Pro určení významu jednotlivých vzorků se užívá distribuční funkce D_t, která na počátku přiřadí všem vzorkům totožnou váhu a následně se opakují následující kroky:

- výběr nebo vytvoření slabého klasifikátoru s nejmenší chybou váženou pomocí D_t,
- výpočet váhy (α_t) slabého klasifikátoru podle velikosti jeho chyby,
- upravení distribuční funkce D_t (převážení dat).

Každý slabý klasifikátor, jak již bylo uvedeno, musí mít přesnost klasifikace vyšší, než kdyby prováděl klasifikaci náhodně. Pokud tedy chyba ε_t překročí hodnotu 0,5, není tato podmínka splněna a není tak zaručeno, že bude algoritmus konvergovat. Převážení dat zajistí, že váha u dobře klasifikovaných dat se sníží a naopak u špatně klasifikovaných dat se zvýší. Díky tomu se nevybere v každém kroku stejný slabý klasifikátor. AdaBoost redukuje trénovací chybu exponenciálně v závislosti na počtu slabých klasifikátorů. Příliš vysoký počet klasifikátorů může vést k přetrénování

(tedy ztrátě schopnosti generalizovat vlivem přílišného zaměření na konkrétní trénovací data). V praxi se však ukázalo, že k tomuto jevu dochází zřídka i pro velmi vysoké počty slabých klasifikátorů [18].

Vstup:
$S = \{(x_1, y_1), \dots, (x_N, y_N)\}$, počet iterací T
Distribuční funkce D_t
Inicializace vah

$$D_1(i) = \frac{1}{N}$$

Cyklus pro $t = 1, \dots, T$:
Výběr klasifikátoru na základě vážené trénovací chyby

$$\varepsilon_j = \sum_{i=1}^{N} D_t(i) I\left[y_i \neq h_j(x_i)\right]$$

$$h_t = \arg\min_{h_j \in H} \varepsilon_j$$

Pokud

$\varepsilon_t = 0$ nebo $\varepsilon_t \geq \frac{1}{2}$, pak konec cyklu

Nastavení

$$\alpha_t = \frac{1}{2} log\left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right)$$

Úprava vah

$D_{t+1}(i) = \frac{D_t(i) e^{-\alpha_t y_i h_t(x_i)}}{Z_t}$, kde

$$Z_t = \sum_{i=1}^{N} D_t(i) e^{-\alpha_t y_i h_t(x_i)}$$

Výstup

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Algoritmus 1.1: Adaptive Boosting (AdaBoost) [10]

1.2.2 Haarovy vlnky

Již bylo řečeno, že principem algoritmu AdaBoost je výběr a spojování slabých klasifikátorů. Je žádoucí, aby těchto slabých klasifikátorů bylo co největší množství. Zvyšuje se tím pravděpodobnost výběru slabého klasifikátoru s vyšší mírou přesnosti. Cílem je tedy získat co největší množství jednoduchých příznaků s minimálními výpočetními nároky. A právě těmto požadavkům vyhovují příznaky založené na principu podobném definici Haarovy vlnky (Haar-like features, obrázek 1.4). Hodnota takového příznaku je rovna rozdílu mezi sumou pixelů odpovídající bílé části a sumou pixelů odpovídající černé časti. Příznaky mohou být tvořeny dvěma (hranové), třemi (čárové) nebo čtyřmi (diagonální) obdélníkovými oblastmi. Aplikovány jsou na celý obraz, přičemž se mění jejich velikost počínaje velikostí 1×1 až na velikost vstupního obrazu. Tedy pro obraz o velikosti 19×19 získáme asi 64 000 hodnot příznaků, z nichž AdaBoost vybere jen malou podmnožinu.

Obrázek 1.4: Haarovy příznaky

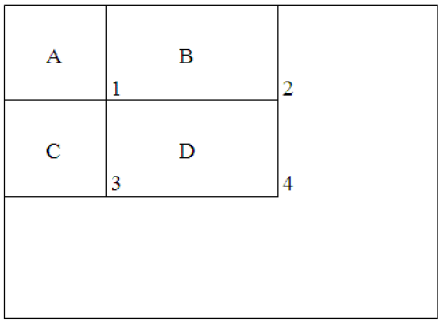
1.2.3 Integrální obraz

Pro zefektivnění výpočtu Haarových příznaků se využívá integrálního obrazu. Jedná se o transformaci vstupního obrazu, kde každý pixel reprezentuje sumu pixelů z obdélníkové oblasti ohraničené levým horním rohem a pozicí daného pixelu. Výpočet integrálního obrazu se řídí následující rovnicí:

$$I_{int}(x,y) = \sum_{\substack{x' \leq x \\ y' \leq y}} I(x',y'), \tag{2}$$

kde $I(x,y)$ představuje hodnoty intenzit jednotlivých pixelů a $I_{int}(x,y)$ jsou jednotlivé hodnoty integrálního obrazu. Výpočet integrálního obrazu lze také uskutečnit s použitím jednoho průchodu přes vstupní obraz.

Výpočet hodnot jednotlivých příznaků se s použitím integrálního obrazu výrazně urychlí, protože na výpočet sumy libovolného obdélníku v obraze postačí dvě operace sčítání, jedna operace odčítaní a čtyři přístupy do paměti, jak je znázorněno na obrázku 1.5. Hodnota integrálního obrazu na pozici 1 je rovna sumě pixelů odpovídající obdélníku A. Hodnota na pozici 2 je rovna A+B, na pozici 3 je hodnota A+C a konečně na pozici 4 je A + B + C + D. Tedy hledaná suma pixelů odpovídající obdélníku D je rovna výrazu 4 + 1 – (2 + 3), kde čísla udávají znázorněné pozice v obrázku.

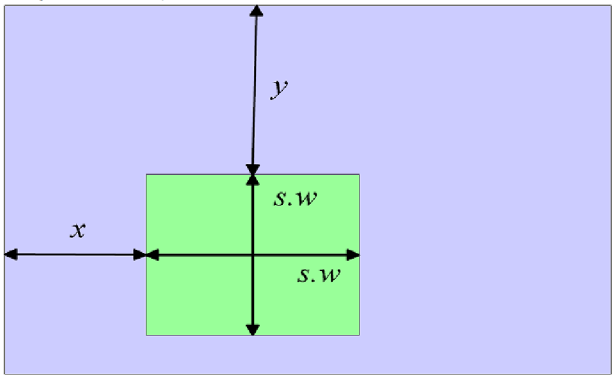


Obrázek 1.5: Výpočet sumy obdélníku pomocí integrálního obrazu

1.2.4 Kaskádové zapojení klasifikátorů

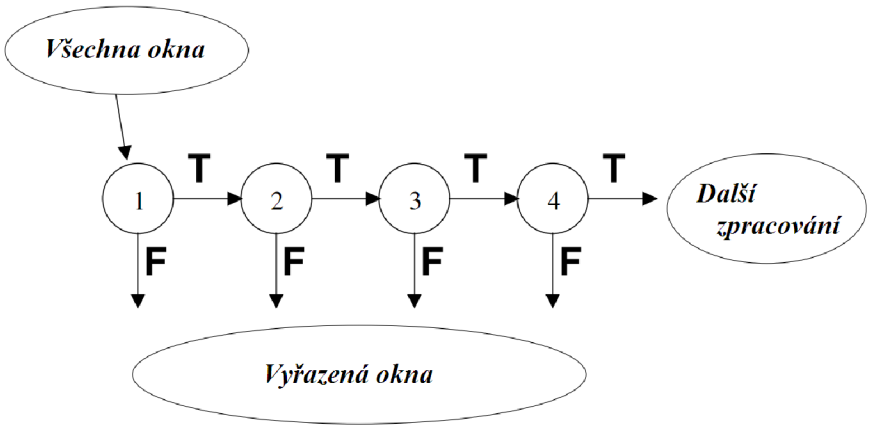
Aby bylo možné detekovat objekt v obraze, je nejprve nezbytné vytvořit trénovací množinu obsahující jak pozitivní vzory objektů (např. obličeje), tak i negativní vzory (pozadí). Všechny vzory by měly mít stejnou velikost $w \times h$ (pro zjednodušení budeme dále předpokládat $w = h$) a zároveň by rozměry měly být co nejmenší (samozřejmě však se zachováním dostatečného optického rozlišení sledovaného objektu). Důvodem je značná výpočetní náročnost trénovacího procesu. Tyto vzory jsou následně využity pro natrénování klasifikačního algoritmu AdaBoost, tedy je získána množina

optimálních příznaků a jim odpovídající množina natrénovaných slabých klasifikátorů společně s jejich váhami.



Obrázek 1.6: Příklad skenovacího okna v obrazu

Při detekci není zpracováván vstupní obraz jako celek, ale po částech. Obraz se skenuje posuvným oknem, které mění nejen svou pozici x, y v obrazu, ale také svou velikost $s \times w$ v závislosti na předpokládané velikosti hledaného objektu. Z důvodů rychlosti detektoru se nevyužívají všechny možné velikosti posuvného okna, ale velikost se zvyšuje skokově. Ukázka je na obrázku 1.6, kde x a y jsou počáteční souřadnice daného okna, w je základní velikost okna a s je faktor zvětšení skenovacího okna. Klasifikátor rozhodne, zda toto okno objekt obsahuje, či nikoli.



Obrázek 1.7: Zapojení kaskády klasifikátorů

Skenovacích oken existuje v obrazu velké množství a detekce se tak stává výpočetně náročná (např. pro obraz o rozměrech 320×240 pixelů a počáteční velikosti okna $w = 19$ je nutné zpracovat asi 500 000 oken). Snížení celkové náročnosti detektoru je realizováno snížením průměrné doby, kterou detektor věnuje klasifikaci každého skenovacího okna. Toho je dosaženo tzv. kaskádovým zapojením klasifikátorů (obrázek 1.7). Tato myšlenka je založena na pozorování, které ukázalo, že k vytvoření klasifikátoru, který dokáže rozpoznat správně téměř všechny pozitivní případy a zároveň významnou část (20-50%) negativních případů, stačí jen malé množství příznaků. Druhou důležitou skutečností je fakt, že většina skenovacích oken neobsahuje hledaný objekt. Kaskádové zapojení klasifikátorů se tak snaží v každém stupni vyřadit co nejvíce negativních oken, a do dalšího zpracování pustit jen pozitivní okna. Většina oken se tak vyřadí již v prvních stupních kaskády a není nutné extrahovat a klasifikovat hodnoty příznaků pro další stupně kaskády.

1.3 WaldBoost

V kapitole 1.2.1 byl popsán algoritmus AdaBoost. Ačkoli zapojení kaskády klasifikátorů urychluje výpočet výsledného klasifikátoru, pro pozitivní vzorky se stále vyhodnocují všechny slabé klasifikátory. Tento problém řeší algoritmus WaldBoost [11]. Klasifikátor natrénovaný touto metodou může být vyhodnocen dříve jak pro záporné, tak i pro kladné vzorky. Pokud se během vyhodnocení dosáhne jisté hranice jistoty, je vyhodnocení ukončeno. WaldBoost je založen na algoritmu AdaBoost pro výběr a řazení slabých klasifikátorů a na metodě SPRT (Sequential Probability Ratio Test [12]). Výhoda v podobě urychlení umožňuje využít WaldBoost v real-time aplikacích.

Při klasifikaci vzorku se v každém kroku provede jedno měření (vyhodnocení jednoho slabého klasifikátoru). Sekvenční rozhodovací strategie (sequential probability ratio test, SPRT) definuje dva prahy A a B. Klasifikační strategie S je definována následovně:

$$\begin{aligned} S &= +1 \quad R_t \leq B \\ S &= -1 \quad R_t \geq A \\ S &= 0 \quad B < R_t < A \end{aligned} \tag{3}$$

kde R_t je poměr likelihoodů:

$$R_t = \frac{p(x_1, \ldots, x_t | y = -1)}{p(x_1, \ldots, x_t | y = +1)} \tag{4}$$

Klasifikace je vyhodnocena jako +1 (pozitivní klasifikace), pokud je R_t vyšší nebo rovno B. Jako -1 je S ohodnocena, pokud je R_t nižší nebo rovno B. Pokud nenastane ani jeden z těchto případů (R_t se nachází mezi prahy A a B), je provedeno další měření. Hodnoty prahů se určují dle maximální přípustné míry chyb false negative (pozitivní vzorek klasifikován jako negativní) a false positive (negativní vzorek je klasifikován jako pozitivní) [12], [13].

Vstup: ohodnocené vzorky $(x_1, y_1), \ldots, (x_n, y_n)$ $x \in X, y \in \{-1, 1\}$. Parametry α a β.

Inicializace: Váhy vzorků $w_{1,i} = \frac{1}{n}$, pro $i \in (1, n)$, $A = \frac{1-\beta}{\alpha}$, $N = \frac{\beta}{1-\alpha}$.

Pro $t = 1 \ldots T$

1. Výběr slabého klasifikátoru s nejmenší chybou (viz AdaBoost).
2. Odhad R_t.

$$R_t = \frac{p(x_1, \ldots, x_t | y = -1)}{p(x_1, \ldots, x_t | z = +1)}$$

3. Nalezení prahů $\Theta_A^{(t)}$ a $\Theta_B^{(t)}$.
4. Odstranění vzorků z trénovací množiny, které odpovídají podmínce $H_t \geq \Theta_B^{(t)}$ nebo $H_t \leq \Theta_A^{(t)}$.
5. Nahrazení odstraněných vzorků novými.

Výstup: Klasifikátor H_T a prahové hodnoty $\Theta_A^{(t)}$ a $\Theta_B^{(t)}$.

Algoritmus 1.2: WaldBoost

Trénování klasifikátoru pomocí algoritmu WaldBoost (algoritmus 1.2) vyžaduje dva parametry α (false negative rate) a β (false positive rate), ze kterých se vypočítají prahy $\Theta_A^{(t)}$ a $\Theta_B^{(t)}$. Trénování probíhá v iteracích. Prvním krokem je výběr slabého klasifikátoru, stejně jako u AdaBoostu. Následuje odhad R_t a nalezení prahů $\Theta_A^{(t)}$ a $\Theta_B^{(t)}$. Na konci se odstraní vzorky, u kterých již stávající klasifikátor dokáže rozhodnout, do které třídy patří a jsou nahrazeny novými vzorky.

1.4 Slabé klasifikátory a příznaky

V předcházejících podkapitolách bylo uvedeno, jakým způsobem jsou vybírány slabé klasifikátory k vytvoření silného klasifikátoru. Slabým klasifikátorem může být libovolná funkce, která rozhoduje lépe než funkce náhodná. Tedy v našem případě binárního rozhodování každá funkce s chybou nižší než 0,5. Základem slabých klasifikátorů používaných v této práci jsou obrazové příznaky (features). Rozeznáváme dvě základní skupiny příznaků – spojité a diskrétní. Spojité příznaky mohou vracet jako výsledek libovolnou reálnou hodnotu. Výsledkem diskrétních příznaku jsou celá čísla. Spojité příznaky lze samozřejmě převést na diskrétní. Avšak existují diskrétní příznaky, které nemají spojitou variantu.

WaldBoost nejprve vytvoří množinu slabých klasifikátorů založených na všech možných příznacích a poté z nich vybírá ty s nejmenší chybou na trénovacích datech. Důvodem, proč se používají příznaky místo hodnot pixelů je právě množství takto vytvořených příznaků a také rychlost jejich vyhodnocení.

V kapitole 1.2.2 jsem se věnoval Haarovým vlnkám, které použili Viola a Jones ve svém detektoru. O dalších příznacích používaných k detekci objektů pojednávají následující podkapitoly.

1.4.1 Local Binary Patterns

Algoritmus Local Binary Patterns (dále LBP) [14] je poměrně mladá metoda vyvinutá na Finské universitě v Oulu. Metoda má bohaté využití, zmínit lze například průmyslové nasazení pro rozpoznávání defektů, analýzu biomedicínských snímků a pro svou vysokou efektivitu lze tuto metodu využít i v aplikacích vyžadující zpracování v reálném čase. Kromě nízkých výpočetních nároků má tato metoda další významnou výhodu, a sice že je invariantní vůči monotónním transformacím. Dokáže si tak poradit např. se změnou osvětlení vlivem denní doby nebo se zastíněním scény.

LBP pracuje s monochromatickým obrazem. V základní variantě se využívá hodnot osmi přiléhajících pixelů (obrázek 1.8a). Každý z těchto okolních bodů je porovnán se středem. Je-li hodnota okolního pixelu nižší, zapíše se na příslušné místo nula, v opačném případě pak jednička. (obrázek 1.8b). Použitou funkci lze zapsat následovně:

$$f(x) = \begin{cases} 0 & x > y \\ 1 & x \leq y \end{cases}, \tag{5}$$

, kde x je hodnota středového pixelu a y hodnota okolního pixelu. Následně se tyto koeficienty vynásobí s odpovídajícími hodnotami váhové matice obsahující prvky rozvoje 2^n pro n=<0,7> (obrázek 1.8c). Posledním krokem je sečtení těchto hodnot a výsledkem je hodnota LBP pro daný pixel (obrázek 1.8d).

32	18	25
12	27	50
81	78	42

(a)

1	0	0
0		1
1	1	1

(b)

1	2	4
128		8
64	32	16

(c)

1	0	0
0		8
64	32	16

(d)

Obrázek 1.8: Demonstrace výpočtu LBP. Hodnota LBP je rovna 121.

Vzhledem k principu LBP je zřejmé, že existuje celkem 2^8, tedy 256 možných variant LBP kódu. Ne všechny však mají stejný význam [20]. Některé varianty by měly být robustnější vůči geometrickým transformacím a klasifikovat tak lépe než ostatní varianty. Tím se dostáváme k definici tzv. uniformních LBP (uLBP, [20]). K tomu je však zapotřebí zavést míru nejednotnosti (measure of nonuniformity) U(LBP), která odpovídá počtu přechodů v bitové reprezentaci LBP kódu. Například LBP kódy 0 (v bitové reprezentaci 00000000) a 255 (11111111) mají hodnotu U=0, LBP kódy 1, 2, 4, 8, 16, 32, 64 a 128 (v bitové reprezentaci 00000001, 00000010 apod.) mají hodnotu U=2, protože v těchto kódech existují právě dva přechody 0/1, respektive 1/0. Podobně pro LBP kódy 00000011, 00000111, 00001111, 00011111, 00111111, 01111111 a jejich bitově rotované varianty je hodnota U=2. Pro ostatní varianty je hodnota U nejméně 4.

Autoři článku [20] uvádí, že čím je počet přechodů v LBP kódu nižší (tedy čím je nižší hodnota U), tím je jeho robustnost vůči geometrickým změnám (např. naklápění či rotace) vyšší. Na základě tohoto tvrzení bylo navrženo použít 9 uniformních LBP kódů s hodnotou U maximálně 2 (00000000, 00000001, 00000011, 00000111, 00001111, 00011111, 00111111, 01111111, 11111111). Takovýchto kódu včetně rotovaných variant existuje 58 z celkového počtu 256. Zbývající varianty jsou spojeny do jediného binu, výsledkem je tedy 59 možných variant LBP kódu.

1.4.2 Histogram gradientů

Tuto metodu použili Navneet Dalal a Bill Triggs[15] pro detekci osob v roce 2005. Ve svém článku využili histogram gradientů (Histogram of Gradients, dále HOG) a dosáhli téměř ideálních výsledků (na originální MIT databázi chodců). Principem této metody je detekce orientovaných gradientů ve výřezech obrazu a tvorba histogramu dle jejich orientace.

Při výpočtu se obraz rozdělí na malé, samostatně zpracovávané oblasti nazývané buňky (cells). Prvním krokem je výpočet gradientů. Nejjednodušším způsobem je aplikace jednorozměrné masky v horizontálním i vertikálním směru:

$$D_x = [-1 \quad 0 \quad 1], \; D_y = [-1 \quad 0 \quad 1]^T$$
$$I_x = I * D_x, \; I_y = I * D_y \tag{6}$$

S použitím těchto masek a vstupního obrazu I získáme gradienty s použitím konvoluce dle rovnice 6. Demonstrace tohoto kroku je znázorněna na obrázku 1.9.

Obrázek 1.9: Znázornění vertikálních (uprostřed) a horizontálních (vpravo) gradientů

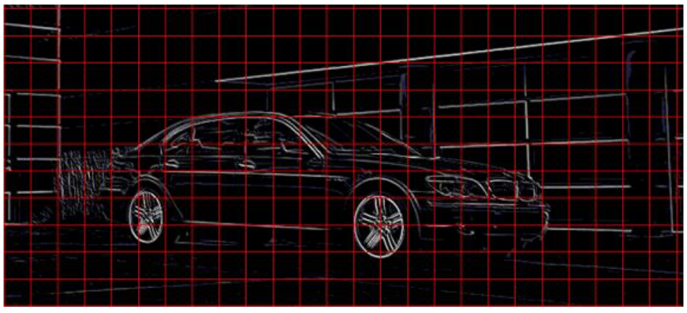
Druhým krokem výpočtu je vytvoření histogramu. Každý pixel v buňce přidá hodnotu velikosti gradientu do patřičného binu dle své orientace (rovnice 7). Histogram je rovnoměrně rozdělen přes 180 nebo 360 stupňů dle toho, jestli se jedná o bezznaménkový (unsigned, 0..180°) nebo znaménkový (signed, 0..360°) gradient. Rozdíl mezi bezznaménkovým a znaménkovým gradientem spočívá v tom, zda uvažujeme i orientaci gradientu. Pokud ignorujeme „znaménko" gradientu, jedná se bezznaménkový gradient. Dalal a Triggs ve své práci uvádějí jako nejlepší variantu použití bezznaménkových gradientů ve spojení s devíti biny (tedy rozdělení histogramu po 20 stupních).

$$\begin{aligned} Velikost\ gradientu\ |G| &= \sqrt{I_x^2 + I_Y^2} \\ Orientace\ gradientu\ \theta &= \arctan\frac{I_y}{I_x} \end{aligned} \tag{7}$$

Na obrázku 1.10 jsou znázorněny všechny výše uvedené kroky. Vstupem je obraz ve stupních šedi (a). Pro každý jeho pixel je spočítán gradient. Pro ilustraci je z těchto gradientů vytvořen obrázek (b). Následujícím krokem je rozdělení obrazu do bloků (c) a vytvoření histogramu pro každý blok.

(a)

(b)



(c)

Obrázek 1.10: Ilustrace kroků pro výpočet histogramu gradientů

Důležitou součástí výpočtu je normalizace bloků. Dalal a Triggs ve svém článku [15] předvedli několik metod normalizace. Před jejich uvedením však pro úplnost uvedu definici normy vektoru. L1 norma vektoru v je definována následovně:

$$|v|_1 = \sum_{i=1}^{N} |v_i|,$$

kde N je počet složek vektoru v. Podobně je definována L2 norma:

$$|v|_2 = \sqrt{\sum_{i=1}^{N} |v_i|^2}$$

Na základě těchto norem a použití ϵ jako malé konstanty lze uvést 4 metody normalizace bloků [15]:

- L2-norm, $v \rightarrow v/\sqrt{|v|_2^2 + \epsilon^2}$
- L2-Hys, L2-norm následována omezením maximální hodnoty v na 0,2 a další normalizací
- L1-norm, $v \rightarrow v/(|v|_1 + \epsilon)$
- L1-sqrt, $v \rightarrow \sqrt{v/(|v|_1 + \epsilon)}$

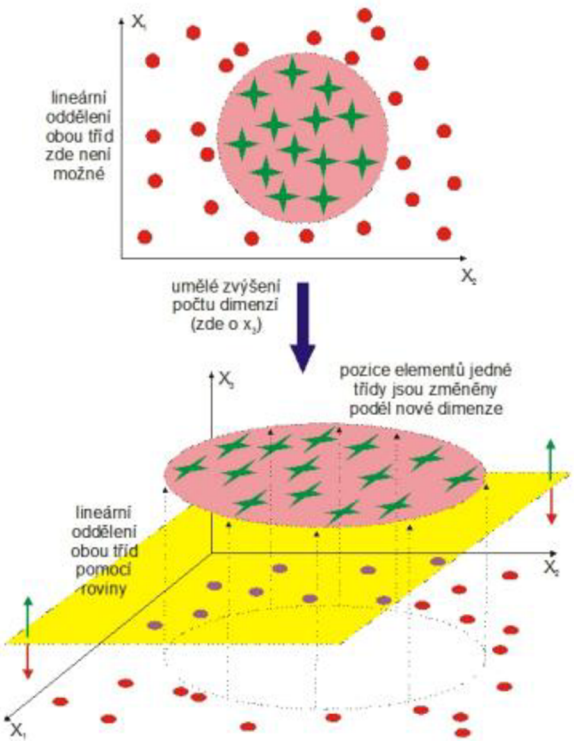
Dle [15] vykazují normalizace L2-norm, L2-Hys a L1-sqrt stejné výsledky, normalizace L1-norm má výsledky horší. Úplné vynechání normalizace má pak významný negativní dopad na výsledky histogramu gradientů. Všechny experimenty v uvedeném článku byly provedeny na datových sadách chodců.

1.4.3 Použití SVM ve spojení s HOG

Jak bylo uvedeno v předchozí kapitole, pro každý blok se vygeneruje histogram gradientů. Otázkou je, jak z tohoto histogramu získat jednu návratovou hodnotu příznaku. Jednou možností je vracet každý bin histogramu jako samostatný příznak, ztrácíme tím však celistvou informaci o bloku. Druhou variantou je použití Support Vector Machines (SVM) [16].

SVM patří mezi tzv. kernel-based metody strojového učení. Tyto metody staví na výhodách efektivních algoritmů pro nalezení lineární hranice a zároveň jsou schopné reprezentovat složité nelineární funkce. Základním principem SVM je převod původního vstupního prostoru do jiného, vícedimensionálního, ve kterém je již možno oddělit třídy lineárně. Tato myšlenka je znázorněna na obrázku 1.11. V původním vstupním prostoru jsou dvě třídy oddělené nelineárně kružnicí. Přidáním další dimenze získají prvky třídy v kružnici třetí souřadnici, která je posune rozdílným směrem od prvků druhé třídy. Nyní lze pro oddělení obou tříd použít rovinu rovnoběžnou s osami x_1 a x_2.

S použitím SVM lze na trénovacích datech natrénovat množinu modelů SVM, kde každý model bude trénován na jisté velikosti a pozici histogramu gradientů. Následně se každý histogram gradientů předá příslušnému modelu SVM jako vektor příznaků (o délce odpovídající počtu binů v histogramu) a výsledkem příznaku je predikce tohoto modelu SVM. Příznaky založené na spojení HOG a SVM jsou dále uváděny pod zkratkou SVMHOG.



Obrázek 1.11: Přidání dimenze umožní lineární separaci dat

2 Navržená rozšíření

V této kapitole seznámím čtenáře s navrženými rozšířeními pro stávající implementaci systému pro detekci objektů na UPGM FIT. Výsledky těchto rozšíření ve formě provedených experimentů jsou obsahem navazující kapitoly. Představeny budou celkem dvě skupiny návrhů, první souvisí se snahou rozšířit datové sady určené pro trénování klasifikátoru pro detekci obličejů algoritmem WaldBoost (viz kapitola 1.3). Druhou skupinou je implementace nových prvků do systému pro detekci objektů se snahou vylepšit jeho výsledky.

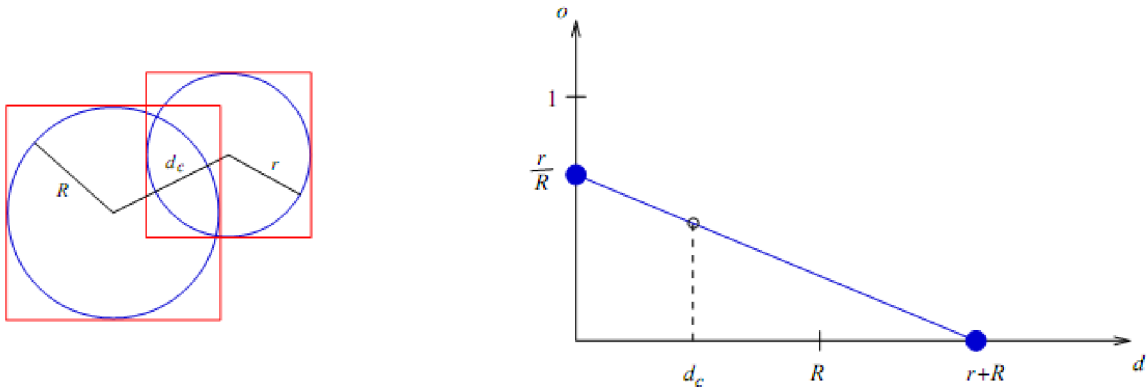
2.1 Náhodné transformace

Pokud detekujeme objekty klasifikátorem pomocí skenovacího okna, rozdíl mezi sousedícími okny v pozici a ve velikosti není nekonečně malý. Z tohoto důvodu je nezbytné, aby byl klasifikátor vůči těmto malým změnám odolný. Dalším důvodem je fakt, že manuální anotace trénovacích dat není ideálně přesná. Na základě těchto fakt se lze domnívat, že je možné aplikovat menší geometrické transformace na anotovaných trénovacích datech, aniž by se snížila výkonnost výsledného klasifikátoru. Na druhé straně mohou tyto transformace produkovat lepší aproximaci skutečného rozložení třídy objektů.

V našem případě jsme pro každý vzorek z anotované datové sady vytvořili více vzorků aplikováním malých změn ve velikosti a pozici. Tyto transformace však musejí mít nějaké omezení a tím je požadavek na minimální překryv. Ke spočítání relativního překryvu s originální anotací jsou použity vepsané kružnice (obrázek 2.1 vlevo). Překryv p je spočítán dle rovnice[17]:

$$p = \frac{r}{R}\left(1 - \frac{d_c}{r+R}\right) \tag{7}$$

kde r (R) je poloměr menší (větší) kružnice a d_c je vzdálenost mezi středy kružnic. Rovnice 7 je aproximace skutečného překryvu bez užití výpočetně náročných goniometrických funkcí. Takto spočítaný překryv je lineární interpolací mezi dvěmi krajními situacemi. První případ nastane, pokud jsou středy obou kružnic shodné. Pak je překryv roven r/R. Pokud mají obě kružnice pouze jeden společný bod, pak se jedná o druhou krajní situaci a překryv je roven 0 (obrázek 2.1 vpravo).



Obrázek 2.1: Ilustrace výpočtu překryvu dvou vzorků (převzato z [17])

2.2 Získání dat z webové galerie

Cílem tohoto pokusu bylo získat velké množství pozitivních vzorů a tím rozšířit trénovací datovou sadu a následně ověřit, zda trénovaní na této obsáhlé trénovací datové sadě vylepší výsledný klasifikátor. Pro získání těchto dat byla vybrána webová aplikace Flickr určená ke sdílení fotek nacházející se na adrese http://flickr.com/. Hlavní důvody vedoucí k výběru aplikace Flickr jsou dva. Prvním je existence tematických skupin (např. Portraiture) čítajících až desetitisíce fotek obsahujících převážně to, co je předmětem dané skupiny (tedy např. námi požadované obličeje). Odpadá tak nutnost ručního výběru vhodných fotek. Ještě důležitější je ovšem fakt, že Flickr nabízí volně k použití knihovnu umožňující operace nad těmito skupinami a obrázky. Pro náš účel je pak zejména důležité, že umožňuje stahovat zadané fotky, ale i celé skupiny.

Obrázek 2.2: Ukázka použitých obrázků z webové galerie Flickr

Výše uvedené vlastnosti umožňují vytvoření programu, schopného automaticky stáhnout všechny obrázky ze zadané skupiny (obrázek 2.2). Program byl implementován v jazyce C a následně použit ke stáhnutí skupiny s názvem Portraiture, která v době stahování obsahovala více než 150 000 obrázků. Z takto získaných dat bylo nutné vytvořit datasety, vhodné k trénování. K jejich tvorbě bylo nutné učinit dva kroky. Prvním krokem je nalezení obličejů a jejich pozicí v získaných obrázcích, tedy vytvoření anotace. K tomu posloužil již existující klasifikátor, který vrátil hodnotu odezvy pro každé skenovací okno v každém z obrázků. Na základě tří určených prahů byly vytvořeny tři anotace, kdy byly jako pozice s obličejem brány okna s odezvou detektoru vyšší, než daný práh. S nižším prahem se vytvořilo více pozitivních vzorů, ale s menší jistotou správnosti a naopak, s vyšším prahem vzniklo méně, ale jistějších vzorů. Druhým krokem je vytvoření datových sad, kdy byly všechny obrázky rozděleny do setů o maximálně 5 000 obrázcích. Navrhnuté experimenty a jejich výsledky jsou obsahem následující kapitoly.

2.3 Existující databáze

Motivací pro tento experiment je opět rozšíření trénovací datové sady, liší se však provedením. Nová data jsou získána z již existujících datových sad BioID, PAL a xm2vts (tabulka 2.1, obrázek 2.3). Na základě těchto dat a existující anotace byly vytvořeny nové vzorky s normalizovanou pozicí obličeje. Celková velikost nově vzniklých obrázků v datové sadě je 150×150 pixelů, velikost samotného obličeje pak 100×100 pixelů. Okraje kolem obličeje jsou ponechány pro případné transformace, pokud ve zdrojovém obrázku chybí, jsou vyplněny černou barvou.

Obrázek 2.3: Ukázka obrázků z databáze BioID (vlevo), PAL (uprostřed) a xm2vts (vpravo)

Název datové sady	Počet obrázků	Webová adresa
BioID	1 521	http://www.bioid.com/downloads/facedb/index.php
PAL	777	https://pal.utdallas.edu/facedb/
Xm2vts	11 898	http://www.ee.surrey.ac.uk/CVSSP/xm2vtsdb/

Tabulka 2.1: Seznam použitých datových sad

2.4 Přetrénování SVM

Při trénování klasifikátorů pomocí AdaBoostu se váhy vzorků mezi jednotlivými iteracemi mění jen částečně a v pozdějších fázích už jen minimálně. Proto lze předpokládat, že není nutné přetrénovat SVM v každé iteraci a lze tak snížit výpočetní náročnost trénování silného klasifikátoru bez zhoršení jeho vlastností.

Pro ověření tohoto předpokladu byly navrženy 3 způsoby přetrénování. Prvním způsobem je natrénování SVM jen v prvním kole a tyto modely použít ve všech následujících kolech. Tento způsob jsem zvolil spíše pro porovnání s ostatními metodami jako extrémní případ. Druhým způsobem je trénování daného zlomku z celkového počtu SVM v každém kole. A konečně třetím způsobem je trénovaní zadaného klesajícího množství SVM modelů. Parametrem $fraction$ se zadá požadované množství, které každé kolo klesá. Pokud je parametr roven 1, trénují se všechny SVM každé kolo. Pokud je $fraction$ menší než 1, je v kole i přetrénováno množství m dle rovnice:

$$m_i = fraction^i \tag{8}$$

Výsledkem je požadovaná část k přetrénování, pro získání absolutního počtu SVM modelů k přetrénování stačí touto hodnotou vynásobit celkový počet SVM modelů.

3 Výsledky experimentů

Tato kapitola shrnuje dosažené výsledky z navržených rozšíření uvedených v kapitole 2. Každá podkapitola obsahuje popis provedeného experimentu a výsledky porovnané s dříve dosaženými výsledky. V experimentech byly využity testovací datové sady MIT+CMU, Matej Smid, INRIA a datová sada s dopravními značkami. Základní informace o těchto sadách lze nalézt v tabulce 3.1.

Při trénování bylo využito algoritmu WaldBoost a příznaků HAAR, LBP, HOG a SVMHOG (viz kapitola 1.2.2 a 1.4). Podrobnější informace o nastavení trénovacího procesu jsou uvedeny u každého experimentu. Ve všech případech, kdy bylo použito normalizace L2-norm bylo ϵ rovno 2.

Dataset	# obrázků	# objektů	# skenovaných pozic
MIT+CMU	113	491	19M
Matej Smid	89	1618	160M
INRIA	1580	1126	2M
Dopravní značky	839	481	27M

Tabulka 3.1: Přehled použitých testovacích sad

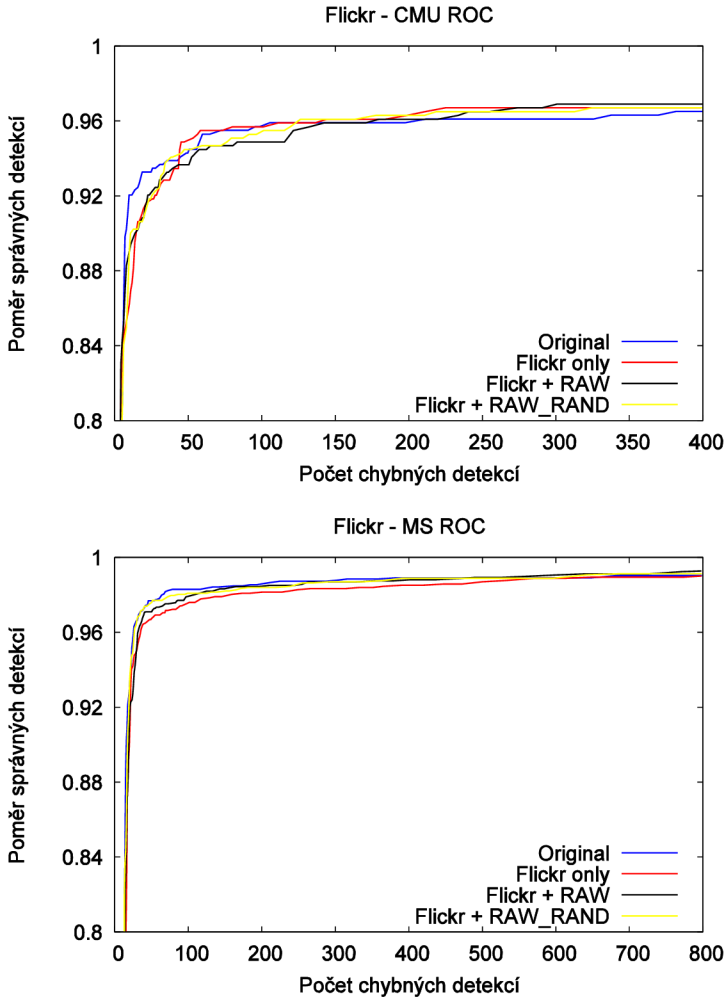
3.1 Získání dat z webové galerie

Pro ověření, zda rozšíření trénovací datové sady automaticky anotovanými daty pomáhá zlepšit kvalitu výsledného klasifikátoru, bylo zvoleno 9 experimentů. Jak bylo zmíněno v kapitole 2.2, byly vytvořeny tři skupiny dle jistoty detekce lišící se v anotaci. Další tři skupiny byly vytvořeny vzájemnou kombinací s již existujícími datovými sadami. Celkový přehled experimentů poskytuje tabulka 3.2. Sloupec Min detekce udává minimální hodnotu detekce, která byla považována za dostatečnou pro označení daného okna za pozitivní vzor (obličej). Ve třetím sloupci je uvedeno, na kterých datech proběhlo trénování a poslední sloupec informuje o celkovém počtu vygenerovaných vzorků. Údaj v závorce u trénovacích dat informuje, kolik vzorků bylo vytvořeno z jednoho pozitivního vzoru s použitím náhodných transformací.

Experiment #	Min detekce	Použitá trénovací data	Celkový počet vzorků
1	25	Pouze nová data (3 vzorky / GT)	200 000
2	25	Nová data + RAW (2 vzorky / GT)	184 155
3	25	Nová data + RAW_RAND (1 vzorek / GT)	200 000
4	100	Pouze nová data (3 vzorky / GT)	200 000
5	100	Nová data + RAW (2 vzorky / GT)	156 131
6	100	Nová data + RAW_RAND (1 vzorek / GT)	200 000
7	150	Pouze nová data (3 vzorky / GT)	200 000
8	150	Nová data + RAW (2 vzorky / GT)	146 322
9	150	Nová data + RAW_RAND (1 vzorek / GT)	200 000

Tabulka 3.2: Přehled provedených experimentů

Klasifikátory byly otestovány na datových sadách MIT+CMU a Matej Smid. Krok v pozici během skenování byl nastaven na 2/24 skenovacího okna a krok ve velikosti na 1,2. Přehled o použitých testovacích datových sadách podává tabulka 3.1. Všechny klasifikátory byly trénovány s použitím LBP příznaků. Jak se v experimentech ukázalo, výsledky jsou téměř totožné nezávisle na zvoleném prahu pro automatickou anotaci. Vybral jsem tedy dva grafy zobrazující klasifikátory (v plné délce 1000 slabých klasifikátorů) natrénované na odlišných datech, viz tabulka 3.2. Graf 3.1 zobrazuje výsledky na datových sadách CMU+MIT a Matej Smid.



Graf 3.1: Výsledky klasifikátorů s minimální hodnotou odezvy pro automatickou anotaci 150 na testovací datové sadě Matej Smid (nahoře) a MIT+CMU (dole)

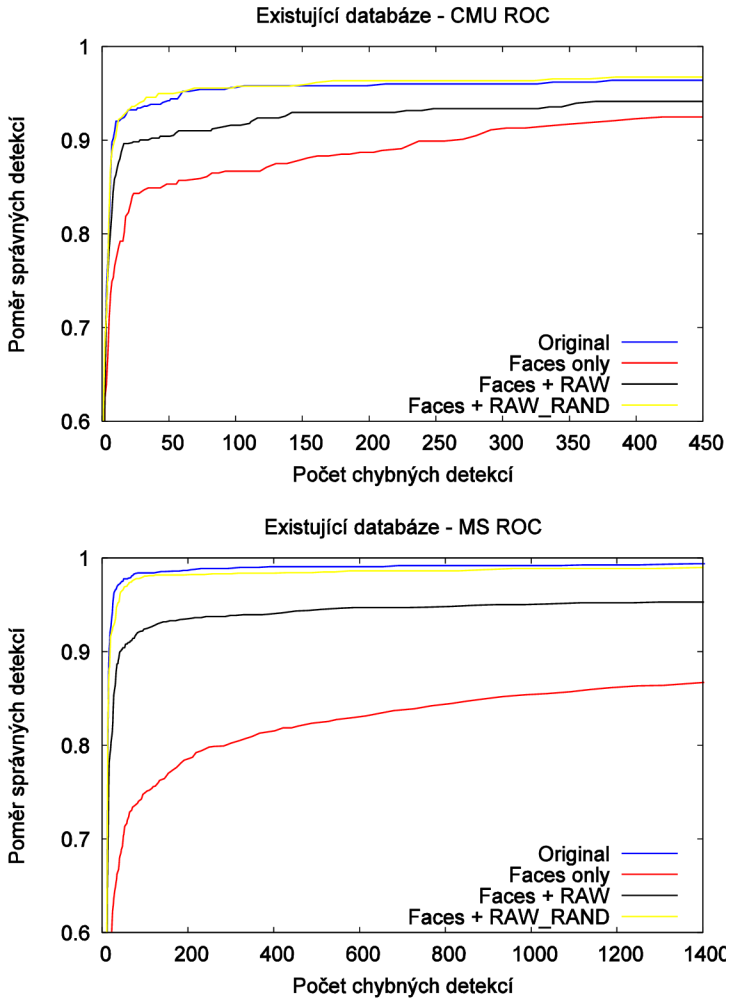
3.2 Existující databáze

Ke zjištění, zda postup uvedený v kapitole 2.3 přináší vylepšení, byly provedeny 3 experimenty s různými kombinacemi datových sad. Trénovací sada vzorků byla rozšířena o data ze zdrojů BioID, PAL a xm2vts. Přehled o provedených experimentech poskytuje tabulka 3.3. Údaj v závorce značí, kolik vzorků se vygenerovalo z jednoho pozitivního vzoru s použitím náhodných transformací.

Experiment #	Použitá trénovací data	Celkový počet vzorků
1	Pouze nová data (45 vzorky / GT)	200 000
2	Nová data + RAW (41 vzorky / GT)	200 000
3	Nová data + RAW_RAND (37 vzorek / GT)	200 000

Tabulka 3.3: Přehled provedených experimentů

Klasifikátory byly otestovány na datasetech MIT+CMU a Matej Smid. Krok v pozici během skenování byl nastaven na 2/24 skenovacího okna a krok ve velikosti na 1,2. Přehled o použitých testovacích datových sadách podává tabulka 3.1. Všechny klasifikátory byly trénovány s použitím LBP příznaků. Graf 3.2 shrnuje výsledky trénování.



Graf 3.2: Výsledky klasifikátorů trénovaných na nových datech z existujících databází na testovacích datových sadách Matej Smid (nahoře) a MIT+CMU (dole)

3.3 Normalizace

Tato podkapitola předkládá výsledky trénování s použitím histogramu gradientů. Cílem je zjistit, který způsob normalizace vykazuje nejlepší výsledky. V experimentech byly vyzkoušeny dva typy normalizace a také použití histogramu gradientů bez normalizace. Prvním typem normalizace je

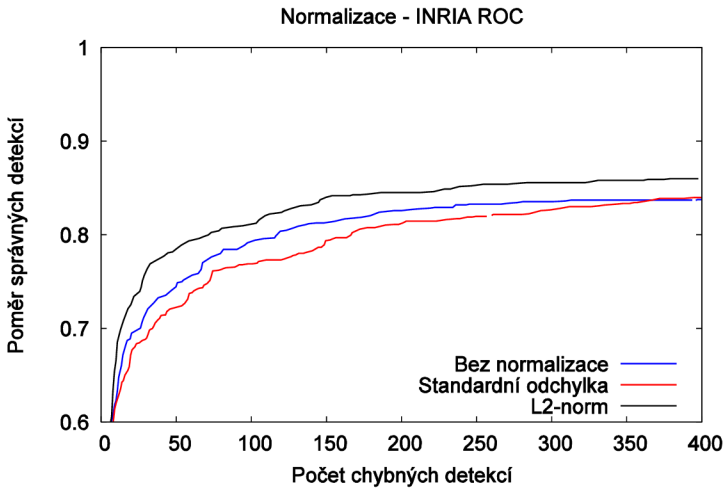
použití standardní odchylky. Standardní odchylka je použita u všech ostatních příznaků jako výchozí metoda, zde slouží pro srovnání. Považujme pixely vzorku za jednorozměrné pole, pak standardní odchylka $stddev$ je spočítána následovně:

$$stddev = \sqrt{\frac{\sum_{i=1}^{n} I(i)^2}{n} - \left(\frac{\sum_{i=1}^{n} I(i)}{n}\right)^2} \tag{9}$$

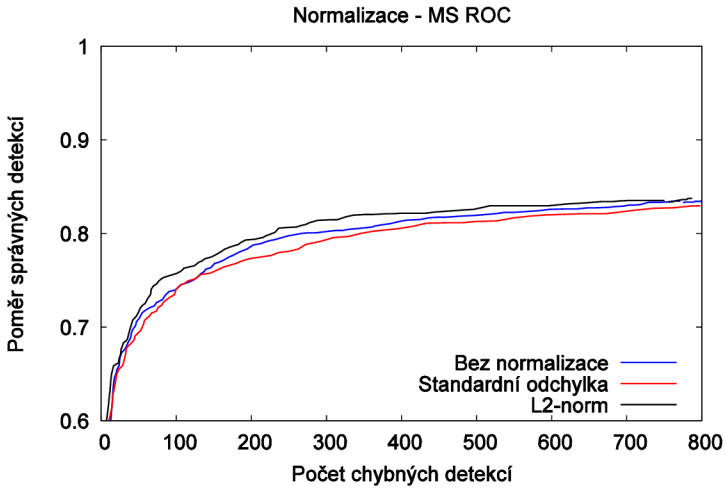
kde I je pole pixelů daného vzorku o délce n. Druhým typem normalizace je L2-norm, uvedená v kapitole 1.4.2. Experimenty byly spuštěny na třech datových sadách a to na obličejích, na chodcích a také na dopravních značkách.

Pro trénování na obličejích byly použity vzorky o velikosti 48x48 pixelů. Histogramy gradientů se generovaly od velikosti 8x8 pixelů do maximální velikosti 16x16 pixelů s krokem 8 pixelů. Krok v pozici byl 4 pixely v obou směrech. V případě chodců bylo nastavení obdobné, jen vzhledem k tvaru detekovaného objektu byly generovány vzorky o velikosti 64x128 pixelů. Ostatní parametry jsou shodné s nastavením pro obličeje. A konečně nastavení pro dopravní značky bylo totožné s nastavením pro obličeje.

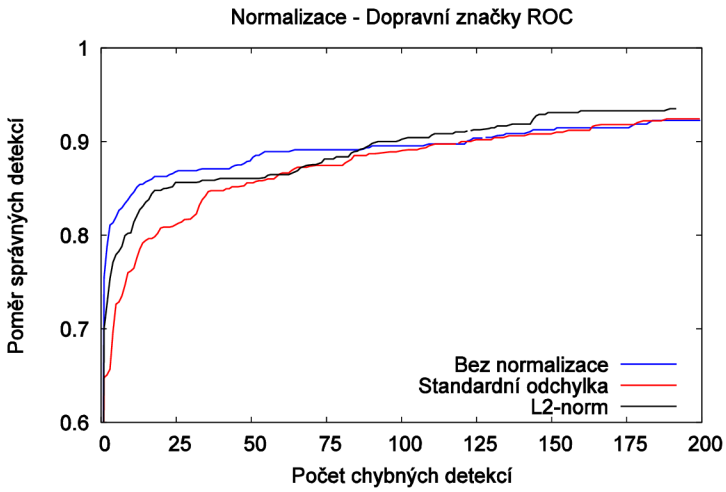
Výsledky uvedených experimentů se liší dle použité datové sady. Z grafu zobrazujícího výsledky na datové sadě chodců (graf 3.3a) je patrné, že nejlepší výsledky byly dosáhnuty s použitím normalizace L2-norm. Naopak, použití standardní odchylky se jeví jako nevhodné, výsledky jsou horší než bez použití normalizace. Při použití datové sady obličejů (graf 3.3b) se potvrzují výsledky dosáhnuté na chodcích. I zde lze vidět zlepšení při použití normalizace L2-norm, nicméně rozdíly nejsou tak patrné. Použití standardní odchylky v porovnání s vynecháním normalizace prakticky nemá vliv na výsledný klasifikátor. Situace se však mění u dopravních značek (graf 3.3c). Obě metody normalizace výsledky zhoršují a nejlepší je tak klasifikátor trénovaný bez použití normalizace. Ze srovnání dvou metod pro normalizaci vychází opět lépe L2-norm. Tyto závěry však platí jen pro nízký počet chybných detekcí. U vyššího počtu chybných detekcí se situace srovnává.



(a)



(b)



(c)

Graf 3.3: ROC křivka zobrazující výsledky normalizace na datové sadě chodců (a), obličejů (b) a dopravních značek (c)

3.4 Přetrénování SVM

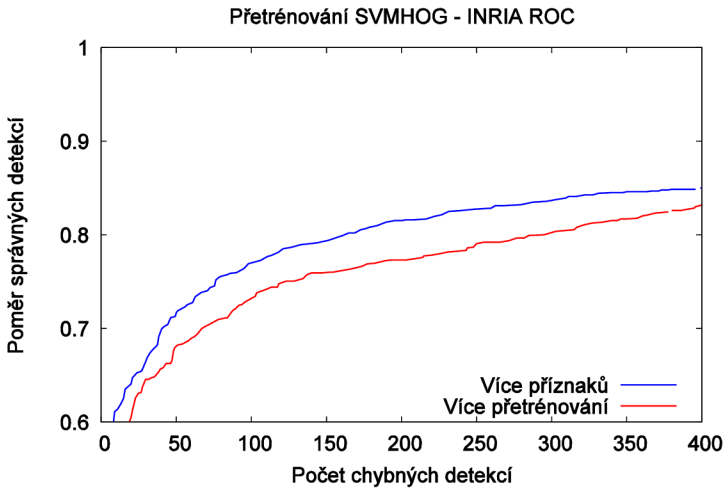
V této kapitole se pokusím vyhodnotit, zda je přínosnější využít SVMHOG (histogram gradientů ve spojení s SVM) s větším počtem příznaků nebo s menším počtem, ale s větší mírou přetrénování. Absolutní počet přetrénovaných SVM byl v každé dvojici experimentů nastaven přibližně stejně, relativní míra přetrénování se však liší vlivem různého počtu příznaků. Tato skutečnost je vyjádřena v tabulce 3.4 uvedením procenta v závorce. Budou tak porovnávány dva experimenty na všech třech datových sadách. Přehled o provedených experimentech je v tabulce 3.4. Velikosti generovaných vzorků byly 64x128 v datové sadě chodců a 48x48 u ostatních datových sad. Zbylá nastavení jsou pro všechny experimenty shodná. Pro normalizaci byla zvolena metoda L2-norm.

Experiment #	Použitá datová sada	Množství přetrénovaných SVM v každém kole	Použité velikosti příznaků	Celkový počet příznaků
1	Chodci	23 (5%)	8x8, 8x16, 16x8, 16x16	465
2	Chodci	26 (25%)	16x16	105
3	Obličeje	6 (5%)	8x8, 8x16, 16x8, 16x16	121
4	Obličeje	6 (25%)	16x16	25
5	Dopr. značky	6 (5%)	8x8, 8x16, 16x8, 16x16	121
6	Dopr. značky	6 (25%)	16x16	25

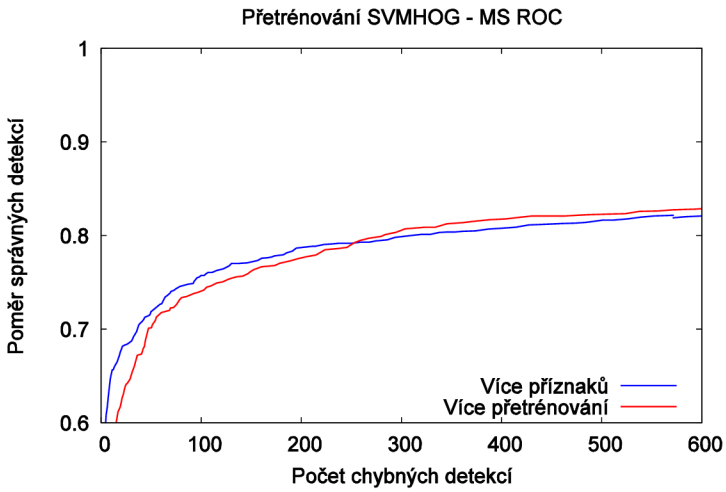
Tabulka 3.4: Přehled experimentů porovnávajících vliv přetrénování SVM a počtu příznaků

Výsledky experimentů 1 a 2 (dle tabulky 3.4) na datové sadě chodců jsou zobrazeny v grafu 3.4a. Z uvedeného grafu je patrné, že lepší variantou je použití více příznaků na úkor přetrénování. Pro trénování klasifikátoru na datové sadě chodců je tedy vhodnější nastavit větší množství příznaků s menším množstvím přetrénování SVM v každém kole.

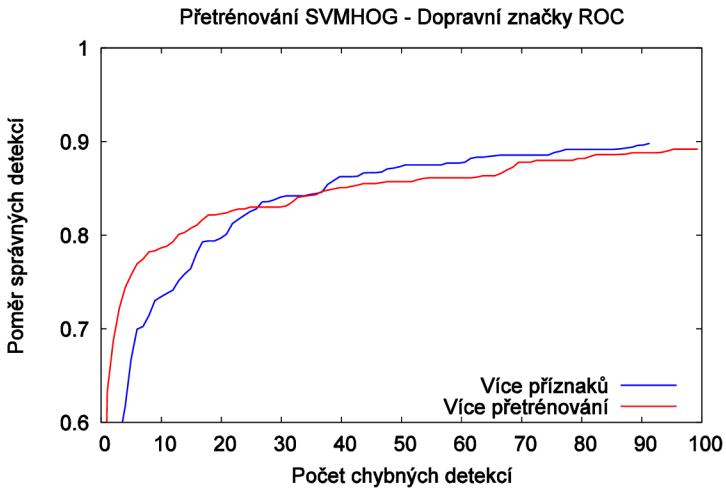
Na datových sadách obličejů (experimenty 3 a 4 dle tabulky 3.4) a dopravních značek (experimenty 5 a 6 dle tabulky 3.4) nelze učinit jednoznačný závěr (graf 3.3b a 3.3c). U dopravních značek lze vidět jistý přínos přetrénování u nízkého počtu chybných detekcí, dále se však již rozdíly smazávají. Z tohoto faktu lze učinit dva závěry. Buď nevýhodu v podobě nižšího počtu příznaků smazává vyšší míra přetrénování, nebo vyšší počet příznaků nemá na těchto datech významný vliv na výsledný klasifikátor.



(a)



(b)



(c)

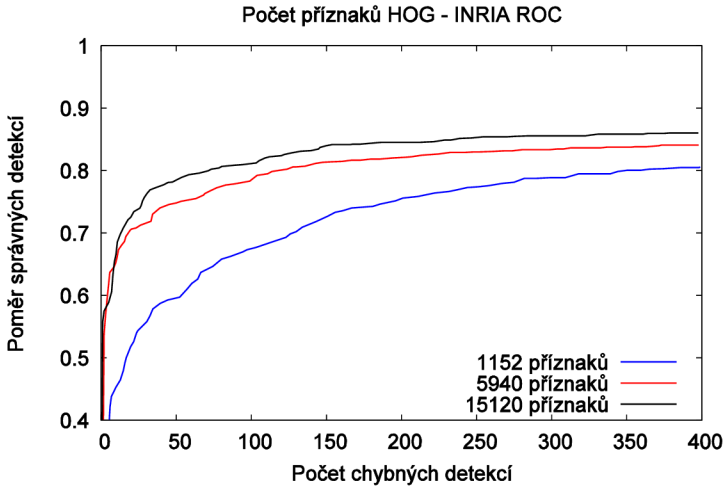
Graf 3.4: ROC křivka zobrazující výsledky přetrénování SVMHOG na datové sadě chodců (a), obličejů (b) a dopravních značek (c)

3.5 Počet příznaků HOG

Tato kapitola se pokusí potvrdit či vyvrátit nás předpoklad, že se zvyšujícím počtem příznaků se zlepšuje výsledný klasifikátor. Z časových důvodů byly provedeny experimenty jen s příznaky HOG, tedy bez spojení s SVM a pouze na datové sadě chodců. Vzorky byly generovány ve velikosti 64x128. Odlišnosti mezi provedenými experimenty jsou obsahem tabulky 3.5. Ve všech experimentech bylo použito histogramu gradientů s 9 biny, tedy celkový počet příznaků je devíti násobkem počtu generovaných velikostí příznaků. Pro normalizaci byla zvolena metoda L2-norm.

Experiment #	Použité velikosti příznaků	Krok v pozici	Celkový počet příznaků
1	8x8	8	1152
2	8x8, 8x16, 16x8, 16x16, 8x32, 16x32	8	5940
3	8x8, 8x16, 16x8, 16x16	4	15120

Tabulka 3.5: Přehled experimentů s počtem příznaků HOG

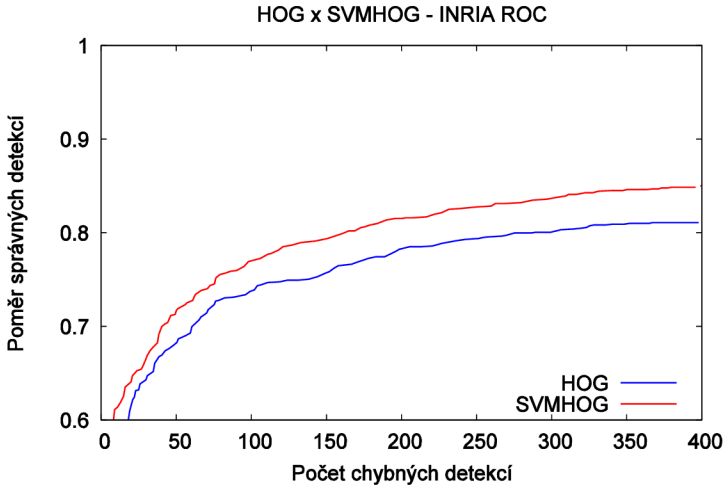


Graf 3.5: ROC křivka s výsledky trénování s různým počtem příznaků HOG na datové sadě chodců

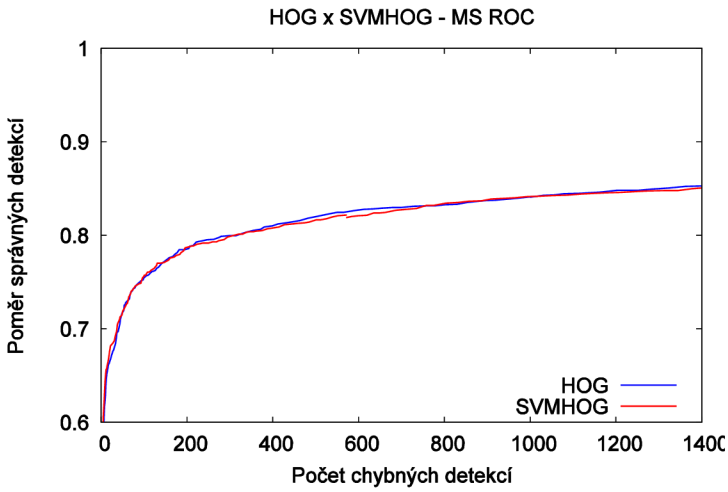
Z grafu 3.5 je patrné, že náš předpoklad byl správný. Klasifikátor trénovaný pouze s příznaky HOG o velikosti 8x8 pixelů je významně horší než klasifikátory trénované na větším počtu příznaků. Přestože klasifikátor trénovaný s největším počtem příznaků neobsahoval příznaky o velikosti 8x32 a 16x32, díky menšímu kroku v pozici (a tedy i většímu překryvu) dosáhl nejlepších výsledků. Avšak rozdíl není výrazný, přestože počet příznaků je 2,5x vyšší.

3.6 Srovnání HOG a SVMHOG

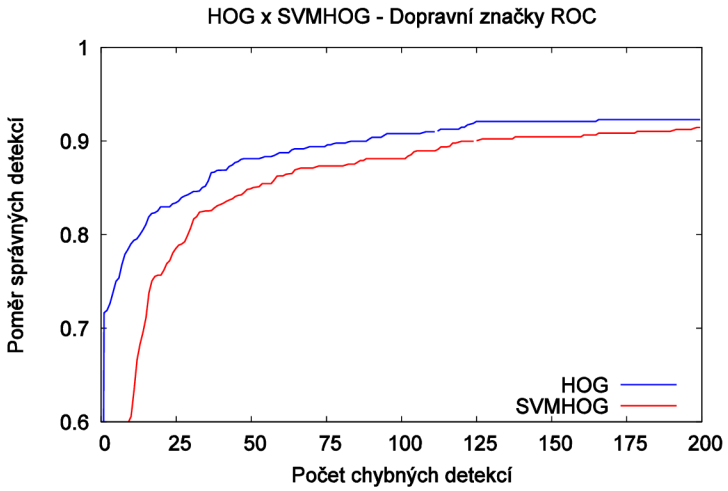
Cílem experimentů uvedených v této kapitole je ověřit přínos SVM ve spojení s histogramem gradientů. Pro tento účel byly použity všechny tři datové sady a na každé z nic byly provedeny dva experimenty lišící se pouze v použití SVM. Použitou normalizací je L2-norm, vzorky byly generovány ve velikostech 64x128 v případě obličejů a 48x48 u ostatních datových sad. Ve všech případech bylo použito normalizace L2-norm a v případě SVMHOG se přetrénovalo 5% SVM každé kolo.



(a)



(b)



(c)

Graf 3.6: ROC křivka zobrazující výsledky trénování s použitím HOG a SVMHOG na datové sadě chodců (a), obličejů (b) a dopravních značek (c)

Pro trénování na datové sadě chodců je přidání SVM přínosem (graf 3.6a). Zlepšení výsledků je znatelné. Jiná situace nastává u datové sady obličejů (graf 3.6b), kde jsou výsledky prakticky totožné a přidání SVM tak výsledky ani nezlepšuje, ale ani nezhoršuje. A opět jiná situace nastává při trénování na datové sadě dopravních značek (graf 3.6c). Z tohoto grafu se jeví spojení SVM s histogramem gradientů pro použití s dopravními značkami jako nevhodné.

3.7 Srovnání s ostatními příznaky

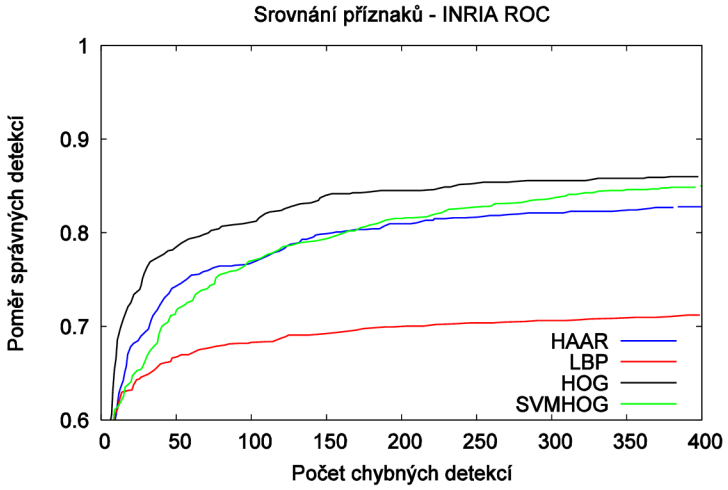
Obsahem této kapitoly je kompletní srovnání Haarových, LBP, HOG a SVMHOG příznaků. Experimenty byly provedeny na všech datových sadách. Je třeba uvést, že z časových důvodů nebyly provedeny všechny experimenty s SVMHOG příznaky, ale jen několik vybraných. Proto při výběru nejlepších výsledků do grafů je výběr u HOG příznaků širší než u SVMHOG příznaků. Parametry těchto příznaků nejsou stejné a není vhodné tak porovnávat HOG a SVMHOG příznaky mezi sebou, k tomu slouží předešlé kapitoly. V této kapitole jde o srovnání nejlepších dostupných výsledků HOG

a SVMHOG příznaku s Haarovými příznaky a LBP. Ve všech experimentech s HOG a SVMHOG byla použita normalizace L2-norm.

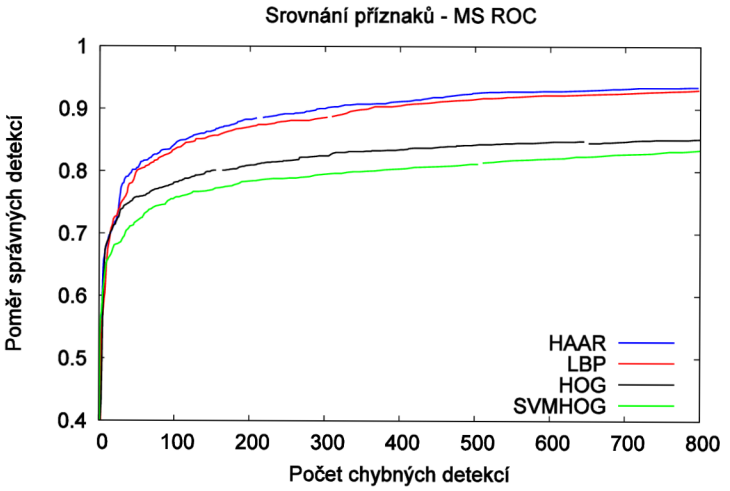
Graf 3.7a zobrazuje výsledky příznaků na datové sadě chodců s velikostí vzorků 64x128. Haarovy příznaky byly generovány ve velikostech od 1x1 do 24x48 pixelů s krokem 2 pixely. Krok v pozici byl nastaven také na 2 pixely. Bylo použito všech základních typů, tedy hranových, čárových i diagonálních haarových vlnek. Pro LBP bylo nastavení velikostí od 1x1 do 64x128 pixelů s krokem 2 pixely, stejně jako pro krok v pozici. Příznaky HOG byly generovány ve velikostech 8x8, 8x16, 16x8 a 16x16 pixelů s krokem v pozici 4 pixely. A konečně u příznaků SVMHOG byly velikosti shodné s příznaky HOG, ale krok byl nastaven na 8 pixelů. Přetrénováno bylo v každém kole 5% SVM. Pro trénování na chodcích se jeví jako nejlepší využít příznaky HOG, které si zde vedou nejlépe. SVMHOG a Haarovy příznaky si vedou obdobně, s odstupem zde nejhůře dopadly LBP příznaky.

Následuje srovnání na datové sadě obličejů. Pro příznaky HOG a SVMHOG byly generovány vzorky o velikosti 48x48 pixelů, pro haarovy příznaky a LBP se ukázaly jako dostačující vzorky o velikosti 24x24 pixelů. Haarovy příznaky byly generovány ve všech možných velikostech na všech možných pozicích. Rovněž byly použity všechny základní typy haarových vlnek. LBP příznaky byly generovány od velikosti 1x1 do velikosti 9x9 s krokem 1 pixel, rovněž i 1 pixel byl nastaven pro krok v pozici. Velikosti HOG příznaků byly 8x8, 8x16, 16x8 a 16x16 pixelů s krokem 4 pixely v pozici. A konečně u příznaků SVMHOG byly velikosti shodné s příznaky HOG, ale krok byl nastaven na 8 pixelů. Přetrénováno bylo v každém kole 5% SVM. Výsledky v grafu 3.7b ukazují, že pro obličeje se příliš histogram gradientů nehodí. Překonávají je jak haarovy příznaky, tak i LBP.

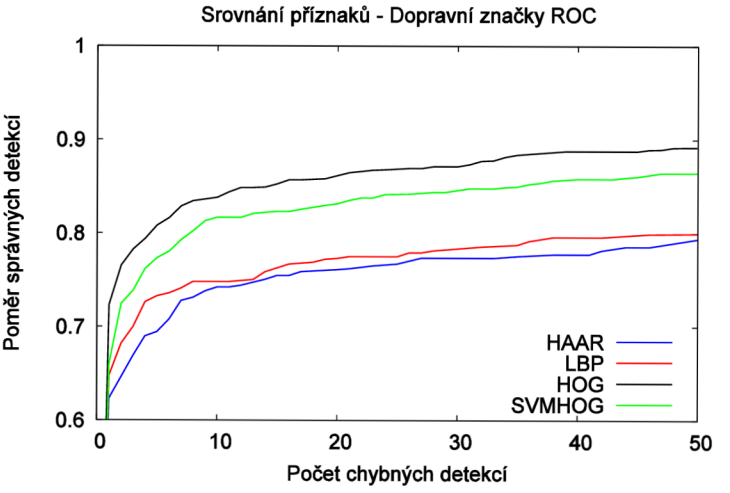
Poslední datovou sadou jsou dopravní značky. Veškeré nastavení se shoduje s nastavením pro datovou sadu obličejů s jediným rozdílem, a tím jsou příznaky SVMHOG. Pro ty bylo vybráno nastavení s velikostí 16x16 pixelů a krokem v pozici 8 pixelů. Přetrénováno bylo v každém kole 25% SVM. Graf 3.7c ukazuje, že pro použití s dopravními značkami jsou histogramy gradientů dobrou volbou. Obě varianty jsou lepší než haarovy příznaky i LBP s poměrně významným rozdílem. Mezi haarovými příznaky a LBP je zanedbatelný rozdíl, ze dvou variant HOG je lepší ta bez použití SVM.



(a)



(b)



(c)

Graf 3.7: ROC křivka zobrazující výsledky trénování s použitím Haarových, LBP, HOG a SVMHOG příznaků na datové sadě chodců (a), obličejů (b) a dopravních značek (c)

Závěr

V práci jsem se věnoval metodám používaných k detekci objektů v obraze. Představil jsem nejčastěji užívané rozdělení těchto metod do čtyř základních skupin. Zvláště důležitou skupinou jsou pak appearance-based metody, do kterých patří i Viola-Jones detektor. Ten lze považovat v jistém smyslu za základ a výchozí bod pro mnoho dalších rozšíření, a je mu tak věnována patřičná pozornost. Jedním z těchto rozšíření je také algoritmus WaldBoost, který je v této práci využíván. Nemalou pozornost jsem také věnoval charakteristice vybraných příznaků, které jsou pro detekci objektů využívány. Ve své práci částečně vycházím z poznatků uvedených ve článku [21].

Další část této práce je shrnutím navrhnutých rozšíření a prezentací jejich výsledků. V případě dat získaných z databáze Flickr sice nedošlo ke zlepšení, zároveň však ani ke zhoršení. Z toho lze vyvodit, že takto získaná a automaticky anotována data jsou k trénování stejně vhodná, jako původní, již existující data. Případné rozdíly jsou v rámci chyby měření. Prahy pro automatickou detekci, tak jak byly zvoleny, nepřinášejí žádné rozdíly. Taktéž vzájemná kombinace nových a původních dat nevytváří žádné významné rozdíly. Použití existujících datových sad se neprokázalo jako přínosné. Trénování jen na takto získaných datech vykazuje oproti referenčním výsledkům významné zhoršení. Problémem bude zřejmě nízká variabilita mezi trénovacími vzorky (na obrazcích se často vyskytují totožné obličeje, velká míra transformací dosahující až 45 vzorků z jediného pozitivního vzoru). V kombinaci s dalšími datovými sadami se situace zlepšuje, stále se však jedná o zhoršení.

Implementace histogramu gradientů se ukázala jako přínosná zejména ve spojení s datovou sadou chodců. Ve spojení s SVM dopadla na této sadě ze všech testovaných příznaků nejlépe. V případě normalizace se ve většině experimentů jeví jako nejlepší L2-norm. Výjimku tvoří trénování na datové sadě dopravních značek, kde bylo dosáhnuto nejlepších výsledků bez použití normalizace. Největší přínos je opět na datové sadě chodců, kde je rozdíl oproti vynechání normalizace výrazný. Z výsledků přetrénování SVM u příznaků SVMHOG nelze činit jednoznačné závěry. Pro ty by bylo třeba spustit více experimentů, trénování s použitím těchto příznaků je však časově velmi náročné a tak bylo vybráno jen několik experimentů. Z těch lze vybrat alespoň jeden poznatek, a sice že v případě trénování na datové sadě chodců je přínosnější trénovat s větším počtem příznaků než častěji přetrénovat SVM. S použitím příznaků HOG se podařilo potvrdit, že větší počet těchto příznaků má pozitivní vliv na výsledky trénování. Celkový přínos SVM je nejednoznačný. Zatímco na datové sadě chodců je přínos zřetelný, u ostatních datových sad je diskutovatelný až záporný. Zde by bylo na místě vyzkoušet další experimenty a pokusit se najít ideální parametry SVM pro každou datovou sadu. A konečné srovnání histogramu gradientů s ostatními příznaky jsem již zmínil. Na datové sadě chodců, ale také dopravních značek si tyto příznaky vedly nejlépe. Pro použití s obličeji se jejich použití jeví jako nevhodné. Pro obličeje jsou zřejmě vhodnější metody pracující s lokální charakteristikou okolí (jako jsou Haarovy příznaky nebo LBP), než detekce na základě gradientů.

V případné další práci bych se zaměřil na SVM. Nabízí se mnoho možností ohledně jeho začlenění do projektu a použití ve spojení s jinými příznaky, než jen s histogramem gradientů. Bylo by také vhodné implementovat nějakou metodu pro automatický odhad optimálních parametrů SVM. Také by bylo možné se zaměřit na obličeje a pokusit se zjistit, z jakého důvodu pro ně není histogram gradientů vhodný, a zda jej nelze nějak rozšířit pro vylepšení výsledků.

Literatura

[1] Yang, M. H., Kriegman, D. J., Ahuja, N.: *Detecting Faces in Images: A Survey.* IEEE Trans. Pattern Analysis and Machine Intelligence 2002.

[2] Yang, G., Huang, T. S.: *Human Face Detection in Complex Background.* Pattern Recognition 1994.

[3] Sirohey, S.A.: *Human Face Segmentation and Identification.* Technical Report CS-TR-3176, Univ. of Maryland 1993.

[4] Canny, J.: *A Computational Approach to Edge Detection.* IEEE Trans. Pattern Analysis and Machine Intelligence 1986.

[5] Augusteijn, M.F., Skujca, T.L.: *Identification of Human Faces through Texture-Based Feature Recognition and Neural Network Technology.* Proc. IEEE Conf. Neural Networks 1993.

[6] Sakai, T., Nagao, M., Fujibayashi, S.: *Line Extraction and Pattern Detection in a Photograph.* Pattern Recognition 1969.

[7] Yuille, A., Hallinan, P., Cohen, D.: *Feature Extraction from Faces Using Deformable Templates.* Int'l J. Computer Vision 1992.

[8] Kohonen, T.: *Self-Organization and Associative Memory*. Springer 1989.

[9] Viola, P., Jones, M.: *Robust Real-time Object Detection.* ICCV, Vancouver, Canada 2001.

[10] Přinosil, J., Krolikowski, M.: *Využití detektoru Viola-Jones pro lokalizaci obličeje a očí v barevných obrazech.* Elektrorevue ISSN 1213-1539, 2008.

[11] Juránek, R.: *Rozpoznávání Vzorů v Obraze Pomocí Klasifikátorů*. diplomová práce, Brno, FIT VUT v Brně, 2007.

[12] Šochman, J., Matas J.: *WaldBoost - Learning for Time Constrained Sequential Detection.* IEEE Computer Society, Washington, DC, USA, 2005.

[13] Wald, A.: *Sequential Analysis*. Dover, New York, 1947.

[14] Mäenpää, T., Pietikäinen, M.: *Texture Analysis with Local Binary Patterns.* Dept. Of Electrical and Information Engineering, University of Oulu, Finland, 2004.

[15] Dalal, N., Triggs, B.: *Histograms of Oriented Gradients for Human Detection.* CVPR'05, 886-893, 2005.

[16] Vapnik, V.: *The Nature of Statistical Learning Theory*. Springer, N.Y., ISBN 0-387-94559-8, 1995.

[17] Šochman, J., Matas, J.: *Learning a Fast Emulator of a Binary Decision Process*. IJCV(83), No. 2, 149-163, 2009.

[18] Freund, Y., Schapire, R. E.: *A Decision-Theoretic Generalization of Online Learning and an Application to Boosting.* Journal of Computer and System Sciences, 55(1), 119-139, 1997.

[19] Drucker, H., Cortes, C.: *Boosting decision trees.* In Advances in Neural Information Processing Systems 8, 479–485, 1996.

[20] Mäenpää, T., Ojala, T., Pietikäinen, M., Soriano, M.: *Robust texture classification by subsets of local binary patterns*.
in Proc. 15th International Conference on Pattern Recognition, Barcelona, Spain, 2000

[21] Kubínek, J.: *Extending training dataset for face detector learning*.
CESCG'09, 2009

Seznam příloh

Příloha 1. CD obsahující zdrojové texty, technickou zprávu v elektronické verzi, plakát a prezentované experimenty