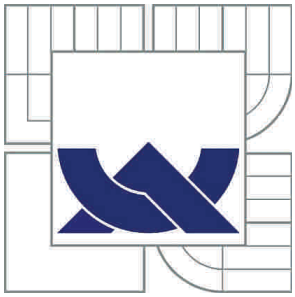
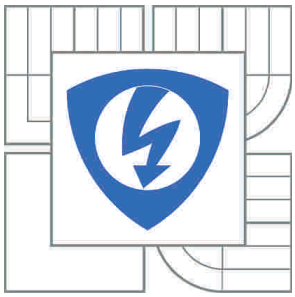


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF CONTROL AND INSTRUMENTATION

VIZUÁLNÍ DETEKCE OSOB V KOMERČNÍCH APLIKACÍCH

HUMAN DETECTION IN COMMERCIAL APPLICATIONS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

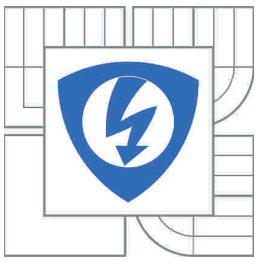
AUTOR PRÁCE — Bc. JAN ČERNÍN

AUTHOR

VEDOUCÍ PRÁCE — Ing. KAREL HORÁK, Ph.D.

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Jan Černín	***ID:***	109641
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Vizuální detekce osob v komerčních aplikacích

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je navrhnout a implementovat metody zpracování obrazu pro detekci osob ve veřejných prostorách. Kamerový systém musí být schopen sledovat pohyb osob v definovaném prostoru a to na obrazu z více kamer a vytvářet časovou trajektorii přesunů osob. Navržené metody musí vykazovat vysokou robustnost a být implementovány v reálném čase. Implementace musí zahrnovat eliminaci okrajových podmínek zejména částečné nebo úplné překrytí sledovaných objektů po krátkou dobu a znovunalezení známého objektu po jeho ztrátě.

DOPORUČENÁ LITERATURA:

[1] VERNON, David. Machine Vision : Automated Visual Inspection and Robot Vision. Hemel Hempstead : Prentice Hall International (UK) Ltd., 1991. 260 p. ISBN 0-13-543398-3.
[2] SONKA, Milan, HLAVAC, Vaclav, BOYLE, Roger. Image Processing, Analysis and Machine Vision. 3rd edition. Toronto : Thomson, 2008. 829 p. ISBN 978-0-495-08252-1.
[3] RUSS, J.C. The Image Processing Handbook. Boca Raton : CRC Press, 1995. 674 p. ISBN 0-8493-2516-1.

Termín zadání:	6.2.2012	***Termín odevzdání:***	21.5.2012

Vedoucí práce: Ing. Karel Horák, Ph.D.
Konzultanti diplomové práce:

doc. Ing. Václav Jirsík, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Předmětem této diplomové práce bylo vybrat a implementovat metody určené pro detekci a sledování osob v komerčních aplikacích. Výsledné řešení bylo voleno jako kombinace moderních přístupů a metod, s kterými přišel výzkum a vývoj poslední doby. Algoritmus je schopen vytvářet trajektorie pohybu osob ve vnitřních prostorách budov, a to i v případě částečného nebo úplného zakrytí sledovaných objektů. Scéna je snímána statickou kamerou umožňující přímý pohled na sledované osoby. Vybrané metody jsou implementovány v programovém jazyce C# s využitím funkcí knihovny OpenCV. Výstup algoritmu je zobrazován ve vytvořeném uživatelském rozhraní.

KLÍČOVÁ SLOVA

Sledování osob, Histogram orientovaného gradientu, Subtrakce pozadí, Kalmanův filtr, Trasovací algoritmus, Trajektorie pohybu

ABSTRACT

The aim of the master thesis is to derive and implement image porcessing methods for people detection and tracking in images or videos. The overall solution was chosen as a combination of modern approaches and methods which were recently presented. The proposed algorithm is able to create trajectory of the person moving in indoor building spaces even under influence of full or partial occlusion for a short period of time. The scene of interest is surveyed by a static camera having direct view on targets. Selected methods are implemented in C# programming language based on OpenCV library. Graphical user interface was created to show the final output of algorithm.

KEYWORDS

People tracking, Histogram of Oriented Gradient, Background Subtraction, Kalman filtering, Tracking algorithm, Trajectory

ČERNÍN, Jan *Vizuální detekce osob v komerčních aplikacích*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2012. 74 s. Vedoucí práce byl Ing. Karel Horák, PhD.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Vizuální detekce osob v komerčních aplikacích“ 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 jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Karlu Horákovi, Ph.D, za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne ....

(podpis autora)

OBSAH

Úvod **11**

1 Současné metody detekce osob **13**
1.1 Subtrakce pozadí 14
1.1.1 Přehled používaných metod 15
1.1.2 Pokročilá metoda subtrakce „Codebook“ 16
1.2 Detektor založený na porovnávání se vzorem 17
1.3 Detektor Viola&Jones s využitím AdaBoost 18
1.4 Normalizovaný obrazový gradient HOG 20
1.4.1 Princip algoritmu 20
1.4.2 Extrakce příznaků 21
1.4.3 Výpočet obrazových gradientů 22
1.4.4 Sestavení histogramu 22
1.4.5 Normalizace a tvar deskriptoru 22
1.5 Klasifikace objektu s využitím HOG deskriptoru 24
1.5.1 SVM klasifikátor 25
1.5.2 Trénování SVM 26
1.5.3 Zhodnocení detekce postav založené na HOG 27
1.6 Sledování objektů 27
1.6.1 Problém korespondence 28
1.6.2 Kalmanův filtr 29
1.6.3 Aplikace Kalmanova filtru 29

2 Vybrané metody **32**
2.1 Histogram orientovaného gradientu 33
2.1.1 Parametry detektoru 33
2.1.2 Velikost obrazu a rychlost algoritmu 34
2.1.3 Normalizace jasu 35
2.1.4 Výpočet gradientů 36
2.1.5 Sestavení histogramu gradientů pro detekční okno 37
2.1.6 Normalizace histogramů bloku 38
2.2 Trasovací algoritmus s využitím Kalmanova filtru 38
2.3 Metody užité pro zvýšení robustnosti systému 39
2.3.1 Kombinace metod subtrakce pozadí a porovnávání se vzorem . . 40
2.3.2 Subtrakce pozadí metodou Codebook 42

3 Použitý hardware a testovací data **43**
3.1 Vývojářská stanice . . . 43
3.2 Kamera MODI . . . 43
3.3 Fotoaparát Canon Powershot . . . 44
3.4 Zpracování testovacích dat . . . 44
3.5 Veřejně dostupné testovací data . . . 44

4 Implementace detekčního algoritmu **46**
4.1 OpenCVSharp 2.3.1 . . . 46
4.2 Struktura programu . . . 46
4.3 Grafické uživatelské rozhraní . . . 47
4.4 Přehled použitých tříd . . . 48
4.4.1 CvClass . . . 49
4.4.2 Třída BodyDetect . . . 50
4.4.3 Třída Person . . . 52
4.4.4 Třída KalmanTracker, implementace Kalmanova filtru . . . 54
4.5 Korespondence sledovaných objektů . . . 57

5 Interpretace výsledků **58**
5.1 Odezva detektoru pro mezní velikosti osob . . . 58
5.2 Selhání detektoru . . . 59
5.3 Selhání trasovacího algoritmu . . . 60
5.3.1 Trasování pohybu skupiny osob . . . 60
5.3.2 Trasování pohybu osob zacloněné překážkou . . . 61

6 Závěr **62**

Literatura **64**

Seznam symbolů, veličin a zkratek **67**

Seznam příloh **68**

A Běh algoritmu na testovacích datech **69**
A.1 Testovací data Kolejní 4, pohled 1 . . . 69
A.2 Testovací data Kolejní 4, pohled 2 . . . 70
A.3 Testovací data Kolejní 4, pohled 3 . . . 71
A.4 Testovací data i-Lids, Londýnské metro . . . 72
A.5 Testovací data Caviar, chodba nákupního centra . . . 73

B Příloha na datových nosičích DVD **74**

SEZNAM OBRÁZKŮ

1.1 Blokový diagram BGS algoritmu [12]. 14
1.2 Blokový diagram BGS algoritmu s možným řetězcem předzpracování [10]. 15
1.3 Průběh intenzity pixelu (obrázek vlevo). Kódovaná tabulka *"coodebook"* ukazující rozsahy hodnot intenzit (obrázek vpravo). V průběhu učení je nový shluk „blok“ vytvořen vždy, když dojde k výrazné změně intenzity (nastavené prahy). Tento blok pomalu narůstá tak, aby pokryl sousední hodnoty (převzato z [1]). 16
1.4 Porovnávání šablony se vzorem. Pro všechny pixely vstupního obrazu je spočítána míra podobnosti okolí tohoto pixelu s danou šablonou. 17
1.5 Příklad jednotlivých obdélníkových filtrů [17]. 19
1.6 Příklad aplikace různě posunutých filtrů na dvou vstupních obrazech velmi malého rozlišení. Obrazy Δ, U, D, R a L jsou výstupem posunutého filtru odpovídajícím směrem [17]. 19
1.7 Řetězec extrakce příznaků a následná detekce podle [3]. 20
1.8 SIFT deskriptor významných bodů. Vlevo velikost a směr gradientů, vpravo normalizovaný histogram jednotlivých oblastí váhovaný gaussovým oknem [3]. 21
1.9 Varianty navrhovaného HOG deskriptoru. Vlevo R-HOG/SIFT, uprostřed C-HOG , vpravo C-HOG s plnou centrální buňkou [2]. 23
1.10 HOG detektor založený především na konturách lidského těla (hlava, ramena a dolní končetiny). (a) Průměrný gradient trénovacího vzoru. (b) Každý "pixel"znázorňuje maximální pozitivní váhu SVM v bloku centrovaném na daném pixelu. (c) Stejné pro negativní váhy SVM. (d) Testovací obrázek. (e) R-HOG deskriptor testovacího obrázku. (f,g) R-HOG deskriptor váhovaný podle pozitivních a negativních SVM vah [3]. 23
1.11 Celkový pohled na řetězec extrakce příznaků. Detekční okno je taženo přes mřížku překrývajících se bloků. Každý blok obsahuje buňky, jejichž hodnoty gradientu přispívají do orientovaného histogramu gradientů. Histogramy jsou lokálně normalizovány a sloučeny do jednoho příznakového vektoru (převzato z [2]) 24
1.12 Hledání nadroviny v 2-D separabilním prostoru. 25
1.13 Strom dělení jednotlivých sledovacích algoritmů [18] 28
1.14 Jednotlivé kroky výpočtu kalmanova algoritmu. 31
2.1 Diagram volání jednotlivých metod 32
2.2 Detekční okno rozdělené na jednotlivé bloky 16x16px vlevo. Vpravo blok rozdělen na buňky o velikosti 8x8px. 34
2.3 Rychlost algoritmu pro různá rozlišení vstupního obrazu. 35

2.4 Gama korekce s příslušnými jasovými křivkami. 36
2.5 Výpočet velikosti a směru gradientů. a) originální obraz, b) absolutní hodnoty gradientů, c) velikost a směr gradientů pro část hlavy a ramen 36
2.6 Sestavení histogramů gradientů pro detekční okno o velikosti 128x64px. Pro zjednodušení je histogram sestavován pro šedotónový obraz. 37
2.7 Výsledná trajektorie pohybu s ukázkou selhání detektoru (simulace). . . . 38
2.8 Výsledná trajektorie pohybu s ukázkou selhání detektoru (Reálná data). . 39
2.9 Zvětšená oblast pro porovnávání se vzorem s trajektorií pohybu (červená). Výstup Kalmanova filtru (modrá). 40
2.10 Algoritmus porovnávání obrazu se vzorem. 41
2.11 Selhání detektoru při porovnávání se vzorem. 41
2.12 Obraz špatně segmentovaného popředí vlevo s provedením morfologických operací vpravo. 42
3.1 Kamera MODI AdoSpy 44
3.2 Testovací snímky pořízené v budově Kolejení 4. 45
3.3 Testovací snímky CAVIAR vlevo, i-Lids vpravo. 45
4.1 Programové GUI 47
4.2 Volání jednotlivých tříd programu 50
4.3 Vývojový diagram třídy BodyDetect. 51
4.4 Vývojový diagram třídy Person. 54
4.5 Řešení problému korespondence. 57
5.1 Modelová situace pro zjištění mezních parametrů velikosti osob v pixelech. 58
5.2 Selhání detektoru pro postavy menší jak mezní velikost. 59
5.3 Selhání trasovacího algoritmu při detekci pohybu skupiny osob. 60
5.4 Selhání trasování osoby zacloněné překážkou. 61
A.1 Aplikování algoritmu na testovací data získané v budově Kolejní 4, pohled 1. 69
A.2 Aplikování algoritmu na testovací data získané v budově Kolejní 4, pohled 2. 70
A.3 Aplikování algoritmu na testovací data získané v budově Kolejní 4, pohled 3. 71
A.4 Aplikování algoritmu na testovací data „i-Lids“, Londýnské metro. 72
A.5 Aplikování algoritmu na testovací data „Caviar“, nákupní centrum. 73

SEZNAM TABULEK

2.1 Průměrné rychlosti detekce pro daná rozlišení (metoda HOG). 35
4.1 Stručný popis programových tříd. 48
5.1 Mezní velikost osob v pixelech pro modelovou scénu. 58
5.2 Parametry detektoru při zjištění mezní velikosti osob. 59

ÚVOD

Problematika počítačového vidění v dnešní době proniká stále hlouběji do běžného lidského života. Člověk, ať už chce nebo nechce, se každým dnem dostává více a více pod drobnohled visuálních kamer. Od aplikací typu zabezpečení prostor budov nebo monitoring pohybu osob ve městech se počítačové vidění velkou rychlostí dostává do lékařství, marketingu nebo domácností. Dříve bylo sledování osob zajištěno operátorem pozorujícím nespočet monitorů zobrazujícími různé prostředí, od venkovních prostor, po chodby a haly interiérů. Nároky kladené na operátora jsou vysoké. Tato práce vyžaduje vysokou koncentraci a v kritických situacích může dojít k selhání lidského faktoru. Postupem času a díky vysokým nárokům na obsluhu, došlo k nahrazení operátora výkonnými výpočetními jednotkami spojených s inteligentními kamerami a jiným hardwarem, který může i nemusí interagovat se sledovanými objekty. Tyto systémy jsou schopny zaznamenávat a analyzovat obrazy z téměř neomezeného množství kamer a na základě získaných dat reagují na podněty a události z venčí. Aplikace tohoto typu jsou hojně nasazovány k zajištění bezpečnosti a chodu letišť, nádražních hal, veřejných prostor měst nebo i ke komerčnímu využití, jako jsou systémy optimalizace prodeje z anglického point-of-sales (POS).

POS aplikace jsou dnes využívány k marketinkovému průzkumu hypermarketů, supermarketů, ale i menších obchodů. Nahrazují často namáhavé a časově náročné průzkumy prováděné z důvodu zvýšení zisků obchodů, ale i k zlepšení pohodlí zákazníka. V těchto aplikacích se často jedná o počítání osob v jednotlivých sektorech obchodu, průzkum prodejnosti produktů, zjišťování vhodného umístění produktů v prostorách obchodu, ale také například zabezpečení proti krádeži a monitorování pohybu osob v pro ně určených prostorách.

Vizuální sledování osob s sebou přineslo mnoho výhod. Odpadá nutnost přímé interakce s člověkem, jako je nošení různých senzorů a zařízení, které je zároveň nepohodlné. K tomu bylo vyvinuto několik systémů a algoritmů.

Tato práce se zabývá vývojem a implementací algoritmů určených pro robustní sledování osob, vhodné především pro marketingové účely. Celý systém by v ideálním případě měl být schopen sledovat neomezený počet osob, zaznamenávat jejich trajektorii pohybu, směr pohledu, kde se zastaví a co si koupí. Algoritmus by měl být natolik robustní, aby se dal použít s již fungujícím kamerovým systémem v kterémkoliv obchodě. Dále budou rozebrány problémy spojené s detekováním osob, které jsou v průběhu sledování částečně nebo úplně zakryty překážkou v pohledu kamery.

Kapitola 1 popisuje moderní algoritmy užívané pro detekci osob jak ve statických, tak i dynamických snímcích. Tyto metody byly vybrány z množství současných přístupů, s kterými přišel vývoj poslední doby. Kapitola obsahuje teoretické poznatky od segmentace obrazu klasickými metodami subtrakce pozadí, porovnávání obrazu s uloženým vzorem, samotnou detekci osob až po algoritmy určené k sledování detekovaných objektů. Kapi-

tola 2 popisuje všechny metody užité při zpracování této práce. Jsou zde shrnuty algoritmy s popisem jednotlivých parametrů a jakým způsobem tyto parametry ovlivňují výslednou detekci, jaká jsou omezení a jejich časová náročnost. V kapitole 4 je popsána samotná implementace vybraných algoritmů v programovém prostředí C#. Popis programu obsahuje vývojové diagramy, které znázorňují strukturu a chod programu. V kapitole 5 jsou shrnuty dosažené výsledky programu při běhu s testovacími daty pořízenými v prostorách Kolejní 4 nebo z veřejně dostupných databází.

1 SOUČASNÉ METODY DETEKCE OSOB

V dnešní době se stále častěji vyskytují aplikace, kde je třeba detekovat jednu, či vetší počet osob. Tyto systémy částečně nahrazují činnost člověka a naleznou své uplatnění v širokém spektru aplikací. V této kapitule budou probrány metody a přístupy k detekci, které přinesl vývoj poslední doby. Algoritmy se ve své podstatě liší ve způsobu zpracování obrazových dat a v jejich následném popisu. Scéna může být snímána z jedné nebo více kamer, kamer statických nebo pohyblivých, s pevným ohniskem, zoomem a dalších. Zpracování dat, segmentace a popis již závisí na zvolených metodách. Kombinací jednotlivých metod lze dosáhnout výsledků s velkou úspěšností detekce a velkou odolností proti chybám.

Vytvořit robustní algoritmus detekující lidskou postavu v sekvencích snímků může být velice komplikovaný problém. Jednoduchý model lidské postavy lze vytvořit za podmínky, že se osoba nachází ve statické scéně a v přímém pohledu směrem od nebo ke kameře. Toto však v případě reálných scén nelze zaručit a řešení se značně komplikuje. Při pohybu se tvar, v případě osob silueta, mění v závislosti na směru pohybu a rychlosti. Úloha detekce se skládá z vytvoření a naučení takového modelu lidské postavy, který bude vůči těmto změnám invariantní. Pokud je toto splněno, model je použit k vytvoření klasifikátoru objektu [6],[13],[15],[16],[19].

Často se proces detekce skládá z následujících operací:

- extrakce pozadí/popředí,
- extrakce objektu,
- klasifikace objektu,
- sledování objektu,
- vhodná reakce na detekovaný objekt.

Mnoho dnešních přístupů klasifikuje objekt jako osobu tak, že se nejprve testuje, zdali nalezený objekt obsahuje významné rysy lidského obličeje. Objekt je klasifikován jako člověk v případě, že byl v sekvenci snímků lidský obličej nalezen více jak jednou. Jak je známo, lidský obličej je snadno rozlišitelným rysem lidského těla což vede k velice spolehlivému detektoru.

Další metody jsou založeny na takzvaném *appearance modelu*, což znamená, že se pro nalezený region hledá nejlépe vyhovující model. Tento model může být reprezentován daty v 2-D nebo 3-D prostoru. Pokud model s určitou přesností vyhovuje, objekt je klasifikován jako lidská postava. Další skupinou jsou metody extrahující významné rysy objektu. Extrahované rysy tvoří příznakový prostor, který je opět porovnáván s naučenými heterogenními daty. Klasifikace je často prováděna užitím metod umělé inteligence (neuronové sítě, genetické algoritmy, Support Vector Machine (dále SVM) atd.). Nejlepšího výsledku lze dosáhnout vhodnou kombinací výše zmíněných metod. Tato řešení jsou pak

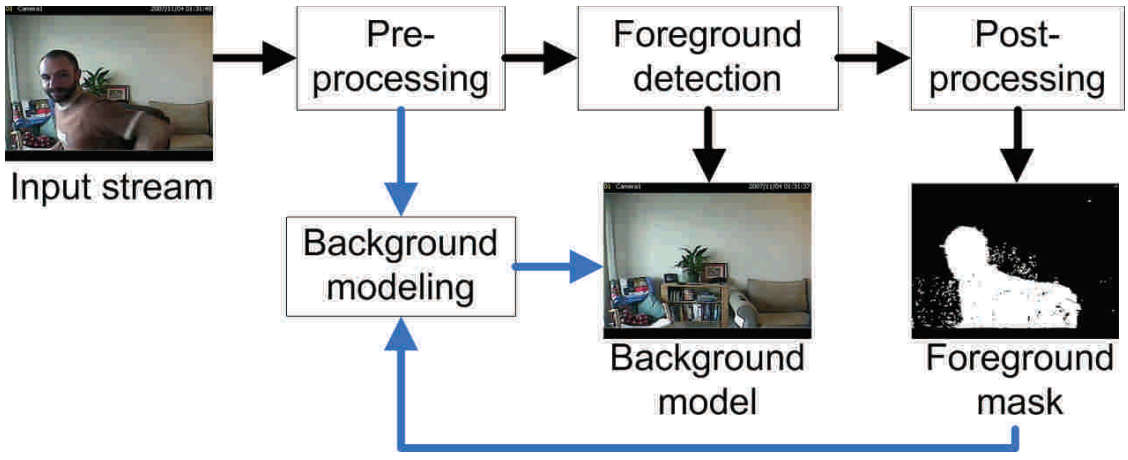
nazývána hybridními metodami, které jsou často výpočetně velmi náročné [9].

1.1 Subtrakce pozadí

Skoro každý detekční algoritmus obvykle začíná detekcí pohybu v obraze a tím extrakcí pohyblivých regionů. Jednou z nejjednodušších a často používaných metod je metoda subtrakce pozadí. Tato metoda je limitována tím, že vyžaduje jak statickou kameru, tak i statické pozadí. V praxi podmínku neměnného pozadí v podstatě nelze splnit a to působí jisté problémy.

Subtrakce pozadí, z anglického *Background Subtraction* (dále BGS), je prvním důležitým krokem v mnoha aplikacích počítačového vidění. Existuje mnoho metod s různými postprocessing technikami s cílem zlepšit jejich výkon. Společnou vlastností každého algoritmu je explicitní model pozadí. Jako pozadí je obvykle považován každý statický nebo periodicky se pohybující objekt, jehož pohyb zůstává statický resp. periodický přes periodu zájmu. Objekty v popředí jsou poté detekovány jako rozdíl aktuálního snímku a modelu. Z toho vyplývá, že jakákoliv změna v pozadí nebo osvětlení může působit problémy. Snahou je tedy sestavit natolik robustní BGS algoritmus, který je schopen tyto vlivy eliminovat a aktualizovat tak model pozadí. Pro real time aplikace musí být samozřejmě výpočetní náročnost dostatečně nízká, aby bylo umožněno zvládnout i následný postprocessing masky popředí. Poté následuje klasifikace objektů jako je člověk, dítě, zvíře, automobil aj., a to podle tvaru, barvy, pohybu nebo jiného významného rysu.

Na následujícím obrázku je znázorněn blokový diagram úloh, který nejčastěji sleduje tuto posloupnost: preprocesing, model pozadí, detekce popředí a postprocessing. Shrnutí preprocesing metod viz [12].



Obr. 1.1: Blokový diagram BGS algoritmu [12].

1.1.1 Přehled používaných metod

Vytvoření modelu pozadí je stěžejní částí algoritmu. Určuje, jakým způsobem bude model pozadí definován a aktualizován. Techniky vytvoření modelu jsou rekurzivní a nerekurzivní. Více k jednotlivým technikám viz [10].

Rekurzivní metody:

- Running Gaussian Average (RGA)
- Gaussian Mixture Model (GMM)
- GMM with adaptive number of Gaussians (AGMM)
- Aproximated Median Filtering (AMF)

Nerekurzivní metody:

- Mediánová filtrace
- Mediod filtering
- Eigenbackgrounds (EigBG)

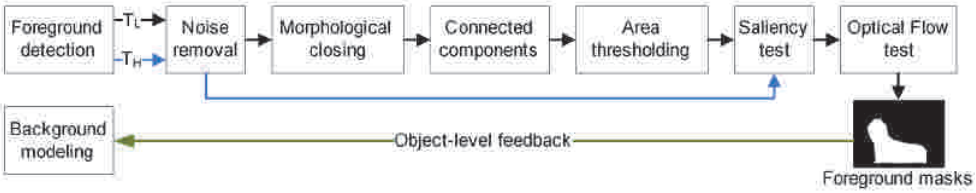
Detekce popředí definuje, které pixely z nového snímku nenáleží modelu pozadí. Metody, postrádající statistický základ (AMF, Mediod, EigBG), klasifikují nový pixel jako část popředí vždy, když je splněno

$$|I_t(x,y) - B_t(x,y)| > T, \tag{1.1}$$

kde je I_t vstupní obraz, B_t model pozadí a T je uživatelem definovaný práh. Největší nevýhodou je, že pro všechny pixely platí jeden práh. Metody založené na statistickém základě (RGA, GMM, AGMM, Median) klasifikují pixely patřící popředí podle

$$p(I_t^c|B_t^c) < T^c = \eta\,\psi^c, \tag{1.2}$$

kde c je libovolný barevný kanál. Práh T^c je roven odhadované odchylce, ψ^c, tak aby bylo zajištěno, že pixel bude klasifikován jako popředí v případě, že je odchylka mimo pravděpodobnostní rozloženi [10].



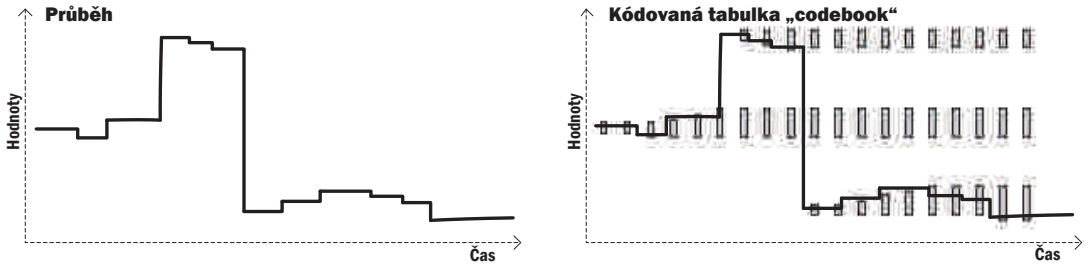
Obr. 1.2: Blokový diagram BGS algoritmu s možným řetězcem předzpracování [10].

1.1.2 Pokročilá metoda subtrakce „Codebook“

Mnoho reálných scén často obsahuje objekty jejichž pohyb nelze snadno popsat. Jedná se například o stromy pohybující se ve větru, pohybující se záclony, rotující ventilátory, atd. V takovýchto scénách se zároveň často mění světelné podmínky, což je způsobeno například pohybujícími se mraky nebo dalšími jevy.

Metoda, jež eliminuje tyto vlivy, přiřazuje jednotlivému pixelu nebo skupině pixelů v obraze sérii modelů, sestavených za periodu času. Takto získaný model se vypořádá s dočasnými změnami, to však za cenu vysoké paměťové a časové náročnosti. Požadavkem real time aplikace je samozřejmě co nejmenší časová náročnost, proto není tento přístup příliš vhodný. Pro tento účel byla vyvinuta metoda, jež se adaptivním metodám subtrakce velice přibližuje. Tato metoda formuje kódovanou tabulku z anglického „*codebook*“, jež kóduje informaci získanou z okolí pixelu jako rozsah hodnot jednotlivých kanálů RGB prostoru.

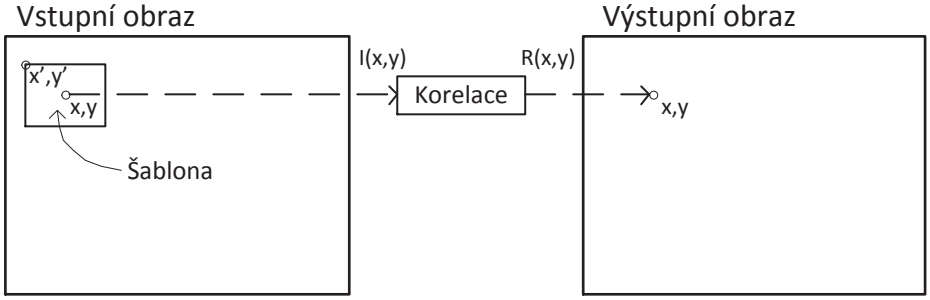
Takto zvolený model se vypořádá s pixely, jejichž intenzita se dramatický mění (listy a větve stromů měnící intenzitu střídavě s intenzitou oblohy v pozadí). V případě RGB prostoru je kódovaná tabulka sestavená z následujících rozsahů hodnot. Pokud se hodnota blíží hodnotě pozorované v čase $t-1$, pak je modelována jako odchylka k dané skupině barev. V opačném případě je pixelu přiřazena nová skupina barev, která je s ním spojena. Na tuto metodu lze pohlížet jako na skupinu shluků plovoucí v RGB prostoru, kde každý shluk reprezentuje pravděpodobnost se kterou patří pozadí. Pro RGB prostor je kódovaná tabulka tvořena shluky v 3-D prostoru (každá složka RGB). V porovnání s metodami subtrakce pozadí využívající průměrování je metoda *”coodebook”* odolná vůči změnám intenzit přes vetší časovou periodu. Model zároveň obsahuje více prahů a je tak imunní vzhledem k šumu obrazu. Pixel je klasifikován jako popředí vždy, když nenáleží ani do jednoho naučeného shluku modelu [1].



Obr. 1.3: Průběh intenzity pixelu (obrázek vlevo). Kódovaná tabulka *”coodebook”* ukazující rozsahy hodnot intenzit (obrázek vpravo). V průběhu učení je nový shluk „blok“ vytvořen vždy, když dojde k výrazné změně intenzity (nastavené prahy). Tento blok pomalu narůstá tak, aby pokryl sousední hodnoty (převzato z [1]).

1.2 Detektor založený na porovnávání se vzorem

Jednou z metod detekce, na kterých staví tato práce, je detektor založený na porovnávání objektů se vzorem. Metoda využívá měření stupně podobnosti hledaného objektu s předem danou šablonou z anglického *Template matching*. Základem je posuvné okno o velikosti šablony, které je přes obraz posouváno viz obrázek 1.4. Hodnota výstupního pixelu udává míru shody, jakou objekt odpovídá šabloně na dané pozici. Jednotlivé metody korelace jsou shrnuty níže.



Obr. 1.4: Porovnávání šablony se vzorem. Pro všechny pixely vstupního obrazu je spočítána míra podobnosti okolí tohoto pixelu s danou šablonou.

V následujícím popisu metod korelace je použito značení I pro vstupní obraz, T pro vzor a R výstupní obraz. Jednou z často užívaných metod je metoda hledání nejmenšího rozdílu čtverců. Tato metoda vrací 0 při pozitivní shodě. Hodnota různá od nuly naopak snižuje pravděpodobnost shody podle následující rovnice

$$R_{sq\_diff}(x,y) = \sum_{x',y'} \left[T\left(x',y'\right) - I\left(x+x',y+y'\right)\right]^2, \tag{1.3}$$

kde x' a y' značí souřadnice vzoru. Další metodou je metoda korelace, která vrací hodnotu blízkou nule při malé pravděpodobnosti shody. Velká hodnota naopak zvyšuje pravděpodobnost shody podle rovnice 1.4

$$R_{ccorr}(x,y) = \sum_{x',y'} \left[T\left(x',y'\right) I\left(x+x',y+y'\right)\right]^2. \tag{1.4}$$

Dalším možným vylepšením výše popsaných metod je metoda, jež porovnává vzor relativně k jeho střední hodnotě a vstupní obraz relativně k jeho střední hodnotě podle následujících rovnic

$$
\begin{aligned}
R_{ccorr}(x,y) &= \sum_{x',y'} \left[T'\left(x',y'\right) I'\left(x+x',y+y'\right)\right]^2, \\
T'\left(x',y'\right) &= T\left(x',y'\right) - \frac{1}{(w.h)\sum_{x'',y''} T\left(x'',y''\right)}, \\
I'\left(x+x',y+y'\right) &= I'\left(x+x',y+y'\right) - \frac{1}{(w.h)\sum_{x'',y''} I\left(x+x'',y+y''\right)},
\end{aligned} \tag{1.5}
$$

kde x'' a y'' jsou pomocné indexy užité při výpočtu střední hodnoty jak vzoru, tak i vstupního obrazu, w je šířka a h výška vzoru. Podle rovnice 1.5 nastává shoda při hodnotě 1 a neshoda při -1. Návratová hodnota blízká 0 znamená, že na pozici x, y není žádná korelace (neúplné vystředění vzoru a objektu).

Pro každou výše zmíněnou metodu existuje jejich normalizovaná verze. Tyto normalizované verze pomáhají redukovat nepřesnosti vzniklé při rozdílných světelných podmínkách vzoru a vstupního obrazu. Normalizace je provedena podělením výše zmíněných rovnic normalizačním koeficientem, který je pro všechny metody stejný a je dán následujícím výpočtem

$$
Z(x,y) = \sqrt{\sum_{x',y'} T\left(x',y'\right)^2 \sum_{x',y'} I\left(x+x',y+y'\right)^2}. \tag{1.6}
$$

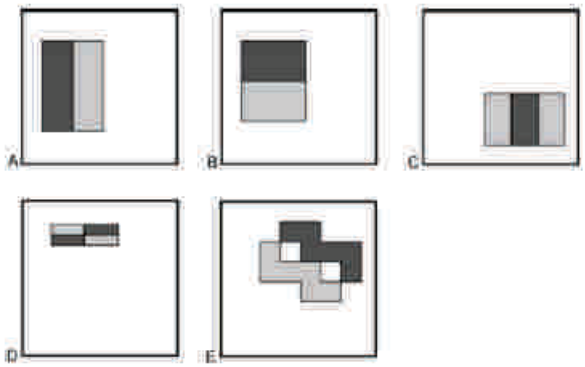
Nevýhodou těchto metod je jejich zřejmá výpočetní náročnost a silná závislost na měřítku a natočení. To lze částečně eliminovat postupem, kdy je vstupní obraz prohledáván pro různá měřítka a úhly natočení vzoru, to však za cenu velkých výpočetních nároků. V případě detekce osob lze tuto metodu užít pouze v omezeném měřítku.

Další nevýhodou je i samotná inicializace metody, kdy je třeba ručně nebo jiným způsobem označit a získat objekt zájmu. Metodu lze také vylepšit například online průměrováním získaných vzorů. Každý nález je tedy uskutečněn pro x předchozích vzorů patřící témuž objektu.

1.3 Detektor Viola&Jones s využitím AdaBoost

Tato metoda staví na algoritmu AdaBoost a aplikací filtrů posunutí podle Viola&Jones na základě Haarových vlnek. Kombinuje znalost modelu pohybu s modelem vzhledu pohybujících se objektů. Takto natrénovaný detektor je vylepšením algoritmu pro detekci obličejů ve statických scénách.

Jak již bylo řečeno, detektor využívá jednoduchých obdélníkových filtrů zobrazených na obrázku 1.5. Původní algoritmus pro statické obrazy byl upraven tak, aby ho bylo možné vhodným aplikováním filtrů použít i na pohyblivých, po sobě jdoucích snímcích.

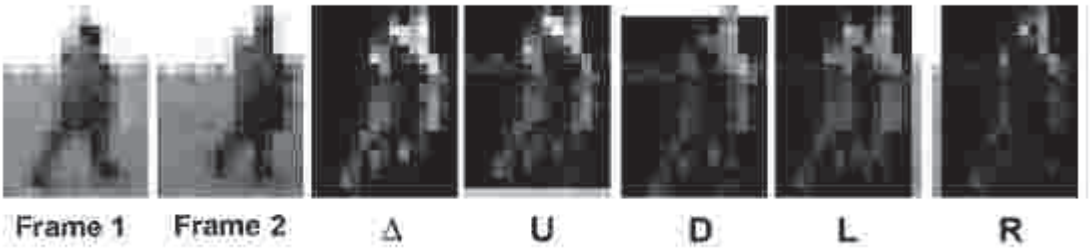


Obr. 1.5: Příklad jednotlivých obdélníkových filtrů [17].

Informace o pohybu je získána rozšířením původních filtrů pro detekci obličejů. Výhodou obdélníkových filtrů Viola&Jones je invariantnost vůči změně velikosti a natočení. Posunutím filtrů ve znázorněných směrech jsou získány následující obrazy (viz. obrázek 1.6):

$$\begin{aligned} \Delta &= abs(I_t - I_{t+1}) \\ U &= abs(I_t - I_{t+1} \uparrow) \\ D &= abs(I_t - I_{t+1} \downarrow) \\ R &= abs(I_t - I_{t+1} \rightarrow) \\ L &= abs(I_t - I_{t+1} \leftarrow), \end{aligned} \tag{1.7}$$

kde šipky naznačují směr posunutí filtru a I_t, I_{t+1} jsou vstupní obrazy. Z těchto obrazů je získána informace o pohybu. Aplikováním vhodného filtru pouze na vstupní obraz I_t získáváme informaci o vzhledu. Jednotlivé výpočty jsou prováděny metodou integrálního obrazu, čímž je zaručena rychlá evaluace.



Obr. 1.6: Příklad aplikace různě posunutých filtrů na dvou vstupních obrazech velmi malého rozlišení. Obrazy Δ, U, D, R a L jsou výstupem posunutého filtru odpovídajícím směrem [17].

Z množství filtrovaných obrazů je sestavena kaskáda slabých klasifikátorů, které poté formují jeden silný, oddělující pozitivní nálezy od negativních. Takto vytvořený klasifikátor je invariantní vůči změně rozlišení a natočení. Zároveň dosahuje velmi vysoké přesnosti detekce. Rychlost detekce pro velmi malé rozlišení (20x15px) je udávána 4fps [17].

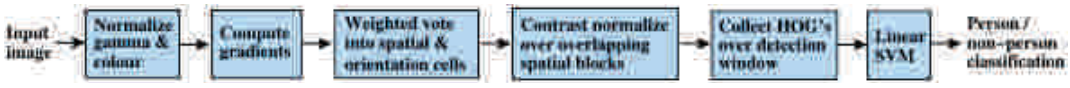
1.4 Normalizovaný obrazový gradient HOG

Jedná se o příznakový detektor využívající lokálně normalizovaného obrazového gradientu z anglického *Histogram of Oriented Gradient* (HOG). Tento deskriptor může být statický nebo dynamický. Statický deskriptor poslouží při detekci ve statických snímcích, zatímco dynamický využívá informaci o pohybu objektu a je používán při detekci v sekvenci snímků. Klasifikátorem příznakového prostoru je jednoduchý kaskádový SVM.

1.4.1 Princip algoritmu

Metoda vyhodnocuje lokálně normalizovaný histogram obrazového gradientu. Základní myšlenkou je fakt, že je často lépe vzhled objektu popsat pomocí lokálního měření gradientu nebo směru hran. Implementace takového deskriptoru je dosažena tak, že je obraz rozdělen na menší spojená podokna, přes která je histogram směru gradientů nebo směr hran sestavován. Každé buňce odpovídá rozložení gradientů nebo hran v 1-D oblasti. Kombinaci těchto histogramů reprezentuje výsledný deskriptor. Pro větší odolnost vůči změnám světelných podmínek a eliminaci chyb vzniklých stíny je kontrast buňky lokálně normalizovaný. Normalizace histogramu je prováděna přes vetší buňky, nazývané ”bloky”. Velikost bloků je obvykle dvakrát vetší než velikost buňky. K normalizaci jednotlivých buněk uvnitř bloku je použita hodnota míry intenzity přes celý blok. Touto normalizací je dosaženo po částech spojitého kontrastu obrazu a invariantnost vůči změnám osvětlení a stínům.

Rozdělením detekčního okna do sítě překrývajících se normalizovaných bloků a použitím kombinovaného příznakového vektoru k natrénování lineárního SVM, vede na řetězec detekce podle obrázku 1.7 [3].



Obr. 1.7: Řetězec extrakce příznaků a následná detekce podle [3].

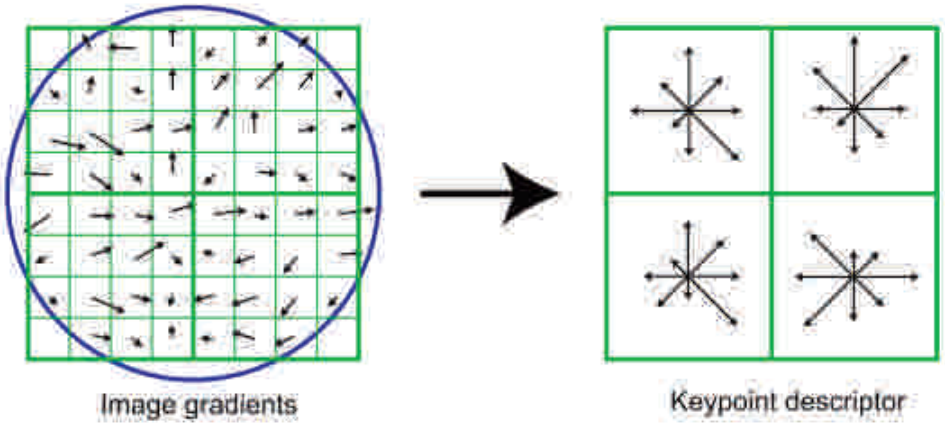
1.4.2 Extrakce příznaků

K sestavení výsledného deskriptoru se hojně užívá metoda SIFT z anglického *Scale Invariant Feature Algorithm*, která je základem mnoha obrazových detektorů. Výsledkem SIFT algoritmu je na měřítku nezávislý deskriptor významných bodů. Díky svým přednostem je SIFT hojně používán v mnoha detekčních algoritmech [8].

Kombinace HOG/SIFT s sebou přináší některé výhody. Při výběru nejlepšího příznaku zohledňuje tu hranu nebo gradient, nesoucí jedinečnou informaci vzhledem ke svému okolí. Zároveň je nutné, aby byl algoritmus odolný vůči malým změnám vlivem translace a rotace částí lidského těla. Tento přístup zároveň těží z toho, že je lidská postava v mnoha případech ve vzpřímené pozici. Lehké pohyby končetin z části eliminuje hrubší prostorové vzorkování a lokální fotometrická normalizace.

Na obrázku 1.8 je znázorněn jednoduchý deskriptor o velikosti 2x2 vytvořený ze vzorů v poli o velikosti 8x8. K dosažení invariance vůči natočení je souřadnicový systém a příslušné gradienty orientovány relativně k významnému bodu. Gaussova váhovací funkce s rozptylem σ rovným polovině šířky deskriptoru je použita k upravení velikosti gradientů jednotlivých bodů. To je ilustrováno modrou kružnicí na levé straně obrázku. Váhovací okno eliminuje změny v deskriptoru vzniklé vlivem posunutí okna a zároveň dává menší váhu gradientům, které jsou dále od středu deskriptoru.

Vpravo na obrázku 1.8 je znázorněn deskriptor významného bodu. V tomto případě je vytvořen směrový histogram přes okolí o velikosti 4x4. Takto zvolené okolí dovoluje i významným posunům v pozici gradientů. Každé pole deskriptoru obsahuje směrový histogram, jehož velikosti šipek odpovídají velikosti odpovídajících vstupních gradientů. Gradient vpravo, jež se posune až o 4 vzorky, bude stále přispívat stejnému histogramu vpravo [8].



Obr. 1.8: SIFT deskriptor významných bodů. Vlevo velikost a směr gradientů, vpravo normalizovaný histogram jednotlivých oblastí váhovaný gaussovým oknem [3].

1.4.3 Výpočet obrazových gradientů

Výkon detektoru silně závisí na kvalitě vypočtených gradientů obrazu. Bylo dokázáno, že klasické jádro masky nahrazující první derivaci $[-1,0,1]$ pro vertikální a $[-1,0,1]^T$ pro horizontální hrany, dokazovalo nejlepších výsledků. Použitím větších masek nebo aplikování Gaussova vyhlazení vždy snížilo výkon detektoru [3].

1.4.4 Sestavení histogramu

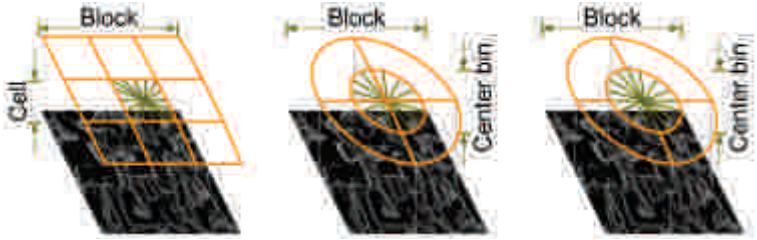
Každý pixel obrazu, respektive gradientu, přispívá svou váhou k danému směrovému histogramu tak jako v kapitole 1.4.2. Jednotlivé histogramy zahrnují příspěvek odpovídajících gradientů příslušné buňky. Buňky samotné mohou být pravoúhlé nebo kruhové. Hodnoty úhlů histogramu jsou rovnoměrně rozprostřeny od 0 do 180 stupňů, nebo od 0 do 360, v závislosti na typu gradientu "signed", nebo "unsigned". Pro detekci osob nejlépe vyhovuje užití unsigned typu gradientu. Míra, jakou bude gradient přispívat do příslušném histogramu gradientu, je určena funkční závislosti jeho velikosti [2],[3].

1.4.5 Normalizace a tvar deskriptoru

Aby byl detektor odolný vůči změnám osvětlení a kontrastu, je velikost gradientu lokálně normalizována. Toho je dosaženo seskupením buněk do větších, prostorově spojených bloků. Výsledný deskriptor je tedy vektor všech složek normalizované buňky ze všech bloků detekčního okna. Bloky se zároveň překrývají, což znamená, že každá buňka přispívá do finálního dekriptoru více než jednou.

Existují dva tvary bloků: pravoúhlý (R-HOG) a kruhový (C-HOG). R-HOG jsou obvykle čtvercové elementy tvořící čtvercovou mřížku, reprezentující tři parametry. První parametr je počet buněk bloku, dalším počet pixelů buňky a posledním počet kanálů histogramu. Optimální velikost buňky je 3x3 pixelů, 6x6 pixelů uvnitř buňky a 9 kanálu histogramu. R-HOG deskriptory jsou podobné SIFT deskriptorům s určitými výjimkami. R-HOG je počítán přes jemnou mřížku pro dané měřítko, bez normalizace podle dominantních úhlů gradientů a implicitně kóduje prostorovou informaci relativně k detekčnímu oknu. SIFT deskriptor je počítán přes hrubou mřížku danou na měřítku nezávislými významnými body, s rotací podle dominantního zastoupení úhlů gradientů a obsahují informaci o daném okolí významného bodu. Stejně jako v kapitole 1.4.2, jsou příspěvky pixelů blízkých hranám váhovaný gaussovým 2-D oknem [2],[3].

C-HOG jsou dvou typů. První typ obsahuje jednu centrovanou buňku, zatímco druhý je složen z více buněk. Dalal a Triggs ve své práci použili pouze první typ dosahující stejných výsledků jako druhý, složitější C-HOG deskriptor. Nejlepších výsledků dosahoval C-HOG deskriptor ve tvaru podle obrázku 1.9 vpravo.

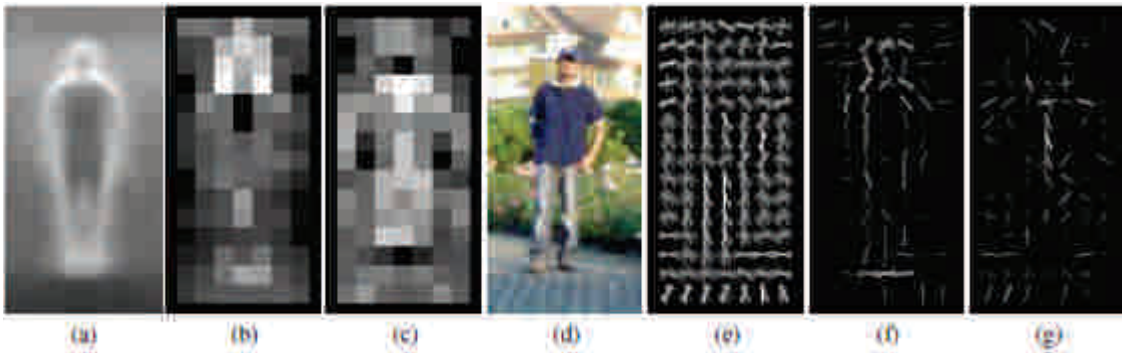


Obr. 1.9: Varianty navrhovaného HOG deskriptoru. Vlevo R-HOG/SIFT, uprostřed C-HOG , vpravo C-HOG s plnou centrální buňkou [2].

Pro lepší výsledky je vektor nenormalizovaných histogramů bloku normalizován. Tato normalizace zlepšuje výkon algoritmu a snižuje procento falešných poplachů (false-positives). Dalal a Triggs ve své práci otestovali čtyři různé metody normalizace. Úplným vynecháním normalizace dojde k výraznému zhoršení výkonu detektoru [3].

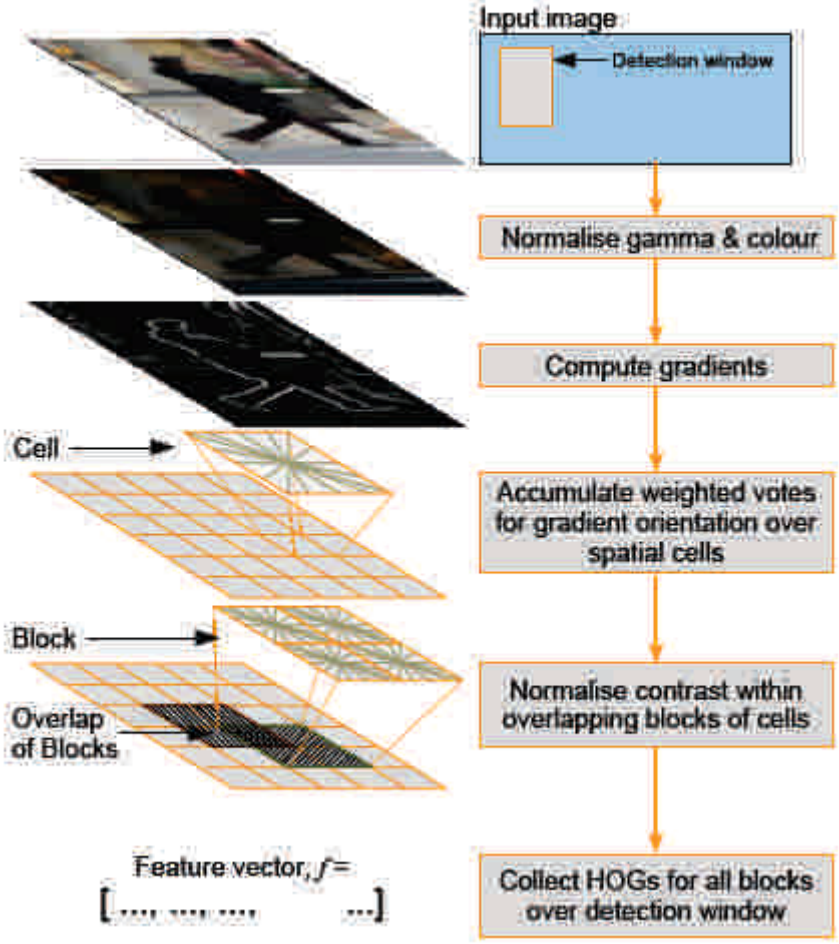
Proces tvorby deskriptoru je znázorněn na obrázku 1.10. Uvažujeme-li R-HOG deskriptor s překrytím jednotlivých bloků, dávají koeficienty natrénovaného lineárního SVM míru, jakou jednotlivé buňky přispívají k výslednému rozhodnutí. Na obrázku 1.10 b), f) je vidět, že nejvýznamnější buňky jsou ty které se nacházejí v oblasti hlavy, ramen a dolních končetin. Podobně, obrázek 1.10 c), g), zvýrazňuje gradienty uvnitř postav (především ty vertikální).

Na následujícím obrázku 1.11 je zobrazen řetězec operací, vykonávající statickou extrakci příznaků. V případě, že se jedná o pohyblivé snímky, je vhodné zkombinovat metody statického a dynamického HOG deskriptoru. Gradient obrazu je v dynamickém HOG



Obr. 1.10: HOG detektor založený především na konturách lidského těla (hlava, ramena a dolní končetiny). (a) Průměrný gradient trénovacího vzoru. (b) Každý ”pixel”znázorňuje maximální pozitivní váhu SVM v bloku centrovaném na daném pixelu. (c) Stejné pro negativní váhy SVM. (d) Testovací obrázek. (e) R-HOG deskriptor testovacího obrázku. (f,g) R-HOG deskriptor váhovaný podle pozitivních a negativních SVM vah [3].

nahrazen optickým tokem v sekvenci po sobě jdoucích snímků. Histogram je vytvořen z velikosti a směru optického toku. Extrakce příznaků je tedy podobná statickému HOG.



Obr. 1.11: Celkový pohled na řetězec extrakce příznaků. Detekční okno je taženo přes mřížku překrývajících se bloků. Každý blok obsahuje buňky, jejichž hodnoty gradientu přispívají do orientovaného histogramu gradientů. Histogramy jsou lokálně normalizovány a sloučeny do jednoho příznakového vektoru (převzato z [2])

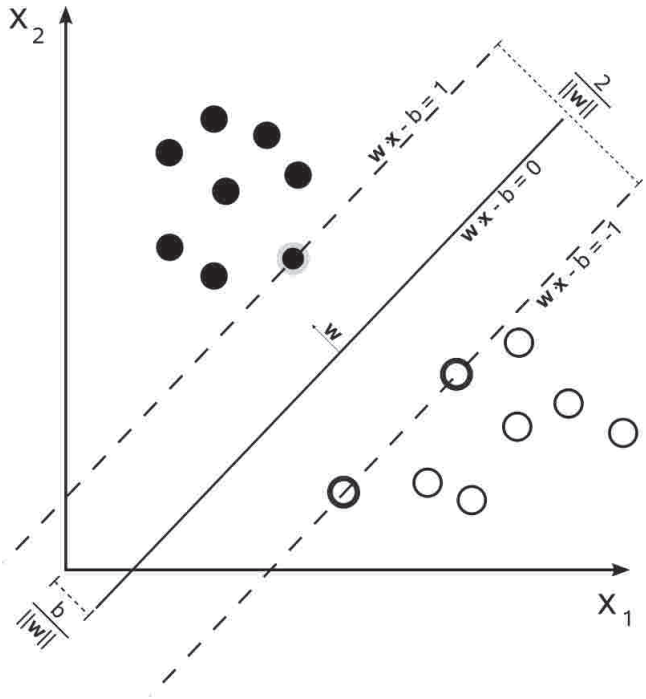
1.5 Klasifikace objektu s využitím HOG deskriptoru

Posledním krokem detekce s využitím HOG deskriptoru je konečná klasifikace objektu. V tomto případě se jedná o dichotomickou klasifikační úlohu člověk/nečlověk. K tomuto účelu je možno využít klasifikátor založený na učení s učitelem. Dalal a Trigs ve své práci použili jednoduchý klasifikátor využívající podpůrné vektory, z anglického „*Support Vector Machine*“ (SVM), natrénovaný pomocí SVMlight (implementováno v OpenCV).

1.5.1 SVM klasifikátor

Principem tohoto klasifikátoru je obvykle lineárně neseparabilní prostor příznaků (deskriptor) transformovat pomocí transformační funkce do prostoru s vyšší dimenzí. V tomto transformovaném prostoru je poté možno provést separaci příznaků pomocí nadroviny nebo množinou nadrovin. Parametry této nadroviny určují pomocné vektory, které jsou v průběhu učení přizpůsobovány.

Lineární SVM hledají nadrovinu $\langle w,x\rangle + b = 0$, která maximalizuje vzdálenost od bodů z lineárně separovatelné trénovací množiny. Trénovací množina je reprezentována vektory $\{(x_1,y_1),(x_2,y_2),...,(x_n,y_n)\}$, kde x_i je daný trénovací vzor a $y_i \in \{-1,+1\}$ je jeho identifikátor třídy. Problém hledání nadroviny tedy spočívá v hledání w a b tak, aby byla vzdálenost mezi paralelními nadrovinami co největší a zároveň stále separovala data tak, jak ukazuje obrázek 1.12.



Obr. 1.12: Hledání nadroviny v 2-D separabilním prostoru.

Řešení vede na přímou úlohu hledání maximální vzdálenosti mezi třídami podle následující rovnice

$$(w,b) = \underset{w,b}{\operatorname{argmin}} \frac{1}{2}||w||^2. \tag{1.8}$$

Po vyřešení rovnice 1.8 je získán vektor w a práh b, jenž určuje hledanou nadrovinu. Tuto přímou úlohu je však lépe převést na tzv. duální úlohu a hledat čísla $\alpha_i, i = 1,...,n$, která jsou řešením úlohy

$$\alpha_i = \underset{\alpha_i}{\operatorname{argmax}} \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \,, \tag{1.9}$$

za podmínky

$$\begin{aligned} \alpha_i &\geq 0, \quad i = 1, 2, ..., n, \\ \sum_{i=1}^{n} \alpha_i y_i &= 0. \end{aligned}$$

V tomto případě se tedy neupravuje vektor w, ale pouze reálná čísla, která jsou dány podpůrnými vektory α_i.

1.5.2 Trénování SVM

Trénink učicího algoritmu patří k nejdůležitějším částem detektorů. Se špatně natrénovaným klasifikátorem roste množství falešných alarmů (false-positives), nebo naopak vůbec nedojde k detekci objektu.

Důležitou částí je rozdělení vstupních dat (obrazů) tak, aby tvořila vzory s pozitivním nálezem (pozitivní obraz) a vzory neobsahující objekt zájmu, např. lidské postavy. Výběr trénovacích dat je složitý a zdlouhavý proces. Část pozitivních obrazů zastihuje osoby téměř za ideálních podmínek ve vzpřímeném, čelním, nebo zadním pohledu. Pro lepší natrénování detektoru je možné vybrat pozitivní obrazy z reálných scén, vyříznutím objektu zájmu. Osoby v těchto snímcích již nejsou zachyceny v ideální pozici. K takto vybraným pozitivním snímkům je ještě možné vytvořit zrcadlově obrácené kopie.

Nedílnou součástí takovéto databáze jsou i testovací obrazy. Tyto obrazy nesmí být vstupem trénovacího algoritmu a slouží k jeho otestování.

K dispozici je množství databází obsahující takto rozdělená trénovací data. Mezi nejznámější patří volně šiřitelná databáze MIT pedestrian dataset, ke stažení [http://cbcl.mit.edu/software-datasets/PedestrianData.html]. Tento dataset obsahuje obrázky s rozlišením 64x128 px s postavami ve vzpřímeném postoji a přímém pohledu. Reálné scény jsou však rozmanité na pózy osob. Pro vytvoření robustního detektoru je třeba mnohem rozmanitějších vstupních dat. Pro tento účel byl vytvořen dataset INRIA, který obsahuje osoby ve vzpřímené pozici a v různých pózách. Dataset INRIA je opět volně stažitelným na [http://pascal.inrialpes.fr/data/human/].

Část trénování nebude dál v této práci rozebírána, protože se jedná o komplexní a zdlouhavý proces. Více o samotném procesu trénování a softwaru SVMlight na stránkách [http://svmlight.joachims.org/].

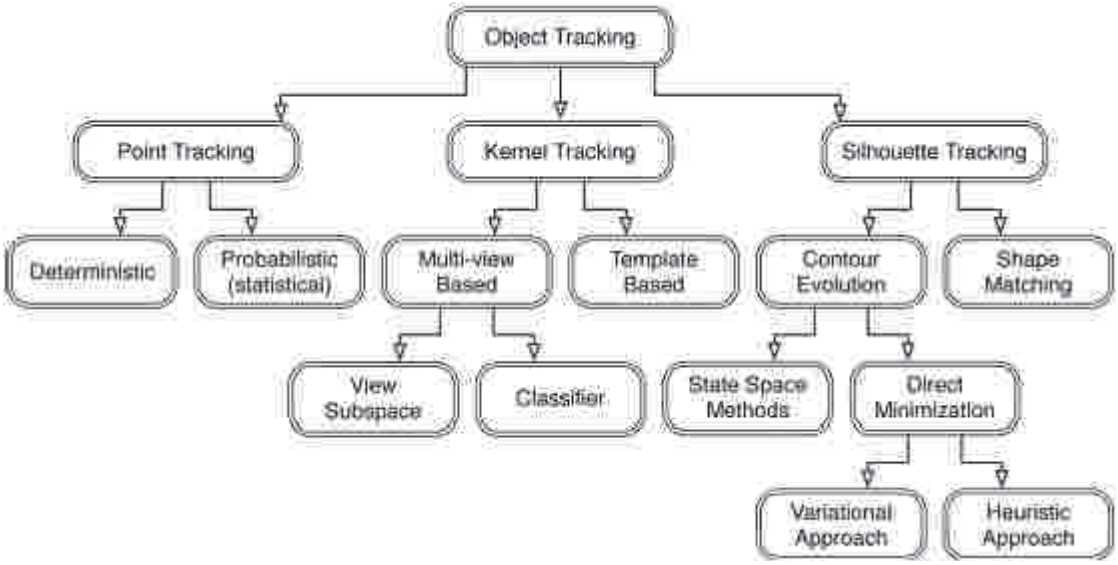
1.5.3 Zhodnocení detekce postav založené na HOG

Pro zhodnocení výkonu algoritmu je třeba předem definovat parametry a vlastnosti snímané scény. V případě detekce osob ve vnitřních prostorách je snímací zařízení často umístěno v podstropové výšce tak, aby bylo zajištěno co největší pokrytí snímané scény. Osoba se ve většině případů pohybuje v blízkosti nebo ve střední vzdálenosti od snímacího zařízení. Rozlišují se tedy objekty blízké, středně vzdálené a vzdálené. V množství aplikací se osoba nachází ve středně až blízké vzdálenosti ke kameře. Velikost, kterou osoba zaujímá v obraze v pixelech, je úměrná vzdálenosti ke kameře. Při rozlišení kamery 640 x 480 zaujímá středně vzdálená osoba 100 a více pixelů na výšku zatímco osoba v kratší vzdálenosti zaujímá přibližně 130 pixelů a více. Dalším faktorem, který je třeba při srovnání brát v potaz je procento vzniklého překrytí. Ve vnitřních prostorách je osoba často zacloněna jinou procházející osobou, ve venkovních prostorách naopak cizími objekty jako jsou například projíždějící auta, stromy nebo značky.

HOG detektor výrazně překonává detektory implementující Haarovy vlnky, PCA-SIFT a tvarem orientované detektory (shape context). Důkladné srovnání provedli Dalal a Triggs ve své práci za použití Recall-Precision (RP) a Average-Precesion (AP) křivek. Srovnání bylo provedeno na vytvořeném datasetu s 2300 označených osob. HOG detektor dosahoval nejlepších výsledků na menší, až střední vzdálenosti s minimálním překrytím [2],[3],[4].

1.6 Sledování objektů

Cílem trasovacího algoritmu je vytvořit časovou trajektorii pohybu sledovaného objektu na základě jeho pozice v každém snímku. Trasovaní je dále rozděleno do dvou skupin. První skupinu tvoří algoritmy, které pracují čistě s regionem nalezeným některým z detekčních mechanismů. Lokace tohoto regionu závisí pouze na aktuálním měření a přesnosti detekčního algoritmu. Druhou skupinu tvoří složitější probabilistické algoritmy, které na základě znalosti pozice v minulém kroku, estimují pozici objektu v kroku následujícím. Společným rysem těchto skupin je reprezentace objektu, která čistě závisí na použitém detektoru. Typ pohybu a možné deformace, které objekt podstupuje, limituje zvolený model reprezentující objekt. Pokud je známa pouze souřadnice nálezu, jedná se o prostý translační pohyb. Parametrické pohybové modely jsou užity v případě, že je znám geometrický tvar objektu. Takovýto objekt je označován jako pevný. Složitější modely objektu, jako je silueta nebo kontura, patří mezi nejpřesnější reprezentace. V takovémto případě mohou být použity jak parametrické, tak i neparametrické modely specifikující jejich pohyb [18].



Obr. 1.13: Strom dělení jednotlivých sledovacích algoritmů [18]

1.6.1 Problém korespondence

Při pohybu objektu v sekvenci snímků mohou nastat následující případy:

- Dojde k překrytí objektů
- Dojde ke ztrátě informace o pozici objektu s následným objevením se na jiném místě
- Ve sledované scéně se nachází více objektů, čímž vzniká problém nejednoznačnosti přiřazení
- Chyby vzniklé při snímání (rozmazání objektu při rychlejších pohybech - vznik nových objektů)
- Chyby vzniklé při detekci objektu

V případě detekce osob v davu může snadno nastat situace, kdy dojde ke křížení trajektorií a tím i překrývání objektů. Tyto situace je možné řešit několika způsoby. Pro určení korespondence odpovídajícího bodu v sekvenci snímků je možné využít znalost o velikosti, rychlosti a směru pohybu. Pokud například dojde ke křížení trajektorií při pohybu osob, dá se předpokládat, že tyto objekty zachovají původní směr a rychlost. Ve 3D scéně navíc objekty mění svou velikost při pohybu směrem k nebo od snímacího zařízení. Tyto změny lze využít při určování korespondence objektu v následujícím snímku [11].

Dalším způsobem je ve vzhledu nebo pohybu objektu nalézt významnou informaci, která poslouží k identifikaci. Při identifikaci stejné osoby na následujícím snímku mohou posloužit příznaky jeho vzhledu (appearance features). Protože s největší pravděpodobností nedojde ke změně vzhledu lidského zevnějšku v sekvenci po sobě jdoucích snímků,

lze této informace využít. Užitím barevné kamery lze například určit barevný histogram sledované osoby. Pokud histogram objektu na následujícím snímku nekoresponduje s histogramem předchozího objektu, s největší pravděpodobností se nejedná o tutéž osobu.

1.6.2 Kalmanův filtr

Určení pozice objektu (měření) je často zatíženo chybou vzniklou šumem. V takovémto případě je možné použít statistické metody sledování objektu. Stav systému, v tomto případě pozice objektu, je predikována na základě měření a známého modelu šumu.

Kalmanův filtr byl vyvinut v 60. letech k filtraci šumu v elektrických signálech a své uplatnění nalezl i v aplikacích počítačového vidění. Jedná se o dynamický filtr. Důležitou součásti Kalmanova filtru je znalost modelu systému, na jehož základě je vytvořena predikce následujícího stavu. Predikce příštího stavu, tedy oblast daná souřadnicemi středu, případně i velikostí objektu, vyznačuje pravděpodobný výskyt objektu. Stav filtru je v každém kroku aktualizován ze získaného měření. Není tedy třeba uchovávat historická data. Pokud není možné stav filtru aktualizovat (ztráta objektu), pokračuje se v predikci pro další krok. Při opětovném nalezení objektu je provedena korekce současné polohy (na základě predikce a měření) [1][11].

1.6.3 Aplikace Kalmanova filtru

V souvislosti s Kalmanovým filtrem hovoříme o třech typech pohybu. Dynamický pohyb je takový, u kterého lze na základě znalosti stavu systému v okamžiku t jednoznačně určit následující stav systému v čase $t+1$. Druhým pohybem je pohyb řízený, jehož směr a rychlost je známá, avšak o stavu systému nejsou dostatečně přesné informace. S tímto pohybem se lze setkat u robotických systémů. Poslední kategorií pohybu je nahodilý pohyb. Nahodilý pohyb obsahuje šum jehož příspěvky nejsou známy a nepodléhají žádnému řízení.

Předpokládejme tedy, že se jedná o druh Gaussova šumu (nahodilá chůze) nebo o šum jež lze jako Gaussův modelovat. Dále předpokládejme že x je stav, z je měření, w je procesní šum a v je šum měření. Šum w a v je nezávislý na stavech a měření. Pak dostáváme [1]

$$
\begin{aligned}
x_{k+1} &= Ax_k + w_k \\
z_k &= Hx_k + v_k \\
w_k &\approx N(0, Q_k) \\
v_k &\approx N(0, R_k) \\
\overline{P_k} &= E\left[\overline{e_k}\overline{e_k^T}\right], \overline{e_k} = x_k - \overline{X_k} \\
P_k &= E\left[e_k e_k^T\right], e_k = x_k - X_k,
\end{aligned} \tag{1.10}
$$

kde P je kovariační matice. Kalmanův filtr predikuje stav x v čase $k+1$, přičemž opravuje predikci užitím měření z v tomto čase, podle následujících rovnic

Aktualizace stavu (predikce):

$$
\begin{aligned}
\overline{x_{k+1}} &= A_k X_k \\
\overline{P_{k+1}} &= A_k P_k A_k^T + Q_k,
\end{aligned} \tag{1.11}
$$

Aktualizace měření (korekce):

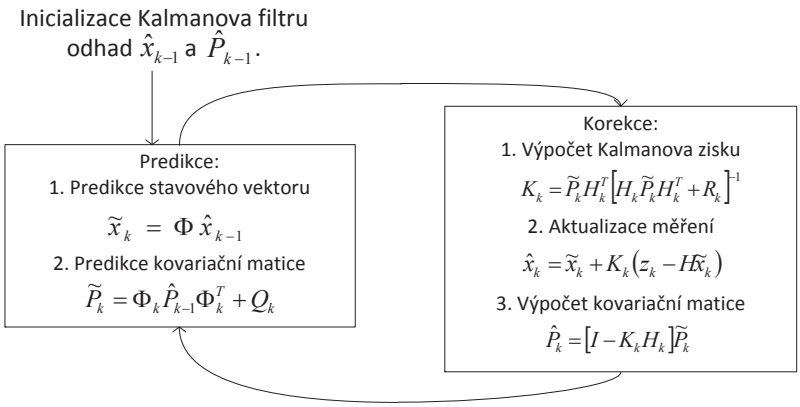
$$
\begin{aligned}
K_k &= \overline{P_k} H_k^T \left(H_k \overline{P_k} H_k^T | R_k\right)^{-1} \\
X_k &= \overline{X_k} + K_k\left(z_k - H_k \overline{X_k}\right) \\
P_k &= (I - K_k H_k) R_k,
\end{aligned} \tag{1.12}
$$

kde K je Kalmanův zisk a veličiny s pruhem jsou predikované hodnoty. Detailní odvození rovnic výše v [1].

Kalmanův filtr je třeba inicializovat. V čase t=0 neznáme předchozí stav systému. Proto se jako stavové proměnné použijí hodnoty aktuálního měření. Zároveň se inicializují matice šumu a Kalmanova zisku. Tyto budou v průběhu výpočtů postupně měněny.

Na obrázku 1.14 jsou znázorněny jednotlivé kroky výpočtu kalmanových matic. Predikce nového stavu se skládá z kroku aktualizace a predikce nových hodnot.

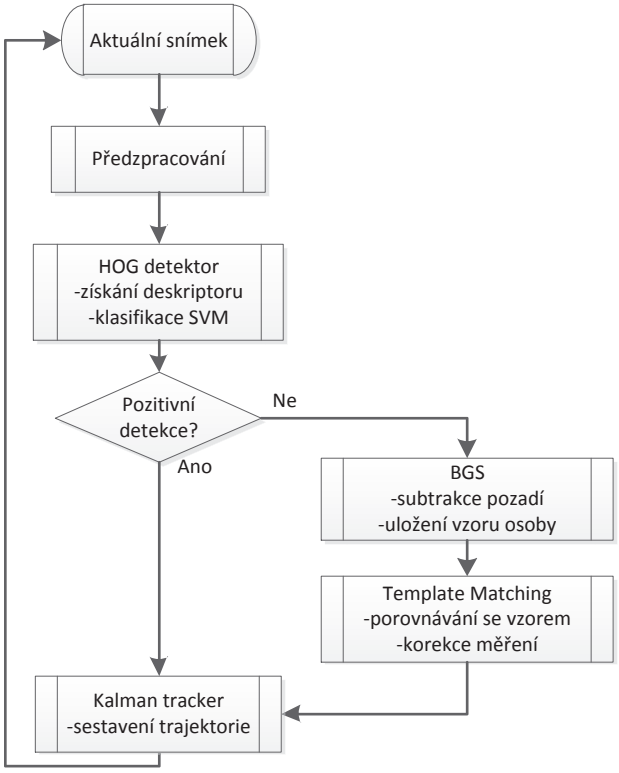
Jak již bylo řečeno, Kalmanův filtr je určen pro lineární systémy. Největší výhodou v oblasti sledování objektu v počítačovém vidění je schopnost predikce. V případě že dojde ke ztrátě informace o poloze objektu, Kalmanův filtr predikuje novou pozici s určitou pravděpodobností. Zároveň vnáší do systému dynamiku, to znamená, že výsledná trajektorie pohybu není zašuměná. Náhradou Kalmanova filtru můžou být Particle filtry nebo nelineární rozšíření Kalmanova filtru (Extended Kalman Filter).



Obr. 1.14: Jednotlivé kroky výpočtu kalmanova algoritmu.

2 VYBRANÉ METODY

V této kapitole budou popsány jednotlivé metody, které byly pro tuto aplikaci vybrány a implementovány. Při zpracování je užito teoretických poznatků z kapitoly 1, jež obsahuje souhrn moderních přístupů, které přinesl výzkum poslední doby. Jádro detekčního algoritmu tvoří HOG detektor a lineární SVM klasifikátor viz kapitola 1.4. Detekce osob je prováděna pro každý snímek sledované scény. Výstupem detektoru je ohraničené okno nálezu, jež je popsáno jeho souřadnicemi, výškou a šířkou. V aplikační vrstvě nad detektorem jsou dále použity metody, které zvýšili spolehlivost trasování pohybu objektů. Celý algoritmus tedy tvoří kombinace různých přístupů.



Obr. 2.1: Diagram volání jednotlivých metod

Diagram na obrázku 2.1 znázorňuje větvení programu a užití jednotlivých metod. HOG detektor dává počáteční odhad pozice objektu. V případě že detektor selže a zároveň byl objekt v předchozích snímcích detekován, zastupují detektor metody porovnávání se vzorem a trasování „blobů“. Blobem je každý pohyblivý region, objekt, jež neodpovídá modelu pozadí. Výstup detektorů je dále zpracováván algoritmem trasování, který sestavuje výslednou trajektorii pohybu objektu. Pro trasování byl zvolen lineární Kalmanův filtr. Tento filtr do systému vnesl potřebnou dynamiku a umožnil tak sledovat objekty, jejichž měření bylo na krátkou dobu ztraceno.

2.1 Histogram orientovaného gradientu

Nejdůležitějším prvkem a stavebním blokem algoritmu je zvolený detektor. Detektor na základě Histogramu orientovaného gradientu je od počátku svého vzniku koncipován pro detekci osob. Lze ho však užít k detekci dalších objektů, které v průběhu sledování nemění svůj tvar. Tento detektor byl vybrán vzhledem k jeho snadné implementaci a celkové robustnosti řešení. Výkon a výsledky detektoru výrazně překonává současné přístupy [4]. Mezi výhodami detektoru je i samotná implementace v knihovně OpenCV.

V následujících kapitolách budou popsány jednotlivé operace HOG detekčního algoritmu. Jak již bylo řečeno v kapitole 1.4, metoda vyhodnocuje vhodně normalizovaný histogram gradientů získaných pro detekční okno, které je po obraze posouváno. Výstupem detektoru je vektor deskriptoru, jehož složky tvoří dané histogramy sestavené pro každou pozici detekčního okna. Výkon a rychlost detekce silně závisí na parametrech detektoru a vstupním rozlišení obrazu.

Celý řetězec detekce se skládá z následujících operací:

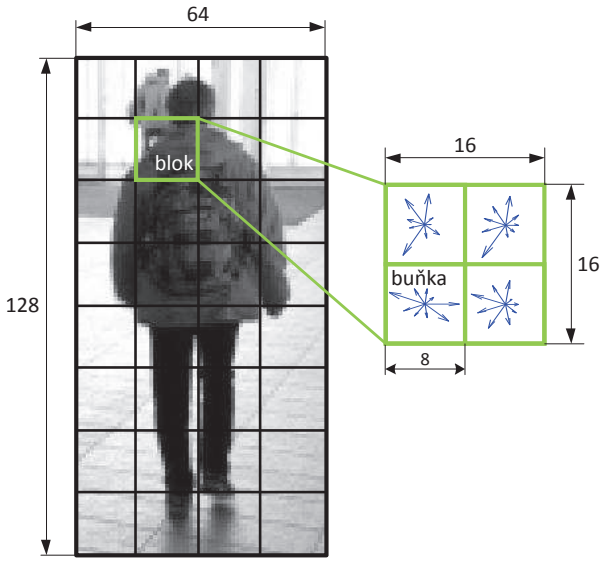
- Globální normalizace jasu
- Kalkulace gradientů
- Sestavení váhovaného histogramu gradientů
- Lokální normalizace kontrastu
- Sestavení deskriptoru z jednotlivých bloků

2.1.1 Parametry detektoru

Kvalitu deskriptoru a tím i jeho následnou klasifikaci určují parametry, které popisují okno deskriptoru a jeho posun po obraze. V implementaci algoritmu bylo použito okno o velikost 128x64 px. Velikost okna proporcionálně odpovídá velikosti osob. Okno s těmito rozměry lze zároveň snadno dělit mocninami dvou na jednotlivé bloky. Velikost bloku tedy bude $2^n x 2^n$. Pro účel aplikace byla velikost bloku zvolena 16x16 px. To je kompromis mezi rychlostí a výkonem detektoru. Tento blok je dále rozdělen na jednotlivé buňky, pro které je sestavován histogram orientovaného gradientu. Tyto buňky mají velikost 8x8px a udávají krok bloku.

V případě detekce různě velikých objektů je deskriptor nutné sestavovat pro různá měřítka detekčního okna. Toto měřítko je dalším parametrem detektoru a udává kolikrát se zvětší detekční okno v dalším průchodu obrazem. V samotném algoritmu byl použit koeficient $scale = 1.13$. Tato hodnota byla určena experimentálně opět jako kompromis mezi výkonem a rychlostí algoritmu. Při zmenšení hodnoty koeficientu scale na $scale = 1.05$, rychlost algoritmu klesá přibližně 2x. Problémem detekce pro různá měřítka je, že vzniká

několik překrývajících se detekovaných regionů. Tyto regiony často náleží jednomu objektu a je nutné je seskupit v jeden. Seskupení je provedeno na základě podobnosti regionů a jejich lokace. Velikost a lokace výsledného regionu je dána jako průměr všech detekovaných oblastí.



Obr. 2.2: Detekční okno rozdělené na jednotlivé bloky 16x16px vlevo. Vpravo blok rozdělen na buňky o velikosti 8x8px.

2.1.2 Velikost obrazu a rychlost algoritmu

Vstupem algoritmu je RGB obraz o velikosti 640x480 pixelů. Tento rozměr je typický pro mnoho kamer. Protože je deskriptor pořizován pro celý obraz, jeho velikost výrazně ovlivňuje rychlost algoritmu. Požadavkem aplikace je samozřejmě i její real time běh. Parametry deskriptoru a velikost vstupního obrazu byly voleny tak, aby byla detekce co nejspolehlivější pro různá měřítka detekovaných objektů. Velikost vstupního obrazu je pro zrychlení zmenšena. Zároveň je třeba, aby byla odezva detektoru pozitivní jak pro objekty v popředí, tak i v pozadí sledované scény. Jak bylo řečeno v kapitole 1.5.3, detektor dosahuje nejlepší odezvy pro středně vzdálené a blízké objekty. Obraz tedy lze zmenšovat pouze do meze dané typem sledované scény.

Tabulka 2.1 ukazuje naměřené rychlosti algoritmu pro různá rozlišení. Vstupní obraz byl upraven tak, aby postava na něm zaujímala stejnou velikost v px pro všechna testovací rozlišení. Rychlost algoritmu klesá lineárně se zvětšujícím se rozlišením obrazu. Scale faktor v tabulce níže znamená, kolikrát je vstupní obraz zmenšen oproti původní velikosti 640x480 px. Obrázek 2.3 zobrazuje zpracovávané obrazy s ohraničeným objektem nálezu.

Obr. 2.3: Rychlost algoritmu pro různá rozlišení vstupního obrazu.

Obrázek 2.3	Scale faktor	Rozlišení	Rychlost
a)	0,5	320x240 px	76 ms
b)	0,7	448x336 px	172 ms
c)	0,8	512x384 px	384 ms
d)	0,9	576x432 px	397 ms
e)	1,0	640x480 px	432 ms

Tab. 2.1: Průměrné rychlosti detekce pro daná rozlišení (metoda HOG).

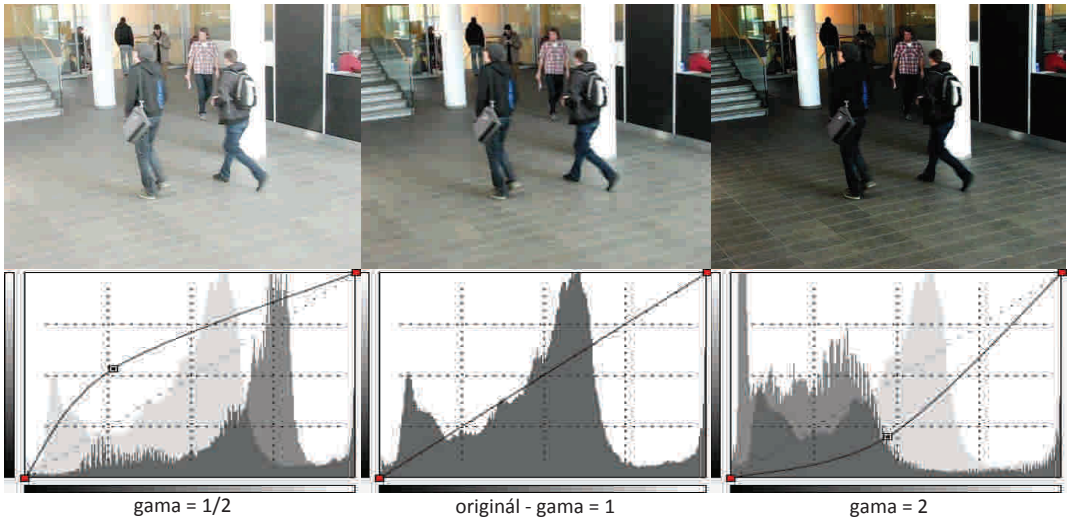
V navrhované aplikaci byl vstupní obraz zmenšen na rozlišení 480x360. Tomu odpovídá scale faktor 0,75 a rychlost 220ms. Toto rozlišení bylo voleno jako kompromis mezi rychlostí algoritmu a pozitivními detekcemi objektu po celé scéně. Schopnost detekce pro daná rozlišení silně závisí na typu sledované scény, tedy v jaké vzdálenosti se detekovaný objekt nachází. Rozlišení obrazu lze v aplikaci měnit. Rychlost algoritmu byla testována na sestavě Windows 7 x64, Intel Core i5 2,67GHz, OpenCV 2.3.1.

2.1.3 Normalizace jasu

Prvním krokem detekce je normalizace jasu všech kanálu RGB obrazu. Jedná se o často používanou korekci, kdy jsou jasové hodnoty jednotlivých kanálů přepočítány podle gama křivek. V tomto případě je gama křivka stejná pro všechny kanály. Tato korekce snižuje náchylnost vůči rozdílným světelným podmínkám. Obrázek 2.4 ukazuje gama korekce obrazu pro hodnoty $gama = 0.5, 1$ a 2 s jejich odpovídajícími gama křivkami. Gama korekce je prováděna podle následujícího vztahu

$$R(x,y) = I(x,y)^{gama}, \tag{2.1}$$

kde I odpovídá vstupnímu obrazu a R je obraz výstupní. Gamma korekce je prováděna pro hodnoty $gama = 2$, čímž dojde k zvýšení kontrastu obrazu [2],[3].



Obr. 2.4: Gama korekce s příslušnými jasovými křivkami.

2.1.4 Výpočet gradientů

Před vypočtením gradientů v jednotlivých osách obrazu x, y není vstupní obraz žádným způsobem zpracováván (s výjimkou gama korekce). Kvalita vypočtených gradientů silně ovlivňuje výkon detektoru. Gradienty jsou počítány pomocí jednoduché sobelovy masky reprezentující první derivaci. Výsledné gradienty a jejich směry znázorňuje obrázek 2.5. V tomto případě jsou gradienty počítány pro obraz o rozlišení 128x64px [3].

Obr. 2.5: Výpočet velikosti a směru gradientů. a) originální obraz, b) absolutní hodnoty gradientů, c) velikost a směr gradientů pro část hlavy a ramen

2.1.5 Sestavení histogramu gradientů pro detekční okno

Po vypočtení gradientů a normalizaci obrazu následuje sestavení jejich histogramů. Jak již bylo řečeno, algoritmus staví na pohyblivém detekčním okně. Toto detekční okno je rozděleno na jednotlivé bloky, pro které je váhovaný histogram sestavován. Obrázek 2.6 zobrazuje vstupní obrazy a jím odpovídající histogram gradientů pro jednotlivé bloky. Detekční okno o velikosti 128x64px je rozděleno na bloky, jejichž velikost je 8x8px. Jednotlivé gradienty, tedy ty gradienty příslušející bloku, přispívají svojí váhou do výsledného 9 kanálového histogramu. Histogramy na obrázku jsou pro zjednodušení počítány pouze pro jeden kanál obrazu (šedotónový obraz). Lepších výsledků dosahuje deskriptor všech kanálu barevného systému RGB. Výsledný deskriptor je vektor hodnot jednotlivých histogramů.

Velikost okna a bloků byla volena experimentálně. Hrubý krok rozdělení bloků způsobuje selhání detekce v případě vzdálených objektů, které jsou vůči pozadí nekontrastní. To je dáno nevýraznou změnou jednotlivých obrysových gradientů.

Obr. 2.6: Sestavení histogramů gradientů pro detekční okno o velikosti 128x64px. Pro zjednodušení je histogram sestavován pro šedotónový obraz.

2.1.6 Normalizace histogramů bloku

Histogramy sestavené v jednotlivých blocích detekčního okna nejsou nijak normalizovány. Normalizace je prováděna metodou L2-norm, respektive její modifikovanou verzí L2-Hys. Nechť je v nenormalizovaný vektor deskriptoru bloku, $||v||_k$ jeho k-tá normalizovaná složka a ε malá konstanta pro ošetření dělení nulou ($\varepsilon = 1e^{-2}$). Potom je L2-norm normalizační konstanta nf dána vztahem

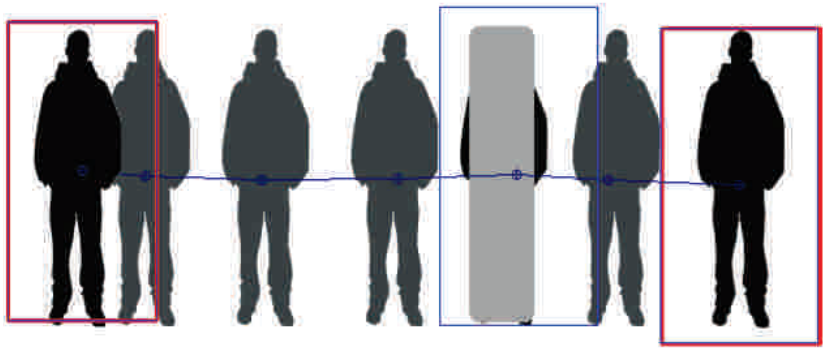
$$nf = \frac{v}{\sqrt{||v||_2^2 + \varepsilon^2}}, \tag{2.2}$$

následována ořezem maximálních hodnot v na hodnotu 0.2. Vektor deskriptoru je zpětně renormalizován což dává modifikovanou normalizační metodu L2-Hys. Tato normalizace zvyšuje výkonnost detektoru. Normalizační metoda L2-Hys je implementována v OpenCV [1],[2],[3],[14].

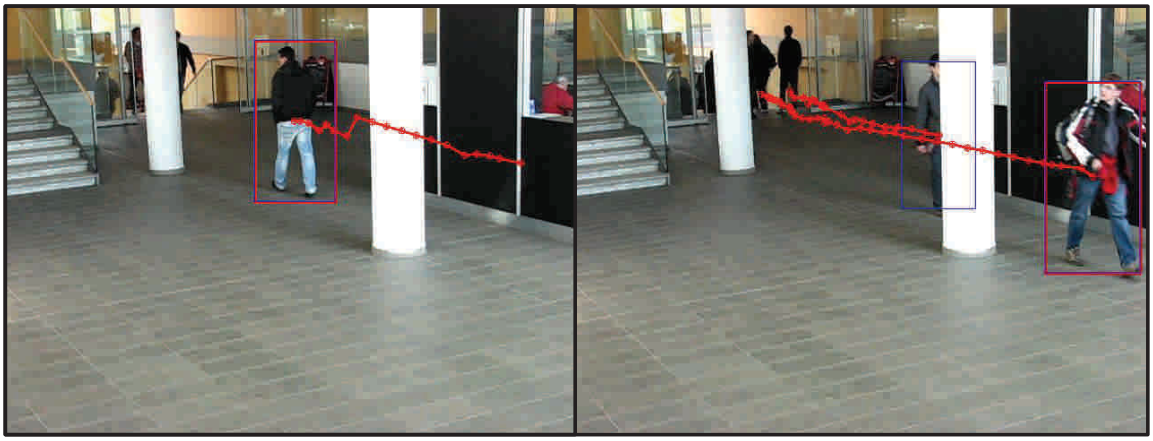
Výsledný vektor deskriptoru je získán sestavením všech normalizovaných histogramů jednotlivých bloků pro celé detekční okno.

2.2 Trasovací algoritmus s využitím Kalmanova filtru

Pro trasování pohybu detekovaných osob je použit algoritmus jehož jádrem je jednoduchý lineární Kalmanův filtr viz kapitola 1.6.2. Tento algoritmus byl vybrán z důvodu jeho jednoduché implementace a celkové robustnosti řešení. V případě selhání detekce Kalmanův filtr predikuje pravděpodobnou pozici objektu aktualizací stavového vektoru. Z výstupu Kalmanova filtru, tedy predikované pozice a velikosti objektu, je sestavena výsledná trajektorie jednoduchým spojením všech bodů nálezu patřící sledované osobě, jak ukazuje následující obrázek 2.8. V tomto testovacím obrázku je nasimulováno překrytí objektu. Červený rámeček znázorňuje výstup detektoru, modrý je predikovaná pozice.



Obr. 2.7: Výsledná trajektorie pohybu s ukázkou selhání detektoru (simulace).



Obr. 2.8: Výsledná trajektorie pohybu s ukázkou selhání detektoru (Reálná data).

V testovacích snímcích pořízených v prostorách budovy Kolejní 4 se často stávalo, že byl sledovaný objekt zacloněn ať už podpěrnými sloupy nacházející se v přístupové hale nebo jinou osobou procházející v popředí. Bez využití Kalmanova filtru nebylo možno sestavit kompletní trajektorii pohybu přes celou sledovanou scénu. Na obrázku 2.8 jsou znázorněny situace, kdy dojde k ztracení a opětovnému nalezení sledované osoby. Červený rámeček je opět výstup detektoru, modrým je naznačena estimovaná pozice Kalmanovým filtrem. Trajektorie je zobrazena jako plná červená čára [5],[20].

2.3 Metody užité pro zvýšení robustnosti systému

Odezva HOG detektoru občas selže. Proto bylo nutné implementovat metody detekce, které zvýší celkovou robustnost řešení. Jednou z možností je sledování pohyblivých regionů v obraze. U tohoto řešení však nastává problém v případě, kdy se jednotlivé separované regiony spojí v jeden. Ke spojení jednotlivých regionů dojde například při křížení trajektorií jednotlivých osob nebo sledování většího počtu osob blízko sebe. V tomto případě nelze jednoznačně definovat pozici osoby v obraze.

Druhým řešením je možnost ukládání jednotlivých vzorů osob v průběhu detekce a poté je porovnávat se vstupním obrazem. Tuto metodu je třeba inicializovat, což je prováděno samotným HOG detektorem. Při pozitivním nálezu dochází k postupnému ukládání vzorů, podle kterých je v případě výpadku HOG detektoru prováděno porovnávání se vzorem. Tato metoda má v případě nestatických snímků své nevýhody. Pohyblivý objekt totiž mění svůj vzhled a pozici, zatímco vzor zůstává tak, jak byl pořízen při poslední pozitivní odezvě HOG detektoru.

2.3.1 Kombinace metod subtrakce pozadí a porovnávání se vzorem

Subtrakce pozadí je prováděna pro každý snímek sledované scény metodou popsanou v kapitole 1.1.2. Výstupem metody je binární obraz obsahující pohyblivé regiony. Těchto regionů je dále využito v kombinaci s výstupem metody porovnávání se vzorem.

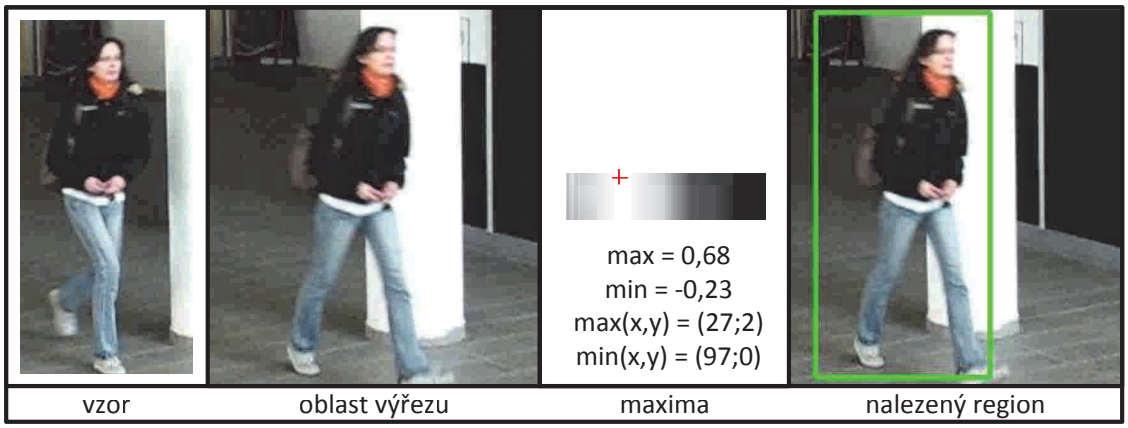
V průběhu detekce je každá detekovaná osoba uložena do databáze vzorů. Při selhání HOG detekce nastává situace, kdy detektor zastupují dvě výše zmíněné metody. Aby nedošlo k výraznému zpomalení algoritmu, je kolem každé osoby vytvářena větší oblast. Tato oblast vzniká upravením výstupu Kalmanova filtru tak, že pokrývá větší část ve směru předpokládaného pohybu osoby. Tento výřez ukazuje obrázek 2.9.

Obr. 2.9: Zvětšená oblast pro porovnávání se vzorem s trajektorií pohybu (červená). Výstup Kalmanova filtru (modrá).

Dalším krokem pokročilé detekce je prohledávání této oblasti a hledání maxima pro daný vzor. K tomu je využita normovaná korelace jak je popsáno v kapitole 1.2. Tato metoda vykazovala na testovacích videích největší úspěšnost. Míra shody každého pixelu se akumuluje v pomocném obraze. Z tohoto pomocného obrazu je metodou hledání maxima navrácena hodnota v místě největší shody.

Obrázek 2.10 zobrazuje hledaný vzor vlevo, uprostřed prohledávanou oblast a vytvořené akumulované pole, vpravo poté výsledek detekce po nalezení maxima v pomocném poli. Maximum je v pomocném poli znázorněno červeným křížkem. Nalezené souřadnice jsou posunuty relativně vůči celému obrazu. Absolutní pozice je dána jako součet souřadnic výřezu a nalezeného maxima v akumulovaném poli.

Tato metoda selhává v případě špatného určení předpokládané pozice a velikosti výřezu s hledanou osobou. Pokud se osoba ve výřezu nenachází, metoda vždy vrátí tu pozici, kde bylo nalezeno maximum v pomocném akumulovaném poli. Při této situaci je třeba zajistit, aby bylo trasování provedeno pouze pro tu osobu, která se po dané trase pohybuje.



Obr. 2.10: Algoritmus porovnávání obrazu se vzorem.

Proto je výstup detektoru porovnávání se vzorem dále korelován se získaným popředím metodou subtrakce „Codebook“.

V případě že detektor vrátí pozici tam, kde se v daném okamžiku nenachází žádný pohyblivý objekt, algoritmus tuto odezvu jednoduše ignoruje. Tím je zajištěno sledování objektu, který se ve scéně pohybuje a zároveň patří sledované osobě. Tuto situaci ilustruje obrázek 2.11. Špatná odezva detektoru je dána především proto, že byl vzor pořízen s velmi výrazným pozadím, které tvoří bílý podpěrný sloup. Z toho tedy vyplývá, že je metoda porovnávání se vzorem nejen závislá na měřítku objektu, ale také na jeho pozadí. Hledaný vzor na obrázku vlevo neodpovídá skutečnému měřítku.

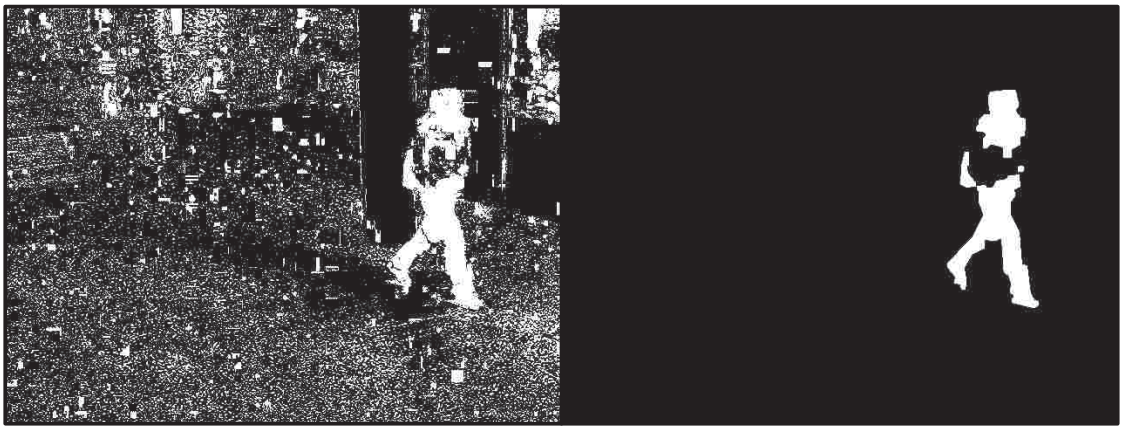
Obr. 2.11: Selhání detektoru při porovnávání se vzorem.

2.3.2 Subtrakce pozadí metodou Codebook

Metoda Codebook je z hlediska výpočetní náročnosti velmi rychlá a pro danou aplikaci naprosto dostačující. Ve vnitřních prostorách za normálních okolností nedochází tak často ke změnám světelných podmínek. Proto není třeba pozadí modelovat složitým a výpočetně náročným modelem (GMM, AGMM). Aktualizace modelu pozadí je prováděna jednou za 20 snímků. Čas, který uplyne mezi jednotlivými aktualizacemi, silně závisí na podmínkách panujících ve sledované scéně. Rychle měnící se pozadí vyžaduje častější aktualizaci modelu.

Získané popředí obsahuje mnoho šumu. Toho je třeba se zbavit. Segmentace popředí spočívá v aplikování jednoduchých morfologických operací jako je *eroze* následovaná *dilatací*. Rozdíl mezi vhodně a nevhodně zpracovaným popředím ukazuje obrázek 2.12.

Subtrahované popředí slouží jako maska, podle které je určeno, zdali se v okolí poslední známé pozice vyskytuje pohyblivý region, který patří dané osobě. V tomto případě může nastat problém při křížení trajektorií osob nebo sledování skupiny lidí. Bylo by vhodné brát v úvahu směr a rychlost pohybu segmentovaného popředí. Korekce Kalmanova filtru poté bude provedena pouze pro ty detekované regiony, které se v čase $t-1$ pohybovaly stejným směrem a stejnou rychlostí jako v čase t.



Obr. 2.12: Obraz špatně segmentovaného popředí vlevo s provedením morfologických operací vpravo.

3 POUŽITÝ HARDWARE A TESTOVACÍ DATA

V první fázi projektu byla tato práce navrhovaná pro užití externího kamerového zařízení. Jednalo se o kameru MODI AdoSpy s jednou kamerou přehledovou a druhou detailní. Typ zpracovávané aplikace však takto složité zařízení nevyžaduje a je možné ho nahradit libovolnou web kamerou s rozlišením 640x480px. V ideálním případě by měl být algoritmus schopen běhu s libovolným snímacím zařízením, avšak za podmínky barevného snímání sledované scény. U pořizovacích zařízení je vhodné zajistit neměnné parametry snímaní jako je clona, počet snímků za sekundu a vyvážení bílé. Algoritmus byl testován při použití klasické integrované web kamery notebooku, externího fotoaparátu CANON a za pomocí veřejných databází obsahující testovací data.

3.1 Vývojářská stanice

Celá aplikace byla vyvíjena a testována na sestavě s procesorem Intel® Core™ i5, CPU M480, 2,67GHz, 4GB RAM s nainstalovaným systémem Windows 7 Professional (Service Pack 1) a vývojovým prostředím Microsoft Visual Studio 2010. Veškeré výpočty jsou vykonávány pouze na procesoru. Výpočetní výkon může být v dalších fázích projektu soustředěn i na procesor grafické karty.

3.2 Kamera MODI

Systém MODI AdoSpy je unikátní kompaktní elektro-optický systém umožňující přehledové sledování velkých ploch a zároveň velmi rychlé pořizování detailních snímků a sekvenci snímků z těchto ploch. Tento systém obsahuje dvě kamery. Jedna je přehledová a druhá detailní. Detailní kamera je schopna ve sledované scéně rychle zachytit velké množství detailních snímku, díky konstrukčnímu řešení snímání přes odrazové pohyblivé zrcátko. Tento systém nalezne své uplatnění při úlohách detekce obličejů či osob ve velkých prostorách.

V této fázi práce není kamera MODI součástí implementovaného algoritmu. Tato kamera může být v budoucnu použita například při určování směru pohledu osob. Snímky získané přehledovou kamerou mohou být zpracovávány navrženým algoritmem, jehož výstupem budou souřadnice detekovaných objektů. Na základě získaných souřadnic může detailní kamera pořizovat přesnější snímky těchto pozic obsahující potřebné informace, které mohou být dále zpracovávány.

Výsledný algoritmus nebyl se systémem MODI zkoušen ani testován.

Obr. 3.1: Kamera MODI AdoSpy

3.3 Fotoaparát Canon Powershot

Pro pořizovaní testovacích sekvencí v prostorách budovy Kolejní 4 byl použit fotoaparát Canon PowerShot S3 IS. Fotoaparát je opatřen objektivem s ohniskovou vzdáleností 6.00 - 72.00 mm. S tímto zařízením bylo pořízeno celkem 6 video sekvencí ve formátu AVI o rozlišení 640x480 px, rychlostí 30 snímků za sekundu při různé ohniskové vzdálenosti. Z těchto sekvencí byly vybrány úseky, které jsou svým způsobem zajímavé z pohledu testovaní aplikace.

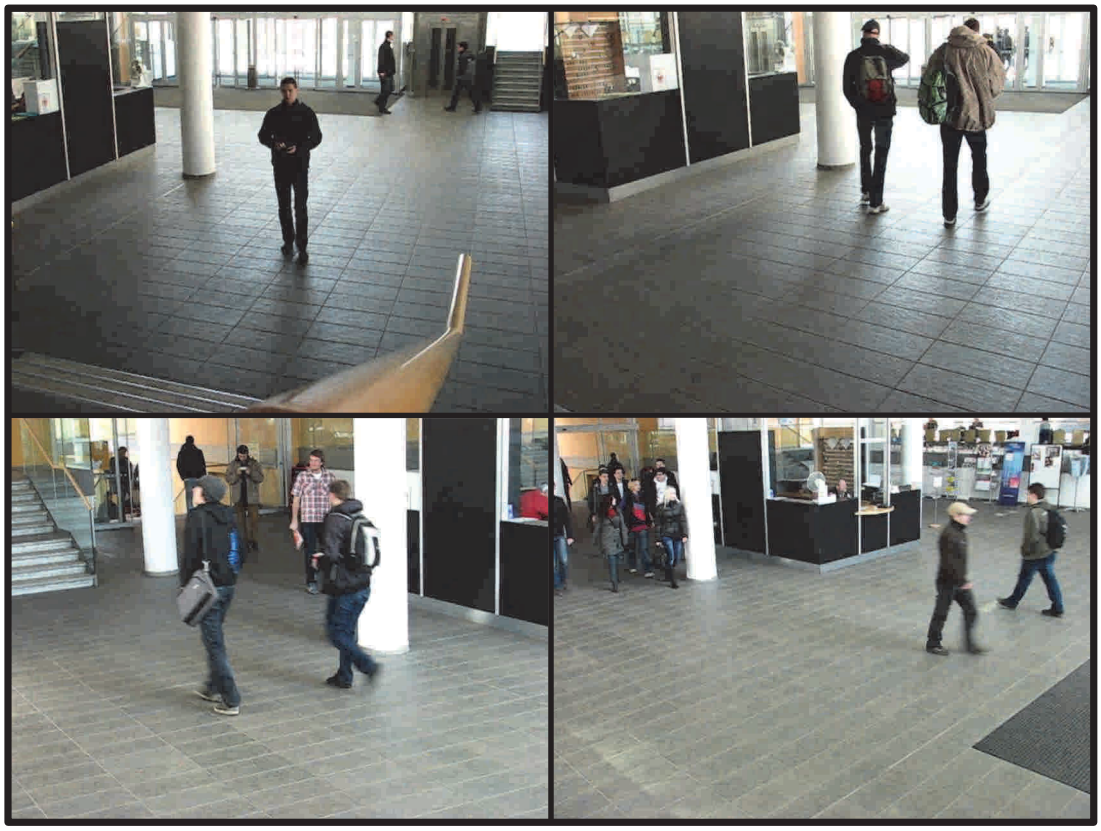
3.4 Zpracování testovacích dat

Pro lepší práci s testovacími daty byly pořízené AVI soubory rozděleny na sekvence po sobě jdoucích snímků ve formátu jpg o stejném rozlišení jako pořízené video (640x480 px). Výsledné soubory nebyly dále nijak upravovány. Při testování bylo nutné simulovat real time běh aplikace a zpracování obrazů. Toho bylo docíleno jednoduchým přeskakováním určitého počtu snímků. Počet přeskočených snímků je dán dobou trvání jedné programové smyčky. Pokud je algoritmus schopen zpracovat v průměru 4 snímky za sekundu, vstupním obrazem bude každý 8. snímek.

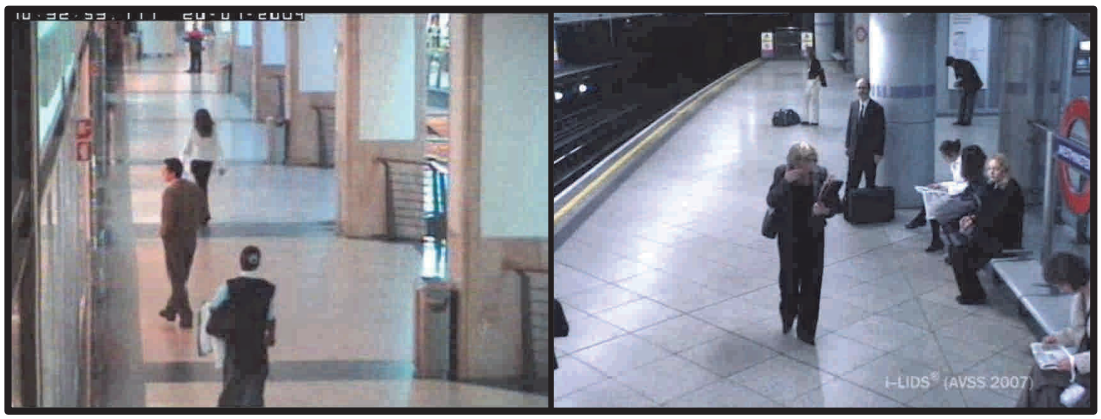
3.5 Veřejně dostupné testovací data

Algoritmus je testován jak na vlastních sekvencích tak i na sekvencích, které jsou veřejně dostupné. První sekvence snímků byla pořízena v rámci projektu „EC Funded Caviar project“, který obsahuje množství videí za různých podmínek a scénářů. Data jsou k dispozici na stránkách projektu http://homepages.inf.ed.ac.uk/rbf/CAVIAR/. Z pohledu testování navrhované aplikace jsou zajímavá data sekvencí „Lisbon Sequences“, které obsahují obrazy koridoru obchodního domu. Tuto scénu zachycuje obrázek 3.3. Dalšími daty

jsou snímky „i-Lids“, které byly pořízeny v prostorách Londýnského metra primárně pro účel testování algoritmu detekující opuštěné zavazadlo. Dataset je veřejně dostupný na stránkách projektu http://tna.europarchive.org/20100413151426/scienceandresearch.homeoffice.gov.uk/hosdb/cctv-imaging-technology/i-lids/index.html.



Obr. 3.2: Testovací snímky pořízené v budově Kolejení 4.



Obr. 3.3: Testovací snímky CAVIAR vlevo, i-Lids vpravo.

4 IMPLEMENTACE DETEKČNÍHO ALGORITMU

Algoritmus je implementován za použití programovacího jazyka C# v prostředí Microsoft Visual Studio 2010. Pro zpracování úloh počítačového vidění byla zvolena otevřená, multiplatformní knihovna OpenCV 2.3.1. Knihovna nyní obsahuje více jak 2500 funkcí, které pokrývají rozsáhlou oblast počítačového vidění. Hlavní výhodou knihovny je možnost použití všech částí ke komerčnímu využití, s velice rozsáhlou podporou ze strany programátorů a uživatelů. Protože je knihovna OpenCV psána v jazyce C/C++, bylo nutné použít wrapper knihovních funkcí.

4.1 OpenCVSharp 2.3.1

OpenCVSharp je multiplatformní wrapper pod licencí GNU Lesser GPL určený pro .NET Framework a je psán v jazyce C#. Programátor tak může používat většinu funkcí i v programových prostředích C#, VB.NET, atd. OpenCVSharp zabaluje více funkcí než ostatní dostupné wrappery, jako je SharperCV a OpenCVDotNet. Kód tohoto wrapperu je snadno čitelný a i přes menší podporu má solidní základnu uživatelů.

Knihovna je složena z několika komponent (knihoven). Stavebním blokem je dynamická knihovna *"OpenCVSharp.dll"*, která zabaluje všechny základní konstanty, struktury, výčtové typy, třídy a jejich metody z knihovny OpenCV. Pokud je v programu volána některá z funkcí, musí být tato knihovna k projektu připojena. Další důležitou knihovnou je *"OpenCVSharpExtern.dll"*. Tato knihovna se musí nacházet ve stejném adresáři jako je spustitelný soubor .exe programu. Dalšími knihovnami jsou *"OpenCVSharp.Blob.dll", "OpenCVSharp.MachineLearning.dll", "OpenCVSharp.CPlusPlus.dll"* nebo *"OpenCVSharp.Extensions.dll"*. Při implementaci je využita pouze knihovna *"OpenCVSharp.CPlusPlus.dll"*, která umožňuje volání a psaní kódu stejně jako je to v C++.

4.2 Struktura programu

Při programování bylo využito vlastností objektově orientovaného programování. Pro testování algoritmu bylo vytvořeno jednoduché grafické uživatelské rozhraní GUI z anglického „Graphical User Interface“, obsahující navigační tlačítka pro řízení chodu programu. Náhled obrazu s možností vykreslování průběhu a výstupu algoritmu je zobrazován v pravé části aplikačního rozhraní. Toto okno zaujímá většinu plochy GUI.

Program je strukturován do tříd, které pomocí svých metod zajišťují vykonávání jednotlivých částí programu. Byla vytvořena jedna hlavní třída `CvClass`, která slouží jako interface mezi GUI a s ostatními třídami sdílí některé důležité objekty a proměnné. Hlavní třída obstarává styk s periferiemi, dochází zde k deklaraci a inicializaci programových

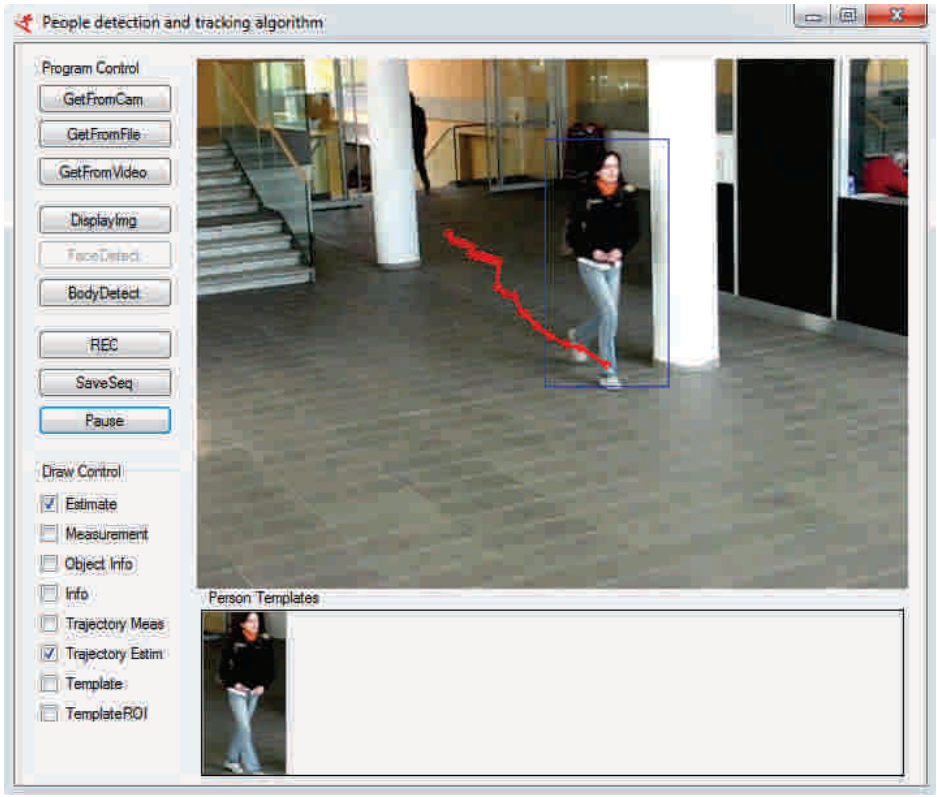
proměnných, zároveň jsou zde volány jednotlivé metody tříd nižší úrovně. Tyto třídy jsou popsány v dalších kapitolách.

Jádrem algoritmu je samotný detektor, jehož výpočty jsou prováděny v samostatném vlákně. V programu je použit mechanismus odchytávání výjimek. Provádění algoritmu tedy při vzniklých chybách nezhavaruje a je možno ho ladit, případně pokračovat v provádění dalších programových rutin.

4.3 Grafické uživatelské rozhraní

Pro snadné ovládání programu bylo vytvořeno jednoduché testovací GUI. Při spuštění programu dojde k inicializaci všech potřebných komponent. Dojde k vytvoření instance hlavní třídy `CvClass` a k volání jeho konstruktoru. Aby byla zajištěna interakce uživatele s programem během jeho zpracovávání, je hlavní `Form` obsluhován při vyvolání přerušení od časovače. Při přerušení zároveň dojde k vykreslení náhledů jako bitmapy pomocí objektu `PictureBox`.

K ovládání programu slouží tlačítka umístěná vlevo od náhledu obrazu. Vykreslování jednotlivých informací do zobrazovaného náhledu, jako je ohraničený výstup detektoru,



Obr. 4.1: Programové GUI

predikovaná pozice, trajektorie pohybu, informace o objektu, informace o počtu detekovaných osob, je řízeno vlevo pomocí zaškrtávacích tlačítek. Ve spodní části jsou v komponentě `Gridbox` zobrazovány náhledy nalezených osob. Zobrazované náhledy všech osob jsou použity při detekci pomocí hledání vzoru ve vstupním obraze.

Obrázek 4.1 ukazuje rozmístění jednotlivých komponent grafického rozhraní. K programu lze připojit externí snímací zařízení pomocí tlačítka *"GetFromCam"*. Algoritmy je možno otestovat pomocí externích testovacích obrazových sekvencí, stisknutím tlačítka *"GetFromFile"*. Tlačítkem *"GetFromVideo"* lze načíst libovolný testovací video soubor. Cesty k souborům musí být předem definovány v konfiguračním textovém dokumentu, který se nachází v programovém adresáři `Debug(Release)\conf\`. Tlačítko *"DisplayImg"* slouží k zmrazení náhledu videa, zatímco program pokračuje ve svých výpočtech na pozadí. K spuštění resp. vypnutí detekce slouží tlačítko *"BodyDetect"*. Při vypnuté detekci program pokračuje v zobrazování náhledu a pořizování dalších snímků. Nechybí ani možnost uložení videa nebo obrazové sekvence do zvoleného formátu. Souborové cesty jsou předem dány při inicializaci hlavní třídy a nelze je definovat za běhu programu.

4.4 Přehled použitých tříd

Třída	**Stručný popis**
`Class CvClass{}`	Interface mezi GUI a programem, obsluha HW (kamery), volání důležitých funkcí zajišťující chod programu, správa tříd nižší úrovně, obsahuje aktuální obrázek `IplImage` a pomocné proměnné.
`Class BodyDetect{}`	Implementuje metodu HOG detektoru a subtrakce pozadí, implementuje funkci `DetectMultiScale`, vrací regiony nálezu postav v poli `CvRect[]`, inicializace a volání metod třídy `Person`.
`Class KalmanTracker{}`	Implementuje sledovací algoritmus založený na Kalmanově filtru, inicializace matic, vrací predikovanou oblast, řeší sledování v případě ztráty měření.
`Class Person{}`	Implementuje metodu porovnávání se vzorem, obsahuje informace o objektu (pozice, velikost, rychlost pohybu), obsahuje proměnné predikce, volání trasovacího algoritmu a jeho obsluha, každý nalezený objekt má svoji instanci třídy `Person`

Tab. 4.1: Stručný popis programových tříd.

Pro snazší orientaci a volání jednotlivých funkcí jsou třídy koncipovány tak, že v nich dochází k obsluze a výpočtům pouze jejich vlastních metod. Každá třída má za úkol zpracovat pouze tu část výpočtů, ke kterým je určena. V programu může existovat několik instancí daných tříd. To závisí na množství detekovaných objektů. Základní popis jednotlivých tříd shrnuje tabulka 4.1.

4.4.1 CvClass

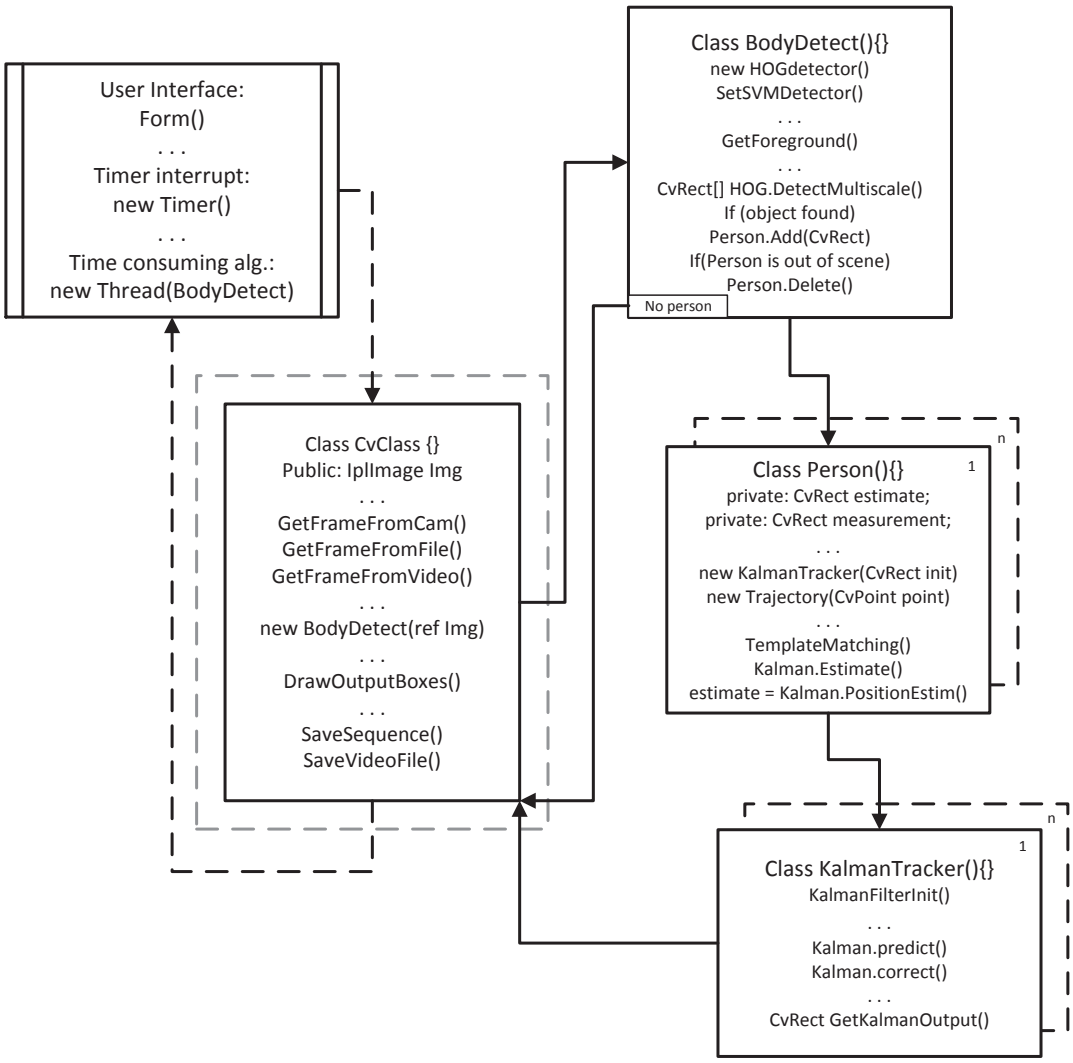
Jak již bylo řečeno, tato třída tvoří rozhraní, mezi programovým GUI a jednotlivými funkcemi. Třída `CvClass` je jádrem celé aplikace a je prováděna v nekonečném cyklu. Některé její metody jsou prováděny v samostatném vlákně, které je vytvořeno v třídě `Form1`. Třída `Form1` tvoří grafické rozhraní a dochází zde k inicializaci grafických komponent. Třída obsahuje statické proměnné, pomocí kterých je řízen chod programu. K těmto proměnným je uživateli umožněno přistupovat přes grafické rozhraní pomocí komponent `CheckBox`. Hlavní třída obstarává také vykreslování všech informací do zpracovávaného obrazu. Při spuštění aplikace dojde k volání konstruktoru v části inicializace, kde lze nastavit s jakými parametry se program spustí. Může existovat několik instancí této třídy. Počet instancí závisí na množství připojených kamer nebo jiných snímacích zařízení. Pro účely testování je vytvořena pouze jedna instance. Dochází zde také k odchytání všech výjimek a chybových hlášení ze všech vrstev aplikace.

Jakmile dojde ke zvolení volby *"GetFromCam"*, *"GetFromFile"* nebo *"GetFromVideo"* v GUI aplikace, inicializuje se připojený hardware nebo příslušné externí soubory. Případná chyba spojení (kamera nenalezena) a chyba při načtení souboru vyvolá výjimku s textem v komponentě `MessageBox`.

Aplikace podporuje načtení video formátů ve video kontejnerech *"MPG"* a *"AVI"*. Externí sekvence snímků může být ve formátu typu *"jpg", "png"* nebo *"bmp"*. Cesta k souboru s jeho názvem je definována v konstruktoru třídy. Název souboru je programově měněn v metodě `SetRWFileName()`. Aplikace také umožňuje uživateli sekvenci již zpracovaných snímků ukládat pod stejným názvem, do předem definovaného adresáře.

Nejdůležitější částí je volání metod HOG detektoru. Při této volbě dojde k inicializaci třídy `BodyDetect`. Jako parametr je konstruktoru předána reference na aktuální snímek sledované scény. Reference je předávána z toho důvodu, že se obrázek v paměti při volání funkce `GetFrameFromCam()` nachází stále na stejném paměťovém místě a není tedy třeba v každém cyklu operovat s množstvím předávaných dat. Zároveň v systému zůstává stále jedna kopie snímku, která je i zobrazována v náhledu.

Na obrázku 4.2 je zobrazen jednoduchý vývojový diagram aplikace s voláním jednotlivých tříd a jejich metod. Běh programu je řízen v hlavní třídě CvClass.

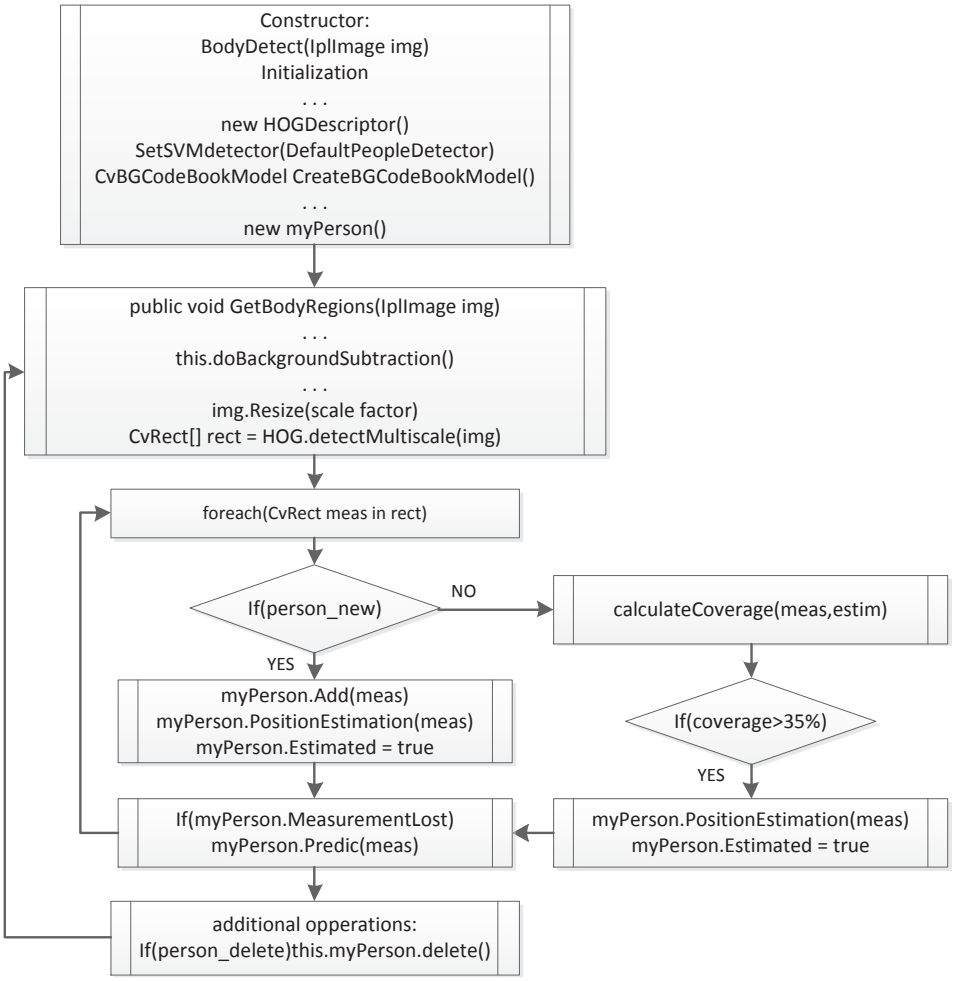


Obr. 4.2: Volání jednotlivých tříd programu

4.4.2 Třída BodyDetect

Stěžejní částí algoritmu je právě detekce osob ve videu. Ve třídě `BodyDetect` je implementován algoritmus detekce osob s využitím metody HOG detektoru popsané v kapitole 1.4. Získaný vektor deskriptoru je vstupem SVM klasifikátoru, který je natrénován podle 1.5.2. Protože se jedná o výkonný algoritmus, OpenCV ho přímo implementuje ve svých funkcích i s naučenými daty SVM klasifikátoru. Natrénovaná data se nacházejí v dodatečném *xml* souboru, který je součástí distribuce OpenCV. Vývojový diagram třídy `BodyDetect` znázorňuje obrázek 4.3.

Při inicializaci této třídy dojde k volání konstruktoru HOG deskriptoru. Vstupem této metody je zmiňovaný *xml* soubor s naučenými daty. V konstruktoru dochází k nastavení parametrů detekce jako je velikost detekčního okna, provedení gamma korekce a nor-



Obr. 4.3: Vývojový diagram třídy BodyDetect.

malizace histogramu. Hodnoty jednotlivých parametrů závisí na typu sledované scény a výkon detektoru lze jejich změnou značně ovlivnit. Dalším krokem je nastavení klasifikátoru voláním metody `SetSVMDetector()`.

O detekci a následné sledování se stará metoda `GetBodyRegion()`. Na začátku je rozlišení vstupního obrazu sníženo na velikost danou scale faktorem, jehož hodnota se nastavuje v konstruktoru třídy. Tím dojde k výraznému snížení času detekce, avšak úspěšnost detekce zůstane zachována změněním parametrů detekčního okna viz kapitola 2.1.2. V knihovně OpenCV je implementována funkce `DetectMultiscale()`, která vstupní obraz prohledává pro různé velikosti detekčního okna. Tím je zajištěna odolnost vůči různě velikým objektům (vpředu nebo vzadu) sledované scény. Volání funkce je následující,

```
CvRect[] DetectMultiscale(Mat Img,double hitThreshold,
        ...CvSize winStride,CvSize padding,double scale,
        ...int groupThreshold);,
```

kde první parametr je vstupní obraz datového typu `Mat`, snížením nebo zvýšením druhého parametru dojde k zvýšení resp. snížení počtu falešných alarmů, třetí parametr je velikost buňky a čtvrtým parametrem je velikost bloku. Pátý parametr typu `double` určuje, jakým násobkem bude detekční okno zvětšováno. Poslední parametr určuje mez, kdy budou vícenásobně detekované objekty sloučeny v jeden výsledný. Funkce vrací pole typu `CvRect[]` se souřadnicemi a velikostí nálezu. Velikost detekovaných regionů je ještě zvětšena, protože velikost okna nálezu navrácená funkcí je o něco menší, než je samotný objekt. Velikost pole udává počet nalezených osob.

Tvůrci knihovny OpenCV doporučují tyto parametry ponechat podle jejich vzorových příkladů. Nastavení je samozřejmě individuální a silně závisí na typu scény, především pak na velikosti hledaných osob v pixelech.

V momentě, kdy je odezva detektoru pozitivní, dojde k vytvoření objektů spojených s danými detekovanými osobami. Každá osoba vlastní instanci třídy `Person`, jež obsahuje informace o sledovaném objektu. Tato třída zároveň obstarává samotné sledování. Nová osoba je přidána vždy při pozitivní odezvě detektoru, avšak za předem definovaných podmínek. Instance třídy vzniká, když se v blízkosti nenachází žádná jiná osoba. V případě že se okno nálezu překrývá s již existující instancí, je nová osoba přidána pouze tehdy, je-li velikost překrývajících se ploch menší jak 50%. Aby bylo přidávání a odebírání osob snadno proveditelné, jsou tyto instance ukládány do datového typu `List`. S tímto seznamem libovolných typů lze poté pracovat jako se zásobníkem.

Vytvořená instance třídy `Person` dále existuje pouze v případě, že se jedná o dvě po sobě jdoucí detekce náležející danému objektu. Tím je ošetřeno přidání objektu, který by vznikl při falešně pozitivní odezvě detektoru.

Tato třída obsahuje i metodu `BGS()`, jež provádí subtrakci pozadí pomoci algoritmu *"Codebook"*. Metoda je volána při každém průchodu detektorem. Model pozadí je nutné inicializovat. Inicializace je provedena při prvních deseti průchodech. Výstupem metody je segmentovaný binární obraz reprezentující pohyblivé regiony. Důležitým prvkem je i interval mezi aktualizacemi modelu pozadí. Tento interval činí 40 snímků a je závislý na podmínkách scény.

4.4.3 Třída Person

Tato třída má za úkol spravovat a řídit úlohu sledování. Při zavolání konstruktoru této třídy dojde k inicializaci Kalmanova filtru s předáním počáteční polohy a velikosti objektu. Obsahuje veškeré informace o sledovaném objektu, jako je jeho pozice a velikost. Tyto parametry jsou uchovávány ve dvou proměnných typu `CvRect`. Jedna proměnná obsahuje data aktuálního měření (detekovaná pozice detektorem), tedy data nutná pro

výpočet korekce Kalmanova filtru. Druhá obsahuje predikovanou pozici, což je naopak výstup Kalmanova filtru.

Tato třída řeší vytváření trajektorií pohybu osoby. Po celou dobu existence objektu asociovaného s danou osobou dochází k ukládání souřadnic bodů trajektorií do fronty typu `Queue`. Tyto body jsou počítány z predikované pozice, jež je výstupem Kalmanova filtru.

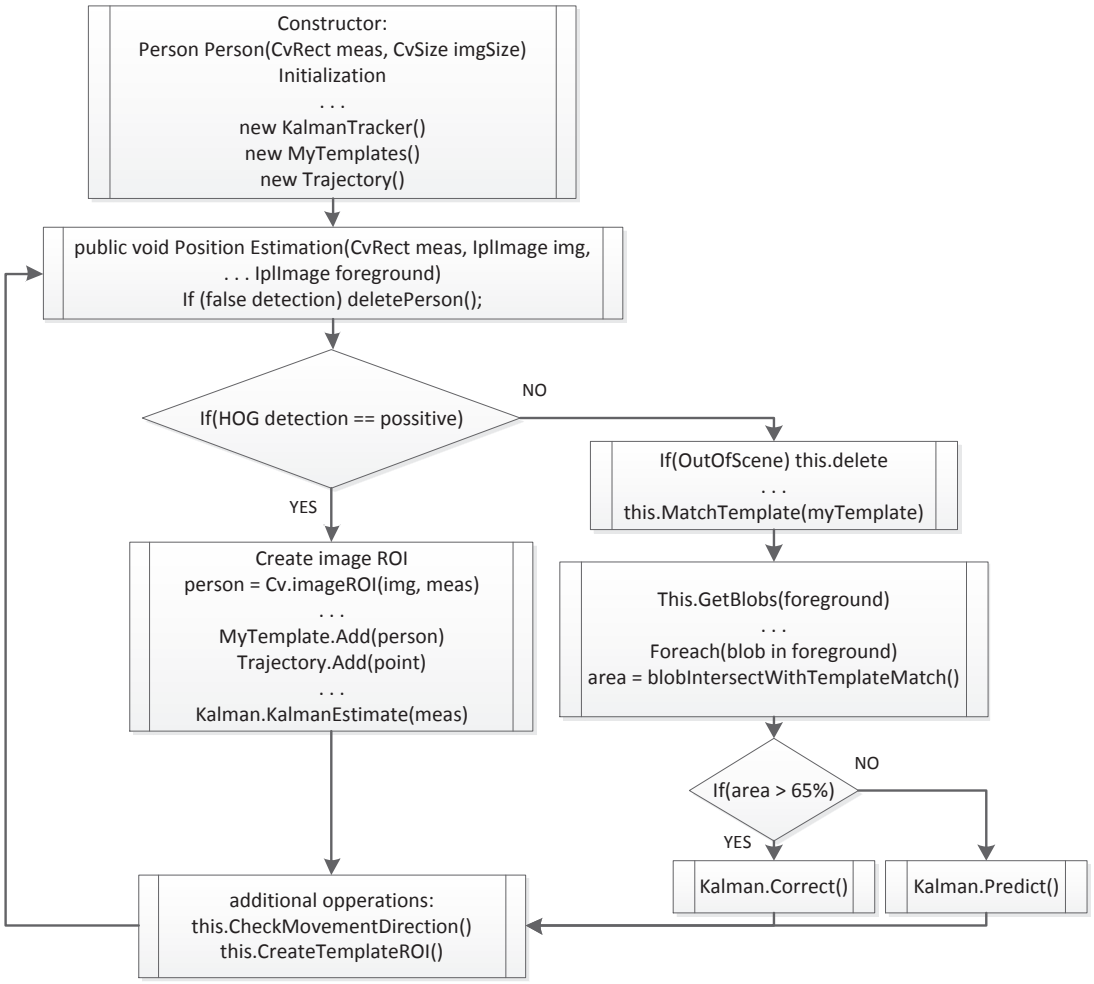
Dále je zde řešena pokročilá detekce osob s využitím metody porovnávání obrazu se vzorem, která je kombinovaná s informacemi ze získaného popředí. Program tuto větev algoritmu provádí pouze při selhání HOG detekce. Aktuální vzor osoby je ukládán do proměnné typu fronta. Tento vzor byl pořízen při posledním pozitivním nálezu HOG detektoru a zároveň je použit při volání funkce `Cv.MatchTemplate()`, jejímž prvním vstupním parametrem je výřez obrazu s pravděpodobným výskytem osoby a druhým je daný vzor. Funkce vrací pole akumulovaných hodnot dané korelací, v němž je hledáno maximum viz kapitola 2.3.1. Volání funkcí je následující,

```
Cv.MatchTemplate(CvArr img, CvArr template,
            ...CvArr result, MatchTemplateMethod method);

Cv.MinMaxLoc(CvArr acc_result,
            ...out double minval, out double maxval,
            ...out CvRect minloc, out CvRect maxloc);
```

Funkce `Cv.MinMaxLoc()` vrací maximální a minimální hodnotu v akumulovaném poli s příslušnými souřadnicemi výskytu. Získaný region slouží ke korekci Kalmanova filtru pouze tehdy, překrývá-li se s pohyblivým regionem popředí. Větvení programu naznačuje vývojový diagram na obrázku 4.4.

V této třídě jsou ošetřeny některé situace, jejichž vznik a následky ovlivňovali chod detekce a sledování. Prvním takovým stavem je situace, kdy objekt mizí ze zorného pole kamery. Pokud se objekt již ve scéně nenachází, metoda vrací hodnotu, která zajistí obsloužení příslušné úlohy. V tomto případě smazání dané instance objektu. Další vyskytující se situace úzce souvisí s problémem korespondence objektu. Docházelo k tomu, že objekt zajišťující sledování přeskočil na jinou osobu v její blízkosti. To znamená, že dojde ke korekci hodnotami, jež danému objektu nenáleží. Vychází se tedy z předpokladu, že osoba pohybující se v čase t daným směrem, pokračuje v tomto pohybu stejným směrem a stejnou rychlostí i v časech $t + 1$. Objekt pak přijímá pouze ta měření (výstup HOG detektoru) spadající do intervalu, který je dán původním směrem pohybu objektu. Toto však neřeší pohyb skupiny osob stejným směrem. V tomto případě algoritmus selhává.



Obr. 4.4: Vývojový diagram třídy Person.

4.4.4 Třída KalmanTracker, implementace Kalmanova filtru

Výhodou Kalmanova filtru je jeho implementace a rychlost maticových výpočtů. Kalmanův filtr je implementován v knihovně počítačového vidění OpenCV. Samozřejmě nechybí ani podpora maticových počtů, která značným způsobem zjednodušuje implementaci těchto algoritmu.

Při implementaci algoritmu se vychází z měření HOG detektoru. Detektor vrací údaje o pozici objektu spolu s jeho velikostí, tedy rozměry obdélníku (x, y, w, h). Dostáváme se tedy k vektoru měření z, který vypadá následovně

$$z_k = \begin{pmatrix} x \\ y \\ w \\ h \end{pmatrix}, \tag{4.1}$$

kde x, y jsou souřadnice nálezu a w, h je šířka a délka okna. Stavový vektor s užitím

rychlosti, tedy derivace polohy má tvar,

$$x_k = \begin{pmatrix} x \\ y \\ w \\ h \\ \dot{x} \\ \dot{y} \end{pmatrix}. \tag{4.2}$$

Lineární změna pohybu:

$$x_{k+1} = Ax_k + v_k \tag{4.3}$$

Model měření:

$$z_k = Hx_k + w_k \tag{4.4}$$

Po dosazení za matici A a H dostáváme:

$$\begin{pmatrix} x_{(k+1)} \\ y_{(k+1)} \\ w_{(k+1)} \\ h_{(k+1)} \\ \dot{x}_{(k+1)} \\ \dot{y}_{(k+1)} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_k \\ y_k \\ w_k \\ h_k \\ \dot{x}_k \\ \dot{y}_k \end{pmatrix} + N(0,Q)$$

$$\begin{pmatrix} x_{obs} \\ y_{obs} \\ w_{obs} \\ h_{obs} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_k \\ y_k \\ w_k \\ h_k \\ \dot{x}_k \\ \dot{y}_k \end{pmatrix} + N(0,R) \tag{4.5}$$

Síla Kalamanova filtru je právě v učinění predikce i v případě výpadku měření. Předchozí znalost pohybu, tedy přírůstek $(\Delta x, \Delta y)$ ve směru osy x a y, způsobí posunutí odhadu do místa, kde by se pravděpodobně objekt nacházel, kdyby nedošlo k jeho ztrátě. Ztrátu měření lze přirovnat k takovému měření, jehož šum má nekonečnou odchylku. V tomto případě se stav systému x v kroku $k+1$ a kovarianční matice P v kroku $k+1$ mění následovně [1],[7],

$$\begin{aligned} x_{k+1} &= x_k \\ P_{k+1} &= P_k \end{aligned} \tag{4.6}$$

Vytvoření Kalmanova filtru a jeho matic je provedeno voláním konstruktoru třídy `CvKalman` následovně,

```
CvKalman CvKalman(int dyn_param, int meas_param,
                  ...int control_param).
```

Podle vektorů 4.1 a 4.2 jsou parametry konstruktoru velikosti vektorů `dyn_param = 6` a `meas_param = 4`. Počet řídících parametrů je 0.

Pokud je měření k dispozici, předpokládaná pozice objektu je získána provedením predikce následovanou korekcí filtru. Programově tedy voláním funkcí:

```
CvMat CvKalman.Predict()
CvMat CvKalman.Corect()
```

Při ztrátě měření:

```
Cv.Copy(kalman.CovariancePre,kalman.CovariancePost)
Cv.Copy(kalman.StatePre,kalman.StatePost).
```

Vliv na funkci a správný odhad Kalmanova filtru má nastavení počátečních parametrů tvořící diagonálu kovariačních matic procesního šumu Q, šumu měření R a kovariační matice P. Prvky hlavní diagonály kovariační matice P dávají odhad chyby daného stavu. Počáteční hodnoty se obvykle nastavují velké a udávají míru nejistoty počátečního stavu. Za běhu filtru jsou tyto hodnoty snižovány aktualizací měření. Počáteční hodnoty matic Q a R byly nastavovány experimentálně a to na následující hodnoty.

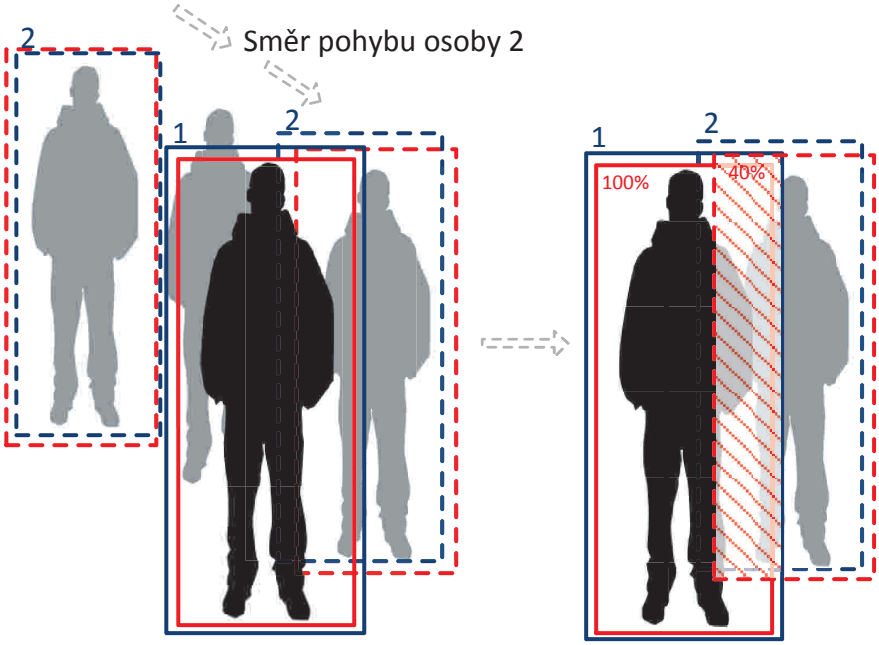
```
kalman.ProcessNoiseCov.SetIdentity(1e-2)      //matrix Q
kalman.MeasurementNoiseCov.SetIdentity(1e-3)   //matrix R
kalman.ErrorCovPost.SetIdentity(10)            //matrix P
```

Změnou počátečních parametrů Q a R je ovlivněna rychlost konvergence a filtrační schopnost algoritmu. Filtrační schopnost však není ve funkci trasovacího algoritmu příliš důležitá. Jde především o správnou predikci následujícího stavu v případě ztráty měření. Hodnoty parametrů jsou nastaveny s ohledem na rychlost zpracování jednotlivých snímků a budou se lišit pro různé časové odezvy implementovaného detektoru.

4.5 Korespondence sledovaných objektů

S přibývajícími objekty v obraze roste i počet instancí Kalmanova filtru. Každý objekt je tedy sledován zvlášť. Problémem je, jakým způsobem rozhodnout, který filtr (tracker) náleží jakému objektu. Tato situace byla vyřešena následujícím způsobem. Pro každý filtr je spočítáno procento překrytí se všemi nalezenými objekty. Ten objekt, který pokrývá největší procento plochy odhadované pozice objektu Kalmanovým filtrem, náleží příslušejícímu měření. Proces korespondence zobrazuje následující obrázek 4.5.

Na obrázku 4.5 vlevo jsou znázorněny dvě osoby ohraničené obdélníky. Červený obdélník značí výstup detektoru (měření), modrý obdélník patří výstupu Kalmanova filtru (predikce). První osoba (1) stojí na místě, druhá osoba (2) míjí stojící osobu v popředí. Vpravo je spočítána plocha, kterou pokrývají jednotlivá měření predikovanou oblast filtru 1. Filtru koresponduje to měření, které pokrývá největší plochu v procentech.



Obr. 4.5: Řešení problému korespondence.

5 INTERPRETACE VÝSLEDKŮ

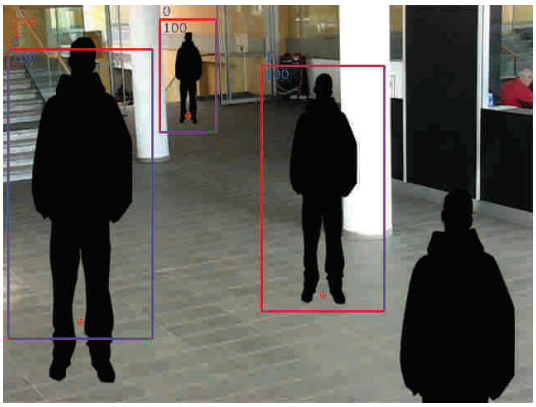
Tato kapitola shrnuje dosažené výsledky algoritmu na testovacích datech s popisem situací, kdy algoritmus selhává. Parametry detektoru (funkce `DetectMultiscale()`) jsou nastaveny jako kompromis mezi rychlostí a schopnosti rozpoznat osoby po celé detekované scéně. V rámci jedné scény zůstávají parametry detektoru neměnné. Příloha A dokumentu obsahuje náhledy výstupu trasovacího algoritmu. Červený rámeček udává měření, modrý rámeček je výstup Kalmanova filtru a červená plná čára naznačuje výslednou trajektorii. Každá řada v přiložené sekvenci obrazů zobrazuje jednu situaci.

5.1 Odezva detektoru pro mezní velikosti osob

Pro zjištění mezních velikostí osob v pixelech, které je detektor schopen rozpoznat, byla namodelována scéna viz obrázek 5.1. Siluety osob vložených do statického pozadí, představují ideální situaci. Velikost osob pro mezní případy, tedy minimální a maximální výška se zachováním proporcí, byla nastavena tak, že detektor pro menší resp. větší osoby již nevrátí pozitivní nález. Modelová scéna také ukazuje případ, kdy nedojde k rozpoznání (pravý dolní roh obázku 5.1). To je dáno tím, že je detektor naučen na celou osobu ve vzpřímené pozici. Mezní velikosti v pixelech shrnuje následující tabulka 5.1.

Mezní velikost	**Výška**	**Šířka**	**Osoba č.**
Minimální	136 px	68 px	0
Maximální	349 px	174 px	2

Tab. 5.1: Mezní velikost osob v pixelech pro modelovou scénu.



Obr. 5.1: Modelová situace pro zjištění mezních parametrů velikosti osob v pixelech.

Minimální mezní velikost je dána velikostí detekčního okna, která činní 128x64 px. Maximální možná velikost detekované osoby je v ideálním případě třikrát vetší. Vzhledem k umístění kamery však osoba maximální velikost nikdy nedosáhne.

Parametry detektoru shrnuje následující tabulka 5.2.

Parametr	**Hodnota**
scale faktor	0,75
hit threshold	-0,35
win stride	8x8 px
padding	16x16 px
scale	1,13

Tab. 5.2: Parametry detektoru při zjištění mezní velikosti osob.

5.2 Selhání detektoru

Na reálných datech pořízených v prostorách školy a z veřejně dostupných databází se stávalo, že algoritmus detekce selhával. To bylo zapříčiněno především nesprávně zvoleným úhlem pohledu nebo nevhodnou ohniskovou vzdáleností objektivu. Osoby tedy ve scéně zaujímali velikost menší, než je minimální mezní velikost detektoru. Možným řešením situace by bylo obraz převzorkovat a zvětšit tak i osoby uvnitř. To však za cenu zvýšené výpočetní náročnosti algoritmu.

Příkladem takové scény může být testovací sekvence SekvenceKolejni4/SEQ1 nebo všechny sekvence Caviar, kdy nedocházelo k detekci vzdálených osob. Soubory se nachází na datovém nosiči.

Obr. 5.2: Selhání detektoru pro postavy menší jak mezní velikost.

Selhání detektoru zachycuje obrázek 5.2. Osoby nacházející se v levém horním rohu (nejdále od kamery) nejsou zachyceny a jejich trajektorie pak nejsou správně sestaveny. Problémy působí i příliš tmavé pozadí sledované scény. Detektor pak selže v případě detekce osoby oblečené v tmavých barvách. Důvodem je fakt, že silueta není dostatečně kontrastní a hodnoty gradientů nestačí k jejímu popsání.

5.3 Selhání trasovacího algoritmu

5.3.1 Trasování pohybu skupiny osob

Trasovací algoritmus selhává za příliš vysokého provozu. Jedná se o situace, kdy se sledovanou scénou pohybuje velká skupina osob. Detektor je schopen detekce pouze osob nacházejících se v popředí skupiny. Odezva detektoru zároveň není stabilní a dochází k vynechání některých již detekovaný osob. To činí problémy při aktualizaci Kalmanova filtru a při určení korespondence objektu.

Nejhorším scénářem trasování je množství osob pohybujících se různými směry. V tomto případě je limitující hlavně rychlost odezvy detektoru. Na obrázku 5.3 je znázorněno selhání trasovacího algoritmu při pohybu skupiny osob na dvou různých scénách. Všechny osoby se pohybují jedním směrem. Při vynechání detektoru v jednom snímku a objevení nového objektu v blízkosti posledně známé pozice trasovací algoritmus přebírá měření nejbližší detekované osoby. To je vidět na vytvořených trajektoriích, které přesně nesledují danou osobu. Tato situace souvisí s korespondencí objektu. Problém by se dal zjednodušit na sledování celého celku.

Parametry detektoru při tomto testu byly stejné jako v tabulce 5.2.

Obr. 5.3: Selhání trasovacího algoritmu při detekci pohybu skupiny osob.

5.3.2 Trasování pohybu osob zacloněné překážkou

Sledovaná scéna v ideálním případě neobsahuje žádné překážky. V reálných situacích však dochází k částečnému nebo úplnému zaclonění osoby. Pokud osoba pokračuje stejným směrem a rychlostí jako před překážkou, algoritmus jednoduše predikuje podle posledně známého pohybu. Nastane-li však situace, kdy se osoba k překážce blíží jedním směrem a jiným směrem naopak odchází, trasovací algoritmus selže. Takovou situaci popisuje obrázek 5.4, kde vrchní část ukazuje správnou predikci pohybu a spodní část selhání algoritmu.

Obr. 5.4: Selhání trasování osoby zacloněné překážkou.

6 ZÁVĚR

Tato práce se zabývá implementací vhodného algoritmu pro sledování osob v komerčních aplikacích. Jednotlivé přístupy jsou shrnuty v prvních kapitolách této práce. Pro detekci byla vybrána metoda orientovaného gradientu HOG, pomocí které je vytvořen deskriptor, na jehož základě je prováděna následná klasifikace. Tato metoda využívá takzvaného detekčního okna, které je posouváno přes překrývající se oblasti obrazu, v kterých je deskriptor vytvářen. Výstupem je vektor deskriptoru pro všechny pozice detekčního okna. Jako klasifikační algoritmus slouží lineární klasifikátor SVM, jehož vstupem je právě vytvořený vektor deskriptoru. Klasifikátor je nezávislý na měřítku a umožňuje detekci postav o obrazové velikosti od 64x128px do 175x296px. Schopnost detekce závisí nejen na parametrech detektoru, ale i na rozlišení vstupního obrazu. Parametry a rozlišení byly voleny jako kompromis mezi výkonem detektoru a rychlostí jeho odezvy. Množství osob nacházejících se ve scéně neovlivňuje čas detekce, protože je deskriptor vytvářen přes celý obraz.

Pro zvýšení výkonu samotného detekčního mechanismu byly implementovány metody detekce porovnávání obrazu se vzorem a subtrakce pozadí. Metoda porovnávání se vzorem není autoinicializační. K její činnosti je třeba znát daný vzor postavy, jež je pořízen při poslední pozitivní odezvě HOG detektoru. Tyto algoritmy tvoří v kombinaci s HOG deskriptorem jádro detektoru postav.

Sledování detekovaných osob zajišťuje implementovaný Kalmanův filtr, jež slouží k trasování objektů. Tento přístup byl zvolen zejména proto, že umožňuje predikovat pozici objektu i při krátkodobém selhání detektoru. Výstupem Kalmanova filtru je estimovaná pozice objektu, z které je tvořena trajektorie pohybu. Algoritmus je schopen sledovat i osoby, jež jsou po krátkou dobu ztraceny předmětem v pohledu kamery. Trasovací algoritmus však selhává při vytváření trajektorií větší skupiny lidí. To je dáno především nedostatečně rychlou odezvou detekčního algoritmu, která činí potíže při aktualizaci Kalmanova filtru. Program je připraven i na připojení více pořizovacích zařízení.

Algoritmus je implementován v jazyce C# s využitím funkcí knihovny OpenCV. Jako interface mezi prostředím C/C++ a C# byl použit wrapper OpenCVSharp, který zabaluje funkce knihovny OpenCV. K programu bylo vytvořeno jednoduché grafické uživatelské rozhraní, které slouží k řízení chodu programu. Zároveň jsou zde zobrazovány náhledy obrazu s informacemi o pozici a trajektorii osob. Program umožňuje načtení testovacích dat z přiložených obrazových sekvencí. Výstup algoritmu je možno ukládat do obrazové sekvence nebo do video souboru.

Implementovaný algoritmus byl testován na testovacích datech získaných z veřejných databází, ale i na datech pořízených v prostorách školy. Části zpracovaných testovacích dat se zakreslenými trajektoriemi obsahuje příloha této práce.

Limitujícím faktorem je rychlost samotného detektoru, která činní 3,4 fps. Ta však

může být zvýšena rozdělením výpočetního výkonu i na procesorové jádro grafické karty. Knihovna OpenCV toto umožňuje v kombinací s výpočetní architekturou CUDA. Výstup systému může být v dalších fázích práce zkombinován se statistickým jádrem, které by mělo za úkol trajektorie vyhladit a vyhodnotit.

LITERATURA

[1] BRADSKI, G., KAEHLER, A. *Learning OpenCV: Computer vision with the OpenCV library*. O'Reilly Media, 2008. 577 s., ISBN 978-0-5965-1613-0
WWW http://books.google.com/books?hl=en&lr=&id=seAgiOfu2EIC&oi=fnd&pg=PR3&dq=Learning+open+cv&ots=hSF56ogDKi&sig=JCClLvaq--mCdiPem5USZsvZ1dY

[2] DALAL, N. *Finding people in images and videos*. Dizertační práce, Institut Nationale Polytechnique de Grenoble, 2006.
WWW http://lear.inrialpes.fr/pubs/2006/Dal06/Dalal-phd-slides.pdf

[3] DALAL, N., TRIGGS, B. Histograms of Oriented Gradients for Human Detection. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, ročník 1, 2005. s. 886–893.
WWW http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1467360

[4] DOLLÁR, P., WOJTEK, C. SCHIELE, B., aj., Pedestrian detection: A benchmark. *IEEE Conference on Computer Vision and Pattern Recognition*, č. 99, 2009. s. 304–311.
WWW http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5206631

[5] FUNK, N. A Study of the Kalman Filter applied to Visual Tracking. University of Alberta, Prosinec 2003. str. 26.

[6] HLAVÁČ, V., ŠONKA, M. Image processing, analysis and machine vision. Toronto: Thomson, 2008. ISBN 978-0-495-08252-1, 829 s.

[7] LIU, X. Kalman filtering with partial observation losses. *Decision and Control, 2004. CDC. 43rd*, 2004.
WWW http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1429408

[8] LOWE, D. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, ročník 60, č. 2, Listopad 2004. s. 91–110. ISSN 0920-5691.

[9] MARTIN, G. *People detection algorithms based on appearance and motion information*. Dizertační práce, Univesidad Autonoma de Madrid, 2009.
WWW http://dymas.ii.uam.es/webvpu/media/docs/publicacion/Tfm/

2009November_peopleDetectionAlgorithmsBasedOnAppearanceAndMotion\
Information.pdf

[10] PARKS, D., FELS, S. Evaluation of Background Subtraction Algorithms with Post-Processing. *2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, Září 2008. s. 192–199.
WWW http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4730412

[11] POTŮČEK, I. Sledování pohybu objektů v sekvenci snímků. *Diplomová práce, Vysoké učení technické v Brně*, 2002. str. 76.
WWW http://www.fit.vutbr.cz/research/view_pub.php?id=7163

[12] RADKE, R., ANDRA, S., AL-KOFAHI, O., aj. Image change detection algorithms: a systematic survey. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, ročník 14, č. 3, Březen 2005. s. 294–307, ISSN 1057-7149.
WWW http://www.ncbi.nlm.nih.gov/pubmed/15762326

[13] RUSS, J.C. *The Image Processing Handbook, Sixth Edition*. North Carolina State University, Raleigh: CRC Press, 2011. 885 s., ISBN 9781439840450.

[14] SUARD, F. RAKOTOMAMONJY, A., BENSRHAIR, A., aj., Pedestrian Detection using Infrared images and Histograms of Oriented Gradients. *2006 IEEE Intelligent Vehicles Symposium*, 2006. s. 206–212, doi:10.1109/IVS.2006.1689629.
WWW http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1689629

[15] UTSUMI, A., TETSUTANI, N. Human tracking using multiple-camera-based head appearance modeling. *IEEE 6th International Conference on Automatic Face and Gesture Recognition*, 2004. s. 2–7.
WWW http://www.computer.org/portal/web/csdl/doi/10.1109/AFGR.2004.1301609

[16] VERNON, D. Machine Vision: Automated Visual Inspection and Robot Vision. Hemel Hempstead: Prentice Hall International (UK) Ltd., 1991. s. 260. ISBN 0-13-543398-3

[17] VIOLA, P., JONES, M., SNOW, D. Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, ročník 63, č. 2, 2005. s. 153–161.
WWW http://www.springerlink.com/index/t61k38u53j531344.pdf

[18] YILMAZ, A., JAVED, O., SHAH, M. Object Tracking: A Survey. *ACM Computing Surveys*, ročník 38, č. 4, Prosinec 2006. s. 1–45, ISSN 03600300, doi: 10.1145/1177352.1177355.
WWW `http://portal.acm.org/citation.cfm?doid=1177352.1177355`

[19] ZHANG, Z., GUNES, H., PICCARDI, M. Tracking People in Crowds by a Part Matching Approach. *2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, Září 2008. s. 88–95, doi:10.1109/AVSS.2008.45.
WWW `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4730389`

[20] ZHOU, J., HOANG, J. Real Time Robust Human Detection and Tracking System. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, 2005. s. 149–149, doi:10.1109/CVPR.2005.517.
WWW `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1565316`

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

POS Systém optimalizace prodeje – Point of Sales

HOG histogram orientovaného gradientu – Histogram of Oriented Gradient

SVM Support Vector Machine

BGS Subtrakce pozadí – Background Subtraciton

RGA Running Gaussian Average

RGA Running Gaussian Average

GMM model mixtury gaussiánů – Gaussian Mixture Model

AGMM GMM s adaptivním počtem gaussiánů – Adaptive GMM

AMF aproximovaný mediánový filtr – Aproximated Median Filtering

RGB RGB barevný model – RGB color model

SIFT na měřítku nezávislý popis významného bodu – Scale Invariant Feature Trasnform

RP srovnávací křivka – Recall-Precision

AP srovnávací křivka – Average-Precision

GPU grafický procesor – Graphic Processing Unit

AVI formát multimediálního kontejneru – Audio Video Interleave

GUI grafické uživatelské rozhraní – Graphical User Interface

SEZNAM PŘÍLOH

A Běh algoritmu na testovacích datech **69**

A.1 Testovací data Kolejní 4, pohled 1 69

A.2 Testovací data Kolejní 4, pohled 2 70

A.3 Testovací data Kolejní 4, pohled 3 71

A.4 Testovací data i-Lids, Londýnské metro 72

A.5 Testovací data Caviar, chodba nákupního centra 73

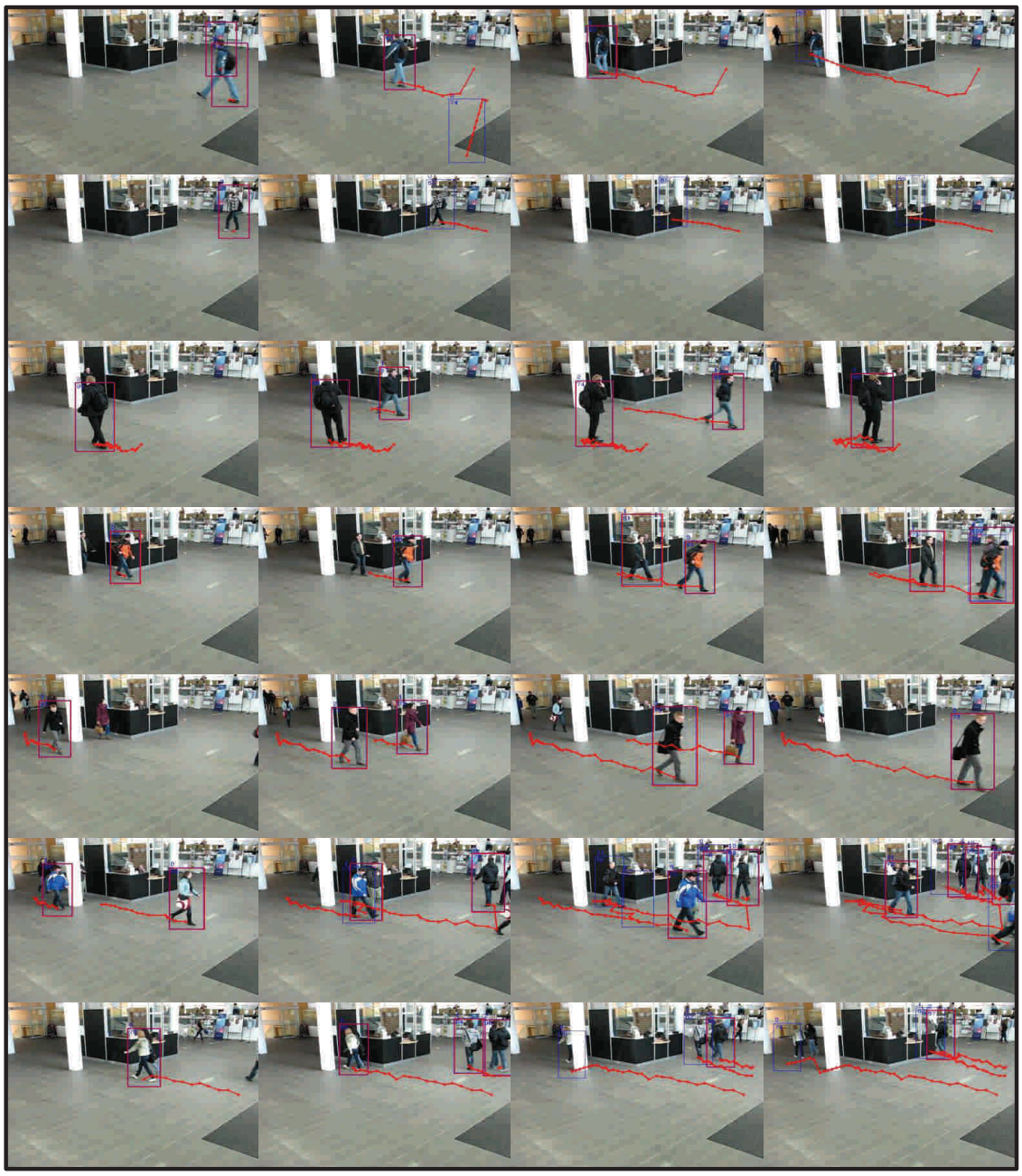
B Příloha na datových nosičích DVD **74**

A BĚH ALGORITMU NA TESTOVACÍCH DATECH

A.1 Testovací data Kolejní 4, pohled 1

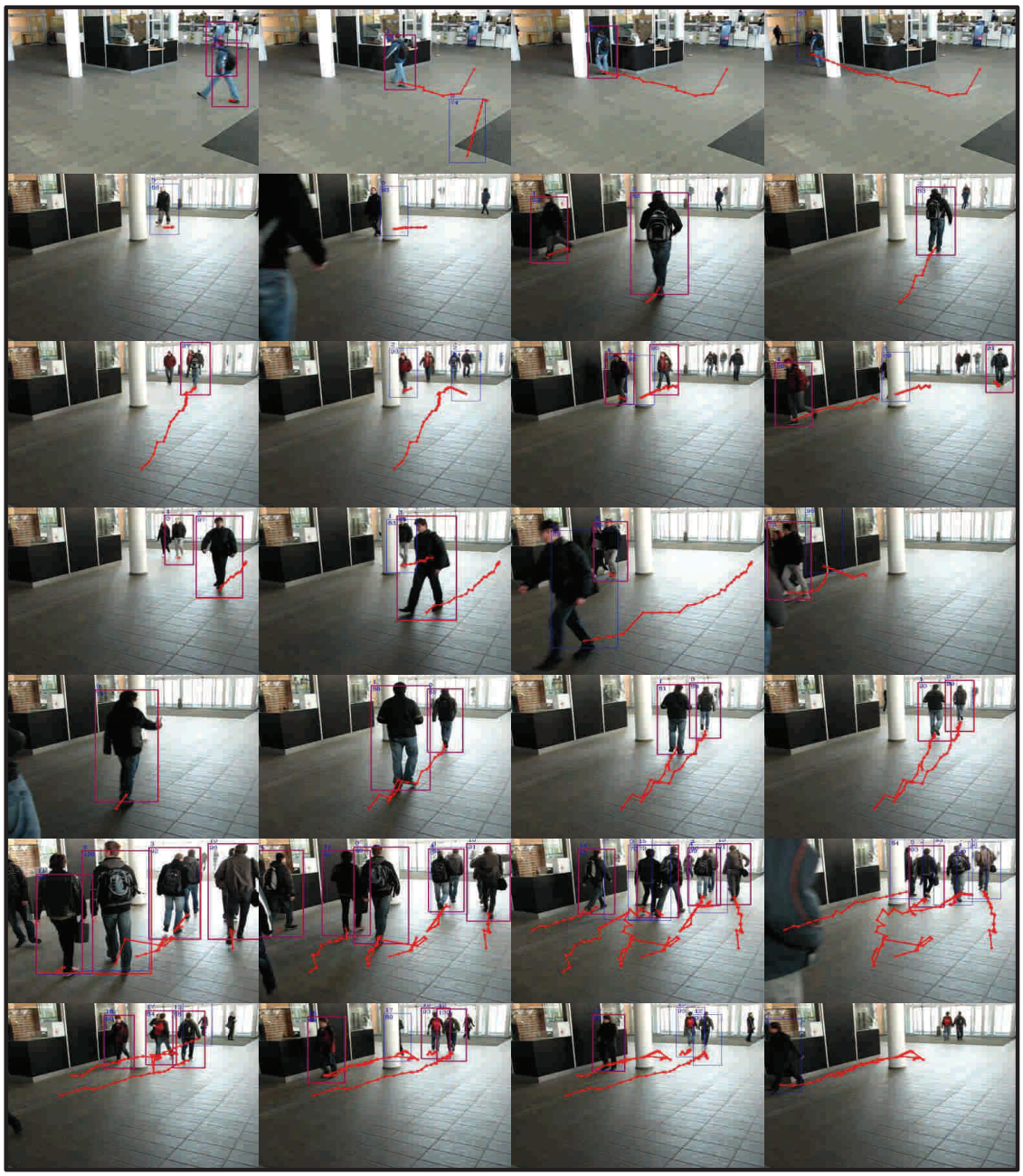
Obr. A.1: Aplikování algoritmu na testovací data získané v budově Kolejní 4, pohled 1.

A.2 Testovací data Kolejní 4, pohled 2



Obr. A.2: Aplikování algoritmu na testovací data získané v budově Kolejní 4, pohled 2.

A.3 Testovací data Kolejní 4, pohled 3



Obr. A.3: Aplikování algoritmu na testovací data získané v budově Kolejní 4, pohled 3.

A.4 Testovací data i-Lids, Londýnské metro

Obr. A.4: Aplikování algoritmu na testovací data „i-Lids“, Londýnské metro.

A.5 Testovací data Caviar, chodba nákupního centra

Obr. A.5: Aplikování algoritmu na testovací data „Caviar“, nákupní centrum.

B PŘÍLOHA NA DATOVÝCH NOSIČÍCH DVD

Součástí tohoto dokumentu je i příloha všech částí práce na datových nosičích DVD. V kořenovém adresáři první přílohy se nachází text diplomové práce cerninDP.pdf. Vypracovaný projekt v prostředí Microsoft VS2010 spolu se všemi potřebnými knihovnami obsahuje adresář Vypracování. Na druhém datovém nosiči se nachází všechny testovací sekvence a videa.