VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

DETEKCE OBJEKTŮ V OBRAZE S POMOCÍ ROZŠÍŘENÉ SADY HAAROVÝCH PŘÍZNAKŮ A HISTOGRAMU

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

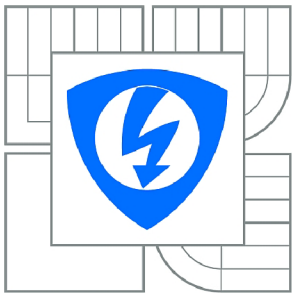
AUTOR PRÁCE
AUTHOR

Bc. MARTIN KRÁLÍK

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

DETEKCE OBJEKTŮ V OBRAZE S POMOCÍ ROZŠÍŘENÉ SADY HAAROVÝCH PŘÍZNAKŮ A HISTOGRAMU

OBJECT DETECTION IN IMAGES USING EXTENDED SET OF HAAR-LIKE FEATURES AND HISTOGRAM-BASED METHOD

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

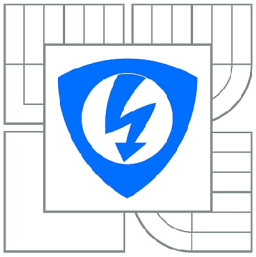
AUTOR PRÁCE — Bc. MARTIN KRÁLÍK

AUTHOR

VEDOUCÍ PRÁCE — Ing. RADIM BURGET, Ph.D.

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student: Bc. Martin Králík ***ID:*** 106556
Ročník: 2 ***Akademický rok:*** 2011/2012

NÁZEV TÉMATU:

Detekce objektů v obraze s pomocí rozšířené sady Haarových příznaků a histogramu

POKYNY PRO VYPRACOVÁNÍ:

Seznamte se s problematikou natočených Haarových příznaků a problematikou histogramů orientovaných gradiendů (HOG) v obrazových datech a implementujte v programovacím jazyce JAVA. S pomocí těchto příznaků natrénujte detektor a vyhodnoťte jejich přesnos z pohledu času a pohledu přesnosti. Výsledky zaneste do grafu a zhodnoťte.

DOPORUČENÁ LITERATURA:

[1] N. Dalal, W. Triggs, Histograms of Oriented Gradients for Human Detection, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR05 (2004)
[2] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, volume 1, pages I–511–I–518, Los Alamitos, CA, USA, April 2001, IEEE Comput. Soc.

Termín zadání: 6.2.2012 ***Termín odevzdání:*** 24.5.2012

Vedoucí práce: Ing. Radim Burget, Ph.D.
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Tato práce je zaměřená na detekci objektů v obraze s využitím rozšířené sady hárových příznaků a histogramu orientovaných gradientů. Nejprve je uveden základní princip extrakce a klasifikace obrazových dat. V další části je představen vlastní koncept příznaku založený na metodě Diffusion distance. Výstupem této práce je implementace těchto metod jako operátory pro aplikaci Rapiminer.

KLÍČOVÁ SLOVA

Adaboost, počítačové vidění, Diffusion distance, příznaky, histogramy, chodci, tepny

ABSTRACT

This diploma thesis is focused on detection in images using extended set of Haar-like features and histogram-based method. At first is introduced a basic concept of extraction and classification image features. The next part bring own concept of image features based on Diffusion distance. Result of this work is implementation this methods in Rapidminer.

KEYWORDS

Adaboost,computer vision,Diffusion distance,features,histogram, pedestrians, artery

KRÁLÍK, Martin *Detekce objektů v obraze s pomocí rozšířené sady Haarových příznaků a histogramu*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2011. 58 s. Vedoucí práce byl Ing. Radim Burget, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Detekce objektů v obraze s pomocí rozšířené sady Haarových příznaků a histogramu“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Radimovi Burgetovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

Brno ....

(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

(podpis autora)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OP Výzkum a vývoj
pro inovace

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

Stačí ráno otevřít oči a v lidském mozku se spustí plejáda algoritmů, které umožňují pochopit viděnou scénu. Za těmito algoritmy stojí miliony let evoluce a přestože stále není jasné jak fungují, roste úsilí předat tuto schopnost strojům.

Získávání znalostí z obrazových dat naráží na jejich složitost a nedostatečnou výkonnost výpočetních prostředků. Způsob jak se s tímto problémem vyrovnat je zvolit vhodný popis, který pomocí několika hodnot popíše podstatně složitější objekt. K popisu je možné využívat příznaky extrahující potřebné charakteristiky.

Tato práce okrajově zmiňuje využití Haarových příznaků, tématickým těžištěm jsou však metody založené na popisu směru růstu hran pomocí HOG (Histogram of Oriented Gradient).

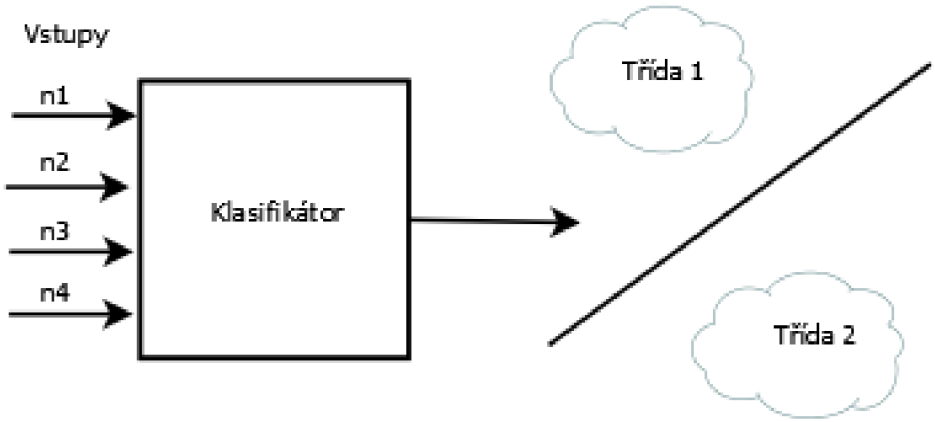
Hlavním přínosem práce je návrh a realizace přístupů, které vedou ke zrychlení detekce, snížení dimenze příznakového prostoru a snížení počtu falešných detekcí. Zrychlení detekce je dosaženo pomocí integrálních obrazů pro reprezentaci obrazových dat. Vysoká dimenze histogramu popisujícího vstupní obraz je redukována metodou Diffusion distance, která dokáže vyjádřit míru podobnosti dvou histogramů. Popis vstupního obrazu pomocí monolitického histogramu je nahrazen množinou příznaků. Výběr jednotlivých příznaků je na základě jejich relevantnosti realizován algoritmem Adaboost. Pro zvýšení detekční rychlosti a snížení počtu falešných detekci je zaveden koncept kaskády klasifikátorů. Díky tomuto kaskádovému řazení je možné sestavit klasifikátor tvořený libovolným klasifikačním modelem. Uvedené metody jsou implementovány jako operátory do prostředí RapidMiner. Díky tomu je možné propojit je s již existujícími operátory specializovanými na zpracování obrazu a dolování dat za účelem tvorby sofistikovanějších metod.

Zbytek práce je strukturován následovně - nejprve je čtenář seznámen s obecným principem extrakce a klasifikace obrazových dat. V další části jsou vysvětleny metody založené na příznacích HOG. Praktická sekce se věnuje implementaci a ověření metody HOG a metody Diffusion HOG v RapidMineru. Ověření je realizováno na úlohách detekce chodců a tepen. Závěrečná část práce je věnována diskuzi a zamyšlení nad použitými postupy a výsledky.

1 EXTRAKCE PŘÍZNAKŮ

1.1 Obrazové příznaky

Významným krokem v řetězci počítačového vidění je transformace obrazové informace na popis zobrazených objektů. Pro popis zobrazených objektů jsou extrahovány příznaky , které spolehlivě popisují objekty dané třídy.[12] Spolehlivý popis znamená, že hodnota příznaku na určité třídě (např. osobní automobil) vykazuje podobné hodnoty s určitou maximální tolerovatelnou chybou. Na jiné třídě objektů (např. chodec) musí příznak vykazovat hodnotu natolik odlišnou, aby klasifikátor dokázal správně rozlišit do jaké třídy objekt spadá. Klasifikátor je tedy matematický stroj, který disponuje n vstupy na něž vstupuje n - rozměrný vektor příznaků x_{i} a jedním diskrétním výstupem signalizujícím třídu objektu $\omega_{\mathrm{r}} = d(x)$. Kde $d(x)$ je funkce nazývaná rozhodovací pravidlo klasifikátoru a ω_r je identifikátor klasifikační třídy.



Obr. 1.1: Binární klasifikátor

Úspěšné rozpoznání objektů je vysoce závislé na volbě vhodného typu příznaku. Obecné požadavky na příznaky je možné shrnout do následujících několika bodů.[12]

- Invariantnost - hodnota příznaku je nezávislá na změně jasu, kontrastu, posuvu, natočení nebo změně měřítka.
- Diskriminabilita - objekty spadající do různých tříd vykazuje odlišné hodnoty příznaků.
- Spolehlivost - objekty spadající do stejných tříd vykazují podobné hodnoty příznaků
- Rychlost výpočtu - Tento požadavek je často protichůdný vůči spolehlivosti příznaku. Vyšší komplexnost a vyjadřovací schopnost příznaku je často vykoupena časově náročnějším výpočtem. Tento konflikt je nutné řešit zejména v real-time aplikacích.

Z hlediska oblasti nad kterou je příznak vypočítáván je důležité jejich rozlišení na globální a lokální. Výpočet globálních je prováděn nad celým obrazem (např. histogram rozložení jasu, Fourierova transformace). Lokální příznak je vypočítáván pro každou oblast v obraze zvlášť (suma intenzity jasu, směr hrany).[12] V souvislosti s rychlostí výpočtu je podstatné rozdělení na příznaky typu dense (příznak je vypočítáván pro každou pozici v obraze) a sparse (příznak je počítán jen pro určité body v obraze). Předchozí text napovídá, že výběr vhodných příznaků se bude zásadně odvíjet od oblasti aplikace a zároveň je zřejmé, že přímo ovlivní úspěšnost klasifikace. Tato práce se zabývá především lokálními příznaky typu sparse. Následující kapitola zmiňuje principy několika, v současnosti často používaných příznaků.

1.2 Extrakce příznaků

1.2.1 Integrální obraz

Průběh extrakce lokálních příznaků lze zobecnit do několika následujících kroků:

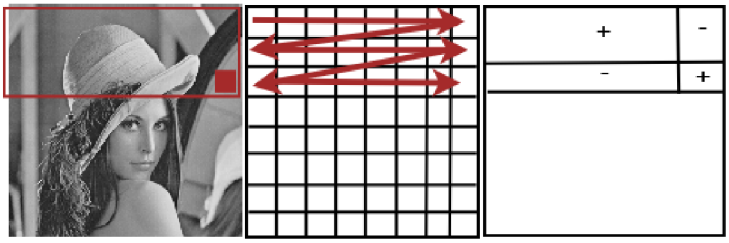
- Výběr vhodné (náhodně nebo heuristikou) pozice v obraze pro extrakci lokálního příznaku.
- Nad touto pozicí je rozvinut blok (nejčastěji pravoúhlý), nebo maska sloužící pro výběr pixelů.
- Pixely získané v předchozím kroku jsou postoupeny do transformace, která z jejich vstupních hodnot získá odezvu příznaku. Transformace je často založena na operacích sumace a diference, ve zcela obecném případě se dá mluvit o konvoluční operaci.

Právě poslední krok extrakce, tedy transformace je často časově náročná zejména kvůli častým přístupům do paměti. Pokud je obrazová informace udržována jako pole koeficientů, pak přístup ke každému pixelu pod rozvinutým blokem představuje přístup do paměti, což vede ke zpomalení aplikace. Výše zmíněný problém lze řešit tak, že vstupní obrazová data jsou transformována na integrální obraz.[14] Integrální obraz je tvořen řádkovým a sloupcovým kumulativním součtem. Hodnotu integrálního obrázku summed area table (SAT) na souřadnicích x, y, kde $I(x, y)$ je hodnota původního pixelu lze spočítat následovně:

$$S_{\text{sat}}(x, y) = S_{\text{sat}}(x, y-1) + S_{\text{sat}}(x-1, y) + I(x, y) - S_{\text{sat}}(x-1, y-1) \tag{1.1}$$

Jakmile je provedena transformace integrální reprezentace, je možné získat sumu libovolné pravoúhlé oblasti $RecSum(r)$ pouze pomocí 4 referencí (označovaných SAT), tento fakt vede k podstatnému snížení přístupu do paměti.[12] [2]

$$\begin{aligned} S_{\text{sum}} = S_{\text{sat}}(x-1, y-1) + S_{\text{sat}}(x+w-1, y+h-1) - \\ S_{\text{sat}}(x-1, y+h-1) - S_{\text{sat}}(x+w-1, y-1) \end{aligned} \tag{1.2}$$



Obr. 1.2: Integrální obraz

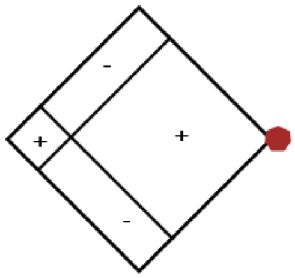
Výše uvedená definice integrálního obrazu je vhodná pro získání sumy pravoúhlé oblasti, která je rovnoběžná s osou x. V případě potřeby sumy oblasti, která není rovnoběžná s osou x) je nutné zavést sumu rotované oblasti rotated summed area table (SAT) - pro každý úhel natočení je třeba vytvořit odpovídající integrální obraz.[5] V praxi se nejčastěji využívá rovnoběžný a 45 stupňů natočený integrální obraz. Rotovaná suma SAT je pro čtyřicetistupňové natočení určena následujícími vztahy.

$$\begin{aligned} S_{\mathrm{rsat1}} &= S_{\mathrm{rsat1}} + S_{\mathrm{rsat1}}(x-1, y-1) - S_{\mathrm{rsat1}}(x-2, y-1) + I(x, y) \\ S_{\mathrm{rsat}} &= S_{\mathrm{rsat1}}(x-2, y+1) - S_{\mathrm{rsat1}}(x-2, y) + I_{\mathrm{R}}(x, y) \end{aligned} \tag{1.3}$$

Výpočet RSAT definovaný rovnicemi 1.3 je oproti 1.1 dvouprůchodový.[5] Tento fakt znamená, že je nejprve prováděn výpočet podle první rovnice vztahu 1.3 a to ve směru zleva doprava a shora dolů. Druhý průchod je počítán podle druhé rovnice ve směru zprava doleva a zdola nahoru. Sumu libovolné oblasti natočené o 45 stupňů je opět možné získat pouze čtyřmi referenčními body.

$$\begin{aligned} S_{\mathrm{sum}} = S_{\mathrm{rsat}}(x-1, y-1) S_{\mathrm{rsat}}(x+w-1, y+h-1) \\ -S_{\mathrm{rsat}}(x-1, y+h-1) - S_{\mathrm{rsat}}(x+w-1, y-1) \end{aligned} \tag{1.4}$$

Je nutné počítat pro každý úhel natočení zvláštní integrální obraz.



Obr. 1.3: Rotovaný integrální obraz

1.2.2 Příznaky

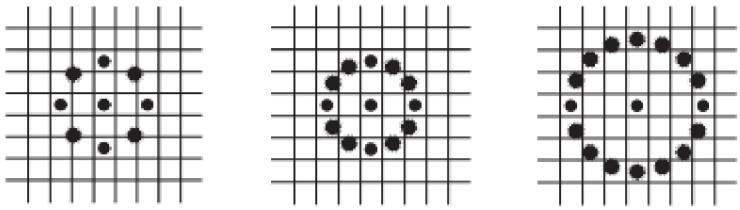
Příznaky LBP (local binary patterns) jsou často využívány pro zpracování informací o textuře.[12] Jejich základní princip spočívá ve zjištění intenzity jasu v okolí vytyčeném kolem středového pixelu. Hodnota jasu středového pixelu slouží k prahování okolních hodnot. Zmíněné okolí je nejčastěji kruhového charakteru a je určeno počtem bodů a poloměrem. Pozice okolních bodů je sekvenčně číslovaná (první bod na pozici 0). Číslování pozice přiřazuje jednotlivým bodům váhu ve formátu 6^P. Výsledná odezva LBP příznaku je dána vztahem $O_{\mathrm{p,r}} = \sum_{p=0}^{p-1} s \times (g_{\mathrm{p}} - g_{\mathrm{c}}) \times 2^P$

$$\phi(t) = \begin{cases} 1 & x \geq 0 \\ 0 & x \leq 0 \end{cases} \tag{1.5}$$

g_{c} - grayscale hodnota středového pixelu

g_{p} - grayscale hodnota pixelu na pozici P

LBP je vhodné pro detekci základních obrazových primitiv jako jsou špičky, konce čar, hrany, rohy. Typ detekované primitivy je závislý na konfiguraci LBP vzoru viz. (následující obrázek). Pro zajištění rotační invariance je nutné příznaky normalizovat. Normalizace vychází z předpokladu, že okolní body lze (po prahování a váhování) vyjádřit jako binární vzor, který je možné rotovat (nejčastěji doprava). Rotací tohoto binárního vzoru je zajištěna rotační invariance přičemž nejnižší hodnota odezvy ze všech odezev získaných rotací představuje správnou detekci. LBP se velmi často využívá ve spolupráci s lokálním histogramem jasu pro popis a segmentování obrazu.[12] [16] Díky relativně snadnému výpočtu jsou tyto příznaky vhodné pro implementaci do real-time aplikací nebo hardwaru (např. FPGA).



Obr. 1.4: LBP příznaky

1.2.3 Haarovy příznaky

Haarovy příznaky jsou založeny na rozdílu jasu mezi obdélníkovými oblastmi.[16][5] Názorná ukázka základního příznaku je na následujícím obrázku. Příznak je tvořen bílým obdélníkem uvnitř kterého je umístěn menší černý. Odezva příznaku je tedy v tomto případě dána následujícím vztahem, kde R_{white} *a* R_{black} představují sumy intenzit pixelů v bílém a černém obdélníku.

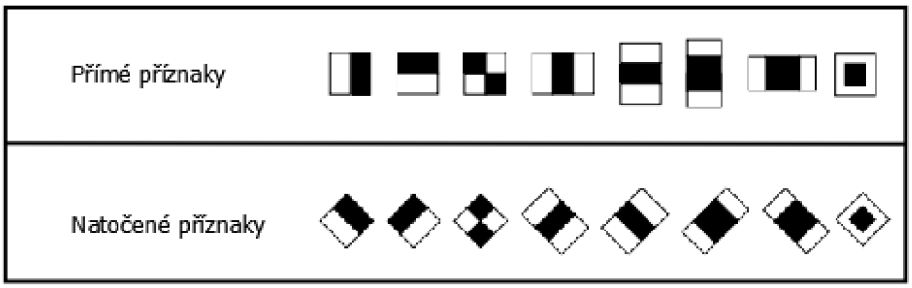
$$F_{\text{haar}} = E(R_{\text{white}}) - E(R_{\text{black}}) \tag{1.6}$$

Haarovy příznaky vycházejí z Haarova waveletu, který je popsán následující wavelet funkcí:

$$\phi(t) = \begin{cases} 1 & 0 \leq t \leq 1/2 \\ -1 & 1/2 \leq t \leq 1 \\ 0 & \text{pokud jinak} \end{cases} \tag{1.7}$$

Předpis wavelet funkce a vztah pro odezvu příznaku naznačuje, že tyto příznaky mohou aproximovat derivaci. Směr derivace je určen kompozicí příznaku. Následující obrázek demonstrují kompozici příznaku a směr derivace, který aproximují.

Haarovy příznaky můžeme dělit do kategorií podle jejich použití (detekce hran, linek, středově orientované), nebo podle chronologie jejich představení odborné veřejnosti. Nejprve byl využíván tzv. základní set (Basic Haar set), ten byl následně rozšířen o příznaky natočené o 45 stupňů.[8][14]



Obr. 1.5: Haarovy příznaky

Výše v textu byl uveden vztah 1.6 představující výchozí výpočet odezvy příznaku. Takový výpočet je vhodný pro demonstrační účely, v reálné praxi je potřeba odezvu příznaku normalizovat a to především kvůli změnám jasu a změně velikosti příznaku. Následující výraz demonstruje právě takovou normalizaci odezvy. [5]

$$F_{\text{haar}} = \frac{E(R_{\text{white}}) - E(R_{\text{black}})}{\sqrt{|E(R_\mu)^2 - E(R_\mu^2)|}} \tag{1.8}$$

Jedna odezva získaná z jednoho příznaku má vypovídací hodnotu o obdélníkové oblasti na souřadnicích x y a velikosti příznaku (šířka w a výška h). Pokud chceme v obrazu detekovat určitý objekt, zavádí se pojem skenovací okno. Skenovací okno je obdélník nebo čtverec s pevně danou šířkou a výškou, toto okno je pak s určitým krokem K posunováno přes celý obraz. V každém kroku tohoto detekčního okna je na jím vymezené ploše generováno N příznaků z nichž je získáno N odezev. Tyto odezvy jsou získány rozdílem sum intenzit jednotlivých obdélníků tvořících příznak

$(E(R_{\text{white}}) - E(R_{\text{black}})$. Pokud uvažujeme na vstupu šedotónový obraz a na souřadnicích $[x, y]$ je umístěn obdélníkový příznak o šířce w a výšce h, pak je suma jednoho obdélníku tvořící příznak definována jako suma pixelů v jím vymezené oblasti. [8]

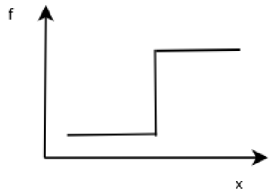
$$s_{\text{u}} = \sum_{x=x_{\text{start}}}^{w} \sum_{y=y_{\text{start}}}^{h} I(x, y) \tag{1.9}$$

Je nutné poznamenat, že vztah 1.9 poukazuje na skutečnost,že získání sumy každého obdélníku je časově náročná operace. Pro obdélník o rozměrech w a h je provedeno $w \times h$ přístupů do paměti a $w \times -1$ sum. Počet těchto operací při detekci pomocí N příznaků a S skenovacích oken (pozic oken) vzroste $S \times N$. Taková situace je problémem zejména pro real-time aplikace. Vhodným řešením pro takový případ je využití integrálního obrázku, díky němuž je možné spočítat sumu jakékoliv obdélníkové oblasti pomocí čtyř přístupů do paměti a tří sum.

1.2.4 Histogramy orientovaných gradientů (HOG)

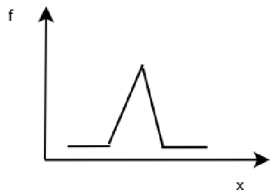
Příznaky HOG slouží k určení gradientu hrany v obraze. Následující řádky definují pojmy hrana, gradient a histogram. Hrana je oblast v obraze, kde dochází k náhlé změně hodnoty jasu. Právě tyto oblasti s přítomnými hranami obsahují více informace než jiná místa v obraze. Hrana je vyjádřením pro změnu rychlosti a směru růstu obrazové funkce $h(x, y)$, toto vyjádření je ve skutečnosti vektor o dvou složkách.[3] Podle průběhu jasové změny jsou rozděleny do několika základních jasových profilů. Nutno poznamenat, že tyto profily představují ideální hrany, v reálném obrazu jsou však zašuměné.[3] [12]

- Skokový profil



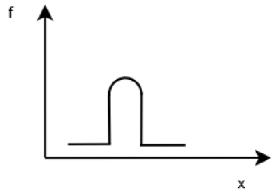
Obr. 1.6: Skokový profil

- Střechový profil



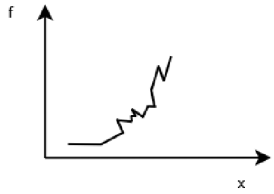
Obr. 1.7: Střechový profil

- Linkový profil



Obr. 1.8: Linkový profil

- Zašuměný profil



Obr. 1.9: Zašuměný profil

Pro detekci hran se nejčastěji využívá hledání maxima prvních derivací, nebo hledání průchodu druhých derivací nulou. Nejjednodušším způsobem jak aproximovat derivaci diskrétní obrazové funkce je provést diferenci sousedních pixelů.[3] Symetrický tvar aproximace derivace v bodě p může mít tvar

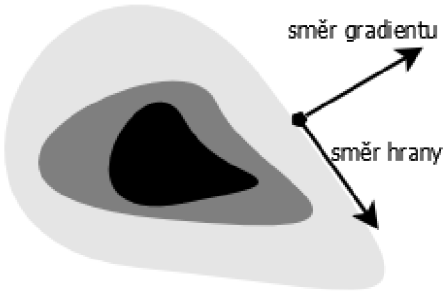
$$f'(p) \approx f(p+1) - f(p-1) \tag{1.10}$$

Samotná derivace se obrazových dat se nejčastěji provádí konvolucí

$$f' \approx [-1, 0, 1] * f \tag{1.11}$$

Vektor $[-1, 0, 1]$ je označován jako konvoluční jádro (kernel). Podle rozměrů jádra ($m_{\text{řádků}} \times n_{sloupců}$) a obsažených koeficientů rozlišujeme několik hranových operátorů. Nejznámější jsou (Roberts, Prewitt, Sobel, Robinson, Kirsch, Laplacián) zmíněné operátory se odlišují především v citlivosti na šum a vhodnosti pro detekci odlišných typů hran. Příkladem může být Sobelův operátor, který se hodí především pro detekci svislých a vodorovných hran. Hlubší rozebírání zmíněných operátorů je mimo rámec této práce, protože pro následující implementaci bude využíváno základní jádro ve tvaru $[-1, 0, 1]$.

Gradient obrazové funkce je definován jako vektor parciálních derivací ve směru x a y.[3] Pokud má obrazový bod velký modul gradientu, jedná se o hranový bod. Směr gradientu je kolmý na směr růstu hrany.



Obr. 1.10: Gradient hrany

Vztah definující gradient

$$\nabla f(x,y) = \left(\frac{\mathrm{d}f}{\mathrm{d}x}, \frac{\mathrm{d}f}{\mathrm{d}y}\right) \tag{1.12}$$

Následující vztah definuje výpočet velikosti modulu gradientu.

$$\|\nabla f(x,y)\| = \sqrt{\left(\frac{\mathrm{d}f}{\mathrm{d}x}\right)^2 + \left(\frac{\mathrm{d}f}{\mathrm{d}y}\right)^2} \tag{1.13}$$

Poslední vzorec umožňuje určení úhlu ψ, který svírá přímka procházející body dx a dy vůči vodorovné ose x.

$$\psi = \arctan\left(\frac{\mathrm{d}f}{\mathrm{d}x} \Big/ \frac{\mathrm{d}f}{\mathrm{d}y}\right) \tag{1.14}$$

Úspěšnost detekce velmi závisí na přítomnosti šumu, tímto problémem trpí zejména metody založené na druhé derivaci. Druhá derivace popisuje rychlost změny hodnot jasu. Jinými slovy druhou derivaci lze označit za změnu změny, z tohoto faktu plyne, že druhá derivace se hodí pro detekci strmých nebo izolovaných hran. Vážným problémem je, že bude zároveň zesilovat šum.[3] A aproximace druhé derivace může vypadat následovně

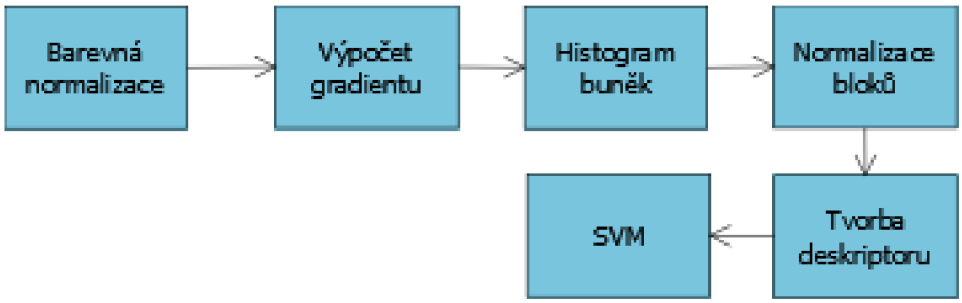
$$f''(p) \approx f(p+1) - f(p-1) - 2 \times f(p) \tag{1.15}$$

Operátor používající druhou derivaci se nazývá Laplaceův operátor, využívá se zejména pro operace ostření obrazu. Vzhledem k citlivosti na šum, může při použití Laplaceova operátoru docházet k nalezení dvojitých hran. Pro zmírnění vlivu šumu se využívá Gaussův operátor, který provede rozmazání obrazu a tím zmírní vliv šumu. Laplaceův a Gausův operátor se kombinují do jedné operace, která se nazývá Laplacian of Gaussian (LoG).

Robustní hranový detektor musí být schopen vyrovnat se s přítomností šumu, měl by dokázat ignorovat nepodstatné (falešné) hrany a zároveň poskytovat dobrou lokalizaci významných hran. Jedním z nejlépe vyhovujících detektorů ve smyslu předchozích požadavků je Cannyho hranový detektor.

Histogram je sloupcové zobrazení intervalových četností, kdy šířka sloupce je šířkou intervalu a výška sloupce představuje četnost výskytu dané třídy. V kontextu předchozího textu, zmiňujícího orientaci gradientu hrany se histogram jeví jako ideální prostředek pro popis distribuce gradientu v libovolně velkém bloku.

Metoda histograms of oriented gradients (HOG) využívá pro popis objektů (v tomto případě postav) příznakové vektory, které popisují směr růstu hran oddělujících objekt od pozadí.[1] Jednotlivé kroky algoritmu popisuje následující text spolu s blokovým schématem algoritmu.



Obr. 1.11: Schéma HOG

- V prvním kroku je nutné vypočítat pro vstupní obraz derivaci ve směru x a y. Derivace se provede pomocí operace konvoluce vstupního obrazu s vhodným konvolučním jádrem. Podle závěrů autorů této metody [odkaz] je nejvhodnější použít jádro $\{1, 0, -1\}$ pro dx a $\{1, 0, -1\}^{\mathrm{T}}$ pro dy.
- Dalším bodem je výpočet gradientu pro každý pixel vstupního obrazu pomocí vztahu 1.14. Výsledná velikost úhlu gradientu s pohybuje podle nastavení v rozsahu 0 - 360 stupňů nebo 0 - 180 stupňů. [1][10] Využití vyššího rozsahu nepřináší žádné podstatné zlepšení a proto je vhodné z hlediska výpočetní náročnosti volit nižší rozsah. Rozsah je následně diskretizován do N tříd. Nejčastější se rozsah úhlu dělí mezi devět tříd, v případě rozsahu 0-180 stupňů připadá pro každou třídu šířka dvaceti stupňů. Každá třída je v dalším textu označována jako bin. Vektor těchto binů je možné připodobnit k histogramu, přičemž hodnota sloupce odpovídajícího binu je tvořena sumou korespondujících modulů gradientu.
- Nad množinou těchto bodů se rozvine síť buněk o velikosti 8×8 pixelů. Hodnoty gradientů jednotlivých pixelů náležících do příslušných buněk jsou podle své velikosti zařazeny do odpovídajícího binu (třídy). Zařazením do odpovídající binu je myšleno iterování sumy odpovídajícího binu o modul zpracovávaného gradientu. Předchozí operace vedou k tomu, že vstupní obraz je rozdělen do buněk a ty jsou popsané histogramy vyjadřujícím distribuci gradientu v jejich vnitřku.
- Histogramy jednotlivých buněk jsou ovlivněny značnou mírou šumu, který je způsoben velkými jasovými rozdíly, kvantizací a nakonec samotnou aproximací derivace při výpočtu gradientu. Z tohoto důvodu je nutno zavést normalizaci. Normalizace je v tomto případě realizovaná jako bloková operace kdy jsou nad

buňkami rozvinuty vzájemně se překrývající bloky (o rozměru 3×3 buněk). V původní práci [citace] jsou rozlišovány dva typy bloků a to obdélníkový a kruhový. Operace s kruhovými bloky jsou výpočetně náročnější a zároveň nepřinášejí významné zlepšení algoritmu, proto je vhodné volit obdélníkové bloky. Z každého bloku získáme vektor $\mathbf{v}$ histogramů nazývaný descriptor. Tento vektor je následně normalizován pomocí L1 nebo L2 normalizace.

$$\text{L1} : \mathbf{v} \rightarrow \mathbf{v} / \sqrt{\|\mathbf{v}\|_1 + \epsilon} \tag{1.16}$$

$$\text{L2} : \mathbf{v} \rightarrow \mathbf{v} / \sqrt{\|\mathbf{v}\|_2^2 + \epsilon^2} \tag{1.17}$$

Algoritmus: *HOG*

Vstup : vstupní obraz i, počet buněk na ose x n_{numBinsX}, počet buněk na ose y m_{numBinsY}, počet binů diskretizující rozsah úhlu gradientu a_{numBinsA},konvoluční jádro k

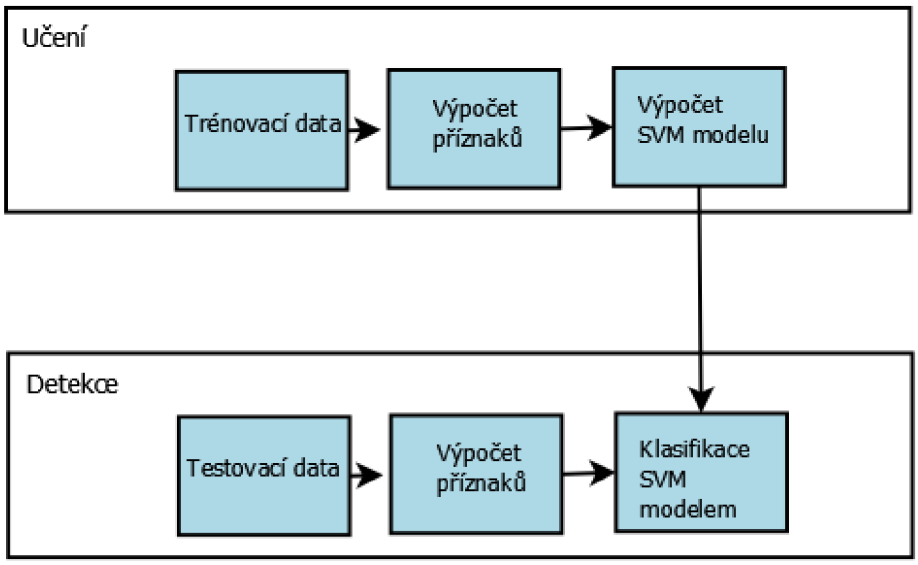
Výstup: D množina descriptorů

```
dx= derivuj(i,k)
dy= derivuj(i,k^T)
foreach pro každý pixel i do
    spočti úhelGradientu(dx,dy)
    spočti modulGradientu(dx,dy)
    i_indexBinu=diskretizuj(u_uhelGrad,a_numBinsA)
    i_indexBunky=zjistiBunku(aktuální souřadnice x,aktuální souřadnice y)
    iterujHistogram(i_indexBunky,i_indexBinu,g_modulGradientu)
end
foreach pro všechny buňky i do
    Zařaď do normalizačního bloku
    Normalizuj
end
Spočti výsledný descriptor D
return D
```

Předchozí kroky vedoucí k získání descriptoru je možné označit jako výpočet příznakového vektoru. Obecné schéma detekčního systému sestává z trénovací a detekční části a právě příznakový vektor je klíčovým faktorem pro oba tyto bloky. Ve fázi trénování je ze vstupního vektoru dat vytvářen model. Model představuje matematické vyjádření hyperplochy, která dokáže správně rozřadit vektor trénovacích dat do odpovídajících předem známých tříd. Míru naučenosti a naučitelnosti modelu

ovlivňuje především vzájemná separovatelnost klasifikovaných tříd a dimenze příznakového vektoru. Ideálním případem je oddělení tříd pomocí jedné přímky, tohoto stavu se však v praxi těžko dosahuje a především u obrazových dat. Jejich příznakové vektory dosahují vysokých dimenzí a proto je nutné zavádět pokročilé klasifikátory schopné separovat nelineární problémy. Příkladem takového pokročilého klasifikátoru je SVM (Support Vector Machine).

Trénovací sada obrázků o rozměrech (64×128) obsahuje pozitivní vzorky (je na nich chodec) a negativní vzorky (bez přítomnosti chodce). Pomocí SVM je z trénovací množiny vytvořen klasifikační model. Tento model je využit v bloku detekce, kde slouží ke klasifikaci vstupního vektoru získaného skenovacím oknem o rozměrech (64×128). Okno je posouváno po vstupním obraze krokem K, v každém kroku je tímto oknem extrahován subobraz, z kterého je pomocí HOG vypočítán příznakový vektor (descriptor). Výsledný descriptor je klasifikován SVM, které rozhodne zda se jedná o chodce či nikoliv.



Obr. 1.12: Schéma učení

1.2.5 Diffusion distance příznaky

Problémem příznaků založených na histogramech je vysoká dimenze příznaků, pojmem dimenze je myšlen počet binů příznaku. Koncepce těchto příznaku je inspirovaná Haarovými příznaky, které jednoduše provádí rozdíl jasu dvou oblastí vymezených příznakem. Pokud se tato myšlenka transformuje na prostor hran popisovaný histogramy, je potřeba určit míru podobnosti mezi dvěma histogramy. Touto operací je dimenze příznaku zredukována na velikost 1. Řešení tohoto úkolu je limitováno dvěma protichůdnými požadavky a to na rychlost a přesnost. Přesností je míněna především schopnost správně změřit podobnost dvou histogramů. V současnosti často používané metody jako χ^2 statistika nebo L_2 metrika vykazují problémy při deformaci hran objektu. Důvodem je, že tyto metody jsou zarovnané. Jinými slovy to znamená, že u dvou histogramů porovnávají jednotlivé odpovídající biny. Pokud je hrana objektu deformovaná, může dojít k přesunu významných hodnot do okolních binů a tedy k chybě. Řešením je využít metody, které berou v úvahu i mezilehlé biny.

Vhodným kandidátem je metoda Diffusion Distance, která naplňuje výše zmíněné požadavky.Princip Diffusion Distance je založen na ustálení teploty mezi dvěma oblastmi. [9] Doba, neboli počet kroků, které bylo nutné provést do okamžiku ustálení představují vlastní metriku. Tyto dvě oblasti různých teplot jsou ony dva histogramy jejichž vzdálenost je třeba určit. Významnou výhodou této metody je, že pracuje v konstantním čase a je silně invariantní vůči deformaci histogramu.[9] Formálně je prezentovaná situace popsaná následovně. Uvažujme izolovanou ohřátou oblast v níž je teplota funkcí $T(x,t)$. V čase 0, tedy v okamžiku ohřátí je teplota určená $T(x,0)$. V čase větším jak 0 se uplatní následující rovnice vedení tepla.

$$\frac{\partial T}{\partial t} = \frac{\partial^2 T}{\partial x^2} \tag{1.18}$$

Řešení rovnice 1.18 je určeno následujícím výrazem.

$$T(x,t) = T_0(x) * \Phi(x,t) \tag{1.19}$$

T_0 představuje počáteční podmínku a $\Phi(x,t)$ je Gaussian filtr definovaný následovně.

$$\Phi(x,t) = \frac{1}{2\pi^{\frac{1}{2}}} exp(-\frac{x^2}{2t^2}) \tag{1.20}$$

Nakonec zbývá zavést výsledný vztah pro výpočet vzdálenosti dvou histogramů h_1 a h_2.

$$\widehat{K}(h_1,h_2) = \int_0^{\mathrm{t}} k(|T(x,t)|)dt \tag{1.21}$$

Symbol k uvedený v rovnici 1.21 značí normu, neboli metriku, která je použitá k výpočtu vzdálenosti. Výše uvedené vztahy představují teoretický rámec, pro praktickou implementaci jsou však zavedena následující zjednodušení. [9]

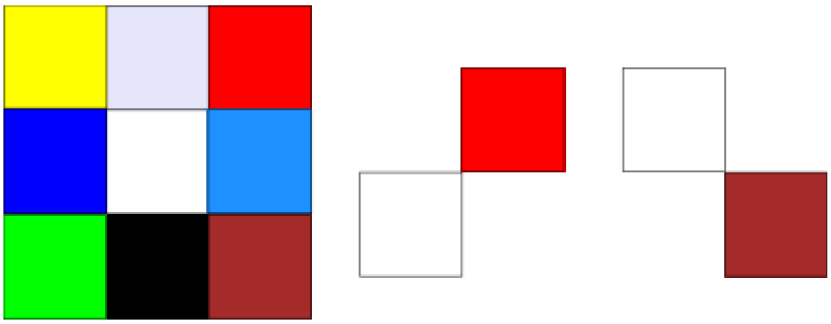
$$K(h_1, h_2) = \sum_{l=0}^{L} k(|d_l(x)|) \tag{1.22}$$

$$d_0(x) = h_1 - h_2 \tag{1.23}$$

$$d_l(x) = [d_{l-1} * \Phi(x, s)] \downarrow_2 \tag{1.24}$$

Výpočet vzdálenosti v čase 0 určený vztahem 1.23 využívá L_1 normu, jinými slovy jde o sumu rozdílů vzájemně odpovídajících binů. V každém následujícím kroku je provedena konvoluce předchozího výsledku spolu s gausovským jádrem a výsledek je podvzorkován. Právě konvoluce a podvzorkování provádí diskretizovanou simulaci systému, kde postupně dochází k rovnoměrnému vyrovnání teploty.

Výstupní odezva těchto příznaků je získaná jako rozdíl dvou histogramů určených dvěma pravoúhlými oblastmi. Pro ilustraci je uveden následující obrázek. Nejprve je určena první pravoúhlá oblast (bíla výplň) a rámci možných pozic v osmiokolí je vygenerována druhá(barevná pole). Mezi histogramy těchto oblastí je vypočítaná vzdálenost pomocí diffusion distance, která právě představuje výstupní odezvu příznaku.



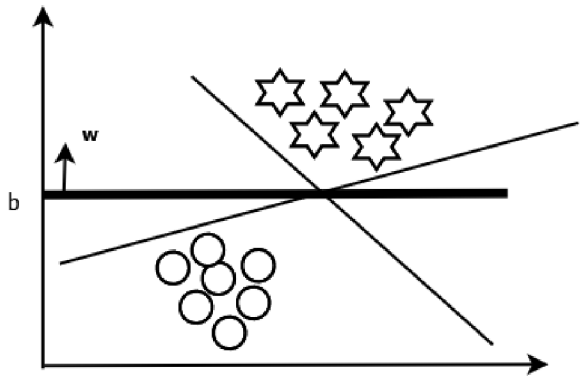
Obr. 1.13: Příznak DiffusionHog

1.3 Klasifikace příznaků

1.3.1 Support Vector Machine (SVM)

SVM je algoritmus využívaný pro klasifikační a regresní úlohy. Přednosti této metody je, že dokáže transformovat nelineární klasifikační úlohu do jiného prostoru (feature space) kde je realizovatelná lineární separace.[11] K transformaci jsou využívány jádrové funkce označované jako kernel trik. Důležitým faktem je, že nový prostor do kterého byla provedena transformace má vždy vyšší dimenzi než prostor původní.

Hlavní myšlenka SVM spočívá v nalezení optimální dělící roviny mezi dvěma třídami. Pod slovem optimální je skrytá snaha maximalizovat vzdálenost dělící roviny od nejbližších vzorků, což vede ke zlepšení výsledného klasifikátoru.



Obr. 1.14: Oddělující rovina

Výchozí úlohou pro SVM je klasifikace lineárně separovatelných dat. Princip funkce SVM je demonstrován následovně.[11] Uvažujme N vstupních vzorků trénovacích dat ve tvaru $\{\mathbf{x_i}, y_i\}$ kde i nabývá hodnot $1 \ldots N$, $\mathbf{x_i}$ je vstupním vektorem příznaků pro i-tý vstupní vzorek a y_i je zařazení do odpovídající klasifikační třídy. V případě binární klasifikace může y_i nabývat $y_i \in \{-1, 1\}$. Vychází se z předpokladu, že existuje hyperplocha, která dokáže separovat trénovací data obou tříd. Pokud platí předchozí předpoklad, tak body vstupního vektoru $\mathbf{x_i}$ ležící na oddělující hyperploše vyhovují vztahu $\mathbf{w} \times \mathbf{x_i} + b = 0$. V předchozím vztahu je nově zaveden normálový vektor hyperplochy vektor $\mathbf{w}$ a posunutí (bias) b. V dalším kroku je definována vzdálenost hyperplochy od počátku souřadného systému, toho je docíleno vztahem $|b|/\|\mathbf{w}\|$, kde $\|\mathbf{w}\|$ je kvadratická (eukleidovská) norma vektoru. Dále jsou zavedeny diference d_p a d_n, které udávají vzdálenost rozdělující hyperplochy od nejbližšího pozitivního a nejbližšího negativního vzorku. Součet těchto diferencí představuje šířku hranice (označováno jako margin), kterou se algoritmus SVM snaží maximalizovat. Operace maximalizující okraj pracují s faktem, že je možné vzájemně měnit velikost normálového vektoru $\|\mathbf{w}\|$ a velikost biasu tak, aniž by došlo k narušení správnosti

klasifikace. Maximalizace hranice tedy vede na následující vztahy, které zajistí že nejbližší body negativní a pozitivní třídy budou odpovídat rovnicím podpůrných hyperploch.

$$\mathbf{w} \times \mathbf{x}_{\mathrm{p}} = +1 \tag{1.25}$$

$$\mathbf{w} \times \mathbf{x}_{\mathrm{n}} = -1 \tag{1.26}$$

Velikost okraje je možné určit pomocí velikosti normálového vektoru $\|\mathbf{w}\|$ následujícím vztahem.

$$\frac{1}{\|\mathbf{w}\|} + \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|} \tag{1.27}$$

Jak je vidět ze vztahu 1.27 maximalizace okraje je závislá na minimalizaci normálového vektoru $\|\mathbf{w}\|$. Všechny vstupní trénovací příznaky $\mathbf{x}$ musí splnit následující vztah, který je možné chápat jako lineární omezující podmínku.

$$y_i(\mathbf{w} \times \mathbf{x} + b) \geq +1 \tag{1.28}$$

Princip SVM tedy spočívá v minimalizaci $\|\mathbf{w}\|$ vzhledem k omezující podmínce 1.28.[11] Algoritmus SVM dále tuto optimalizační úlohu transformuje tak, aby mohla být řešena jako kvadratický optimalizační problém, to už je však nad rámec této práce.

Klasifikace neznámého vektoru $\mathbf{x}$ ve fázi detekce je provedena následující klasifikační funkcí.

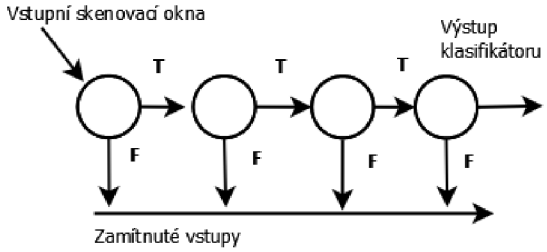
$$f(\mathbf{x}) = \mathbf{w} \times \mathbf{x} + b \tag{1.29}$$

Výsledkem klasifikace je číslo, které je buď větší nebo rovno nule, v tom případě jde o pozitivně zařazený vzorek. V případě kdy je výstup menší jak nula, je vstupní vzorek klasifikován jako negativní.

1.3.2 Kaskáda klasifikátorů

Princip kaskády klasifikátorů je založen na spojování slabých klasifikátorů do jednoho silného.[16][7] Příkladem slabého klasifikátoru může být odezva jednoho Haarova příznaku nebo jednoho HOG descriptoru. Ani jeden z nich však nedokáže sám o sobě úspěšně klasifikovat vstupní obraz. Uspokojivých výsledků je dosaženo až v okamžiku, kdy je k dispozici N těchto příznaků. Jedná se tedy o situaci kdy N slabých klasifikátorů vytváří silný klasifikátor.

Klasifikace příznakového prostoru, který dosahuje vysoké dimenze je výpočetně i časově drahá záležitost. Tento problém se projeví zejména při klasifikaci negativních vzorků, kdy je vyhodnoceno všech N klasifikátorů. Kaskáda těží z principu slabých klasifikátorů, to znamená, že klasifikátory jsou rozčleněny do sekcí a jsou vyhodnocovány postupně. Sekvenční vyhodnocování umožňuje ukončit klasifikaci na



Obr. 1.15: Kaskáda klasifikátorů

jakémkoliv stupni kaskády a to v okamžiku kdy je klasifikován negativně. Každý stupeň kaskády obsahuje více klasifikátoru než předchozí, takže přesnost predikce je postupně zvyšovaná.

Pro zrychlení klasifikace se hodí zejména sériová kaskáda, další typy kaskád se využívají pro vícetřídní klasifikaci, někdy se označuje jako vektorová klasifikace. Příkladem vícetřídní klasifikace může být například úloha, která má rozhodnout zda je na obrázku člověk a z jakého profilu je vyfocen. Pro takový případ bude vhodná serioparalelní kaskáda, která bude mít na svém výstupu vektor určující zařazení objektu do subtřídy.

1.3.3 Boosting

Boosting je metoda založená na kombinování klasifikátorů se špatnou úspěšností do výsledného kvalitního klasifikátoru. Výsledný klasifikátor$H(x)$ je možné zapsat jako lineární kombinací slabých klasifikátorů $h_{\mathrm{t}}(x)$

$$H(x) = sign(\sum_{t=1}^{T} \alpha_{\mathrm{t}} h_{\mathrm{t}}(x)) \tag{1.30}$$

Trénovací data pro tento algoritmus jsou definována jako uspořádaná dvojice učebního vzoru a zařazení do klasifikační třídy. Formální definice může vypadat například následovně $\{(x_1, y_1), ..., (x_{\mathrm{m}}, y_{\mathrm{m}})\}, x_{\mathrm{i}} \in X, y_{\mathrm{i}} \in Y$

Klasifikátor je matematický stroj, který vzoru ze vstupní množiny přiřazuje prvek z výstupní množiny $h : X \to Y$. Pokud je uvažována binární klasifikace, obsahuje výstupní množina pouze dva prvky. $Y = \{-1, +1\}$

Slabý klasifikátor bývá často reprezentován jednoduše a rychle získatelnou hodnotou. Příkladem může být prahová hodnota příznaku, ale dá se použít i výstup neuronové sítě, nebo rozhodovací strom. Možnosti výběru klasifikátoru jsou rozsáhlé. Jediným omezujícím kritériem je, aby chyba slabého klasifikátoru byla na trénovací množině menší než 50%.

Nejznámější zástupce skupiny algoritmů založených na principu boostingu je Adaboost. Původní verze Adaboost prováděla pouze binární klasifikaci, v současnosti

existují modifikované varianty umožňující klasifikaci reálným výstupem. Reálný výstup vykazuje míru příslušnosti do dané třídy, tato vlastnost vede většinou ke konstrukci přesnějších klasifikátorů. Příkladem takové modifikace je například algoritmus Real AdaBoost. V této práci je využíván algoritmus Adaboost, jeho funkce je definována následovně:

- Vytvoří slabý klasifikátor, který minimalizuje chybu na trénovacích datech. Tyto data jsou vážena pomocí váhovací funkce $D_{\mathrm{t}}(i)$.
- Pro klasifikátor vytvořený v předchozím kroku je proveden výpočet váhy α. Hodnota této váhy je závislá na chybě klasifikátoru skrze trénovací množinu. Tato váha se uplatňuje při lineární kombinaci tvořící výsledný klasifikátor.
- Po výpočtu váhy příznaku následuje převážení trénovacích dat distribuční funkcí $D_{\mathrm{t}}(i)$. U dobře klasifikovaných vzorů dojde ke snížení váhy a u chybně klasifikovaných je váha zvýšena. Smyslem této operace je, aby v byl v každém kroku vybrán jiný klasifikátor.

Podrobnou představu o dění uvnitř algoritmu Adaboost představuje následující pseudokód.

Algoritmus: *Adaboost*

Vstup : uspořádaná dvojice $(x_1, y_1), ..., (x_{\mathrm{m}}, y_{\mathrm{m}})$; $x_{\mathrm{i}} \in X, y_{\mathrm{i}} \in \{-1, 1\}$

Výstup: silný klasifikátor $H(x) = sign(\sum_{t=1}^{T} \alpha_{\mathrm{t}} h_{\mathrm{t}}(x))$

1 *inicializace vah* $D_1(i) = 1/m$

2 **foreach** $t = 1, \ldots, T$ **do**

3 *najdi klasifikátor* $h_{\mathrm{t}} = \min(\epsilon_{\mathrm{j}}); \epsilon_{\mathrm{j}} = \sum_{t=1}^{m} D_{\mathrm{t}}(i) I[y_{\mathrm{i}} \neq h_{\mathrm{j}}(x_{\mathrm{i}})]$

4 *If* $\epsilon_{\mathrm{t}} \geq 0.5$ *then konec*

5 $\alpha_{\mathrm{t}} = 0,5 \log \frac{1-\epsilon_{\mathrm{t}}}{\epsilon_{\mathrm{t}}}$

6 *aktualizace vah* $D_{\mathrm{t+1}}(i) = \frac{D_{\mathrm{t}}(i) \exp(-\alpha_{\mathrm{t}} y_{\mathrm{i}} h_{\mathrm{t}}(x_{\mathrm{i}}))}{\sum_{i=1}^{m} D_{\mathrm{t}}(i) \exp(-\alpha_{\mathrm{t}} y_{\mathrm{i}} h_{\mathrm{t}}(x_{\mathrm{i}}))}$

7 **end**

8 *Vytvořen silný klasifikátor* $H(x) = sign(\sum_{t=1}^{T} \alpha_{\mathrm{t}} h_{\mathrm{t}}(x))$

9 **return** $H(x)$

Algoritmus Adaboost dokáže lineární kombinací jednoduchých klasifikátorů vytvořit nelineární klasifikátor. Významným plusem této metody je, že je použitelná s širokým spektrem příznaků a neinklinuje k přetrénování jako například neuronové sítě. Určitým problémem může být, že Adaboost není příliš invariantní vůči šumu v trénovacích datech.

1.3.4 Hodnocení kvality modelu

Významnou součástí procesu získávání znalostí je vyhodnocení relevantnosti nalezených znalostí. Důležitým ukazatelem predikujícím kvalitu nalezených dat jsou charakteristiky vykazující míru naučenosti a schopnost zobecňovat.

V této práci jsou pro trénování modelu využívány metody učení s učitelem. Princip je založen na tom, že učícímu se algoritmu jsou předkládaná trénovací data se známým zařazením do klasifikační třídy. Učící se algoritmus pak iterativním způsobem hledá optimální klasifikační model. Míra naučenosti takového modelu se pak posuzuje v rámci úspěšnosti klasifikace na trénovacích a testovacích datech. Použití trénovacích dat pro testování má nižší vypovídací schopnost než testování na datech. Při testování na trénovacích datech se vystavujeme riziku, že neodhalíme přeučený model. Přeučený model vykazuje na trénovacích datech velmi dobré výsledky, ale díky ztrátě schopnosti zobecňovat na nezávislých datech propadne. Nejčastěji používané přístupy pro testování je možné rozdělit následovně.

- Testování v celých trénovacích datech. Jak už bylo zmíněno výše, není vhodné spoléhat se pouze na testy provedené na trénovací sadě.
- Křížová validace je metoda při níž jsou trénovací data rozdělena do x skupin, přičemž $\frac{1}{x}$ dat se použije pro testování a zbylých $x - \frac{1}{x}$ je použito pro trénování. Celý tento proces je zopakován x-krát.
- Metoda leave-one-out je velmi podobná předchozí metodě. Rozdílem je, že jeden příklad je použit pro testování a zbytek dat pro učení. Cyklus je zopakován tolikrát, kolik je trénovacích dat.
- Metoda bootstrap je opět velmi podobná křížové validaci, hlavním rozdílem je, že stejná data mohou být pro učení vybrána opakovaně. Podle velikosti trénovací sady vybere tato metoda přibližně 63% dat pro učení a 37% dat pro učení.
- Testování ne testovacích datech představuje v mnoha případech nejvhodnější způsob ověření modelu. Problém vyvstává pokud je nedostatek dat, nebo pokud je obtížné stanovit přípustnou míru podobnosti mezi trénovacími a testovacími daty.

Výstupem celého testovacího procesu je zjištění v kolika případech klasifikátor data zařadil do správné třídy a kolikrát selhal. Tyto informace se nejčastěji interpretují pomocí matice záměn. Sloupce představují výsledky klasifikace modelem a řádky nesou správné informace od učitele. Následující tabulka 3.1 představuje ukázku takové matice záměn pro binární klasifikaci. Správně pozitivní klasifikace je značena jako TP a obdobně správně negativní je značena TN. Chybně pozitivní klasifikace je označena FP a chybně negativní klasifikace je FN.

Tab. 1.1: Matice záměn

	Klasifikace modelem	
Správná klasifikace	Pozitivní	Negativní
Pozitivní	TP	FN
Negativní	FP	TN

Z matice záměn poznáme především kolik testovacích dat bylo klasifikováno správně a kolik chybně. Informace v matici záměn pomohou stanovit zda například model inklinuje k pesimistickým nebo optimistickým předpokladům. Důležitost této informace je závislá na řešené úloze. Cílem úlohy je například detekovat všechny objekty pozitivní třídy a výskyt falešně pozitivních výsledků není považován za zásadní problém. V takovém případě je vhodně vytvořit právě optimistický model. Pro minimalizaci falešně pozitivních detekcí směřujeme k pesimistickému modelu. Z parametrů matice záměn lze vypočítat charakteristiky, které dávají jasnější představu o chování modelu. Několik z nich je přiblíženo na následujících řádcích.

- Správnost (Acc) podává informaci o tom, jak přesně byly klasifikovány jednotlivé třídy.

$$A_{\mathrm{cc(positive)}} = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}} + N_{\mathrm{FP}}} \tag{1.31}$$

$$A_{\mathrm{cc(negative)}} = \frac{N_{\mathrm{TN}}}{N_{\mathrm{TN}} + N_{\mathrm{FN}}} \tag{1.32}$$

- Úplnost (recall) je charakteristika, která říká kolik z hledaných objektů bylo úspěšně nalezeno.

$$R = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}} + N_{\mathrm{FN}}} \tag{1.33}$$

- Souhrnná charakteristika, která dává do souvislosti správnost pro danou třídu a úplnost se nazývá F-míra(F-measure).

$$F = \frac{2N_{\mathrm{TP}}}{2N_{\mathrm{TP}} + N_{\mathrm{FP}} + N_{\mathrm{FN}}} \tag{1.34}$$

- Následující dvě charakteristiky pocházejí z oblasti medicíny jsou to Sensitivita a Specificita. Sensitivita je ve skutečnosti jiné pojmenování pro úplnost. Specificita v medicíně sleduje, zda daný model je skutečně jen na konkrétní nemoc. To znamená, v případě detekce objektů v obraze, že skutečně vybírá jen ty objekty, na které byl naučen.

$$S_{\text{sen}} = \frac{N_{\text{TP}}}{N_{\text{TP+FN}}} \tag{1.35}$$

$$S_{\text{spe}} = \frac{N_{\text{TN}}}{N_{\text{TN+FP}}} \tag{1.36}$$

2 IMPLEMENTACE

2.1 Natočené Haarovy příznaky

Haarovy příznaky jsou užity v detektoru objektů Viola Jones [9], kde spolu s integrálním obrazem a učícím algoritmem AdaBoost umožňují rychlou a přesnou detekci. Jejich výhodou je, že jsou snadno vyčíslitelné a výstupem je jednoduchá číselná odezva definovaná vztahem 1.6. Na základě této odezvy je Adaboost schopen vybrat optimální příznaky napříč celou trénovací sadou.

Výchozí množina příznaků z nichž se vybírá je na obrázku 1.5 označena jako přímé příznaky. Každý z těchto příznaků má pevně definované rozměry podle článku [14] a je umístěn v levém horním rohu okna o rozměrech 20×20. Z každého příznaku je operacemi posunu (po ose x a y) a změnou měřítka rozměrů (zvětšováním v původním poměru) vytvořena podmnožina příznaků. Problémem přímých příznaků je, že nedokáží adekvátně rozlišit natočené objekty. Z tohoto důvodu byly v práci [8] představeny příznaky natočené o 45 stupňů viz obrázek 1.5.

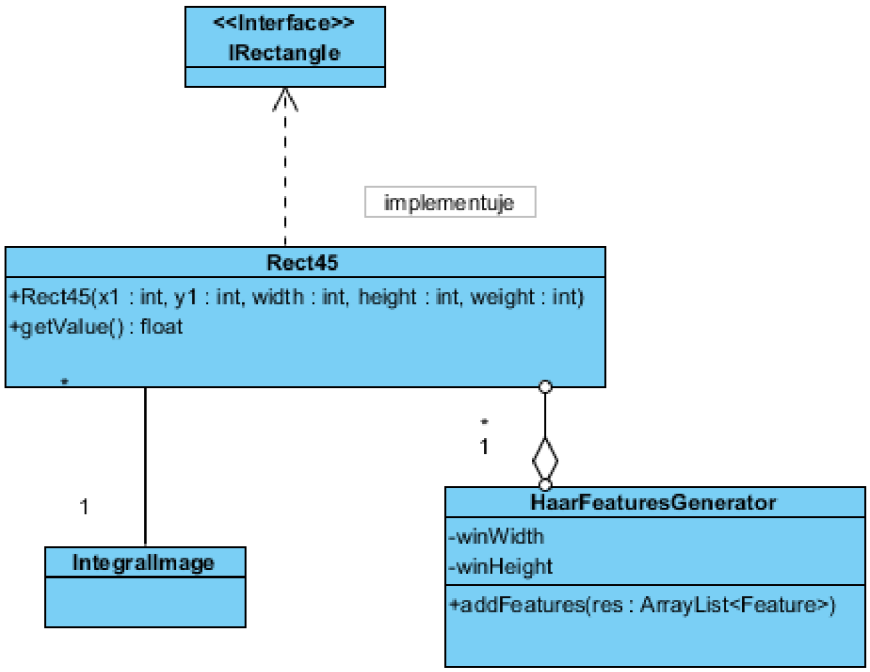
Implementací natočených příznaků do stávajícího Viola Jones detektoru přítomného v Rapidmineru (vizuální nástroj pro dolováni dat) lze dosáhnout úspěšnější detekce na datech s vyšší variabilitou než tomu tak je pouze s přímými příznaky. Implementace natočených příznaků se skládá z následujících kroků:

- Imlementace generátoru příznaků.
- Implementace natočeného integrálního obrazu viz vztah 1.3.
- Implementace odezvy natočeného příznaku,kterou lze získat vztahem 1.4.

Model tříd realizovaného systému je zobrazen na obrázku 2.1. O generování množiny příznaků se stará objekt třídy HaarFeaturesGenerator. Ta nejprve vytvoří příznak ve své základní podobě v levém horním okraji okna o rozměrech 20×20 a následným posouváním a změnou měřítka vytvoří jednotlivé podmnožiny pro všechny základní příznaky. Jednotlivé příznaky jsou charakterizované svým umístěním (pozice na souřadnicích x a y), rozměry stran (šířka, výška) a nakonec hodnotou odezvy.

Pro reprezentaci výše zmíněných vlastností byla vytvořena třída Rect45.Díky metodě *getValue()* je každý příznak tvořený objektem třídy Rect45 schopen na vyžádání odevzdat hodnotu své odezvy. Vzhledem k faktu, že detektor využívá natočené i přímé příznaky, je třeba pro ně použít různě implementované metody *getValue()*. Z tohoto důvodu bylo vytvořeno rozhraní IRectangle, které pak třídy Rect45 a Rect(přímé příznaky) implementují. Celá množina takto vygenerovaných příznaků je uložena v objektu HaarFeaturesGenerator v datové struktuře typu ArrayList.

Pro rychlé získání odezvy příznaku je využit integrální obraz viz 1.3. Výpočet integrální obrazu je implementován ve třídě IntegralImage.



Obr. 2.1: Generátor natočených příznaků

2.2 Histogram orientovaných gradientů

V teoretickém rozboru bylo zmíněno, že histogramy orientovaných gradientů se využívají pro popis objektů pomocí gradientů hran. Výstupem příznaku HOG je histogram popisující distribuci gradientu v lokalitě vymezené příznakem. Původní metoda HOG provádí seskupení těchto příznaků pomocí vzájemně se překrývajících bloků do deskriptoru. Deskriptor tedy popisuje celý obrázek a z tohoto důvodu dosahuje vysokého počtu příznaků.

Podrobný rozbor algoritmu je uveden v pseudokódu 1 a na obrázku 1.11, v této části budou uvedeny pouze informace nezbytné pro pochopení implementace.
Implementace metody HOG se skládá z následujících kroků:

- Výpočet úhlu a modulu gradientu pro jednotlivé body vstupního obrazu.
- Diskretizace úhlu gradientu do odpovídajícího binu (nejčastěji se volí rozsah 9 binů pro 180 stupňů).
- Seskupení pixelů do buněk (šířka 8 pixelů, výška 16 pixelů).
- Vytvoření histogramů pro jednotlivé buňky.

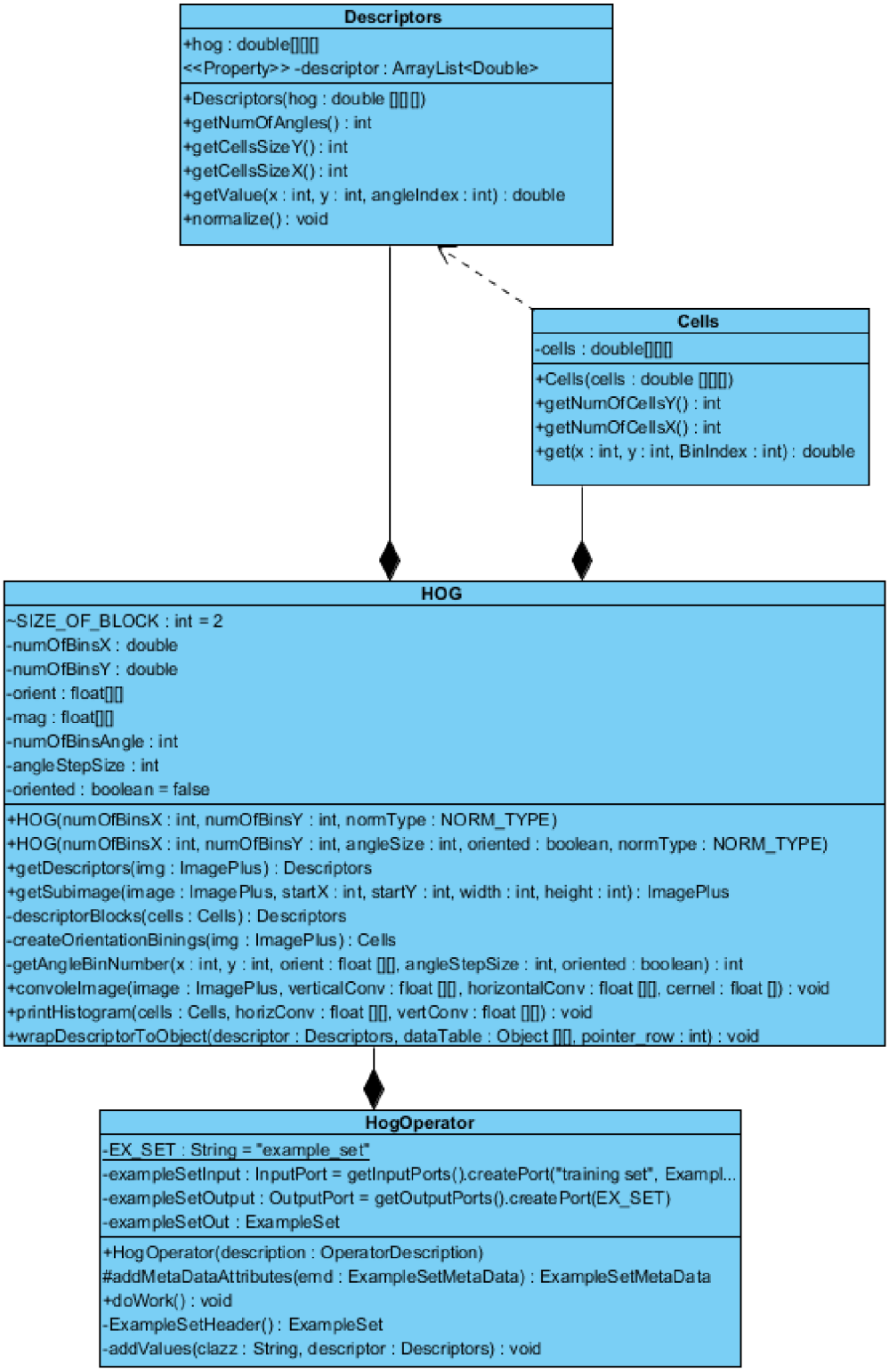
- Seskupení buněk do bloků a vytvoření výstupních deskriptorů.
- Zapouzdření metody HOG do operátoru vhodného pro aplikaci Rapidminer.

Model tříd popisující realizaci HOG je zobrazen na následujícím obrázku 2.2. Základem celého řešení je třída HOG, její zodpovědnost se zaměřuje především na výpočet gradientu (pro každý vstupní pixel) a na diskretizaci úhlu gradientu do jednotlivých binů. Pro výpočet gradientu je nejprve nutné provést konvoluci vstupního obrazu s jádrem [1,0,-1] v horizontálním a poté s transponovaným jádrem ve vertikálním směru. Tato funkcionalita je zajištěna metodou *convoleImage*, která na vstupu přijímá zpracovávaný obrázek (Image), reference na pole obsahující výsledek horizontální a vertikální konvoluce (horizConv a vertConv) a nakonec pole definující konvoluční jádro (cernel). Diskretizaci úhlu gradientu do jednotlivých binů provádí metoda *createOrientationBinings*. Dalším důležitým úkolem třídy HOG je, že udržuje asociativní vazby na třídy Descriptor a Cells.

Objekt třídy Cells reprezentuje buňky, které jsou rozvinuty nad původními obrazovými body. Síť buněk je reprezentovaná třídimenzionálním pole, kde první dvě dimenze určují souřadnice buňky v rovině a třetí dimenze odpovídá histogramu pro danou buňku. Histogram je vytvořen tak, že jednotlivé indexy pole odpovídají binu a hodnota uložená na odpovídajícím indexu (binu) je modulem gradientu právě pro tuto orientaci úhlu. Ve třídě Cells je důležitá metoda *get*, která na požádání vrací modul gradientu o orientaci určené indexem binu na souřadnicích buňky(x,y).

Objekt třídy Descriptor slouží pro vytvoření výsledného deskriptoru z objektu třídy Cells. Hlavní úkol metod implementovaných v této třídě spočívá v rozvinutí vzájemně se překrývajících bloků nad vstupní množinou buněk. Uvnitř těchto bloků je nutné provést normalizaci histogramu z důvodu různého kontrastu vstupního obrazu. Pro normalizaci je využívaná L1 (vztah 1.16) nebo L2 (vztah 1.17) normalizace v závislosti na uživatelském nastavení. Ze všech takto normalizovaných bloků a ze všech histogramů buněk do nich náležících je složen reprezentativní vektor popisující vstupní obrázek. Tento vektor je tedy výstupem deskriptoru a slouží jako vstup pro klasifikační (učící) algoritmus.

Přínosem tohoto řešení je implementace metody HOG do prostředí Rapidminer, kde může být jako operátor kombinována s dalšími metodami zpracování obrazu nebo dolování dat. Výchozím krokem pro vytvoření operátoru je zavedení agregačního vztahu mezi třídou HOG a třídou HogOperator. Ta slouží jako prostředník mezi ostatními operátory a vnitřním kódem třídy HOG.



Obr. 2.2: Schéma HOG

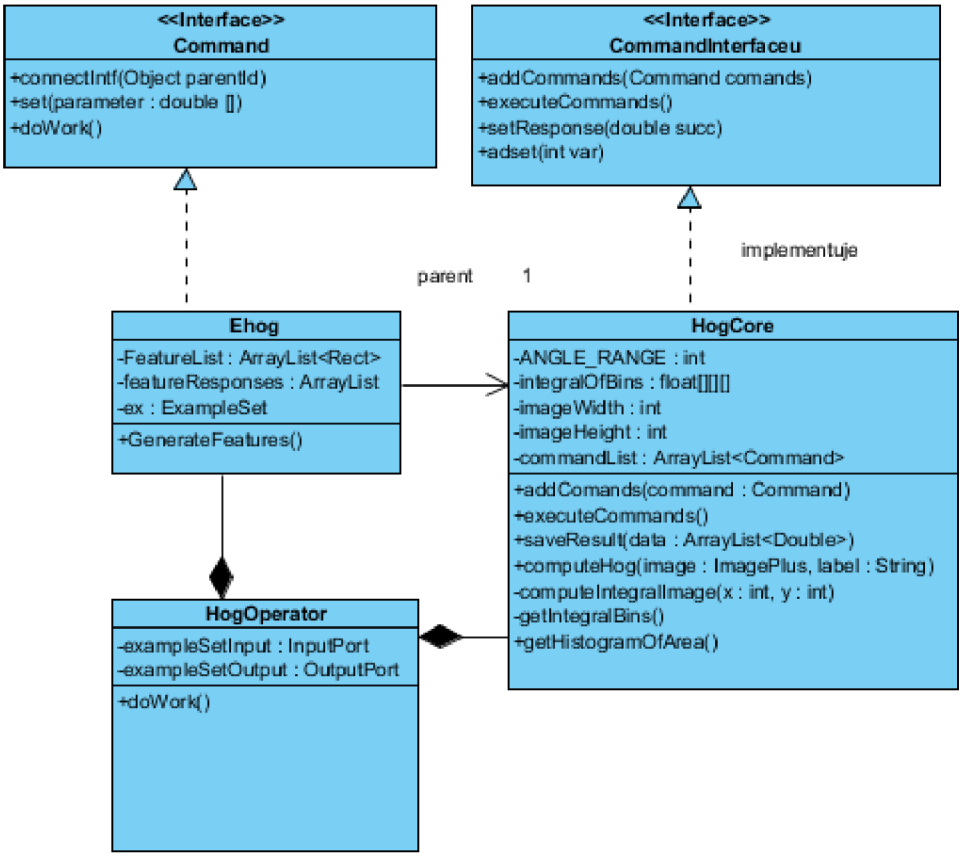
2.3 Modulární HOG

Původní objektový návrh metody HOG se ukázal být pro další modifikace nedostatečně flexibilní. Důvodem k modifikaci původního algoritmu je například fakt, že metoda opomíjí jakoukoliv heuristiku pro výběr oblastí zájmu a namísto toho počítá výsledný deskriptor pro celý obraz. Jakákoliv úprava vyžadovala zásadní zásah do celého kódu což vedlo ke vzniku chyb a nízké produktivitě. Nový návrh (viz obrázek 2.3) zanechává z původního konceptu pouze výpočet úhlu a modulu gradientu pro každý obrazový bod. Tuto funkcionalitu řeší metoda *computeHog* ve třídě HogCore, která jako vstupní parametr přijímá aktuální obraz. Operace jako rozdělení na bloky a na buňky jsou v jádře tohoto systému vynechány.

Architektura vychází z návrhového vzoru Command. Myšlenka vzoru Command spočívá v zanechání minimálního nutného kódu v jádře třídy (třída HogCore) a vytvoření rozhraní, které umožní rychlé přidávání další funkcionality. Zde slouží pro rozšiřování rozhraní Command. Na schématu 2.3 je vidět, že rozhraní předepisuje pouze tři nutné metody. Metoda *connectIntf* přijímá jako parametr odkaz na nadřízený objekt (objekt třídy HogCore) a slouží pro připojení metod nadřízeného objektu. Metoda *set* slouží pro nastavení vstupních parametrů. Nejdůležitější metodou je *doWork*, ve které je skryta vlastní funkcionalita.

Například třída Ehog vytváří deskriptor z buněk proměnlivých velikostí. Třída HogCore implementuje rozhraní CommandInterface,které umožňuje přidávat a spouštět objekty implementující
rozhraní Command. V souhrnu to tedy vypadá tak, že objekt Ehog je metodou *addCommands* přidán do objektu HogCore, kde čeká až bude spuštěn. Tímto způsobem je možné za sebe sériově zapojovat další rozšiřující funkce. Pokud tedy třída Ehog slouží k výběru buněk, je možné implementací rozhraní Command vytvořit další objekt, který bude vybrané buňky ohodnocovat validační funkcí. Přínosy tohoto návrhu jsou shrnuty v následujících bodech.

- Pro každou orientaci gradientu je spočítán samostatný integrální obraz přes celý obrázek. V případě rozlišení na devět binů je vytvořeno devět integrálních obrazů. Tato modifikace značně zrychluje vytváření histogramu a je zcela nezávislá na velikosti zpracovávané oblasti.
- Umožňuje snadnou a rychlou rozšiřitelnost stávajícího algoritmu.
- Separováním funkcí je snadnější provádět automatizované testy funkčnosti.



Obr. 2.3: Modulární HOG

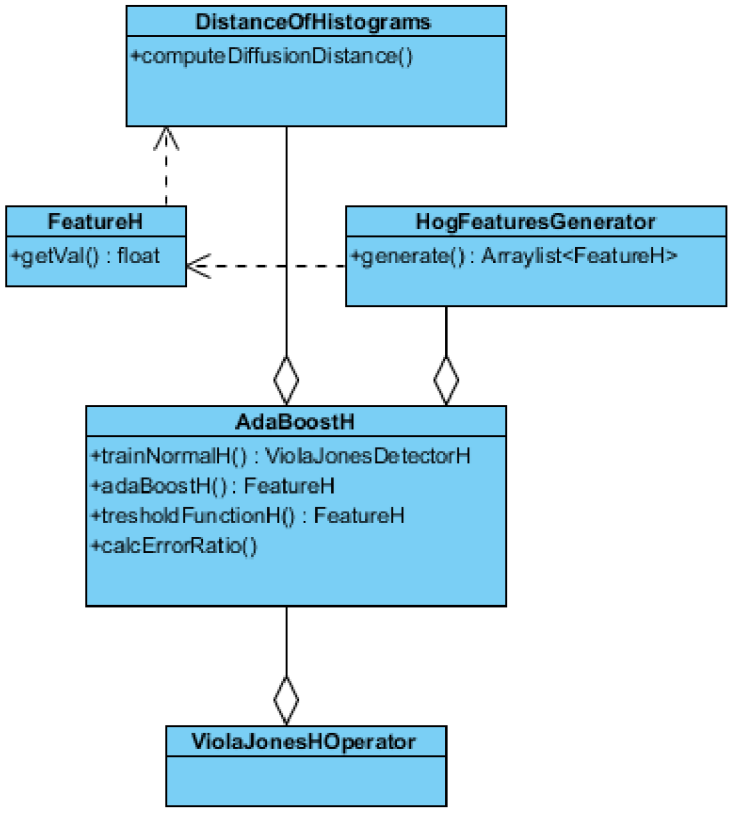
2.4 Příznaky Diffusion HOG

Výše zmíněné metody extrahují histogramy popisující hrany z celého vstupního obrazu. Takto získaný popis vykazuje vysokou dimenzi příznakového prostoru a zahrnuje i oblasti, které nemusí být pro následnou detekci objektu důležité. Zjednodušeně řečeno předchozí koncepty fungují na principu extrahuj histogramy z celého vstupního obrazu a vlož je do učícího se algoritmu (SVM). Výsledkem takového procesu je klasifikační model, který je sice komplexní, ale při následné detekční úloze pomalý. Ve snaze zrychlit proces detekce je v následujícím textu navrhnut koncept využívající algoritmů Adaboost a Diffusion distance. Zjednodušený princip vypadá následovně:

- Nejprve je vygenerována množina příznaků, kdy každý příznak obsahuje dvě pravoúhle oblasti jako na obrázku 1.13. Bloky těchto příznaků jsou generovány s různým rozměrem i počátečním umístěním. Cílem je získat co nejrozsáhlejší množinu vstupních příznaků.
- Vygenerované příznaky spolu s množinou trénovacích obrazů je předána ke

zpracování algoritmu Adaboost viz 2. Adaboost celé této množině příznaků stanoví klasifikační práh, na jehož základě je rozhodnuto do jaké třídy bude objekt zařazen. Klasifikační práh se stanovuje na základě odezvy příznaku, která je získaná jako rozdíl dvou histogramů (metoda Diffusion distance). V dalším kroku u každého příznaku spočítaná jeho klasifikační chyba. Pojem klasifikační chyba je vyjádření toho v jaké míře se příznak na celé trénovací množině mýlil. Posledním krokem jedné iterace je, že Adaboost z množiny příznaků vybere příznak, který má nejmenší chybu. Celý proces je prováděn do okamžiku než bude vybrán požadovaný počet příznaků nebo do okamžiku, kdy minimální chyba bude rovna čí větší hodnotě 0,5.

- Jednotlivé vybrané příznaky jsou za sebe sériově řazeny do stupňů kaskády. Princip kaskády viz 1.15 umožňuje vyhodnocovat klasifikátor po částech. Díky tomu lze dosáhnout rychlejší detekce, protože markantní část negativních vstupů je vyřazena hned na prvních stupních kaskády.
  Následující UML schéma nastiňuje zjednodušenou strukturu tříd implementující tuto metodu.



Obr. 2.4: Boosted HOG

2.5 Implementace do Rapidmineru

Rapidminer je softwarový nástroj pro dolování informací z dat. Značnou výhodou je, že vstupní data mohou nabývat různorodé povahy, takže jimi může být kurs akcií nebo obrazový či zvukový záznam. Tato variabilita je zajištěna díky rozšiřitelnosti pomocí pluginů, které do RapidMineru přidávají další funkce. Pracovní prostředí je vizuální a umožňuje mezi sebou propojovat různé funkční bloky, které jsou označovány jako operátory. Pod pojmem operátor je možné představit si standardizovanou obálku, která v sobě zapouzdřuje určitou metodu například HOG. Obálka slouží pro dodržení jednotné komunikace mezi jednotlivými operátory a Rapidminerem.

Při vytváření operátoru je nutné, aby nově vytvářený zdědil vlastnosti třídy Operator. Pro umožnění komunikace s okolními operátory je třeba nadefinovat vstupní a výstupní porty. Při jejich vytváření se musí určit jaký datový objekt mohou přijmout nebo odeslat. Pro přenos dat existují již definované třídy,nebo lze vytvořit vlastní, které slouží jako přepravky. Přiřazení přepravního objektu portu doprovází přidání metadat. Metadata popisují jaká data mohou být na port doručena, tento fakt se projeví pokud se propojí operátory přeposílající si nekompatibilní datové přepravky. V takovém případě RapidMiner varuje chybovým hlášením a po spuštění procesu dojde nejspíš k chybě.

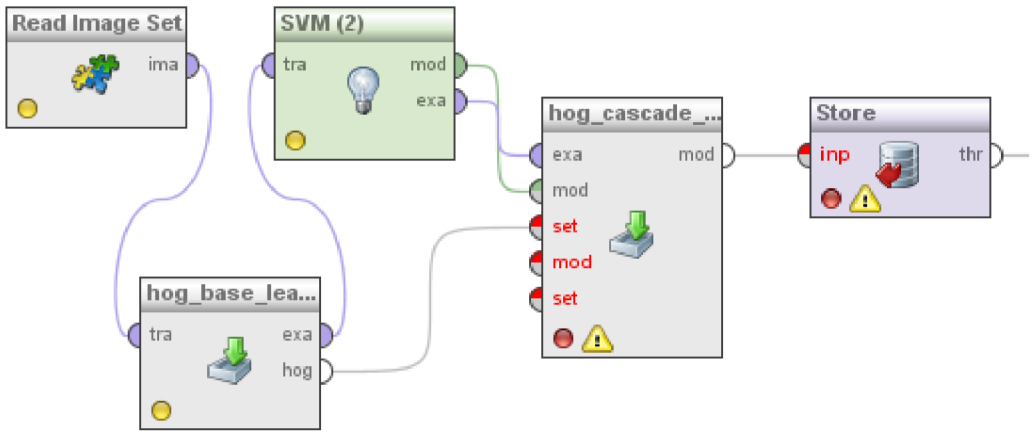
Důležitý a univerzální formát pro výměnu dat v rámci rapidmineru je reprezentován rozhraním ExampleSet. Tento datový kontejner je možné připodobnit k tabulce o několika vrstvách, první vrstva je logický kontext v němž se data zobrazují jako v běžné tabulce. Druhá vrstva (fyzický kontext) je tvořena binární reprezentaci dat a s představou tabulky nemá nic společného. Tento přístup je vhodný například pro řídká data, která mohou být na fyzické úrovni uložena mnohem úsporněji. Sloupce ExampleSetu jsou označovány jako atributy, každý je označen jménem a typem hodnot. Volitelně se přiřazuje tzv. role, tyto rozšiřující vlastnosti se hodí pro některé specializované operátory. Příkladem významné role je *label*, která říká, že hodnota takto označeného atributu specifikuje klasifikační třídu dat. Atribut označuje proměnou, která je ukládána do ExampleSetu. Řádek v ExampleSetu je označován jako Example a určuje jeden konkrétní vzorek dat. Například pro metodu HOG znamená jeden řádek v Examplesetu deskriptor pro jeden obrázek.

Na obrázku 2.5 je zobrazené schéma, které ze vstupní trénovací množiny vytvoří model schopný detekovat postavu člověka.Pro tuto úlohu byly vytvořeny operátory hog_base_learn a hog_cascade_operator. Operace prováděné jednotlivými operátory jsou následující:

- Operátor read_imageSet slouží k výběru množiny pozitivních a negativních obrázků. Každému z nich je přiřazena klasifikační třída do které náleží(true/false).

Na vstup následujícího operátoru je odeslán ExampleSet obsahující cesty k jednotlivým obrázkům a zařazení do klasifikační třídy. Je nutné poznamenat, že všechny obrázky musí mít stejné rozměry. Těmto rozměrům pak při následné detekci bude odpovídat velikost skenovacího okna.

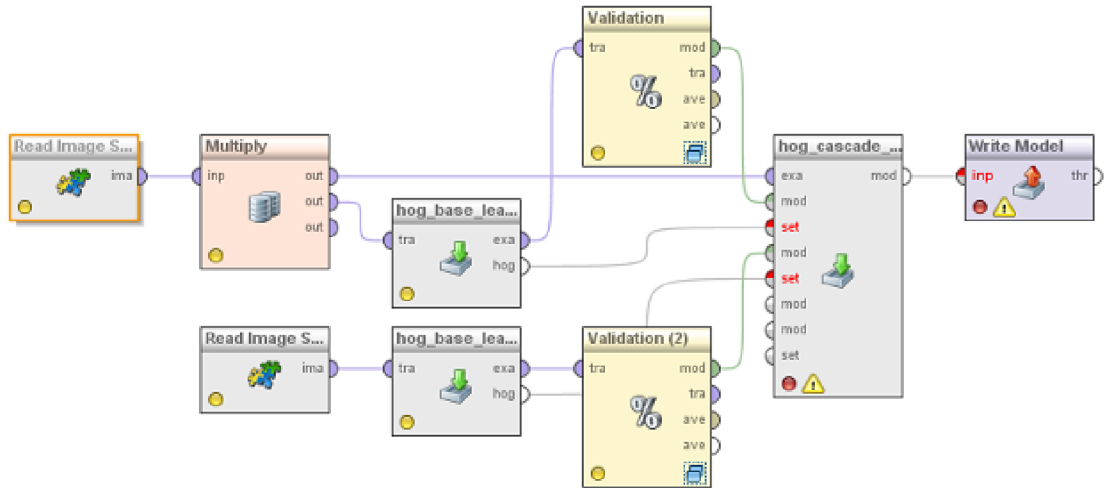
- Operátor hog_base_learn provede pro každý obrázek výpočet deskriptoru metodou histogramu orientovaných gradientů. Každý sloupec představuje jeden bin z histogramů obsažených v deskriptoru. Jeden řádek je Example a odpovídá deskriptoru celého obrázku.
- Operátor SVM je vektorový klasifikátor, který dokáže najít oddělující hyperplochu i mezi daty vysoké dimenze. Výstupem tohoto operátoru je klasifikační model, který v sobě obsahuje váhy jednotlivých podpůrných vektorů. Tento model je následně aplikován ve fázi detekce pro ohodnocení neznámého vstupního obrazu.
- Operátor hog_cascade_operator vytváří kaskádu klasifikátorů. Jeho účelem je řetězit za sebou jednotlivé modely a vytvářet z nich jeden klasifikátor. Smyslem takového přístupu je možnost kombinace různých klasifikačních metod. Podle potřeby je možné použít například neuronovou síť spolu s SVM či KNN. Dalším podstatným důvodem proč je vhodné za sebe řetězit modely je klasterizace vstupních dat. V případě rozsáhlé trénovací sady a obtížné klasifikační úlohy je obtížné najít optimální oddělující hyperplochu. Pokud se však trénovací data rozdělí do skupin s obdobnými parametry, dosáhne skupina modelů podstatně nižšího počtu falešných detekcí.



Obr. 2.5: Trénování modelu

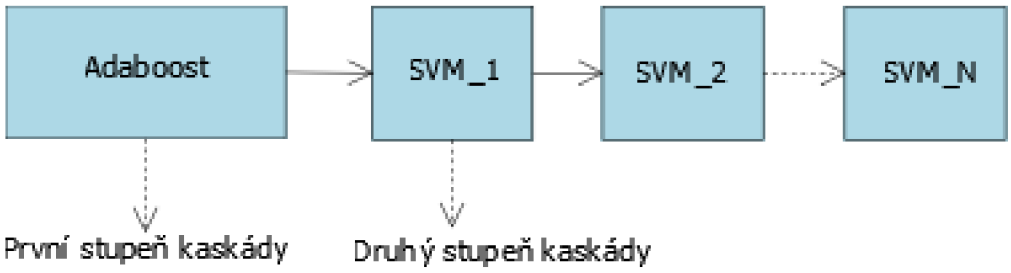
Schéma na obrázku 2.5 vytváří kaskádu pouze z jednoho modelu a výstup z op-

erátoru hog_base_learn je odesílán přímo do operátoru SVM. Pro vytvoření kvalitnějšího modelu je vhodné použít metodu crossvalidace. Princip této metody spočívá v tom, že trénovací data jsou rozdělována na trénovací a testovací sadu. Popsanou situaci zobrazuje schéma 2.6. Zde je ilustrováno vytvoření kaskády dvou SVM modelů a oproti 2.5 přibyl operátor Validation. Validation má v sobě umístěný operátor SVM a zajišťuje tak vytváření modelu metodou crossvalidace.



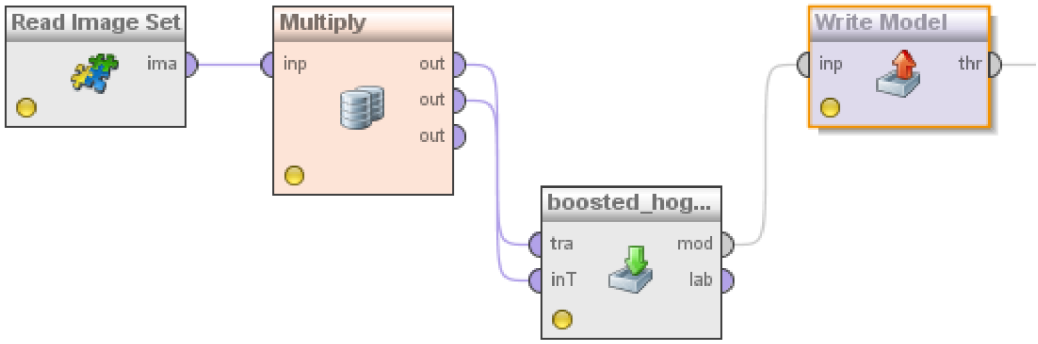
Obr. 2.6: Trénování kaskády modelů

Výše uvedená schémata pracují s implementací která je v této práci označena jako modulární HOG viz podkapitola Modulární HOG. Nedostatkem této metody byla především detekční doba, která v závislosti na velikosti vstupního obrázku a měřítku dosahovala několika minut. Z tohoto důvodu byl zpracován koncept nové metody, která využívá učícího algoritmu Adaboost a příznaků Diffusion distance (viz podkapitola Diffusion distance příznaky). Tato metoda oproti své předchůdkyni prokázala znatelné zrychlení detekční rychlosti (v průměru stokrát). Zvýšení detekční rychlosti však vedlo k zvýšení falešně pozitivních detekcí. V průběhu testů se ukázalo, že pro praktické použití je tato metoda sama o sobě nepoužitelná. Na základě těchto poznatků byl zpracován finální koncept, který vzájemně kombinuje oba předchozí přístupy. Model vytvořený Diffusion HOG se aplikuje na prvním stupni kaskády a vybere kandidáty na pozitivní klasifikaci. Tito kandidáti jsou následně validováni pomocí pomalejší metody. Výsledné zrychlení této hybridní metodu oproti původnímu modulárnímu HOG je v průměru třicetinásobné.Je nutné poznamenat, že model vytvořený pomocí Diffusion HOG je natrénován optimisticky, to znamená, že nesnižuje úspěšnost detekce následnému stupni kaskády.



Obr. 2.7: Schéma kaskády

Důležité je, že veškeré údaje o době detekce či trénování a samozřejmě úspěšnosti detekce jsou velmi závislé na trénovací sadě a na samotném nastavení procesu. Z tohoto důvodu jsou zde údaje uváděny spíše vágně a pro představu o chování zmíněných metod. Konkrétní údaje jsou prezentovány v kapitole Detekce objektů. Následující schéma zobrazuje zapojení pro vytvoření modelu Diffusion HOG.



Obr. 2.8: Trénování Diffusion HOG

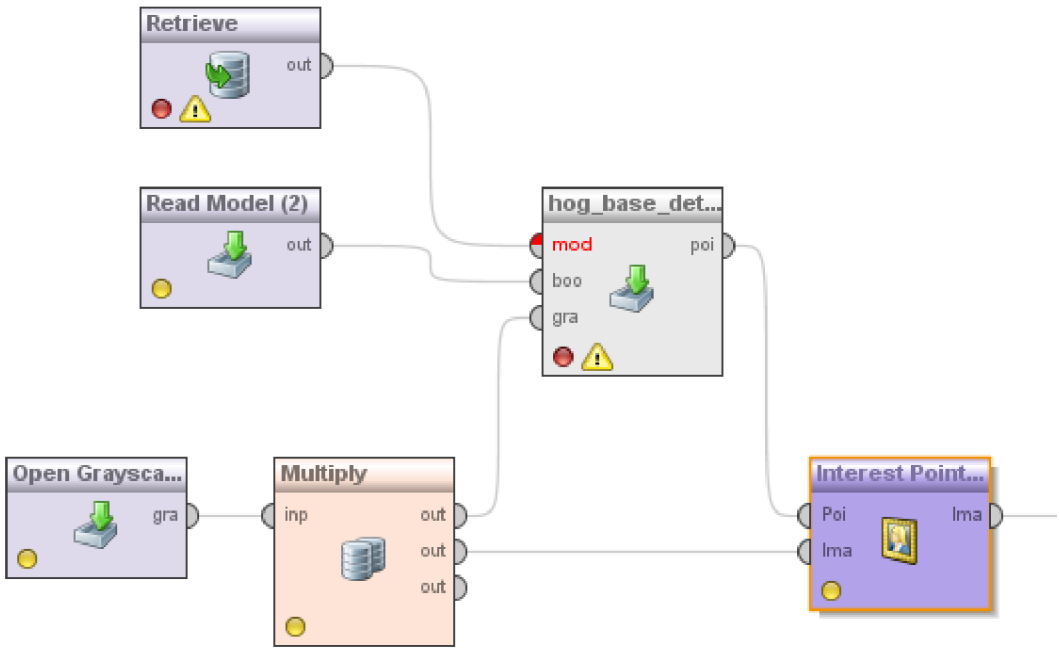
Důležitým vstupním parametrem pro tento operátor je textový soubor s popisem kaskády. Cesta k tomuto souboru se nastavuje v kontextovém menu po kliknutí na operátor. Popis kaskády definuje kolik příznaků bude obsaženo v jednotlivém stupni a jaké bude procentuální navýšení práhu kaskády. Nastavení kaskády významně ovlivňuje úspěšnost výsledného modelu.

K procesu trénování je vhodné na tomto místě poznamenat, že je velmi náročný na paměťovou kapacitu výpočetního stroje. Množství zkonzumované paměti se opět odvíjí od velikosti trénovací sady, při velikostech nad dva tisíce obrázků dosahuje až 3 GB paměti RAM.

Předchozí schémata prezentovala zapojení pro tvorbu klasifikačního modelu. Pro potřebu detekce k objektu byl vytvořen detektor, který je schopen aplikovat před-

chozí vytvořené modely na vstupní obraz.Na obrázku 2.9 je zobrazeno schéma klasifikace neznámého obrázku za účelem zjištění přítomnosti hledaného objektu. Pro tuto úlohu byl vytvořen vlastní operátor Detector_hog. Operace prováděné jednotlivými operátory jsou následující:

- Operátor OpenGrayscaleImage slouží k otevření šedotónového obrazu.
- Operátor ReadModel načítá klasifikační model (Diffusion HOG) vytvořený ve fázi trénování.
- Operátor Retreive načítá taktéž klasifikační model (HOG) vytvořený ve fázi trénování.
- Operátor Multiply jednoduše kopíruje vstup na několik výstupů.
- Operátor hog_base_detector sekvenčně prochází vstupní obraz skenovacím oknem (rozměry odpovídají rozměrům trénovacích dat), z každého výřezu okna je vypočítán HOG deskriptor. Ten je klasifikován pomocí naučeného modelu, který rozhodne o přítomnosti hledaného objektu. Vstupem tohoto operátoru jsou dva naučené modely. První model připojený na port mod je tvořený kaskádou lineárních SVM. Model připojený na port boo je vytvořen metodou Diffusion HOG. Detektor tedy může pracovat ve dvou režimech, v prvním jsou připojeny oba modely nebo ve druhém, kdy je připojen pouze SVM kaskáda.
- Operátor InterestPointsVisualizer slouží k vykreslení



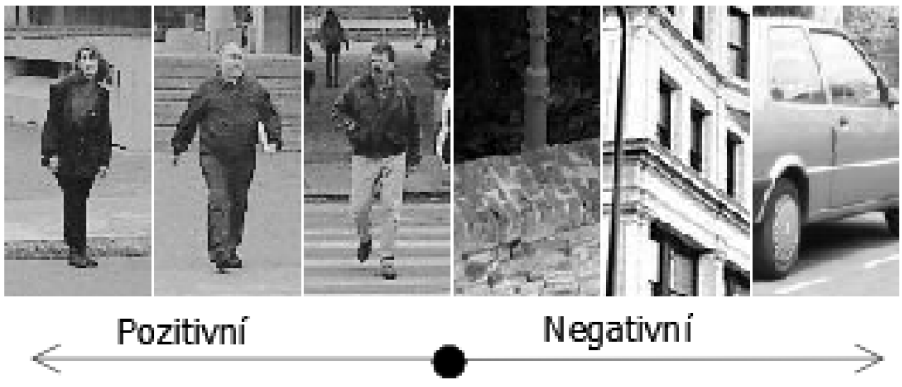
Obr. 2.9: Detektor HOG

3 DETEKCE OBJEKTŮ

V této části práce budou prezentovány některé zajímavé výsledky, kterých bylo dosaženo při praktickém ověřování implementovaných algoritmů. Praktické ověření spočívalo ve vytvoření detektoru postav a tepen. Obě úlohy jsou dostatečně odlišné na to, aby se dalo uvažovat o univerzálnosti popisovaných metod. Úloha detekce chodců se oproti detekci tepen vyznačuje velkou variabilitou okolního prostředí. U detekce tepen se naopak výrazněji projevuje šum. Celkové shrnutí výsledků a poukázání na nedostatky je provedeno v podkapitole Diskuze výsledků.

3.1 Detekce chodců

Pro trénování modelu rozpoznávajícího postavy byl použit dataset MIT pedestrian dataset, který obsahuje 924 obrázků chodců o rozměrech 64×128 pixelů (šířka a výška). Popředí je tvořeno centrovaně umístěnou postavou převážně z frontálního pohledu. Na pozadí jsou zachyceny artefakty typické pro město (části domů, silnice atd.). Původní trénovací sada byla kvůli usnadnění výpočtu převedena do šedotónové reprezentace. Sada negativních obrázků byla vytvořena z obrazového datasetu Inria, který obsahuje obrázky z prostředí města a přírody bez přítomnosti osob. Z náhodně vybraných negativních obrázků bylo vytvořeno 4770 výřezů o rozměru 64×128 , které jsou použity pro samotné trénování. Závěrečné otestování naučeného modelu je prováděno na testovacích obrázcích ze sady Inria persons. Testovací sada je tvořena 175 obrázky o rozměrech 64×128 kde je přítomna jedna postava. Další testovací sada je tvořena 64 obrázky kde se vyskytují různé počty postav v různém měřítku.



Obr. 3.1: Trénovací data

V kapitole věnované algoritmu HOG bylo uvedeno, že významný vliv na histogram získaný ze vstupního obrázku má především velikost buňky a velikost bloku. Nastavení těchto parametrů nelze zobecňovat. Pro různé detekční úlohy je třeba hledat jejich vhodné nastavení. Následující tabulka prezentuje míru naučenosti modelu v závislosti na velikosti buněk a bloků. Tyto data byla získána z procesu trénování za použití crossvalidace. Symbol P (precision) představuje správnost a jeho index definuje zda se jedná o pozitivní či negativní třídu. Spec je označením pro specificitu a obdobně Sen vyjadřuje sesitivitu. Fm je označení pro souhrnou míru F-measure. V některých tabulkách se vyskytují parametry úplnost a správnost, oba parametry jsou vztaženy k pozitivní třídě. Úplnost udává kolik procent relevantních objektů bylo nalezeno vzhledem k celé množině. Správnost říká, kolik procent z nalezených objektů je skutečně hledaný objekt.

Tab. 3.1: Parametry HOG (trénovací sada)

Buňky [px]	**Blok [buňky]**	$\mathbf{P_N}$ **[%]**	$\mathbf{P_P}$ **[%]**	**Spec**	**Sen**	**Fm**
6	3	99,13	98,71	98,51	99,24	98,98
6	5	98,74	97,65	97,27	98,92	98,29
8	3	98,88	98,39	98,14	99,03	98,71
8	4	98,25	97,85	97,52	98,48	98,16
8	5	96,53	96,76	96,28	96,97	96,87
12	2	98,62	97,65	97,27	98,81	98,23
16	2	95,34	96,72	96,28	95,88	96,31

Tab. 3.2: Parametry HOG (testovací sada)

Buňky [px]	**Blok [buňky]**	$\mathbf{P_N}$ **[%]**	$\mathbf{P_P}$ **[%]**	**Spec**	**Sen**	**Fm**
6	3	91,78	98,23	99,72	63,43	77,08
6	5	94,27	93,62	98,74	75,43	83,54
8	3	93,70	97,69	99,58	72,57	83,28
8	4	93,94	96,99	99,44	73,71	83,77
8	5	94,88	91,33	98,19	78,29	84,31
12	2	95,05	95,83	99,16	78,86	86,52
16	2	99,15	91,85	97,91	96,57	94,15

Pří výběru vhodného nastavení je nutné zvážit především přesnost klasifikátoru v dané třídě. Dalším důležitým faktorem je rychlost detekce. Z předchozích tabulek tento fakt sice není zřejmý, ale s větším počtem buněk a bloků stoupá detekční doba. Dále je důležité uvědomit si, že menší velikost buněk (tedy větší rozlišení) mnohdy

nevede k lepším výsledků. Menší buňky jsou více náchylnější na přítomnost šumu. Pro detekci postav se jako nejoptimálnější nastavení jevilo využívat buňky o velikosti osm pixelů a bloky složení ze tří buněk.

Kaskáda klasifikátorů byla zavedena pro snížení výskytu falešně pozitivních detekcí. Snížení falešně pozitivních detekcí zobrazuje následující tabulka. Pro snížení falešných detekcí byl zaveden ještě další mechanismus nazvaný minimální počet sousedů. Vychází z předpokladu, že v místě skutečného výskytu objektu dojde k mnohonásobnému výskytu detekcí. Počet výskytu těchto detekcí je samozřejmě závislý na kroku detekčního okna, které skenuje obrázek.

Tab. 3.3: Počet stupňů SVM kaskády

Stupňů kaskády	**Chodců**	$\mathbf{S_P}$	$\mathbf{F_P}$	**Úplnost** [%]	**Správnost** [%]	**Sousedů**
13	107	93	37	86,91	71,54	2
13	107	91	20	85,04	81,98	3
13	107	87	11	81,30	88,78	4
10	107	93	58	86,91	61,59	2
10	107	92	35	85,98	72,44	3
10	107	87	20	81,30	81,31	4
7	107	94	67	87,85	58,39	2
7	107	92	40	85,98	69,70	3
7	107	87	24	81,30	78,38	4

Následující tabulka oproti tabulce 3.4 prezentuje vliv minimálního počtu sousedů na detekci modelem Diffusion HOG. Oproti předchozí tabulce je na první pohled zřetelná vysoká míra falešně pozitivních detekcí. Míra falešně pozitivních detekcí je natolik vysoká, že se ji nedaří snižovat ani zvyšováním počtu minimálních sousedů.

Tab. 3.4: Příznaky Diffusion HOG - počet sousedů

Počet příznaků	**Chodců**	$\mathbf{S_P}$	$\mathbf{F_P}$	**Úplnost** [%]	**Správnost** [%]	**Sousedů**
61	107	93	211	86,91	30,59	3
61	107	93	211	86,91	30,59	4
61	107	93	211	86,91	30,59	5

Vzhledem k vysokému počtu falešných detekcí není metoda Diffusion HOG vhodná pro samostatné použití. Jejím hlavním plusem oproti použití HOG a SVM je několikanásobně vyšší detekční rychlost. Tohoto faktu bylo využito pro nasazení Diffusion HOG do prvního stupně kaskády. Dojde tak velice rychle k redukci prohledávaného

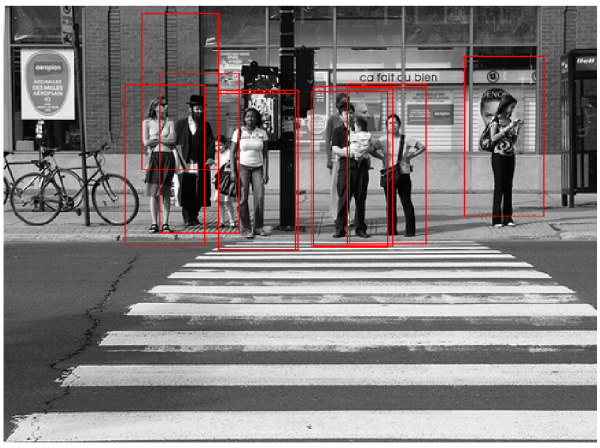
prostoru. Vybrané oblasti zájmu jsou následně validovány kaskádou modelů SVM. Tabulka 3.5 prezentuje míru zrychlení detekce díky tomuto hybridnímu uspořádání.

Tab. 3.5: Detekce Diffusion HOG + SVM

Rozlišení [pixelů]	$\mathbf{T_{SVM}}$ **[ms]**	$\mathbf{T_{Diff}}$ **[ms]**	**zrychlení**
640×480	82280	1076	76,47
391×293	10651	440	24,21
397×289	13519	457	29,58
621×466	139573	943	148,01
212×159	8337	269	30,99
276×207	14721	384	38,34
276×207	13834	602	22,98
256×192	27518	383	71,85
192×144	6189	602	10,28
218×164	9803	508	19,30

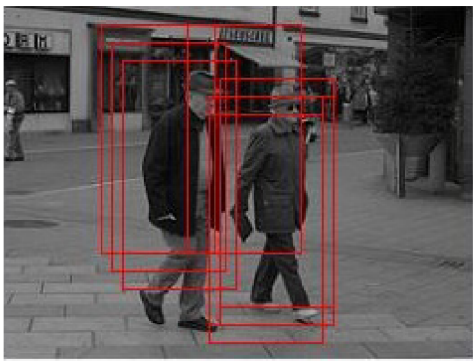
Z hodnot uvedených v tabulce je jasně vidět, že dojde k výraznému zrychlení detekce. Pokud provedeme zprůměrování hodnot udávajících nárůst detekční rychlosti, zjistíme, že došlo průměrně ke 47,20 násobnému zrychlení.

Posledním, ale neméně důležitým parametrem ovlivňujícím míru naučenosti modelu je nastavení parametrů SVM. Pokud je použito lineární SVM jedná se především o parametr C. Ten slouží u klasifikátoru SVM pro nastavení míry tolerance vůči špatně klasifikovaným vzorkům. Jinými slovy se dá říci, že čím je nižší hodnota C, tím je klasifikátor obecnější a tolerantnější vůči chybně klasifikovaným vzorkům. Je nutné poznamenat, že pro klasifikaci postav se při použití lineárního SVM osvědčilo používat malé hodnoty C. Doporučený interval se pohybuje od 0.03 do -1 včetně. V této práci bylo pracováno s hodnotou 0.01. Hodnota -1 specifikuje, že budou pro pozitivní a negativní třídu použita rozdílná C. V této práci bylo experimentováno i s nelineárními SVM jádry, jednalo se především o radiální a gaussovské. V žádném z testovaných případů se však nepodařilo dosáhnout lepších výsledků v úspěšnosti detekce. Další nevýhodou nelineárních jader byla velikost modelu a několikanásobně větší učební a detekční doba.



Obr. 3.2: Detekce

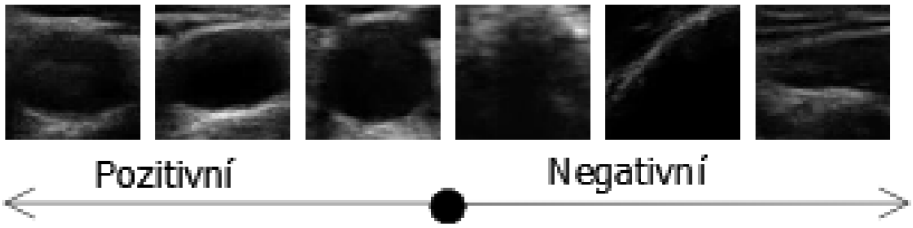
Obr. 3.3: Detekce vyhlídka



Obr. 3.4: Detekce

3.2 Detekce tepen

Pro úlohu detekce tepen byla trénovací data získaná z databáze UltrasonixTrain. Pozitivních trénovacích snímků bylo 284 a negativní množina sestávala z 903 snímků.



Obr. 3.5: Trénovací data

Následující dvě tabulky dávají do souvislosti výsledky naučeného modelu na trénovací a testovací sadě. První řádek tabulky 3.6 uvádí, že model má pouze jeden stupeň kaskády. To znamená, že se jedná o běžný monolitický model natrénovaný na celistvé sadě pozitivních a negativních snímků. Pokud bychom se zaměřili pouze na tabulku 3.6 mohlo by se zdát, že tento model naprosto vyhovuje a není třeba dalších stupňů. Ovšem hned druhým pohledem na tabulku 3.7 zjistíme, že model s jedním stupněm vykazuje 279 falešných detekcí. Tento příklad hezky ilustruje odlišné chování modelu na trénovacích a testovacích datech.

Tab. 3.6: Počet stupňů kaskády (trénovací sada)

Stupňů kaskády	$\mathbf{P_N}$ [%]	$\mathbf{P_P}$ [%]	**Spec**	**Sen**	**Fm**
1	99,89	97,25	99,11	99,65	98,43
5	96,83	65,08	84,61	91,20	75,95
9	89,95	72,90	92,14	67,25	69,96
13	89,67	100,00	100,00	63,38	77,59
16	89,67	100,00	100,00	63,38	77,59

Tabulka 3.7 ukazuje jak s postupný zvyšováním stupňů kaskády klesá počet falešných detekci a zároveň dochází ke snižování hodnoty úplnosti. Jinými slovy se dá říci, že čím je model přísnější, dochází ke ztrátám pozitivních objektů, které se vyskytují blízko rozhodovacího prahu. Základní snahou by tedy mělo být dosažní co nejvyšší a úplnosti a přesnosti.

Tab. 3.7: Počet stupňů SVM kaskády (skenovací okno)

Stupňů kaskády	**Tepen**	$\mathbf{S_P}$	$\mathbf{F_P}$	Úplnost [%]	**Správnost** [%]
1	82	82	279	100,00	22,71
5	82	80	243	97,56	24,77
9	82	79	30	96,34	72,48
13	82	78	7	95,12	91,76
16	82	78	3	95,12	96,30

Obdobné chování lze vysledovat i v následující tabulce 3.8, která zachycuje zvyšování počtu příznaků pro metodu Diffusion HOG. Stejně jako předešlá metoda dochází při zvyšování počtu příznaků k poklesu úplnosti a zvyšování správnosti. Zásadním problémem však je, že zatímco úplnost klesá relativně rychle, správnost stoupá pomalu. Například v tabulce 3.7 je pro 30 falešných detekcí úplnost rovna 96,34 procent a správnost 72,48 procent. Nejbližší počet falešných detekcí v druhé tabulce odpovídá 50. Tomuto řádku také odpovídá úplnost 89,05 procent a správnost 59,35 procent.

Tab. 3.8: Počet příznaků (Diffusion HOG)

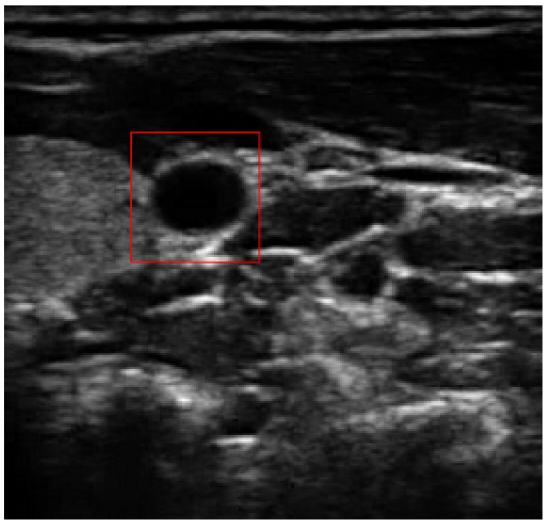
Počet příznaků	**Tepen**	$\mathbf{S_P}$	$\mathbf{F_P}$	Úplnost [%]	**Správnost** [%]
42	82	79	210	96,34	27,34
50	82	79	184	96,34	30,04
67	82	77	102	93,90	43,02
94	82	75	67	91,46	52,82
115	82	74	67	90,24	52,48
133	82	73	50	89,02	59,35

Tabulka 3.9 dává do kontextu dobu trénování a detekce v závislosti na použité metodě a na počtu stupňů kaskády. Z tabulky je zřejmé, že metoda Diffusion distance využívající algoritmus adaboost vyžaduje delší dobu pro trénovaní. Na druhou stranu doba nutná pro detekci je podstatně nižší.

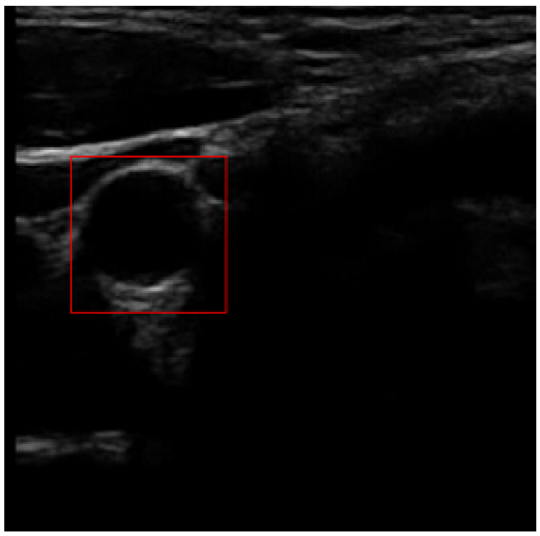
Tab. 3.9: Doba trénování/detekce

Model	$\mathbf{T_{tren}}$ **[min:s]**	$T_{\mathbf{det}}$ **[s]**	**Okno**	**Krok [px]**
SVM (16 modelů)	01:35	1,869	45×45	5
SVM (13 modelů)	01:06	1,424	45×45	5
SVM (9 modelů)	00:58	1,143	45×45	5
Adaboost(133 příznaků)	12:37	0,083	45×45	5
Adaboost(115 příznaků)	11:51	0,060	45×45	5
Adaboost(94 příznaků)	09:27	0,057	45×45	5
Adaboost(67 příznaků)	07:48	0,057	45×45	5
Adaboost(60 příznaků)	06:00	0,057	45×45	5
Adaboost(42 příznaků)	05:10	0,057	45×45	5

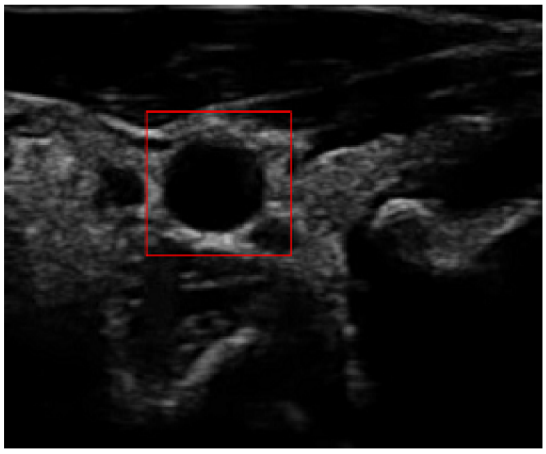
Následující obrázky jsou ukázkou detekce tepny. Zobrazené obrázky pochází z testovací sady na níž bylo prováděno experimentální ověření.



Obr. 3.6: Detekce



Obr. 3.7: Detekce



Obr. 3.8: Detekce

3.3 Diskuze výsledků

Metoda HOG dosahuje dobrých výsledků v různorodých úlohách detekce. Nejčastěji se jedná o detekci postav,automobilů a zvířat. Přestože původní metoda funguje relativně dobře, je zde široké pole možností pro zlepšování. V následujících bodech jsou shrnuty hlavní nedostatky a návrhy, ke kterým se dospělo během experimentování.

- HOG nevyužívá žádnou heuristiku pro určení významných bodů, ze kterých je vypočítán deskriptor. Jinými slovy to znamená, že výpočet se provádí vždy pro celý obrázek bez ohledu na fakt zda je buňka umístěna na oblasti zájmu nebo na pozadí. Tento nedostatek byl v této práci řešen využitím algoritmu

Adaboost a generováním příznaků, adaboost vybírá pouze relevantní příznaky. Dále se nabízí možnost předřadit samotnému učebnímu procesu metodu schopnou detekovat významné body. Příkladem takové metody může být například Hessian detektor. Jak by takové předzpracování mohlo vypadat je vidět na následujícím obrázku.

Obr. 3.9: Významné body

- Ve snaze vytvořit příznaky popisující změnu histogramu byla využita metoda Diffusion distance. Průběžné testy ukázaly, že tyto příznaky vykazují sklon k vytváření falešně pozitivních závěrů. Konstrukce těchto příznaků je tvořena dvěma pravoúhlými oblastmi z nichž jsou extrahovány histogramy. Jako řešení pro zvýšení správnosti příznaků se jeví generovat několik nespojitých pravoúhlých oblastí. Následně z těchto oblastí poskládat dva histogramy a provést jejich rozdíl. V podstatě by se jednal o rozdíl vícedimenzionálních histogramů.

- Nabízí se i zamyšlení nad formou použití algoritmu adaboost. Přístup použitý v této práci předpokládá, že každý příznak poskytne jednoduchou číselnou odezvu na jejímž základě bude adaboost rozhodovat o zařazení do příslušné třídy. Odlišným přístupem může být vytváření klasifikátoru pro každý bin histogramu. Příznak by tedy neposkytoval jednoduchou odezvu, ale celý histogram. Dá se říci, že každý histogram by tvořil skupinu klasifikátorů, jejichž počet by odpovídal počtu binů. Výsledné hlasování o zařazení do příslušné třídy pak probíhá na základě hlasování jednotlivých klasifikátorů.

- Úspěšnost a výsledek detekce se odvíjí jak od samotného algoritmu a nastavení učících a detekčních procesů, tak od trénovacích dat. Výsledek detekce je závislý na velikosti trénovacího souboru, přítomnosti šumu, variabilitě trénovacích dat a samozřejmě podobnosti trénovacích a testovacích dat. Zatím lze jen velmi těžko předpokládat vytvoření univerzálního detektoru, tento fakt se ukázal zejména při detekci chodců. Prostředí, ve kterém se mohou vyskytnout je natolik variabilní, že dosažení univerzální detekce je ještě vzdálené. Ve

vztahu k trénovacím datům se ukázalo, že je vhodné provádět klasterizaci a vytvářet sériově řazené modely. Klasterizace umožní rozdělit data do podobnostních podtříd a na jejich lze vytvořit přesnější model. Nabízí se otázka zda takový model neztrácí na obecnosti. Odpověď na tuto otázku by měla být posuzována podle konkrétní situace.

4 ZÁVĚR

V této práci byly implementovány a ověřeny metody využívající natočené Haarovy příznaky a příznaky založené na histogramech orientovaných gradientů. Pro oba tyto typy byly vytvořeny operátory umožňující jejich použití v RapidMineru. V kombinaci se stávajícími metodami v RapidMineru byla vytvořena schémata pro naučení modelu detekujícího postavy a tepny. Z pohledu učení modelu byly popsány zásadní parametry, které měly vliv na míru naučenosti výsledného modelu. Pomocí těchto optimálních parametrů byl natrénovány detektory objektů jejichž úspěšnost byla otestovaná na nezávislých testovacích sadách.

V průběhu realizace práce byly odhaleny významné nedostatky původní metody histogramů orientovaných gradientů. Mezi nejdůležitější patřila především vysoká dimenze příznaku, absence jakékoliv heuristiky vybírající vhodnou oblast zájmu nebo optimální příznak. Problémem byla také zdlouhavá detekce v řádech minut.

Hlavním přínosem této práce je především návrh a realizace přístupů, které kompenzují výše zmíněné nedostatky. Zvýšení rychlosti procesů učení a detekce bylo dosaženo využitím integrálních obrazů pro reprezentaci obrazových dat. Vysoká dimenze histogramu popisujícího vstupní obraz byla redukována metodou Diffusion distance, která dokáže vyjádřit míru podobnosti dvou histogramů. Popis vstupního obrazu monolitickým histogramem byl nahrazen množinou příznaků. Výběr jednotlivých příznaků je na základě jejich relevantnosti realizován algoritmem Adaboost. Pro zvýšení detekční rychlosti a snížení počtu falešných detekci byl implementován koncept kaskády klasifikátorů. Díky tomuto kaskádovému řazení bylo možno sestavit klasifikátor tvořený modelem SVM a Adaboost.Uvedené metody byly implementovány jako operátory do prostředí RapidMiner. Díky tomu je lze propojovat s dalšími metodami pro zpracování obrazu a dolování dat. Mohla tak být vytvořena zapojení umožňující natrénování a aplikaci klasifikačního modelu. Úspěšnost detekce byla ověřena na úlohách detekce chodců a tepen. V této práci bylo zmíněno a realizováno několik vylepšení, která by měla odstranit nedostatky původní metody a zvýšit tak její efektivitu. Byla vytvořena zapojení, která dokáží v obraze úspěšně detekovat postavy a tepny. V závěrečné diskuzi byla navrženy další možné přístupy, které by mohly vést k získání ještě přesněji fungujících detektorů.

LITERATURA

[1] DALAL, Navneet; William Triggs. *Histograms of Oriented Gradients for Human Detection.* In CVPR05 [online]. 2004, 1, [cit. 2011-12-12]. Dostupný z WWW: <eprints.pascal-network.org>.

[2] DERPANIS, Konstantinos. *Integral image - based representations.* York University [online]. 2007, 1, [cit. 2011-12-12]. Dostupný z WWW: <www.cse.yorku.ca>.

[3] DOBEŠ, Michal. *Zpracování obrazu : a algoritmy v C#.* 1. Praha : BEN, 2008. 140 s. ISBN 978-80-7300-233-6.

[4] GAO, Wei; HAIZHOU, Ai. *Adaptive Contour Features in Oriented Granular Space for Human Detection and Segmentation.* In CVPR09 [online]. IEEE, 2010 [cit. 2011-12-12]. Dostupné z WWW: <ieeexplore.ieee.org>.

[5] GERÓNIMO, David. *Haar-likeFeatures and Integral ImageRepresentation [online].* Barcelona : Universitat Autonoma de Barcelona, 2009. 46 s. Výuková prezentace. Universitat Autonoma de Barcelona. Dostupné z WWW: <www.cvc.uab.es/adas>.

[6] GUNAY, Asuman; NABIYEV, Vasif . *A new image division for LBP method to improve face recognition under varying lighting conditions.* In [online]. 23. [s.l.] : [s.n.], 2008 [cit. 2011-12-15]. Dostupné z WWW: <ieeexplore.ieee.org>. ISBN 9781424428816.

[7] Hui-Xing Jia, Yu-Jin Zhang. *Fast Human Detection by Boosting Histograms of Oriented Gradients.* In CVPR06 [online]. IEEE, 2006, 1, [cit. 2011-12-12]. Dostupný z WWW:<ieeexplore.ieee.org>.

[8] LIENHART, Rainer; MAYDT, Jochen. *An extended set of Haar-like features for rapid object detection - Proceedings International Conference on Image Processing (2002).* [online]. 1. [s.l.] : Ieee, 2002 [cit. 2011-12-12]. Dostupné z WWW: <ieeexplore.ieee.org>.

[9] LING, Haibin; OKADA, Kazunori. *Diffusion Distance for Histogram Comparison* In CVPR06 [online]. 2006. [s.l.] : [s.n.], 2006 [cit. 2011-12-15]. Dostupné z WWW: <ieeexplore.ieee.org>. ISBN 0769525970.

[10] SATPATHY, A; XUDONG, Jiang; HOW, Lung Eng. *Extended Histogram of Gradients with Asymmetric Principal Component and Discriminant Analyses for Human Detection.* In Computer and Robot Vision (CRV), 2011 Canadian

Conference on [online]. 2011. Singapore : Nanyang Technol. Univ., 2011 [cit. 2011-12-13]. Dostupné z WWW: <ieeexplore.ieee.org>. ISBN 978-1-61284-430-5.

[11] STEINWART, Ingo ; CHRISTMANN, Andreas . *Support vector machines.* 1 : Springer, 2008 [cit. 2011-12-15].ISBN 0387772413.

[12] SZEILESKI, Richard. *Computer Vision : Algorithms and Applications.* New York : Springer, 2011. 811 s. ISBN 978-1-84882-934-3.

[13] TUZEL, Oncel. *Human Detection via Classification on Riemannian Manifolds.* In CVPR07 [online]. [s.l.] : [s.n.], 2007 [cit. 2011-12-15]. Dostupné z WWW: <ieeexplore.ieee.org>.

[14] VIOLA, Paul; JONES, Michael. *RapidObjectDetectionusingaBoostedCascade-ofSimple Features.* In CVPR01 [online]. 2001, 1, [cit. 2011-12-12]. Dostupný z WWW: <ieeexplore.ieee.org>. ISSN 10636919.

[15] WU, Jianxin ; BRUBAKER, Charles ; MULLIN, Matthew . *Fast asymmetric learning for cascade face detection.* In [online]. 30. [s.l.] : IEEE Computer Society, 2008 [cit. 2011-12-15]. Dostupné z WWW: <www.ncbi.nlm.nih.gov>.

[16] ZENG Chengbin, Ma Huadong, Ming Anlong. *Fast human detection using misVM and a cascade of HOG-LBP features* In CVPR10 [online]. IEEE, 2010 [cit. 2011-12-12]. Dostupné z WWW: <ieeexplore.ieee.org>.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

DSP číslicové zpracování signálů – Digital Signal Processing

f_{vz} vzorkovací kmitočet

HOG histograms of oriented gradients

LBP local binary patterns

SAT summed area table

SAT rotated summed area table

LoG Laplacian of Gaussian

SVM Support Vector Machine