VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

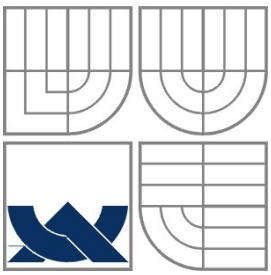
SLEDOVÁNÍ POHYBU OSOB VE VIDEO SEKVENCI

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

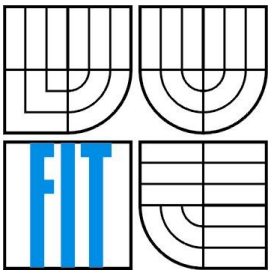
AUTOR PRÁCE
AUTHOR

DANIELA JOHANOVÁ

BRNO 2013



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

SLEDOVÁNÍ POHYBU OSOB VE VIDEO SEKVENCI

MOVING PERSONS DETECTION AND TRACKING

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

DANIELA JOHANOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL ŠPANĚL, Ph.D.

BRNO 2013

Abstrakt

Tato bakalářská práce se zabývá detekcí postavy ve videu pomocí Kinectu. Detektor lidských postav je založený na metodě ComboHod, která využívá barevné i hloubkové informace z obrazu. Cílem práce bylo vytvořit detektor postav, jehož funkci ukáži na statistické aplikaci. Aplikace shromažďuje statistické informace o lidech, kteří prošli kolem výlohy obchodu. Závěrem práce jsou popsány experimenty s detektorem v různých prostředích a podmínkách.

Abstract

This bachelor's thesis deals with the person detection using RGB-D Microsoft Kinect sensor. Human body detector is based on the method Combohod which uses both color and depth information from Kinect sensor. The aim of the thesis was to create a person detector whose functionality is demonstrated by proposing a statistical application that collects statistical information about the people who passed the shop window. At the end of the thesis the experiments with the detector under varios different conditions are described.

Klíčová slova

Detekce osob, Kinect, RGB-D, C/C++, ComboHod, HOG, HOD, OpenCV, OpenNI, deskriptory, SVM klasifikátor, ROC křivka.

Keywords

People detection, Kinect, RGB-D, C/C++, ComboHod, HOG, HOD, OpenCV, OpenNI, descriptors, SVM classifier, ROC curve.

Citace

Daniela Johanová: Sledování osob ve video sekvenci, bakalářská práce, Brno, FIT VUT v Brně, 2013

Sledování pohybu osob ve video sekvenci

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně pod vedením Ing. Michala Španěla, Ph.D. Uvedla jsem všechny literární prameny a publikace, ze kterých jsem čerpala.

……
Daniela Johanová
15.5.2013

Poděkování

Ráda bych poděkovala svému vedoucímu Ing. Michalu Španělovi, Ph.D. za skvělé vedení při vytváření této práce, za přínosné rady a užitečná doporučení podkladů pro mou práci. Dále chci poděkovat své rodině za podporu, kterou mi projevovali po celou dobu mého studia, a přátelům za pomoc při vytváření datasetů pro tuto práci.

Obsah

1 Úvod

Zrak je pro člověka jedním z nejdůležitějších smyslů, ba bych si dovolila říci, tím nejdůležitějším. Vidění nám dává možnost snadno a rychle se orientovat v našem okolí, překonávat různé překážky či výhodně volit nejlepší cestu. Není to však jen vnímání barev a textur, co nám naše oči poskytují. Hlavní složka pro lidskou rozpoznávací schopnost je tvar pozorovaného objektu. Už od raného dětského věku se učíme, jaký tvar jaké předměty běžně mívají. Víme, že míč je kulatý, že kostky stavebnice mají čtvercový nebo obdélníkový tvar, víme, že pastelka je úzká a dlouhá. Člověk se rychle naučí rozpoznávat i složitější tvary, jako je například lidská postava. V dnešní době jde technika kupředu mílovými kroky a v mnoha věcech se již počítače dokáží vyrovnat člověku. Není žádným problémem, aby počítač díky kameře rozpoznával jednoduché tvary jako například míč nebo tužku. Naučit počítač, aby dokázal rozpoznat siluetu lidské postavy, by bylo také velice výhodné. Detekce lidí na videu nabývá čím dál více na důležitosti a to nejen z hlediska bezpečnostního, ale také komerčního využití, do něhož jsem zařadila i svou bakalářskou práci.

Cílem práce bylo vytvořit detektor lidské postavy, jehož funkčnost demonstruji na aplikaci, která sbírá informace o pohybu lidí kolem statické kamery a získaná statistická data poté zobrazí jako graf. Aplikace je rozdělena na dvě části - detekční a statistickou. Jako snímací zařízení jsem si pro svoji práci zvolila přístroj Kinect, který poskytuje obrazová a hloubková data. Na těchto dvou druzích dat, které kombinuji, jsem vystavěla detektor postav, jež mi poskytuje vstupní informace pro statistickou část programu. Aplikace je určena pro kamenné obchody, kde bude sledovat pohyb lidí před výlohou. Program by bylo možné snadno modifikovat na snímání a získávání dat z vnitřních prostorů obchodu.

Tato práce je rozdělena do několika kapitol. V následujícím oddílu představím zařízení Kinect. Popisuji jeho stavbu, názorně ukáži rozmístění detekčních senzorů a nastíním způsob získávání prostorových dat.

Ve třetí kapitole se věnuji popisu různých způsobů detekce objektů z obrazových a hloubkových dat. Zevrubně popisuji několik metod detekce. Podrobněji se věnuji postupům, jež dále využívám pro implementaci své práce.

Čtvrtá kapitola pojednává o celkovém návrhu aplikace. Přibližuji několik zajímavých částí programu a popisuji navrhovaná řešení.

Začátkem páté kapitoly zmiňuji postup získávání testovacích dat. Dále se zde zabývám popisem implementačních detailů mé práce.

Šestá kapitola obsahuje vylíčení experimentů a porovnání dosažených výsledků při různých nastaveních částí programu.

Závěrem zhodnocuji dosažené výsledky a funkčnost detektoru jako celku i jeho jednotlivých částí. Práce končí diskusí nad možnými vylepšeními detektoru i celé aplikace.

2 Zařízení Microsoft Kinect

Detekce lidí je stále snazší díky moderní technice, která poskytuje přesnější videozáznamy a obecně více dat o sledovaném prostoru. S průlomovým zařízením v oblasti mapování prostoru přišla na konci roku 2010 firma Microsoft. Představila senzor Kinect a konzoli s názvem Xbox 360 (Obr. 2.1). Tyto dvě zařízení dohromady dokáží v reálném čase mapovat prostor před senzorem a přenášet obraz i zvuk pomocí konzole Xbox na téměř jakoukoli televizi nebo monitor. Pomocí kombinace kamery a hloubkových senzorů, které popíši níže, dokáže soustava zařízení sledovat trojrozměrný prostor před senzorem a doplňovat tak obraz.

Obrázek 2.1: Zařízení Kinect a Xbox 360 (převzato z [18])

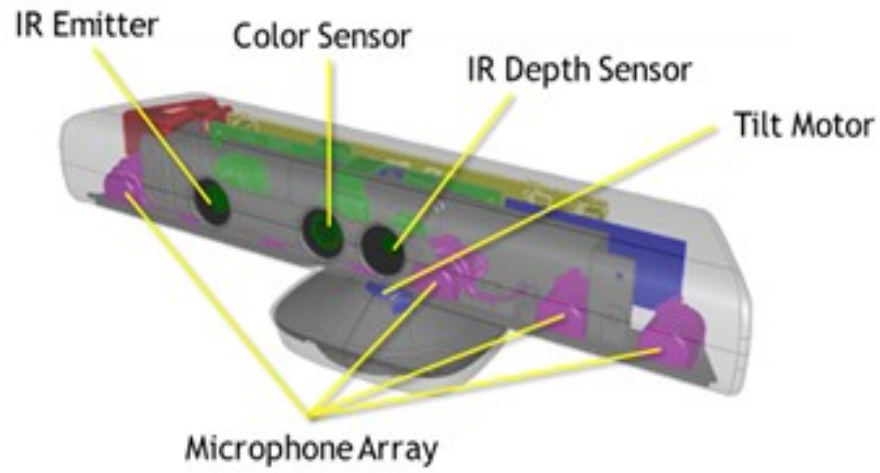
Snímač Kinect byl původně určen (ve spojení s konzolí Xbox) pro běžné použití jako herní zařízení. Avšak krátce po uvedení na trh se objevily knihovny umožňující připojení Kinectu k počítači. Microsoft na to reagoval tím, že vydal vlastní ovladače pro připojení Kinectu k operačnímu systému Windows. I tak dále vznikla řada volně šiřitelných knihoven zejména pro jiné systémy. Nejznámější jsou knihovny OpenNI, Nite nebo OpenKinect.

Senzorické zařízení Kinect se dnes nevyužívá pouze jako herní pomůcka, ale má spoustu jiných možností využití. Člověk může z pohodlí svého křesla pomocí pohybů ruky ovládat prohlížení internetu na televizi nebo vleže před kamerou posouvat právě běžící film. Kinect lze ovládat také hlasem, nesmí však být v místnosti výrazný hluk. Pokud pohlédneme na využití Kinectu společně s počítačem, můžeme jako nejvýznamnější novinky jmenovat ovládání pc pomocí gest, sledování mimiky obličeje a především využití Kinectu v robotice. Nová řada zařízení má být schopna odezírání z úst nebo rozeznávání nálady člověka, který stojí před kamerou. Ke všem těmto úkonům se používá kombinace informací z RGB kamery a IR senzoru.

Potenciál kombinace hloubkové a obrazové informace si rychle uvědomili i jiné firmy, takže dnes máme na trhu další podobný přístroj značky Asus. Základ monitorovacích zařízení je však velice podobný (obě snímací zařízení navrhovala firma PrimeSence). Dále popíši zařízení Kinect, které jsem použila ve své bakalářské práci.

Vidění Kinectu zajišťuje především jedna klasická RGB kamera, umístěná uprostřed přístroje (Obr 2.2). Prostorové informace získává zařízení díky emitoru IR částic, které jsou po odražení od objektů před senzorem zachyceny hloubkovým čidlem. Infračervené záření nevychází z Kinectu stále v plošném rozptylu. Laser pseudonáhodně promítne mračno bodů do prostoru, vnitřně si Kinect uchová obraz těchto vyslaných bodů (Obr. 2.3). IR hloubkový senzor zachytí odražený obraz bodů. Tyto dva obrazy se poté použijí ke stereo triangulaci vzdálenosti objektů od kamery. Kinect dokáže vytvořit 30 framu za sekundu s rozlišením 640x480 [6].

Výhodou hloubkových senzorů je, že nejsou závislé na osvětlení prostoru, kde se Kinect nachází. K nevýhodám infračervených laserů patří to, že některé materiály dokáží IR záření pohltit či pozměnit jeho odraz. V zachyceném odraženém obraze poté daný pohlcený bod chybí a v obraze poté vznikají nežádoucí artefakty. Tento šum lze z větší části odstranit použitím nějakého nelineárního obrazového filtru. Příkladem může být mediánový filtr nebo konzervativní vyhlazení. Mediánový filtr funguje na principu nahrazení chybějící hodnoty v matici mediánem, který se vypočte z hodnot okolí daného pixelu. Konzervativní vyhlazení nahrazuje chybějící či vadnou hodnotu pixelu hodnotou prahu (maxima nebo minima), který byl pro daný pixel překročen [12].



Obrázek 2.2: Zařízení Kinect se zvýrazněnými hlavními komponentami (převzato z [19])

Kinect samotný stojí na malém podstavci, jež je spojen s hlavní částí kloubem, kterým lze pohybovat pomocí zabudovaného servomotoru. Kinect lze natočit pomocí motoru o 27° nahoru nebo dolů z kolmé iniciační polohy. Dále zařízení obsahuje čtyři mikrofony rozmístěné po celé délce přípravku. Motorek ani mikrofony jsem pro svou práci nevyužila.

Obrázek 2.3: Příklad mračna bodů vyslaných Kinectem (převzato z [16])

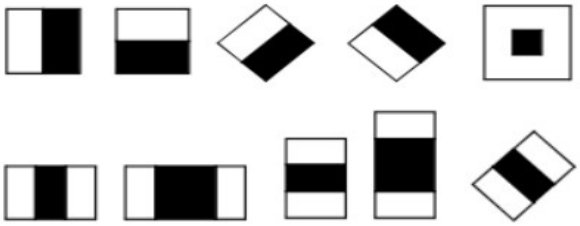
3 Detekce objektů v obraze

Tato kapitola přibližuje některé postupy, využívané při detekci vzorů v obraze. Nejprve popíši typy obrazových příznaků, poté uvedu nejčastěji používané klasifikátory a na závěr kapitoly se zaměřím na algoritmy detekce objektů v obraze. Blíže popisuji principy použité v této práci.

3.1 Obrazové příznaky

Aby bylo možné detekovat objekty v obraze, je nutné obraz předzpracovat. K tomu slouží obrazové příznaky a z nich spočtené deskriptory, které podávají ucelenou formou informace o tom, jaký objekt nebo textura se na dané části obrazu nachází. Existuje celá řada obrazových příznaků. Mezi nejznámější a nejpoužívanější příznaky patří Haarovy příznaky (Haar-like) a Histogramy orientovaných gradientů (HOG).

Nejznámější použití Haarových příznaků je ve spojení s metodou zvanou Viola-Jones [8], která je založená na spojení Haarových příznaků a algoritmu Adaboost (viz kapitola 3.2). Tato metoda je často používaná pro detekci obličejů v obraze a to především pro svoji rychlost a spolehlivost.



Obrázek 3.1: Příklady Haarových příznaků (převzato z [10])

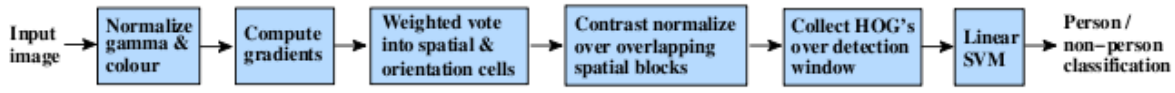
Haarovy příznaky pracují s intenzitou jasu oblasti obrazu, která se nachází pod zvoleným typem obdélníkového příznaku. Nejprve se vypočítá hodnota jasu oblasti pod bílou částí příznaku. Od této hodnoty se odečte suma jasu pod černou částí obrazu a tím se získá ohodnocení dané části obrazu [8][10]. Příklad některých používaných typů příznaků je na obrázku 3.1.

a) b)

Obrázek 3.2: a) Příklad vstupního obrázku pro získání HOG deskriptorů
b) Gradient ze vzorového obrázku (převzato z [4])

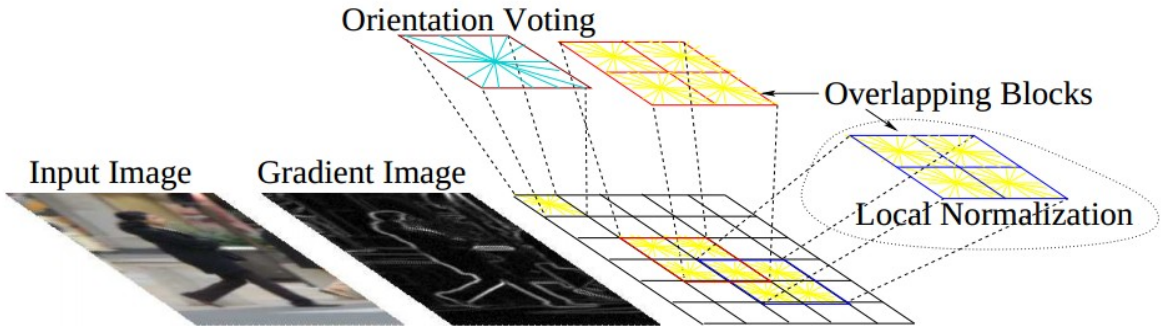
3.1.1 HOG

Histogramy orientovaných gradientů jsou příznaky, které se nejvíce hodí pro vyhledávání lidské postavy. Pracují se změnou jasu v určité části obrazu, především tedy na hranách (viz Obr. 3.2). Nejvíce je známá metoda, která využívá HOG příznaky v kombinaci s SVM klasifikátorem. Tuto metody popsali pánové Navneet Dalal a Bill Triggs. Schéma jejich navržené metody je ukázáno na obrázku 3.3. V následujících odstavcích je dle jejich článku Histograms of Oriented Gradients for Human Detection parafrázován popis extrakce histogramů orientovaných gradientů [4].



Obrázek 3.3: Schéma Dalal-Triggsovy metody (převzato z [4])

Pro získání deskriptoru musíme nejprve obrázek rozdělit na malé části zvané "buňky" (většinou 8x8 pixelů). Pro každou buňku je následně spočítám 1-D histogram orientovaných gradientů nebo směr růstu hran. Na obrázku 3.4 je zhruba ukázáno, jak se obrázek zpracovává. Na obrázku nejvíce vpravo je zvětšený příklad rozdělení obrázku na buňky a zobrazení jejich gradientů.



Obrázek 3.4: Příklad zpracování obrázku - vlevo je vstupní obrázek, uprostřed obrázek gradientů vstupu, vpravo je příklad rozdělení obrázku na buňky a bloky (převzato z [4])

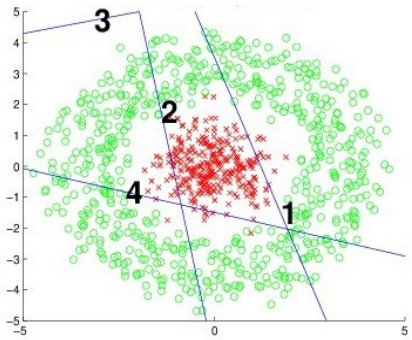
Pro lepší schopnost vyrovnat se se změnami stínování nebo osvětlení je vhodné histogram buňky normalizovat. Normalizace se provádí nad větší plochou obrazu - přes sousedící buňky. Tuto oblast, která bývá dvakrát větší než buňka, nazýváme "blok". Každá buňka v bloku se normalizuje hodnotou míry intenzity celého bloku. Touto úpravou hodnot dosáhneme spojitého kontrastu obrazu. Výsledný deskriptor obrazu se vytvoří kombinací histogramů jednotlivých buněk. Pro dokončení Dalal-Triggsovy metody už zbývá jen pomocí deskriptorů z trénovacích dat natrénovat SVM klasifikátor (blíže popsán v kapitole 3.2.1).

3.2 Klasifikátory

Pro rozpoznávání objektů z připravených deskriptorů obrazů slouží klasifikátory. Je jich celá řada, nejvíce používané jsou Neuronové sítě, metoda Adaboost a Support Vector Machine. Tuto metodu popíši blíže, protože ji využívám ve své práci.

Klasifikátor Adaboost (Adaptive Boosting) představili pánové Yoav Freund a Robert E. Schapire [13]. Klasifikátor se nejčastěji používá ve spojení s Haarovými příznaky (viz kapitola 3.1). Je založen na sloučení více slabých klasifikátorů (úspěšnost do 50%) do kaskády klasifikátorů, které poté rozhodují o příslušnosti vstupních dat do jednotlivých tříd. Tento algoritmus má spoustu modifikací, hojně používaný je např. Real Adaboost.

Příklad klasifikace dat vidíme na obrázku 3.5. Nejprve jsou data rozdělena prvním slabým klasifikátorem, jež je reprezentován čárou číslo 1. Po úspěšné klasifikaci je přidán klasifikátor č. 2., a následně obdobným způsobem klasifikátor č. 3 a č. 4. Všechny čáry tvoří dohromady kaskádu klasifikátorů. Po dokončení základní klasifikace vznikne na obrázku ohraničená oblast oddělující přibližně červené vzorky od zelených.

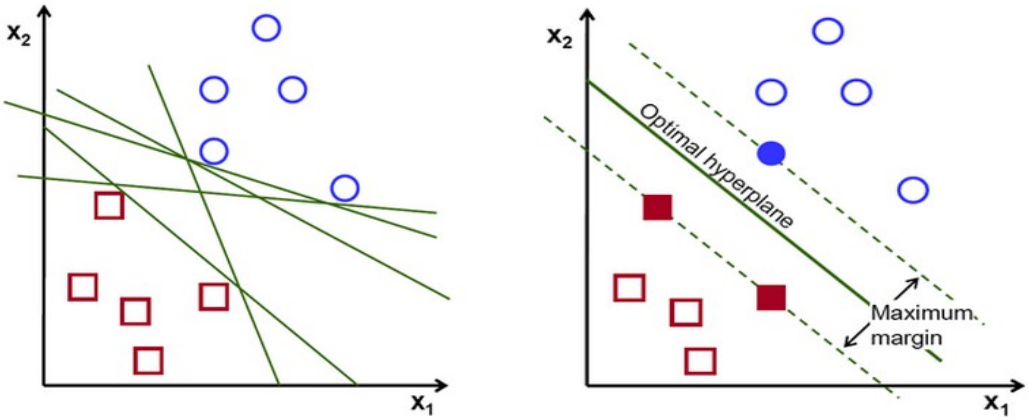


Obrázek 3.5: Příklad klasifikace metodou Adaboost (převzato z [17])

3.2.1 SVM

Support Vector Machine (SVM) [3] je další z řady klasifikátorů, spadající do třídy učení s učitelem. Jak již bylo zmíněno v kapitole 3.1.1, SVM klasifikátor se často používá ve spojení s HOG příznaky [4]. Po natrénování dokáže klasifikátor dobře rozpoznávat naučený vzor objektu, hodí se především na identifikaci chodců v obraze.

Tento klasifikátor se používá především proto, že dokáže pomocí transformace prostoru oddělit lineární funkcí prvky daného prostoru, které před transformací nebylo možné lineárně rozdělit [3] (viz obrázek 3.6-a). Transformací prostoru se přidá nová dimenze, která umožní oddělitelnost prvků. K transformaci se využívá jádrové funkce klasifikátoru (já pro svoji práci využila lineální jádrovou funkci). Z prvků ve vyšší dimenzi může být poté provedena separace příznaků pomocí nadroviny.



a) b)

Obrázek 3.6: a) Příklad pokusných rozdělení roviny
b) Vložení pomocné nadroviny (převzato z [14])

Cílem algoritmu je co nejvíce vzdálit pomocnou nadrovinu od prvků ze separovaných tříd. Vzdálenost nejbližšího prvku jedné množiny od nadroviny by měla být ideálně stejná jako vzdálenost nejbližšího prvku druhé množiny od nadroviny. Příklad vložení nadroviny do množiny prvků je vidět na obrázku 3.6 b). Metoda Support Vector Machine, jak už název napovídá, používá pro klasifikaci podpůrné vektory, které jsou získány z prvků, které leží nejblíže určené nadroviny. V obrázku 3.6 b) jsou těmito vektory prvky, které leží na čárkované ose. Čím větší vzdálenost je mezi třídami, tím lepší je klasifikátor.

3.3 Metody detekce objektu

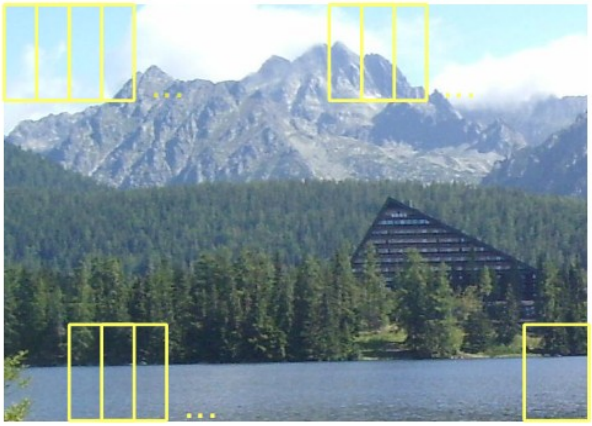
K identifikaci vzorů v obraze neslouží pouze obrazové deskriptory a klasifikátory. Je nutné znát také přesný postup získávání informací o obraze a o manipulaci s těmito daty. Při identifikaci objektů některé metody používají také poznatky o tom, v jaké části obrazu se vzor nachází. Pokud tuto informaci neznáme, lze obraz prohledat celý a tím získat informaci o poloze. Následující kapitola popisuje algoritmus vyhledání vzoru v obraze, který je použit v této práci.

3.3.1 Sliding Window

Tato metoda určuje to, jakým způsobem se bude obrázek prohledávat. Sama o sobě nemá žádnou detekční schopnost. Jedná se o metodu, která nám umožní prohledat obrázek bez ohledu na to, že předem neznáme velikost hledaného objektu [5]. Využívá se společně s HOG příznaky a SVM klasifikátorem. Dále popíši spojení těchto algoritmů.

Metoda spočívá v tom, že se vstupní obrázek projde s několika velikostmi detekčního okna. Tento popis však není úplně přesný, neboť se doopravdy nezvětšuje prohledávací okno, nýbrž se podvzorkovává prohledávaný obraz. To se provádí proto, že klasifikátory jsou natrénovány na konstantní velikost vstupního obrazu. Zvětšování okna je tedy nahrazeno zmenšováním obrazu.

Získaná část obrazu je poté předložena k ohodnocení pomocí HOG příznaků a dále se zpracuje SVM klasifikátorem [4]. Příklad průchodu okna obrazem je znázorněn na obrázku 3.7.



Obrázek 3.7: Příklad průchodu obrázku klouzajícím okénkem

3.4 Detekce objektů z hloubkového obrazu

Detekce objektů z hloubkové informace obrazu probíhá podobně jako z obrazové části. Metod detekce lidí a různých objektů vůbec je široká škála. Následující podkapitoly popisují algoritmy použité v této práci.

3.4.1 HOD detekce

Tato metoda (Histogram of Oriented Depths) je založená na získávání informací z hloubkového obrazu. Pracuje podobně jako Dalal-Triggsova metoda [4]. Jejím základem jen získat histogram z hloubkového obrazu.

Obraz získaný od hloubkového čidla (např. ze zařízení Microsoft Kinect) se musí před použitím předzpracovat [1]. Prvním krokem je výpočet metrické hloubky obrazu. To se provádí podle

rovnice 3.1. Celá rovnice se násobí ještě zlomkem M/D_{max}. Autoři článku People Detection in RGB-D Data tuto úpravu uvádí jako nutnost pro numerickou stabilitu výpočtu [1].

$$d = \frac{(8 * B * F_x)}{(V_{max} - v)} * \frac{M}{D_{max}} \tag{3.1}$$

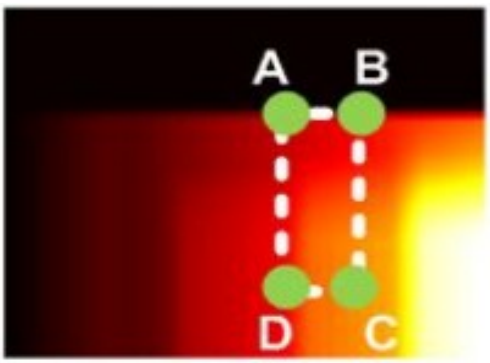
Proměnné v rovnici mají následující význam. Hledaná hodnota - metrická hloubka je reprezentována písmenem *d*. Vzdálenost mezi IR projektorem a IR kamerou v metrech značí proměnná *B*. F_x je horizontální ohnisková vzdálenost IR kamery, V_{max} znamená maximální konverzi pro Kinect. Hodnota *v* se mění s každým pixelem - je to hodnota přicházející z Kinectu pro daný pixel obrazu. Druhý zlomek se skládá ze dvou konstant - *M* je konstantní nárůst a D_{max} je maximální dosah Kinectu v metrech. Pokud vyjde hodnota proměnné *d* záporná, výsledek se zahodí a je vhodné ho nahradit hodnotou okolního pozadí.

Když je matice naplněná výslednou metrickou hloubkou obrazu, můžeme ji dále upravit podle rovnice 3.2. Touto rovnicí získáme pro každý pixel jeho váhu *s*. Výsledek rovnice se uloží do matice.

$$s = \frac{F_y * H_m}{d} * \frac{1}{H_w} \tag{3.2}$$

Proměnná F_y značí vertikální ohniskovou vzdálenost IR kamery, H_m je průměrná výška člověka v metrech a písmeno *d* je dříve spočtená metrická hloubka pro každý pixel. Celá rovnice se nakonec vynásobí zlomkem, kde participuje výška detekčního okna při prvním prohledávání (H_w) v metrech. Po získání matice s vahami obrazu začne prohledávání. To je navrženo tak, že se metodou Sliding window projde celý obraz a k nalezení objektu se použije nerovnice 3.3.

$$(A + C) - (B + D) > 0 \tag{3.3}$$



Obrázek 3.8: Zobrazení pozice bodů užitých v nerovnici 3.3 (převzato z [2])

V článku People Detection in RGB-D Data detekují pozitivní okno, pokud je výsledek nerovnice větší než nula [1]. Pozice pixelů (označených písmeny A, B, C a D) užitých v nerovnici, jsou ukázány na obrázku 3.8. Okna s pozitivním nálezem se uloží do vektoru. Díky výpočtu pozitivních oken se značně redukuje počet částí obrazu, jež se předkládají klasifikátoru k ohodnocení a tím se algoritmus výrazně urychlí.

3.4.2 ComboHod

Metoda ComboHod, navržená od pana Luciana Spinella a Kaia Arrase, má hlavní myšlenku v tom, že spojí výstupy dvou jiných detektorů [1]. Jedná se o výše popsané metody HOG detekce (viz kapitola 3.1.1 a 3.2.1) a HOD detekce (viz předchozí kapitola). Postup výpočtu metody je znázorněn na obrázku 3.9.

Tyto metody (HOG a HOD detekce) jsou používány ve své podstatě úplně stejně, jak jsou popsány výše. Jediný rozdíl je v tom, že konečný klasifikátor SVM nevrací příslušnost obrázku do třídy (tedy název třídy), ale vrací hodnotu ohodnocení obrázku. S těmito hodnotami se dále pracuje dle rovnice 3.4, kde písmeno *p* značí výslednou pravděpodobnost výskytu objektu v obraze.

$$p = p_D + k * (p_G - p_D) \tag{3.4}$$

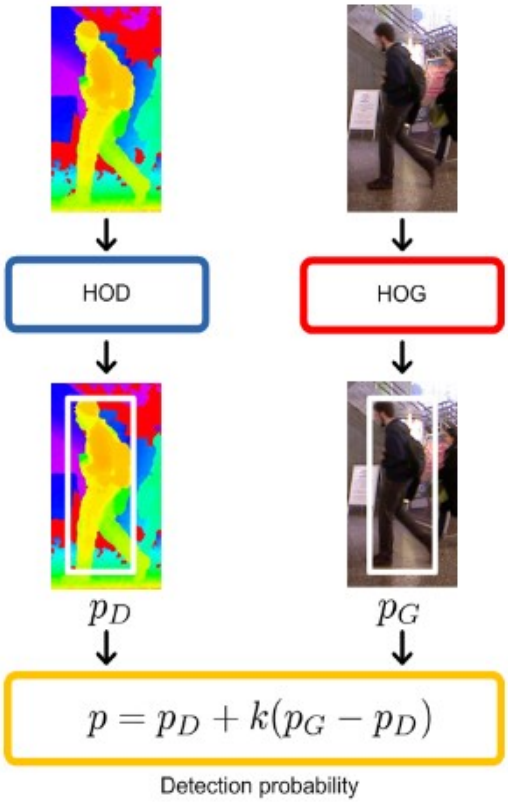
Jak je vidět na obrázku 3.9, proměnná p_D značí ohodnocení obrázku z HOD detektoru. Proměnná p_G značí ohodnocení obrázku od metody používající HOG příznaky. Proměnná *k* představuje konstantu, která se spočítá z ROC křivek dvou spojovaných detektorů. Pro výpočet této konstanty se použije rovnice 3.5.

$$k = \frac{\sigma_D^2}{\sigma_D^2 + \sigma_G^2} \tag{3.5}$$

Proměnná σ_D se spočítá tak, že se vydělí hodnota chybovosti při EER (equal error rate) pro HOD detektor hodnotou chybovosti HOG detektoru pro EER. Proměnná σ_G je definována rovnicí 3.6.

$$\sigma_G^2 = 1 - \sigma_D^2 \tag{3.6}$$

Hodnota výsledné proměnné *p* se porovná s určeným prahem detektoru (práh se určí pomocí ROC křivky). Hodnota větší než zvolený práh znamená přítomnost objektu v obraze.



Obrázek 3.9: Schéma metody ComboHod (převzato z [1])

4 Návrh detektoru postavy

Začátkem kapitoly představuji program jako celek, jeho předpokládanou funkčnost a popisuji základní rozdělení aplikace na části. Poté popisuji algoritmy detekce lidí a způsob získávání informací z detekovaných objektů. Dále uvádím detaily návrhu statistické části programu a způsob, kterým zamýšlím prezentovat získaná data.

Mým cílem bylo vytvořit detektor lidské postavy, jenž využívá obrazová i hloubková data a tím se stane robustnějším. Detektor je poté prakticky využit pro realizaci aplikace, která shromažďuje statistické informace o lidech, pohybujících se ve sledovaném prostoru. Program počítá lidi, kteří prošli před statickou kamerou, umístěnou ve výloze obchodu. Získané informace poté dle volby uživatele zobrazuje ve formě grafů. Rozdělila jsem tedy aplikaci na dvě hlavní části, které fungují nezávisle na sobě. Program obsahuje tyto části:

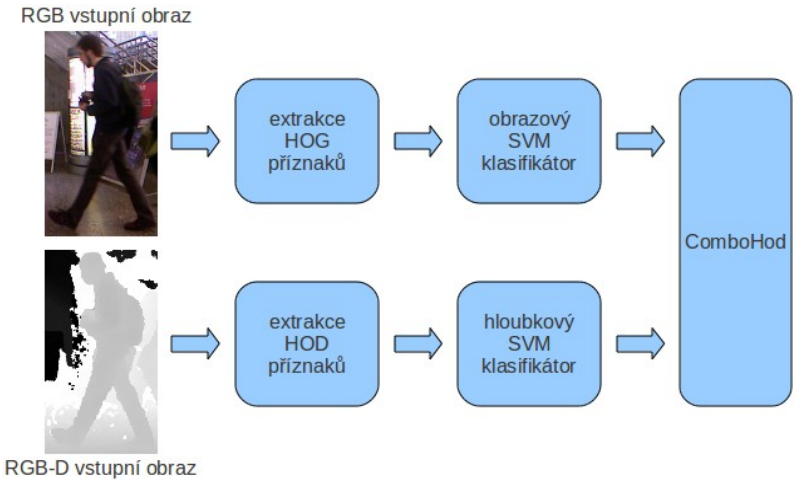
- Detekční část - zaměřena na detekci lidí v obraze pomocí metody ComboHod (viz kapitola 3.4.2), jež slučuje výstupy dvou nezávislých klasifikací obrazu [1].
- Statistická část - prezentuje výsledná statistická data získaná navrženým detektorem. Tuto část představuje grafické uživatelské rozhraní, ve kterém uživatel vybere požadovaná data, která se zobrazí grafem.

Důležitou položkou pro můj program jsou také testovací data, která blíže popisuji v kapitole 4.3.

Při pohledu na aplikaci jako celek by bylo ideální realizovat výstup detekční části aplikace do databáze. Databáze však není předmětem mé bakalářské práce, a proto budu ukládat informace do souborů v *.xml* formátu.

4.1 Detekční část aplikace

Detektor, který identifikuje lidskou postavu na videu, jsem navrhla podle popisu pana Luciana Spinella a Kaia Arrase [1]. Jejich návrh počítá s detekcí z obrazových i hloubkových dat (viz kapitola 3.4.2). Detekce probíhá zároveň dvěma podobnými způsoby (viz Obr. 4.1). První detektor pracuje s obrazovými daty, využívá kombinaci HOG příznaků a klasifikátoru SVM. Druhý způsob je detekce z hloubkových dat, kde se využívá HOD příznaků a SVM klasifikátoru. Klasifikátor pro hloubková data bylo nutné si ručně natrénovat. Výstupy z těchto dvou klasifikátorů se poté sloučí do jednoho výsledku, který podle zvoleného prahu označí, zda se osoba v obraze opravdu nachází či nikoli. Ke sloučení výstupů z detektorů se využívá také konstanta `k`, kterou si předem vypočtu pomocí grafů ROC křivek detektorů. Získávání konstanty je popsáno v kapitole 3.4.2.



Obrázek 4.1: Schéma metody ComboHod

Pro nasazení detektoru v praxi je nutné, aby se zpracování dat provádělo ihned. Proto jsem detektor navrhla pro práci v reálném čase.

4.1.1 HOG

Detektor založený na HOG příznacích pracuje s obrazovými daty, která se použitím Kinectu nijak nemění od obrazů z běžné kamery. Podrobně jsou HOG příznaky popsán v kapitole 3.1.1. Pomocí funkcí z knihovny OpenCV a HOG příznaků získávám z framu deskriptory. Tyto deskriptory předkládám již zmíněnému naučenému SVM klasifikátoru.

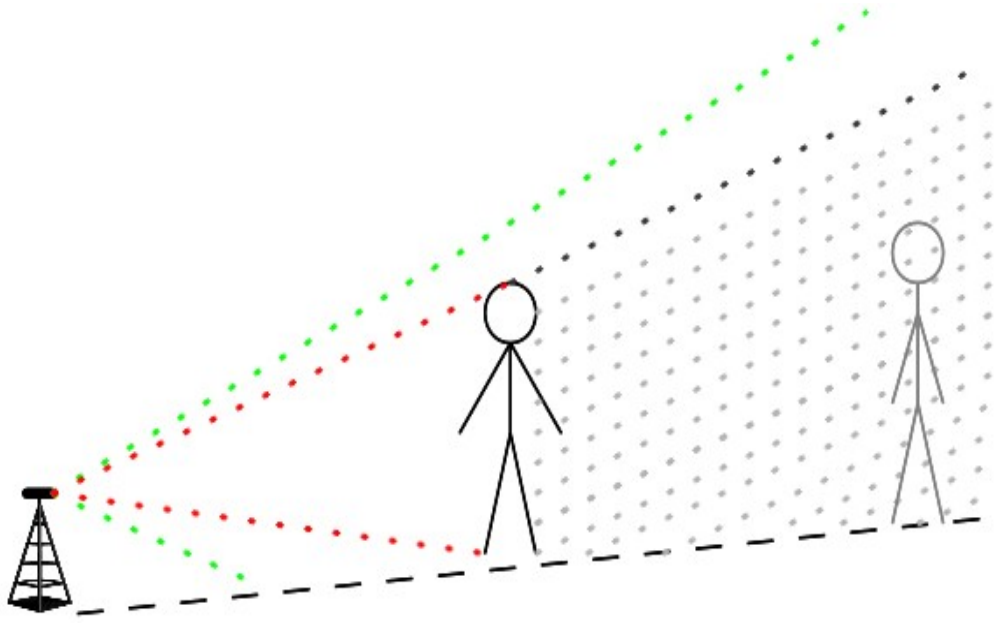
Nevýhoda této metody spočívá v tom, že se do klasifikátoru posílá velké množství dat. Při kombinaci metody Sliding window s detekcí multi-scale je deskriptorů tolik, že se klasifikace znatelně zpomalí. Naproti tomu je tento způsob detekce velice odolný proti vzdálenosti hledaného objektu od kamery.

4.1.2 HOD

Ke zpracování hloubkových dat je třeba natrénovat si vlastní klasifikátor. K trénování jsem použila datovou sadu získanou z internetu [1][2] v kombinaci s vlastní datovou sadou (popis datových sad přiblížen v kapitole 4.3).

Hloubková data zpracuji podobně jako obrazová, což je podrobně popsáno v kapitole 3.4.1. Nejprve se však musí upravit formát vstupních dat. Příchozí matice hloubkových dat je v odlišném formátu, než který vyžadují funkce, které následně budu používat pro další výpočty.

K urychlení detekce vypočítávám nejprve váhu obrazu pro každý pixel. Tyto váhy se ukládají do speciální matice. Tuto matici poté procházím metodou klouzajícího okénka, viz kapitola 3.3.1. Dle článku People Detection in RGB-D Data [1] jsem navrhla vektor, do kterého uložím pouze ty části obrazu, kde se nacházejí váhy s pozitivní hodnotou pro výskyt člověka. Tento výpočet vah provedu při metodě Sliding window dle nerovnice 3.3.



Obrázek 4.2: Dohlednost Kinectu - Černá přerušovaná čára značí podlahu. Zelená tečkovaná čára značí zorné pole Kinectu, který je umístěn na podstavci (cca jeden metr nad zemí). Červená tečkovaná čára značí zorné pole, ve kterém se vyskytla postava. Šedě potečkovaná plocha za první postavou značí místo, které je v zákrytu za touto postavou.

4.1.3 ComboHod

Tuto metodu jsem převzala podle popisu pana Spinella [1] a upravila jsem ji pro svou potřebu. Metoda spočívá ve spojení výstupů dvou SVM klasifikátorů. Pan Spinello v článku tyto výstupy ještě dále upravuje pomocí sigmoidy, ale pro moji práci tato úprava nebude nutná. Metoda je přiblížena v kapitole 3.4.2.

Po provedení všech výpočetních kroků metody se výsledný vektor s obdélníky ještě upravuje. Úprava spočívá v tom, že se upraví velikosti obdélníků, které se z větší části překrývají. Procházím celý vektor a porovnám každý prvek z ostatními. Pokud jsou v zákrytu, sloučí se tyto dva obdélníky. Pokud jsou dva obdélníky nad sebou, sloučí se také. Tento krok si mohu dovolit proto, že nikdy v reálném světě nemůže nastat, že budou dva objekty přesně (nebo velmi blízko) nad sebou. To je dáno především umístěním Kinectu (ideálně 1 metr nad zemí) a perspektivou objektů - bližší objekt by vždy překryl vzdálený vzor (viz obrázek 4.2). Tento vzor poté nemůže být zachycen hloubkovým čidlem ani obrazovou kamerou.

4.1.4 Sledování objektů

Každý objekt je uložen v seznamu detekovaných postav. Sledování lidských postav probíhá tak, že se porovnávají souřadnice a parametry zaznamenaných objektů. Při nové detekci se prochází seznam již detekovaných objektů, zda se nejedná pouze o přesun již zaznamenaného objektu. Mohou nastat tři situace:

- Pokud není nalezena shoda, vloží se nový objekt do seznamu. Při získávání informací o novém objektu se uloží jeho parametry a také čas, kdy byl detekován. Navíc se uloží ještě jeden pomocný čas, který se mění s každou novou detekcí daného objektu a je využíván pro zjišťování, zda se daný objekt pohybuje. Při první detekci člověka se také dle jeho x-ové souřadnice určí směr, ze kterého přišel před kameru.
- Pokud jsou parametry nového objektu velice podobné nějakému již zaznamenanému objektu (především tedy x-ová souřadnice a šířka objektu), jedná se o tentýž objekt, který se pouze posunul. Přepíší se tudíž souřadnice a velikostní parametry starého objektu na nové. Výpočet pro pohyb objektu po x-ové souřadnici pro objekt je popsán koncem této kapitoly.
- Pokud jsou rozdíly mezi parametry nového a starého objektu minimální, program prohlásí objekt za statický. Při první detekci stagnace se uloží čas, kdy k zastavení došlo. Zároveň se také podle x-ové souřadnice objektu určí část výlohy, ve které k zastavení došlo. Program dále provede změnu parametrů starého objektu tak, jak bylo popsáno v předešlém bodu. Hodnota doby stagnace se získá odečtením času začátku a konce stagnace. Zastaví-li se objekt znovu, pak se před uložením počtu sekund nehybnosti porovná stávající uložená hodnota a nová získaná hodnota. Nakonec se uloží větší číslo a informace s ním spojené.

Při zvolení limitu pro pohyb člověka po x-ové ose jsem vycházela z průměrné rychlosti lidské chůze, za což je obecně brána rychlost 4km/h (= 1,1 m/s). Předpokládám, že moje aplikace bude zvládat kvůli pomalým algoritmům zpracovat jeden frame za vteřinu. Vezmeme-li v úvahu parametry chodby obchodního centra, zabírá Kinect do šířky prostor zhruba o osmi metrech. Vypočetla jsem tedy poměrnou velikost, jakou znamená jeden pixel pro tuto délku (dle rovnice 4.1).

$$\frac{délka\ prostoru[mm]}{rozlišení\ obrazu[pix]} = \frac{8000}{640} = 12{,}5\ \text{mm/pix} \qquad (4.1)$$

Pokud beru, že zpracuji 1 frame za sekundu, člověk za tuto dobu ujde 1,1 metru. Výpočet, o kolik pixelů se člověk zhruba pohne v rámci dvou odlišných obrazů, jsem provedla dle následující rovnice (4.2):

$$\frac{délka\ ušlé\ trasy\ za\ 1s[mm]}{délka\ ,kterou\ značí\ \ jeden\ pixel[mm]} = \frac{1100}{12{,}5} = 88\ \text{pix} \qquad (4.2)$$

Spočtená hodnota označuje větší polovinu šířky člověka. Budu tedy brát jako mezní pohyb polovinu šířky objektu.

Pro detekci se také využívá jeden pomocný čas, ve kterém se uchovává doba, kdy byl naposledy objekt detekován. Limit nastavím na 3 sekundy (za tuto dobu program zpracuje přibližně tři framy). Pokud během nich není objekt znovu detekován, nastavím danému objektu příznak a dále se s ním nepočítá. Za každým 60. framem se používají metody, které ukládají informace z vyřazených objektů z listu do souboru. Po zápisu jsou vyřazené objekty ze seznamu smazány.

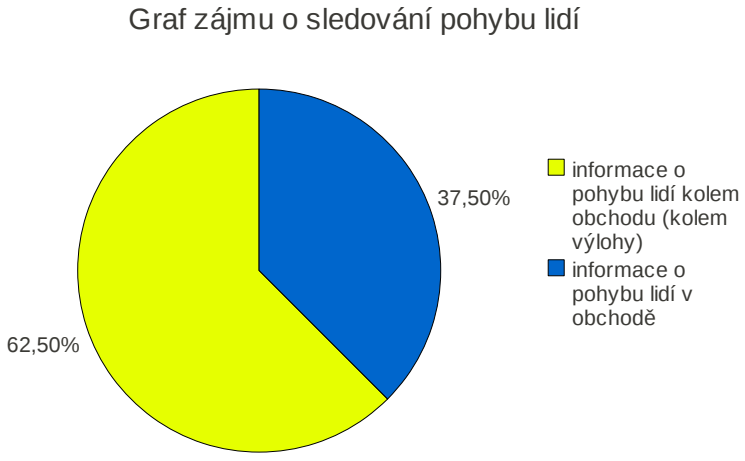
4.1.5 Získávání informací o detekovaném objektu

Informace o detekovaném objektu získávám pomocí jednoduchých výpočtů. Vycházím přitom z velikosti a pozice detekované části obrazu.

Údaje o pozici člověka v okně pro statistiku číslo jedna a pět získávám jednoduše podle pozice objektu získané z detektoru. Statistiky jsou probírány v následující kapitole. Při detekci nového objektu zaznamenám čas detekce, který se využívá pro druhou statistiku. Pokud zaznamenám, že se objekt přestal hýbat, uloží se druhý čas a jeden pomocný. Tyto časy se použijí ve třetí a pro výpočet čtvrté statistiky. Informace pro čtvrtou statistiku se spočítají tak, že se mění pomocný čas po dobu stagnace objektu.

4.2 Statistická část programu

Aplikace počítá osoby, které se vyskytnou před kamerou. Dále jsem se zamyslela, jaké informace by se daly z detekovaného objektu získat a pro jakou sféru by byly vhodné. Navrhla jsem tedy aplikaci, která je díky detektoru schopna počítat lidské postavy, které se vyskytnou před výlohou kamenného obchodu, umístěného v obchodním centru. Umístění v obchodním centru je důležitá položka pro tuto aplikaci, jelikož Kinect, jež využiji pro tuto práci, je citlivý na změnu osvětlení. Chodby v nákupních centrech jsou konstantně osvětlovány stejnou intenzitou světla (většinou se svítí zářivkami i ve dne) a jsou široké zhruba šest až osm metrů. Tyto parametry jsou pro Kinect téměř ideální.



Obrázek 4.3: Výsledky dotazníku - prostor pro sledování lidí

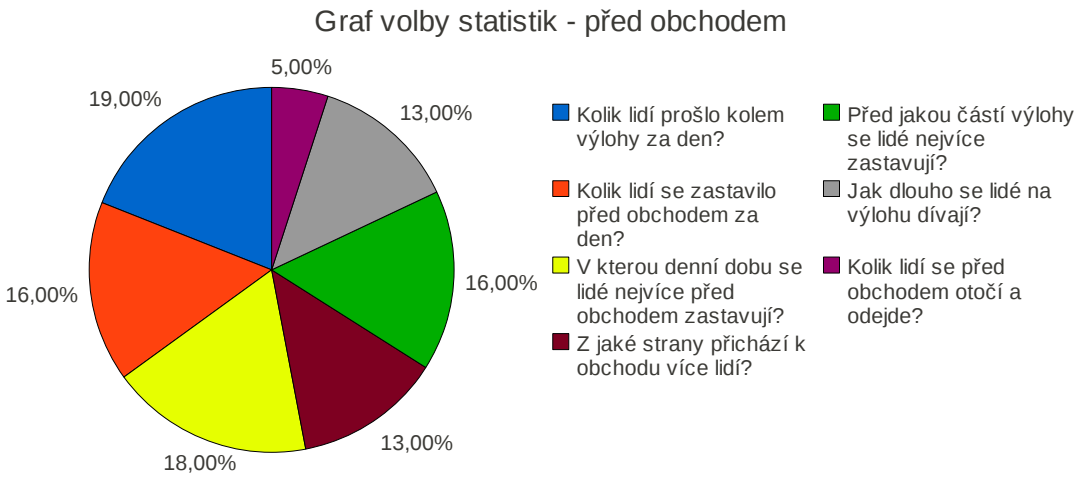
Interakci programu s uživatelem zajišťuje grafické uživatelské rozhraní, které je přizpůsobené očekávané kvalifikaci uživatele (mírně pokročilá až středně pokročilá). Rozhraní nabízí uživateli možnosti volby z pěti statistik.

Volbu statistik jsem provedla následovně. Při začátku práce na tomto projektu jsem vytvořila dotazníky zaměřené na lidi, kteří vlastní nebo pracují v nějakém obchodě. Anketu jsem poté rozeslala po internetu. Ptala jsem se cílové skupiny, které statistiky by považovali za nejpřínosnější. Další

položka se týkala umístění kamery (před nebo v obchodě). Dále jsem do dotazníku zařadila dotaz, za jaké období by se měly statistiky ideálně zobrazovat. Na základě výsledků z tohoto dotazníku jsem navrhla konečnou podobu statistické části programu. Nejdůležitější výsledky dotazníku jsou uvedeny v grafech na obrázku 4.3, 4.4 a 4.5. Zbylá část vyhodnocení dotazníku se nachází na konci této práce (viz Příloha 2).

Statistiky zařazené do aplikace:

- Statistika ST1 - zobrazuje graf, z jaké strany výlohy lidé nejvíce k obchodu přicházejí. Šířku videa jsem si rozdělila na tři části. Zleva a zprava jsem určila interval 120 pixelů. Pokud se postava ve videu prvně objeví v tomto rozmezí, je určen příslušný směr. Jinak je bráno, že se postava ke kameře přibližuje zpříma.
- Statistika ST2 - zobrazí počet lidí, kteří prošly kolem výlohy za zvolený čas. Den jsem rozdělila na osm úseků po třech hodinách. Toto rozdělení využívám i ve třetí statistice.
- Statistika ST3 - ukazuje počet lidí, kteří se během dne zastaví před výlohou obchodu.
- Statistika ST4 - znázorní, na jak dlouho se lidé před výlohou zastavili. Rozpětí času je rozděleno do pěti skupin po třech vteřinách.
- Statistika ST5 - ukazuje, v jaké části výlohy se lidé nejvíce zastavují. Výlohu (celou šířku videa) jsem rozdělila taktéž na pět částí. Úseky jsou stejně velké a označila jsem je slovně dle umístění ve videu.



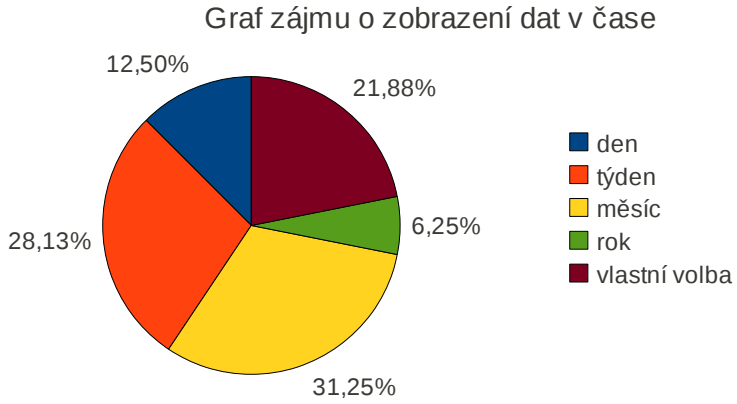
Obrázek 4.4: Výsledky dotazníku - volba statistik

4.2.1 GUI

K modelování rozhraní využívám knihoven a frameworku Qt, jelikož nabízí spoustu možností pro modelování vizuální stránky projektu. Důraz je kladen především na jednoduchost a intuitivnost ovládání této části programu. Rozhraní jsem navrhla tak, aby odpovídalo formální normě. Použité barvy by neměly nijak rozptylovat pozornost člověka, který s programem pracuje.

Rozhraní dává na výběr typ grafu, kterým se poté zobrazí vybraná statistika. Na výběr je z pěti statistik, k nimž se vztahují vysvětlivky ve spodní části GUI.

Za jediné místo v uživatelském rozhraní, které by mohlo být složitější na pochopení ovládání, považuji volbu dní, ze kterých se zobrazuje statistika. Do rozhraní je vložen malý kalendář klasické podoby. Při kliknutí na vybraný termín v kalendáři se po potvrzení zobrazí data z vybraného dne. Nebo je zde možnost vybrat si zobrazení dat za minulý měsíc či minulý týden od data označeného v kalendáři (tuto možnost jsem zvolila na základě výsledků z dotazníku - viz Obr. 4.5). Data z vybraných statistik lze uložit do textového souboru. Tato možnost je uživateli přístupná z hlavního menu.



Obrázek 4.5: Výsledky dotazníku - seskupení dat do časových úseků

Obrázek 4.4: Příklad trénovacích dat [1][2]
vlevo je hloubkový obraz, vpravo je frame z RGB kamery

4.3 Datová sada

Detektor pro hloubková data jsem si musela sama natrénovat. K trénování jsem se rozhodla použít datovou sadu od pana Spinella [1][2] (Obr. 4.4) a přidala jsem sadu vlastních anotovaných obrázků. Sada anotovaných obrázků od pana Spinella je pro mne výhodná v tom, že je obrázků více jak tři tisíce a tím bude SVM klasifikátor lepší. Avšak některé obrázky jsou anotovány i pro malou hloubkovou změnu (např. výskyt objektu ve vzdálenosti cca 15 metrů od Kinect), což může v mých zamýšlených podmínkách způsobit chybnou detekci. Můj klasifikátor nemusí být tak jemný, protože ideálně nebudu snímat data do hloubky více jak deseti metrů, proto jsem nakonec využila jen zhruba polovinu datové sady od pana Spinella.

Ke zjednodušení testování aplikace jsem natočila sadu vlastních videí, kde jsem se snažila napodobit podmínky, které by měly být ideálně v obchodním domě. Videa jsem natočila pomocí příkladové aplikace z OpenNI knihovny.

Druhá testovací sada obsahuje videa, kde jsem se snažila napodobit extrémní podmínky, které mohou při používání aplikace nastat (viz Obr. 4.5).

Obrázek 4.5: Příklad obrázků z datové sady videí, pořízené za nepříznivých podmínek
a) ostré slunce - způsobuje rušení hloubkového obrazu,
b) velký hloubkový prostor - místo pozadí hloubková data chybí,
c) noc - chybí RGB obraz (obrázek vlevo s pozadím, obrázek vpravo s větší hloubkou obrazu),
d) šero - způsobuje rušení pro RGB kameru,
e) zářivkové světlo (ideální stav).

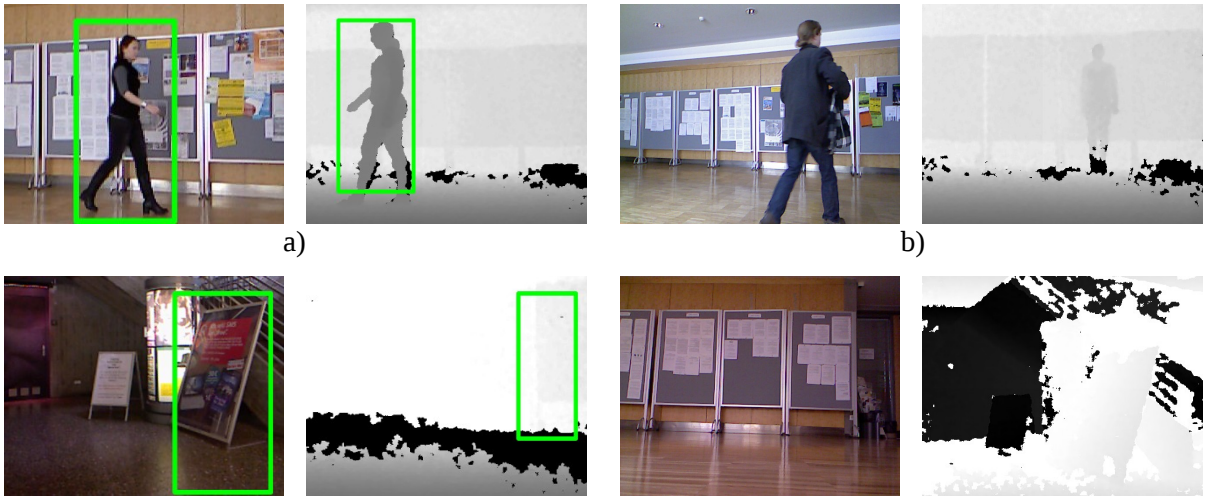
5 Implementační detaily

Tato kapitola přibližuje postup práce na projektu, problémy, které nastaly při vývoji aplikace, využité algoritmy a metody, uplatněné v této práci. Postupně popisuji nejdůležitější třídy, které implementují hlavní část funkčnosti mého projektu.

Aplikaci jsem vytvářela pod systémem Linux (Ubuntu 10.04). Jako vývojářské prostředí jsem si zvolila Qt framework, neboť je dobře dostupný. Používám ho často pro vývoj jiných projektů a obsahuje efektivní prostředky pro modelování grafického uživatelského rozhraní. Pro zobrazování grafů jsem vybrala program Gnuplot verze 4.6, který je multiplatformní a snadný na ovládání. Tento program je taktéž volně dostupný na internetu. Pro detekční část programu využívám dvě speciální volně dostupné knihovny. Počítačové vidění umožňuje knihovna OpenCV [14][9], OpenNI je knihovna zajišťující interakci se zařízením Kinect [15]. Je vhodné při instalaci nejprve přidat knihovnu OpenNI a poté OpenCV.

5.1 Trénování SVM klasifikátoru

Pro vytvoření HOD detektoru jsem si musela natrénovat vlastní klasifikátor obrázků. V článku People Detection in RGB-D Data je doporučen lineární SVM klasifikátor [1] (viz kapitola 3.2.1). Dle dokumentace knihovny OpenCV dostupné na internetu a článku A Tutorial on Support Vector Machines for Pattern Recognition [3], jsem si vytvořila aplikaci k trénování klasifikátoru. Aplikace zprvu načte obrázek a jeho anotaci z předem připraveného textového souboru. Poté spočítá deskriptor dané výseče obrazu. Deskriptory se ukládají do matice, která se později použije pro trénování klasifikátoru. Společně s deskriptory se ukládá do speciální matice, která má jen jeden sloupec, třídní zařazení obrázku. Jeden řádek trénovací matice odpovídá řádku v matici s třídním ohodnocením. Těmito maticemi jsem následně natrénovala SVM klasifikátor. K trénování a testování jsem použila datovou sadu popsanou v kapitole 4.3. Na obrázku 5.1 jsou příklady správně a falešně detekovaných lidských postav.



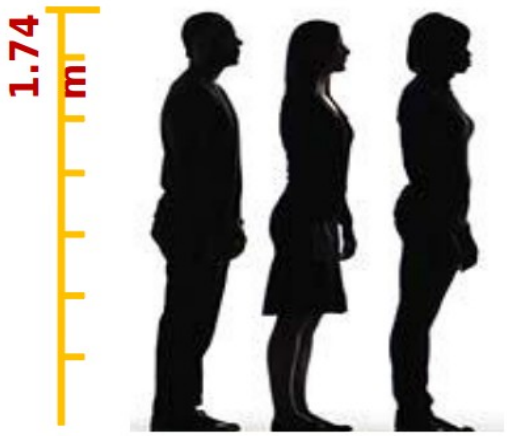
Obrázek 5.1: a) správně pozitivně detekovaný obrázek, b) falešně negativní obrázek, c) falešně pozitivní obrázek, d) správně negativní obrázek (převzato z [1])

5.2 Třída detectors

Tato třída obsahuje hlavní funkční jádro projektu. Jedná se o implementaci dvou detektorů, které se následně spojí v jeden výstup. Pro implementaci první části detektoru, který je založený na HOG příznacích z RGB obrazu, jsem se rozhodla využít již natrénovaný detektor z knihovny OpenCV . Pro můj projekt byla nutná úprava detekční funkce tohoto objektu, což je přiblíženo v kapitole 5.2.1. Vytvoření druhého detektoru, založeného na získávání obrazových příznaků z hloubkových obrázků, popisuji v následujících odstavcích.

Postup natrénování klasifikátoru pro hloubkový obraz jsem popsala v předešlé kapitole. Rozdíl mezi dvěma klasifikátory je ten, že SVM pro hloubková data neklasifikuje každý kousek obrazu pod každou velikostí detekčního okna, tak jako detektor z knihovny OpenCV. Do tohoto klasifikátoru posílám pouze oblasti, které předzpracující funkce označila za oblast, kde je pravděpodobně umístěn hledaný objekt. Tato předzpracující funkce značně zkrátí čas vyhodnocování.

Do předzpracující funkce jsem zahrnula výpočty metrické hloubky a váhy pixelu, popsané v kapitole 3.4.1. Hodnoty proměnných v rovnici pro výpočet metrické hloubky jsem převzala z článku People Detection in RGB-D Data [1] (konkrétně hodnoty pro maximální konverzi pro Kinect, pro konstantní zesílení, pro maximální dosah Kinectu a pro průměrnou výšku člověka - viz Obr. 5.2). Díky spojení OpenNI a OpenCV knihovny jsou dostupné příznaky pro objekt, který představuje zařízení Kinect. Přes tyto příznaky získávám hodnoty pro proměnnou značící vzdálenost mezi IR projektorem a IR kamerou a proměnnou pro horizontální ohniskovou vzdálenost. Vertikální ohniskovou vzdálenost jsem zjistila na internetu [6]. Hodnota ohniskové vzdálenosti v milimetrech je získána ze stránek ČVUT [11]. Konstantu `k`, užívanou při spojení výstupů detektorů, jsem určila po získání ROC křivek detektorů. Nejprve jsem pro každý detektor vypočítala práh, při kterém je chybovost falešně pozitivních a falešně negativních detekcí stejná. Tyto hodnoty jsem poté vydělila, jak je popsáno v 3.4.2. Hodnota chybovosti při EER pro HOG detektor byla 17, pro HOD detektor 9.



Obrázek 5.2: Průměrná výška člověka z testovacích dat, převzato z [1]

Po získání matice s váhami pixelů ji dále zpracovávám tak, že porovnávám váhy sousedních pixelů (viz nerovnice 3.3). Při větším rozdílu hodnot uložím příslušnou část obrazu do vektoru. Rozdíl hodnot vah je v článku People Detection in RGB-D Data [1] porovnáván s nulou. Pro moji práci lze však výsledek porovnávat s hodnotou 4 (určeno experimentálně). To je zapříčiněno tím, že počítám s omezenou hloubkou videa, která je určena účelem aplikace. V rámci experimentování s hloubkovým dosahem detektoru budu tuto konstantu měnit.

Při procházení obrazu odshora logicky nejprve narazím na hlavu člověka. Proto se příslušnou částí obrazu, zmíněnou v předchozím odstavci, rozumí to, že do vektoru uložím část obrazu o velikosti minimálního rozměru detekčního okna, které má detekovaný bod v polovině své šířky

(neboť hlava bývá ve většině případů při chůzi na nebo blízko svislé osy lidského těla). Při dalším porovnání se před uložením hodnot do vektoru dotazuji, jestli nový detekovaný bod neleží v nějaké již detekované oblasti, která má podobnou hodnotu vah. Při větším rozdílu vah se jedná o dva objekty v prostoru, co se právě překrývají a proto se uloží nový záznam do vektoru. Toho, že by minimální okno nepojalo celé lidské tělo se nemusím bát, jelikož na konci funkce procházím tento vektor s různou velikostí oken. Velikosti detekčních oken jsou předmětem experimentu popsaného v kapitole 6.1. Výsledkem předzpracující funkce je vektor, který obsahuje obdélníky, kde je nejpravděpodobnější výskyt hledaného objektu.

Po předzpracování ohodnotím úseky obrazu, uložené ve vektoru. Váhy, získané od SVM klasifikátoru, uložím do vektoru. Místa možného výskytu seskupím[1] a ponechám pouze váhy, které místům přísluší. Dané váhy, získané z HOD detektoru, jsou vynásobeny číslem -1, protože pozitivní nález objektu pro HOD detekci je označen vahou menší než nula, ale pro HOG detekci je pozitivní nález větší než nula. Hodnoty vah z HOG detektoru se pohybují v rozmezí[2] od -1 do +1, rozmezí vah HOD detektoru je -2 až +2. Proto aby se škály hodnot vyrovnaly, násobím váhy z HODu navíc ještě jednou polovinou.

Nyní mám připravené výstupy z HOD i HOG detektoru - dva vektory s vahami a dva vektory s možnými oblastmi výskytu. Tyto výstupy spojuji tak, jak je popsáno v kapitole 3.4.2. Identifikaci oken možných výskytů z jednotlivých vektorů provádím podle toho, jak moc se blízká okna překrývají. Pokud není nalezena shoda pro nějaké okno z vektoru možných výskytů, je místo chybějící hodnoty vah v druhém vektoru dosazena nula (nula značí stav, kdy si klasifikátor není jistý, zda objekt na místě je či není). Výsledek slučovací rovnice se porovnává s hodnotou 0,05 (práh při EER pro detektor ComboHod - určeno při experimentování). Je li výsledek rovnice větší, jsou vypočteny souřadnice rámečku a ten je uložen do finálního vektoru. Po ohodnocení všech oken získaných z HOG a HOD detektoru, se vektor ještě finálně poupraví (popsáno v kapitole 4.1.3). Tato úprava zajišťuje, že nejsou ve vektoru duplicity ani obdélníky s velkým překrytím. Po dokončení úprav parametrů obdélníků z vektoru se získají o výsledných objektech informace důležité pro statistickou část.

5.2.1 Třída MyHogDescriptor

Kvůli spojení výstupů dvou detektorů bylo zapotřebí získat ohodnocení dané části obrazu přímo z výstupu ohodnocovací funkce a nikoli získávat popisovací hodnotu třídy, do které byl obrázek přiřazen SVM klasifikátorem. Knihovna OpenCV nenabízela funkci, která by vracela přímé ohodnocení obrazu pro danou výseč obrazu[3]. Proto jsem vytvořila třídu, která zdědila vlastnosti objektu `HogDescriptor`. Následně jsem přepsala metodu `detectMultiScale` tak, aby navracela požadovanou kombinaci výsledku, přičemž jsem si určila, že pořadí prvků vektoru s ohodnoceními bude závislé na pořadí prvků vektoru, který vrací pozitivní části obrazu. Obsah knihovny *myhogdescriptor.h* je převzat z knihovny OpenCV, metoda `myDetectMultiScale` je upravena pro účely této práce.

Redukci ohodnocení, získaných po detekci, provádím výpočtem podle `x`-ové souřadnice obdélníku, který je určen jako místo nálezu detekovaného objektu. Před získáním jedinečného detekčního obdélníku si tento vektor nálezů zkopíruji. Na první vektor s obdélníky použiji funkci z knihovny OpenCV, která vyhodnotí četnost oken v jednotlivých oblastech obrazu, a vrátí pro danou část obrazu jen jeden obdélník, který detekuje objekt. Poté identifikuji vybraný obdélník v kopii prvotního vektoru. Ohodnocení daného obdélníku spočítám tak, že pro jeho `x`-ovou souřadnici vezmu okolí 10 pixelů. V tomto okolí najdu nejvyšší ohodnocení a to přiřadím danému obdélníku. Pozice

1 Seskupením se myslí smazání menších obdélníků, které většinou svého obsahu leží v jiném obdélníku.

2 Hodnoty nejsou přesně omezeny dále popsanými intervaly, avšak překračují je jen zřídka kdy. Pokud se interval překročí, zpravidla je to v obou případech (překročení u HODu i HOGu).

3 Knihovna OpenCV obsahuje funkce, které vrací ohodnocení obrazu. Ale při metodě DetectMuliScale vrací vektor hodnotící číslo pro všechny pozitivní detekční okna, nicméně nedokáže určit konkrétní číslo pro výseč obrazu, kterou vrátí jako pozitivní nález objektu.

ohodnocení ve vektoru je stejná jako pozice detekovaného obdélníku ve vektoru detekovaných oken, ke kterému číslo přísluší. Upravená metoda `myDetectMultiScale` vrací vektor možných oblastí výskytu člověka a vektor příslušných vah obrazu.

```
<?xml version="1.0" encoding="UTF-8"?>
<PERSON>
   <DIR>F</DIR><STOP>L</STOP><COME>22:21</COME><LOOK>22:22</LOOK><TIME>5</TIME>
</PERSON>
<PERSON>
   <DIR>L</DIR><STOP>R</STOP><COME>08:16</COME><LOOK>08:16</LOOK><TIME>9</TIME>
</PERSON>
```

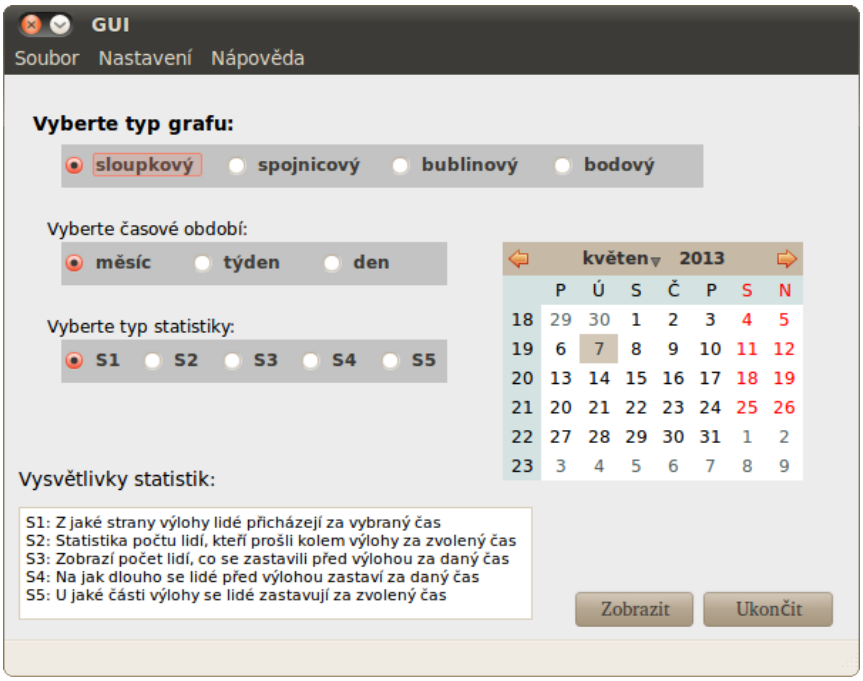
Kód 5.1: Příklad výstupního souboru z detektoru

5.3 Třídy pro ukládání informací o objektech

Operace s daty jako ukládání a načítání ze souboru provádí třída `Op_with_data`. Pro usnadnění práce s daty jsem si navrhla vlastní tagy pro ukládání do *.xml* souborů (viz Kód 5.1). Název těchto souborů vytvářím pomocí data, kdy byly informace získány (ve formátu DD_MM_RRRR), a klíčového slova *data_file*, které identifikuje daný speciální soubor.

Informace o lidské postavě ukládám předně ve třídě nazvané `People_base`. Tato třída má funkci pouze jako objekt udržující základní data k zobrazení pomocí grafu, využívá se v části pro zobrazení statistik. Od této třídy dědí vlastnosti třída `People`. Tento objekt je využit v detekční části aplikace. Jsou v něm udržována pomocná data pro získávání informací o detekovaném objektu.

Třída `People` implementuje funkce pro získání informací potřebných pro zobrazení statistik. Detekované objekty se udržují v seznamu objektů. Princip vkládání objektů do seznamu je implementován dle kapitoly 4.1.4. Identifikace objektů neprobíhá úplně ideálně. Vlivem nedokonalostí detektoru se zaznamená více objektů, než kolik jich v obraze doopravdy bylo. Po každých 60ti zpracovaných framech se vyvolá zápis dat z objektů do souboru. Zapíší se pouze data z objektů, které mají nastavený příznak expirace. Po zápisu se dané objekty smažou ze seznamu.



Obrázek 5.6: Grafické uživatelské rozhraní pro volbu statistik

5.4 Třídy pro GUI a zobrazování grafů

Pro vytvoření grafického uživatelského rozhraní jsem využila Qt knihoven. Pomocné funkce a implementace činnosti tlačítek jsou umístěny ve třídě s názvem `gui`.

Data, které uživatel požaduje, získávám funkcemi implementovanými ve třídě `graph`. Tyto funkce manipulují s objekty typu `people_base`, popsanými výše. Funkce na zobrazování statistických dat pracují na jednoduchém principu seskupení požadovaných dat ze základních objektů. Požadovaná data poté odešlou buď na zapsání do souboru, nebo na vykreslení pomocí grafu. Tento úkon je řízen dle volby uživatele v grafickém uživatelském rozhraní.

Po spuštění GUI aplikace čeká, až si uživatel vybere statistiku a období, za které chce data zobrazit. Po potvrzení výběru načtu data ze souboru a v podobě objektů typu `people_base` uložím do seznamu. Při opětovném potvrzení volby statistiky (i různé od poslední volby) se nejprve zkontroluje datum, na který se vztahuje nový požadavek. Pokud je datum totožné s poslední volbou, načtou se data z uloženého seznamu. Pokud je datum různé, uložený seznam se smaže a naplní novými objekty z požadovaného data.

Uživatel má na výběr z pěti statistik, ke kterým je v dolním rohu GUI umístěna nápověda. Dále si uživatel může zvolit typ grafu, kterým budou data reprezentována. Příklad konečné vizuální podoby GUI k mé aplikaci je uveden na obrázku 5.6.

5.4.1 Nástroj Gnuplot

Pro zobrazení grafů využívám program Gnuplot. Tento nástroj se mi hodí především proto, že je snadno ovladatelný z příkazové řádky a má širokou škálu možností využití. Program obsahuje předdefinované styly grafů. Tyto styly využívám a dále upravuji. Jejich stylistiku mám uloženou v pomocných souborech.

Při vytváření práce jsem však narazila na problém, který spočívá nejspíše v ovládání programu Gnuplot z Qt frameworku. Při odesílání příkazů pro vykreslení grafu se někde překódovávají data a proto tento program není schopný vypisovat text příslušící grafům v českém jazyce s diakritikou. Proto jsou popisky grafů psány bez diakritiky.

6 Experimenty a vyhodnocení

Tato kapitola se věnuje experimentům s detekční částí aplikace. Na začátku každé podkapitoly popisuji návrh a způsob experimentování. Poté ukazuji a hodnotím výsledky, které jsem popsanými pokusy získala. K vyhodnocení experimentů používám ROC křivky, které vynáší do grafu poměr úspěšně detekovaných prvků vůči mylně detekovaným [7]. Provedla jsem tři sady experimentů:

- První sadou experimentů určuji velikosti detekčních oken jedné třetiny škály pro prohledávání hloubkového obrazu, které povedou k nejlepší funkčnosti detektoru vzhledem ke zvolenému prostředí.
- Druhá sada experimentů má za úkol porovnat funkčnost částí detektoru s finálním detektorem Combohod.
- Ve třetí sadě experimentů měním podmínky při detekci (zářivka, ostré sluneční světlo, šero, úplná tma, zaměření do větší vzdálenosti) a zjistím rozsah použitelnosti vytvořeného detektoru.

6.1 Detekční okna pro HOD

První typ experimentů se zabývá volbou velikosti detekčního okna pro HOD detektor. Pro detektor založený na HOG příznacích z RGB dat jsem využila funkce z knihovny OpenCV a zde je již implementována funkce, která prohledá obrázek se škálou přibližně dvaceti velikostí detekčních oken. Autoři článku People Detection in RGB-D Data píší, že je vhodné prohledávat hloubkový obrázek pouze s jednou třetinou celé škály detekčních oken [1]. Touto úpravou se má docílit znatelného snížení využití paměti. V článku nejsou uvedeny hodnoty, které by se měly v dané jedné třetině zachovat. Což je logické, neboť tím autoři rozšiřují možnost využití daného algoritmu. První sadou experimentů tedy určím velikost základního detekčního okna[4] a počet jeho zvětšení (škálu detekčního okna). V experimentu sleduji úspěšnost detekce a její čas a pokouším se určit nejmenší možné množství prohledávacích oken. Se snížením počtu prohledávacích oken by se měla úměrně sníží i paměťová náročnost.

	Počet navýšení a průměrný čas, který zabere následná detekce [ms]						
počet lidí v obraze	4x	6x	8x	10x	12x	14x	16x
0	28	43	74	92	142	292	349
1	56	73	103	152	207	436	712
2	67	98	121	191	289	626	832
3	96	145	167	265	410	705	972

Tabulka 6.1: Detekce s různým počtem navýšení oken

Při experimentování s částí detektoru, zaměřenou na hloubková data, jsem zjistila, že nejmenší vhodná velikost základního detekčního okna je 128x256 pixelů. Tato velikost je vztažena na to, jak daleko se objekty od kamery pohybují (se zvětšující se hloubkou obrazu se snižují rozdíly mezi objektem a pozadím a to znemožňuje detekci) a také k jejich průměrné výšce. Základní okno je zvoleno tak, aby bylo co nejtěsněji kolem objektu. Při procházení výslednou stupnicí se velikost okna navýší a i větší objekty jsou detekovány. Pokud se před kamerou ocitne člověk menšího vzrůstu, základní detekční okno bude vyhovovat nejvíce. Menší velikosti základního okna nejsou příliš vhodná, jelikož se vzrůstající hloubkou obrazu ztrácejí data svoji přesnost, častěji dochází k artefaktům v obraze a tím se detekce znemožňuje. Při velké hloubce obrazu data úplně chybí.

4 Základním detekčním oknem je myšlena iniciační velikost okna, která se poté dále navyšuje o zvolený přírůstek.

Jako inkrementační hodnoty jsem pro šířku okna určila číslo 16 a pro výšku 32. Tyto hodnoty jsem musela zvolit s ohledem na omezení funkcí knihovny OpenCV (velikost okna musí být násobkem bloku). Menší krok byl v poměru ke zvolenému iniciačnímu oknu zanedbatelný.

Počet navýšení oken	4x	6x	8x	10x	12x	14x	16x
Úspěšnost detektoru	84%	88%	90%	90%	90%	90%	90%

Tabulka 6.2: Procentuální úspěšnost detekce vztažená ke škále detekčních oken

Experimentování se škálou oken jsem prováděla tak, že jsem měnila konstantu určující, kolikrát k navýšení základního okna dojde. Některé výsledky experimentů jsou uvedeny v tabulce 6.1, kde je vidět, jaký čas zabere detekce s danou škálou oken pro danou obtížnost detekce (= počet objektů v obraze). Tabulka 6.2 ukazuje, jak se měnila úspěšnost detekce s tím, jak klesala škála prohledávacích oken. Z experimentů jsem určila, že budu v projektu procházet okno se škálou osmi oken. Zatížení paměti se mezi jednotlivými testovanými hodnotami uvedenými v tabulce příliš nemění.

Čas detekce byl pro zvolenou hodnotu, tedy osm, při jednom objektu v obraze přibližně 100 milisekund a úspěšnost detekce postavy se s touto hodnotou nijak neomezila. Menší hodnota stupnice byla vyloučena z důvodu, že i při maximálním navýšení detekční okno nepokrylo celou výšku obrazu, takže objekty dosahující maximální velikosti obrazu byly detekovány jako malé a docházelo k zkreslení výsledného detekčního rámečku.

6.2 Funkčnost jednotlivých částí detektoru

Druhá sada experimentů má za úkol porovnat funkčnost finálního detektoru a jeho jednotlivých částí zvlášť - nejprve zhodnotím obrazový detektor, poté pouze hloubkový detektor, nakonec porovnám hodnoty s cílovým detektorem ComboHod. V úvahu vezmu čas detekce a úspěšnost, s jakou detektor určí objekt v obraze za běžných podmínek. Výsledky úspěšnosti detektorů ukáži na ROC křivkách.

Po prvním natrénování klasifikátoru, který je založený na hloubkových datech, nebyla detekce úspěšná dle očekávání. Klasifikátor pracoval hůře (zhruba 60%). Ačkoliv je SVM klasifikátor do určité míry tolerantní k natočení objektu, zhodnotila jsem, že je možné, že za zhoršenou funkčnost může to, že sada od pana Spinella [1][2] je o 90° otočena doprava. Proto jsem zkusila natrénovat klasifikátor pouze na své nepočetné anotované sadě. Klasifikátor fungoval přibližně na 76%, této úspěšnosti bylo dosaženo na mé testovací anotované sadě obrázků (klasifikátor nepracoval lépe z toho důvodu, že nebyl natrénován na rozsáhlejší sadě dat). Proto jsem na začátek aplikace přidala funkce na otočení obrázků od pana Spinella a na výpočet nových anotačních souřadnic. Po úpravě datové sady se úspěšnost klasifikátoru zlepšila.

Předtrénovaný klasifikátor z knihovny OpenCV, který jsem nijak neupravovala, dosáhl na mé testovací sadě úspěšnosti 81%. Mnou natrénovaný klasifikátor fungoval na testovací sadě na 90%. Výsledné ROC křivky obou detektorů jsou znázorněny na obrázku 6.1.

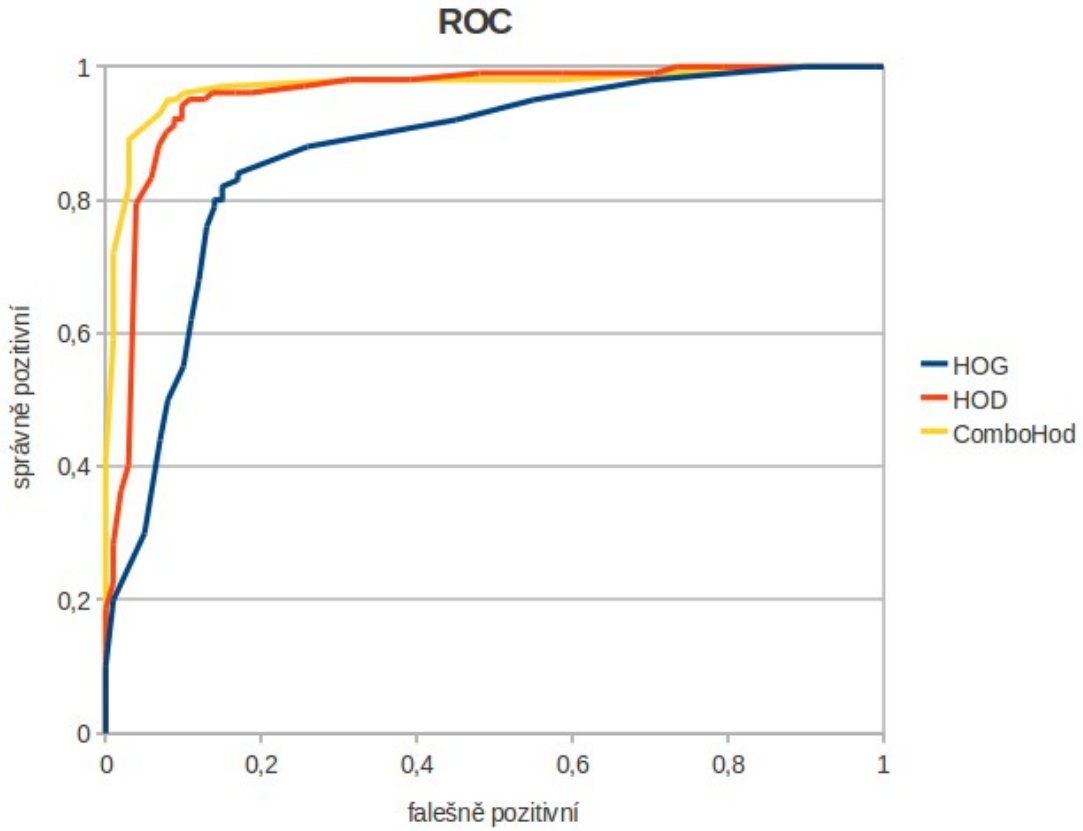
	Detektory a čas jejich detekce [ms]		
počet lidí v obraze	HOG	HOD	ComboHod
0	850	74	942
1	860	103	978
2	870	121	1036
3	890	167	1075

Tabulka 6.3: Průměrné časy[5] detekce objektu jednotlivých detektorů

5 Průměrný čas detekce je spočítán aritmetickým průměrem z jednotlivých časů 5 pozorovaných detekcí.

ROC křivka výsledného detektoru ComboHod je vidět také na obrázku 6.1. Detektor dosáhl na testovací sadě úspěšnosti 92,5%. Práh pro porovnání úspěchu detekce (EER) byl určen na číslo 0,05.

Výsledný detektor nemá o moc lepší úspěšnost, než jeho jednotlivé části. To je nejspíš dáno tím, že je testovací sada vytvořena dle toho, jaký mám předpoklad ideálního prostředí použití. Pro zamýšlený účel je vhodný detektor zaměřený na menší vzdálenost, na což je specializován HOD detektor. HOG detekce je přínosná pokud se člověk od kamery vzdálí. Spojením těchto dvou detektorů vznikl dobrý detektor na vzdálenost do přibližně deseti metrů.



Obrázek 6.1: Graf ROC křivek pro jednotlivé detektory

HOG detekce má občas problémy s rozpoznáním vzoru, pokud je nepříznivě natočen ke kameře. Tím vzniká nesouvislá detekce objektu. Naproti tomu HOD detektor snáší natočení vzoru vůči kameře poměrně dobře, díky čemuž je i výsledný detektor ComboHod vůči natočení objektu poměrně odolný (do maximální vzdálenosti, kdy ještě získá kinect kvalitní hloubková data).

Čas detekce HOG detektoru se s přibývajícími objekty v obraze příliš nenavyšuje, zatímco HOD detekce se s každým objektem navíc navýší zhruba o třetinu. Proto i výsledný ComboHod detektor je závislý na počtu vzorů v obraze. Výsledný detektor zpracuje průměrně jeden frame za vteřinu. Časy detektorů jsou vidět v tabulce 6.3.

Při menší vzdálenosti objektu od kamery je HOG detektor průměrně funkční a detekce trvá poměrně dlouhou dobu (při samotné detekci HOG se zpracuje přibližně pět obrazů za čtyři vteřiny). HOD detektor je na menší vzdálenost lépe funkční než HOG detektor, ale čas jeho detekce je závislý na počtu objektů v obraze a hůře detekuje objekty na krajích scény. S větší hloubkou obrazu přestává HOD detektor fungovat. Výsledný detektor ComboHod slučuje výhody obou detektorů za cenu delšího času stráveného detekcí. Přebírá také chyby, které mají oba detektory stejné - a to je, že pokud je pozadí hodně členité a různobarevné, dochází k označení obrazu jako pozitivního, i když se v dané části obrazu hledaný objekt nenachází.

6.3 Nepříznivé podmínky při detekci

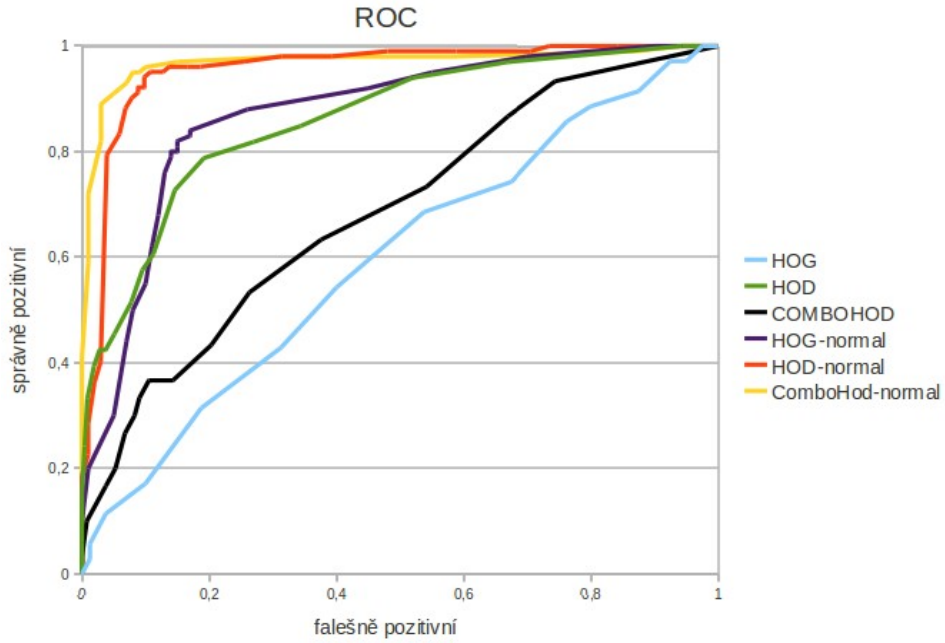
Třetí sada experimentů spočívá v tom, že budu měnit podmínky pro detektor. Program nechám ideálně nastavený podle výsledků z předchozích experimentů. Pro tyto testy použiji sadu vlastních testovacích videí (viz kapitola 4.3). Sady videí se liší v míře osvětlení prostoru (zářivka, ostré sluneční světlo, šero, úplná tma) a ve vzdálenosti, do které je Kinect zaměřen. Detektor budu hodnotit ROC křivkami v porovnání s ideálním stavem (zářivkové světlo). Podle chování detektoru za zmíněných podmínek zhodnotím rozsah použitelnosti vytvořeného detektoru.

6.3.1 Přímé sluneční světlo

Sluneční záření neprospívá ani jednomu typu kamery. Pokud slunce svítí na RGB kameru, dochází k přesvětlení částí obrazu, odkud záření vychází. Zbylé části obrazu ztrácí své vlastnosti (barvy jsou zašedlé, jsou špatně rozpoznatelné textury) a může docházek k rozostření kontur objektů.

Při dopadu přímého záření na Kinect fungují hloubková čidla velmi špatně. V místech, odkud záření vychází, se vytvoří kruhovitá černá skvrna. To je zapříčiněno tím, že infračervená část slunečního záření zkreslí záření vyslané IR emitorem Kinectu. Hloubku postranních částí obrazu (těch, které jsou nějak v zákrytu nebo daleko od zdroje záření) Kinect zaznamená, avšak skvrna od slunce je zpravidla přes většinu framu. Pokud je scéna jen extrémně osvětlena sluncem, avšak přímý zdroj záření je zakryt, černá skvrna je zhruba přes polovinu framu a možnost úspěchu detekce se tím zlepšuje. Výslednou funkcionalitu detektoru a jeho částí demonstruje graf na obrázku 6.2.

Celková funkčnost se nejvíce zhoršila HOG detektoru a to z důvodu, že v tomto prostředí splývají kontury objektů z okolím (vzor je správně zaznamenán jen zřídka). Navíc při tomto nasvícení prostoru vzniká spousta falešných detekcí, které se podepisují i na výsledné ROC křivce.



Obrázek 6.2: ROC křivky detektorů za přímého slunečního záření
(k porovnání jsou uvedeny i křivky za normálních podmínek)

Funkčnost HOD detektoru se sice snížila, ale nedosáhla tak špatných výsledků, jak bylo očekáváno. To je zapříčiněno tím, že části obrazů osvětlené sluncem neobsahují žádnou hodnotu. Tím se znemožní pozitivní nález ale také falešná detekce. Objekty jsou detekovány jen po stranách obrazu, velice záleží na míře osvětlení prostoru. Dochází zde k mylným detekcím v případech, kdy je černá

skvrna od slunce přes většinu obrazu, ale nedosahuje například až na spodní okraj, takže jsou vidět nohy člověka, který projde před kamerou.

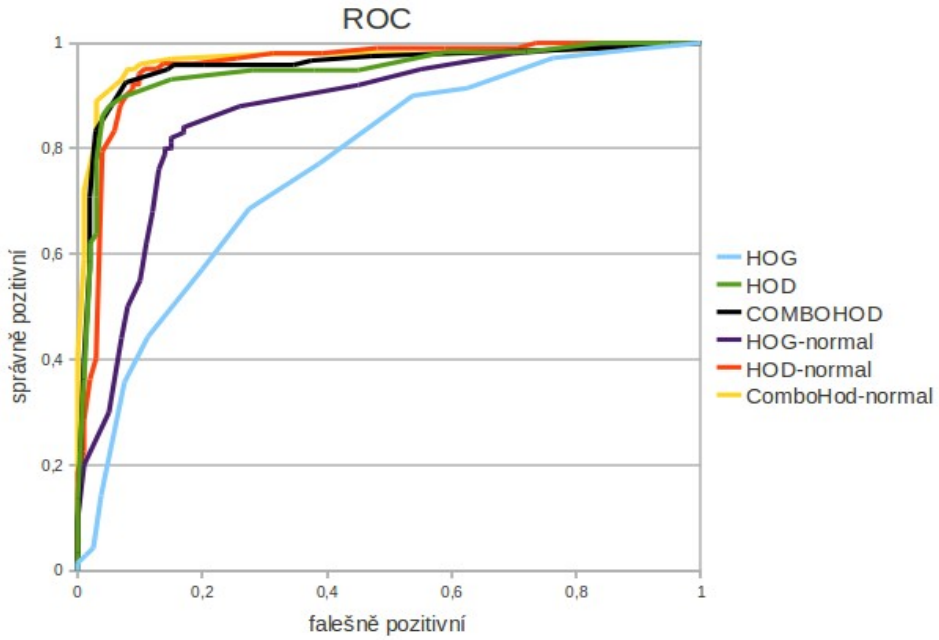
Jak je zřejmé z obrázku 6.2, při špatné činnosti částí detektoru ComboHodu ani on nedává uspokojivé výsledky. Detektor ComboHod zde trpí především absencí pozitivních hodnot od HOD detektoru. Z toho při chybějících datech přichází větší záporné hodnoty, takže po dosazení do výpočetní rovnice ComboHodu se vyruší s pozitivní vahou hodnot z HOG detektoru. To způsobuje časté nedetekování postavy ale také to znatelně eliminuje falešné detekce přicházející z HOGU.

6.3.2 Šero

Při zhoršeném osvětlení scény trpí především obraz z RGB kamery. Na hloubková čidla tato změna běžné viditelnosti nemá znatelný vliv, jelikož nejsou závislá na intenzitě osvětlení scény (pokud není extrémní, jak je popsáno v předchozí kapitole). Při špatném osvětlení scény dochází k částečnému zašumění barev. Čím dál je sledovaný objekt od kamery, tím hůře je rozeznatelná jeho barva a textura. Pozadí je zašedlé - pokud je objekt tmavých barev, může splývat s okolím. Výhodu v tomto prostředí pro RGB obraz mají objekty jasných zářivých barev (i po zašumění jsou dobře rozpoznatelné).

Hloubkový detektor HOD funguje při těchto podmínkách srovnatelně s účinností za optimálních podmínek. Snížená běžná viditelnost se na získávání obrazových dat nijak neprojevila.

Obrazový detektor HOG má zhoršenou úspěšnost zhruba o 15%. Toto zhoršení je způsobeno tím, že jsou častěji hledané objekty klasifikovány jako falešně negativní (z důvodu splývání s okolím). Při těchto podmínkách se snížil i počet falešně pozitivních detekcí. Pokud je vzor dál od kamery (více jak cca 5 metrů), je pravděpodobné že bude klasifikován jako falešně negativní.



Obrázek 6.3: ROC křivky detektorů použitých za špatných osvětlovacích podmínek

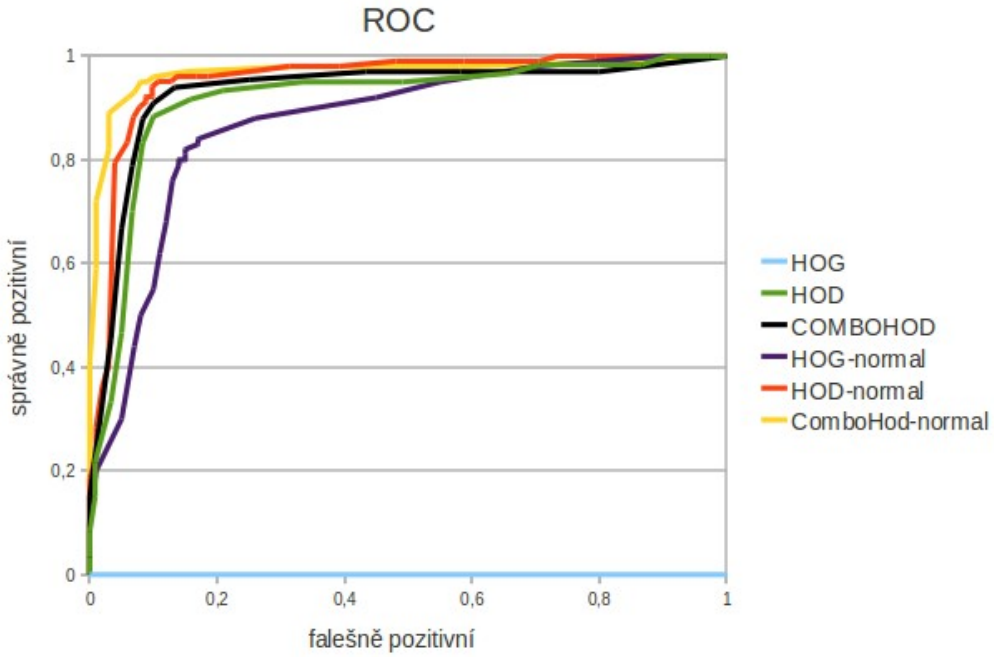
Úspěšnost detektoru ComboHod je jen lehce snížená než za běžných podmínek a to díky tomu, že detektor HOD funguje srovnatelně s účinností za běžných podmínek (viz Obr. 6.3). Toto jen nepatrné zhoršení ComboHodu je způsobeno HOG detektorem. Ten má sice sám o sobě středně zhoršenou funkčnost, nicméně jeho ohodnocovací výsledky nejsou nijak extrémní. Většina vah obrazu se pohybuje kolem zvoleného prahu (rozpětí hodnot je $\pm 0{,}3$), takže ve slučovací funkci ComboHodu tyto hodnoty nepřehluší jednoznačné výsledky z metody HOD.

6.3.3 Úplná tma

Pokud scéna není vůbec prozářena viditelným světlem, HOG detektor logicky nefunguje vůbec. V tom případě funkčnost výsledného detektoru ComboHod spočívá čistě na HOD detekci. Hloubkový detektor je založen na infračerveném světle, takže mu úplná absence viditelného záření nijak nevadí.

Detektor HOD funguje za těchto podmínek srovnatelně s účinností za běžných podmínek. Při zaměření zařízení do větší vzdálenosti se detekce zhoršuje se vzrůstající vzdáleností objektu od IR emitoru (při větší hloubce obrazu chybí hodnoty pozadí a častěji vznikají obrazové artefakty na zaznamenaných objektech).

Detektor ComboHod pracuje jen o málo hůře než za běžných podmínek. Je to dáno absencí ohodnocení obrazu od HOG detektoru ve výpočetní rovnici. Při absenci těchto hodnot (ohodnocení z HOG detektoru) se vždy bude ohodnocení od HOD detektoru o polovinu snižovat, takže se tím krátí hloubka, do které je detektor schopný zaznamenat objekt a také se tím zhorší detekce objektu, pokud není celým obsahem v obraze. Má to ale také pozitivní vliv a to ten, že se výrazně eliminují falešně pozitivní detekce z HOD detektoru. Výsledné ROC křivky jsou znázorněny na obrázku 6.4.



Obrázek 6.3: ROC křivky detektorů při úplné tmě

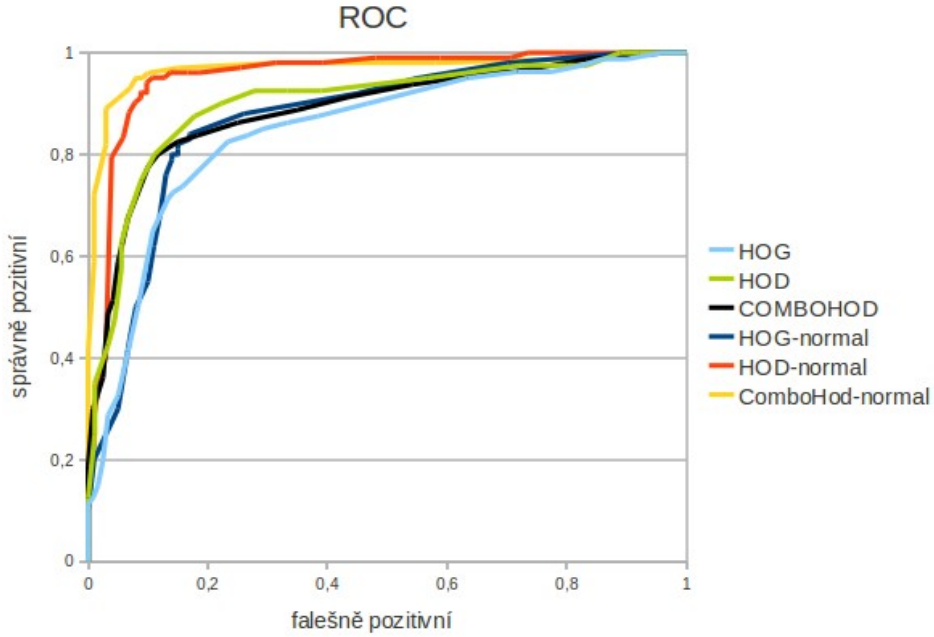
6.3.4 Velká hloubka obrazu

Vzdálenost, do které je detektor ComboHod schopný zaznamenat objekty, je spjata především s HOG detektorem. Ten ovšem také ovlivňují další faktory, jako například osvětlení vzdáleného prostoru a barva objektů, které se v něm pohybují.

HOD detektor funguje nejlépe na omezenou vzdálenost, pokud je zaměřen do větší dálky nepodává tak dobré výsledky. Data o vzdálené prostoru často chybí, takže i když se v něm pohybuje objekt, hloubkové čidlo ho nedokáže zaznamenat. Objekty, které se pohybují více jak dvanáct metrů od emitoru IR částic, nelze ve většině případů znamenat[6]. Po přiblížení postavy na zhruba deset se úspěšnost detekce zlepšuje s každým posunem vzoru směrem ke kameře. Při těchto podmínkách

6 Pokud má objekt povrch, který dobře odráží infračervený paprsek, lze jeho přítomnost zachytit. Ovšem člověk většinou nebývá pokryt od hlavy až k patě podobnou látkou (či materiálem). Program tedy dostane informaci pouze o určité části těla, pokryté daným materiálem, a to nestačí k tomu, aby byl objekt identifikován jako člověk.

pracuje detektor přibližně o deset procent hůře. Toto zhoršení je způsobeno mírným navýšením počtu falešně pozitivních obrazů a tím, že postavy na hraně dohlednosti bývají označeny za falešně negativní (objekty mívají nepřesné kontury a vyskytuje se na nich více artefaktů, než na bližších objektech).



Obrázek 5.4: ROC křivky detektorů při zaměření do větší vzdálenosti

HOG detektor pracuje při zaměření do větší vzdálenosti jen zhruba o 5% hůře než za běžných podmínek. Toto snížení účinnosti je způsobeno tím, že objekty v dálce zhruba 20 metrů jsou hodně závislé na osvětlení prostoru, který je obklopuje. Pokud jsou objekty ještě vzdálenější, často jsou označeny jako falešně negativní. Při těchto podmínkách se nezvýšil počet falešně pozitivních detekcí, ale zvýšil se počet falešně negativních objektů (především tedy těch velice vzdálených).

Detektor ComboHod funguje při těchto podmínkách na 81%. Je ovlivňován především hodnotami z HOD detektoru, který vzdálené objekty nezachytí. Proto se místo ohodnocení z HOD detektoru dosazuje do výpočetní rovnice nula, pokud ohodnocení obrazu chybí (jestliže ohodnocení obrazu existuje, nabývá pak záporné hodnoty, což má ještě větší snižující efekt na hodnotu z HOG detektoru). Výsledný detektor získal stejnou míru falešně pozitivních obrazů jako HOD detektor. Jako HOG detektor se poté chová při zpracování obrazu se vzdáleným objektem. V tomto obraze nebývají falešně pozitivní detekce, spíše úspěšnost trpí falešně negativními označeními vzorů.

6.4 Zhodnocení aplikace

Při implementaci jsem se zaměřila především na detektor lidských postav. Identifikace pohybujících se objektů je zde řešena okrajově. Pokud se trajektorie pohybujících se objektů protínají, dochází k chybové identifikaci objektů (přičte se osoba navíc). Vyřešení této chyby vyžaduje implementaci algoritmů na sledování pohybujících se objektů. Toto je ale již nad rámec mé bakalářské práce, je možné to brát jako perspektivu dalšího vývoje projektu.

Program v současném stavu zpracuje přibližně jeden obraz za vteřinu, což je poměrně nedostatečné pro reálné použití v praxi. Tato časová náročnost je způsobena především pomalým algoritmem HOG detekce, která na jednom framu trvá více jak půl vteřiny. Urychlení by bylo možné s pomocnými algoritmy nebo s akcelerací na grafické kartě.

Pokud by aplikace měla zaznamenat i velký počet kolemjdoucích lidí, bylo by třeba přidat navíc jedno snímací zařízení (postačila by obrazová kamera), aby bylo možné zachytit i lidi jdoucí v úplném zákrytu.

7 Závěr

Tématem bakalářské práce byla detekce postavy s využitím hloubkových dat. V rámci práce jsem vytvořila detektor lidských postav založený na metodě ComboHod, která spojuje výstupy dvou jednotlivých detektorů - HOG detektoru a HOD detektoru. První detektor (HOG) pracuje s obrazovými daty, druhý (HOD) zpracovává hloubkové informace obrazu. Hloubkový detektor jsem vytvořila za pomoci datové sady získané na internetu. Pro svou práci jsem využila zařízení Microsoft Kinect, jež mi poskytovalo oba typy vstupních dat pro detektor lidských postav. Díky tomuto zařízení jsem vytvořila vlastní testovací sadu videí, určenou pro detektor postav. K prezentaci funkčnosti detektoru jsem vytvořila aplikaci, která na základě detekce lidské postavy získává statistické údaje o sledovaném prostoru. Získaná statistická data program ukládá do souboru. Uživatel si může získaná data jednoduše zobrazit ve formě grafů. Komunikaci uživatele s programem zajišťuje jednoduché grafické uživatelské rozhraní.

V praxi je program určen k vytvoření představy majitele obchodu o tom, jaké množství lidí prochází kolem, v jakou denní dobu, případně u jaké části výlohy se zastaví. Tyto informace jsou užitečné například k úpravě reklam ve výlohách.

Aplikace je určena pro pozorování pohybu lidí před výlohou obchodu. Primární podmínky pro zařízení Kinect jsou prostory uvnitř budov. V těchto podmínkách dosahoval detektor postav nejlepších výsledků. Proto je aplikace určena především do obchodů v obchodních centrech. V experimentech jsem testovala použití detekční části aplikace za různých podmínek. Pokud by byl obchod umístěn například na náměstí, měl by sledovaný prostor větší hloubku a také nestálé osvětlovací podmínky. Z experimentů jsem zjistila, že na přímém slunci detektor nepřináší uspokojivé výsledky. Detekce z barevné části obrazu je zde snižována špatnými ohodnoceními z detekce z hloubkových dat a také vypočtenou konstantou poměru. Nedomnívám se však, že by bylo užitečné tuto konstantu měnit, neboť i samotná detekce z obrazových dat pracuje za těchto podmínek velice špatně.

Pokud se zařízení nachází v absolutní tmě, funguje detektor zhruba o 30% hůře než v optimálním prostředí. Zde závisí úspěšnost detekce na hloubce sledovaného prostoru. Se zvětšujícím se prostorem se úspěšnost snižuje. Za šera pracuje detektor srovnatelně jako za běžných podmínek díky nezměněné funkčnosti hloubkového detektoru. Při větším hloubkovém zaměření se funkčnost výsledného detektoru jen lehce zhorší. V těchto podmínkách se nejlépe projevuje, jak se dané dva detektory doplňují. Pokud by bylo snímací zařízení umístěno v prostředí bez možnosti dopadu přímého slunečního světla, mohlo by být užíváno i pro zaměření do venkovních prostor. Úspěšnost detekce člověka metodou ComboHod neklesla pod 80% vyjma experimentů s přímým slunečním světlem.

Jako další vylepšení programu vidím implementaci algoritmů na zrychlení HOG detektoru. Dále by bylo vhodné snížit paměťovou náročnost aplikace. Následně bych navrhovala upravit implementaci tak, aby bylo možné pracovat s další kamerou, díky které by aplikace zaznamenala i velký počet kolemjdoucích lidí, kteří se překrývají.

Literatura

[1] Spinello, L., Arras, K.: People Detection in RGB-D Data. Int. Conf. on Intelligent Robots and Systems (IROS), Freiburg, Germany, p. 3838-3843, September 2011 [cit. 12.2.2013], ISBN 978-1-61284-454-1. Dostupné z: http://www.informatik.uni-freiburg.de/~spinello/spinelloIROS11.pdf

[2] Luber, M., Spinello, L., Arras, K.: People Tracking in RGB-D Data With On-line Boosted Target Models. IEEE Int. Conf. on Intelligent Robots and Systems (IROS), Freiburg, Germany, p. 3844-3849, September 2011 [cit. 12.2.2013], ISBN 978-1-61284-454-1. Dostupné z: http://www.informatik.uni-freiburg.de/~spinello/luberIROS11.pdf

[3] Burges, Ch.: A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, Hingham, MA, USA, p. 121-167, 1998 [cit. 1.4.2013]. Dostupné z: http://www.svms.org/tutorials/Burges1998.pdf

[4] Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. Conference on Computer Vision and Pattern Recognition, Montbonnot, France, Vol. 1, p. 886-893, June 2005 [cit. 24.4.2013], ISBN 0-7695-2372-2. Dostupné z: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=1467360

[5] Lampert, C., Blaschko, M., Hofmann, T.: Beyond Sliding Windows: Object Localization by Efficient Subwindow Search. Conference on Computer Vision and Pattern Recognition, Tubingen, Germany, p. 1-8, June 2008 [cit. 24.4.2013], ISBN 978-1-4244-2242-5. Dostupné z: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4587586&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D4587586

[6] Wikipedia: Kinect [online]. 2013 [cit. 12.2.2013]. Dostupné z: http://en.wikipedia.org/wiki/Kinect

[7] Langdon, W.: Receiver Operating Characteristics (ROC) [online]. 10. května 2011 [cit. 26.3.2013]. Dostupné z: http://www0.cs.ucl.ac.uk/staff/ucacbbl/roc/

[8] Viola, P., Jones, M.: Rapid Object Detection using a Boosted Cascade of Simple Features. Conference on Computer Vision and Pattern Recognition (CVPR), Kauai, Hawai, USA, Vol. 1, p. 511-518, December 2001 [cit. 21.4.2013], ISBN 0-7695-1272-0. Dostupné z: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=990517&contentType=Conference+Publications&searchField%3DSearch_All%26queryText%3DRapid+Object+Detection+using+a+Boosted+Cascade+of+Simple+Features

[9] Bradski, G., Kaebler, A.: Learning OpenCV: Computer Vision with the OpenCV Library. Sebastopol, O'Reilly Media, Inc., 2008 [cit. 12.2.2013], ISBN 978-0-596-51613-0.

[10] Lienhart, R., Maydt, J.: An Extended Set of Haar–like Features for Rapid Object Detection. International Conference on Image Processing, Santa Clara, CA, USA, Vol. 1, p. 900-903, September 2002 [cit. 24.4.2013], ISBN 0-7803-7622-6. Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.86.9433

[11] Smíšek, J., Jančošek, M., Pajdla, T.: 3D with Kinect. International Conference on Computer Vision Workshops (ICCV Workshops), Prague, Czech Repulic, IEEE Computer Society Press, p. 1154-1160, November 2011 [cit. 21.3.2013], ISBN 978-1-4673-0063-6. Dostupné z: ftp://cmp.felk.cvut.cz/pub/cmp/articles/pajdla/Smisek-CDC4CV-2011.pdf

[12] Žára, J., Beneš, B., Sochor, J., Felkel, P.: Moderní počítačová grafika. Brno, Computer Press, 2004 [cit. 21.4.2013], ISBN 80-251-0454-0.

[13] Freund, Y., Schapire, R.: A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting. Proceeding of the Second European Conference on Computational Learning Theory, Springer-Verlag, London, UK, p. 23-37, September 1995 [cit. 24.4.2013], ISBN: 3-540-59119-2. Dostupné z: http://dl.acm.org/citation.cfm?id=712093#

[14] OpenCV documentation [online], 2013 [cit. 21.3.2013]. Dostupné z: http://docs.opencv.org/

[15] OpenNI documentation [online], 2013 [cit. 21.3.2013]. Dostupné z: http://www.openni.org/about/#.UViVoZBdU-R

[16] Microsoft Research [online]. 2013 [cit. 26.3.2013]. Dostupné z: http://research.microsoft.com/en-us/projects/touchless/

[17] Paisitkriangkrai, S., Shen, C., Zhang, J.: Efficiently Training A Better Visual Detector With Sparse Eigenvectors. Conference on Computer Vision nad Pattern Recognition, Sydney, NSW, Australia, p. 1129-1136, June 2009 [cit. 24.4.2013], ISBN 978-1-4244-3992-8. Dostupné z: http://cs.adelaide.edu.au/~paulp/publications/pubs/cvpr2009.pdf

[18] Svět microsoftu [online]. 2013 [cit. 12.2.2013]. Dostupné z: http://svetmicrosoftu.cz/wp-content/uploads/2012/06/xbox-360-kinect-1300191794.jpg

[19] Kinect for Windows Blog [online]. 2013 [cit. 26.3.2013]. Dostupné z: http://blogs.msdn.com/b/kinectforwindows/

Seznam příloh

Příloha 1

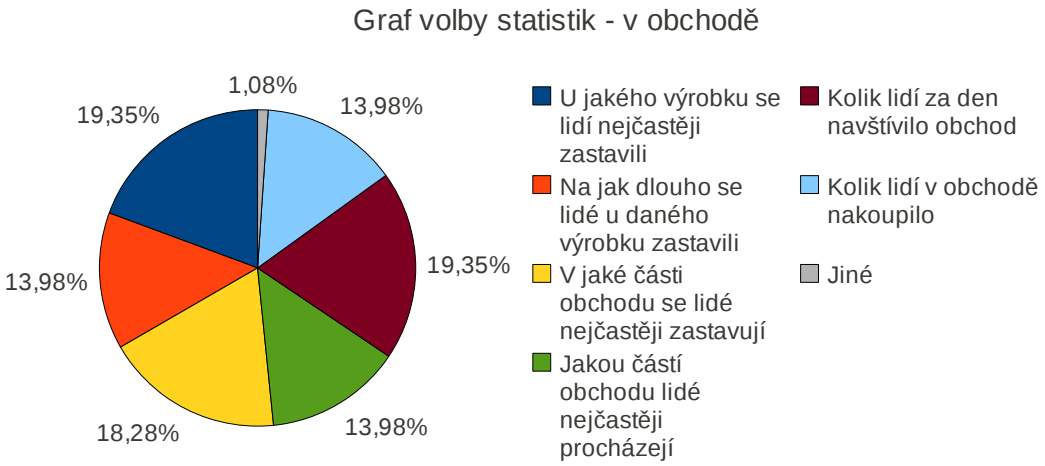
Manuál pro spouštění aplikace z konzole

Detekční i statistická část programu se spouští s konzole pomocí jednoduchých přepínačů.
Seznam přepínačů a příklady použití:

- -h, --help - zobrazení nápovědy
- -s - spuštění statistické části programu
- -r - spuštění detektoru pro získávání dat v reálném čase (nutné mít připojený Kinect), aplikace se ukončí stiskem libovolné klávesy
- -r <video.oni> - spuštění detektoru, za parametrem musí následovat mezera a jméno souboru s videem ve formátu .oni, aplikace se ukončí stiskem libovolné klávesy

Příloha 2

Vyhodnocení dotazníku



Příloha 3

Obsah DVD

Seznam souborů na přiloženém DVD:

- technická zpráva ve formátu .pdf a její zdrojový soubor
- zdrojové kódy aplikace
- testovací videa
- příklad trénovacích dat
- README - je zde popsána instalace potřebných knihoven a programů, dále je zde postup jak přeložit a spustit aplikaci
- spouštěcí skript
- plakát

Příloha 4

Plakát

Detekce lidské postavy na videu

Autor: Daniela Johanová

Vedoucí: Ing. Michal Španěl, Ph.D.

Detektor lidských postav

- zpracování hloubkové a obrazové infomace z obrazu
- metoda ComboHod - složená z:
  - HOG detekce
  - HOD detekce

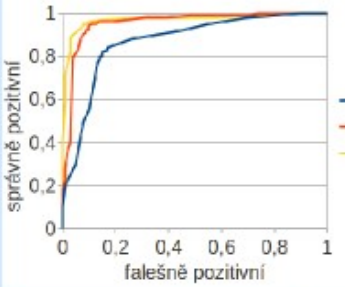
Zpracování dat

- vstupní data pro detektor - využití zařízení Microsoft Kinect
- zpracování dat a komunikace s Kinectem - knihovny OpenCV a OpenNI

Schéma metody ComboHod

- výstupy z klasifikátorů se slučují do výsledné pravděpodobnosti výskytu hledaného vzoru v obraze

Experimenty: ROC křivky HOGu, HODu a ComboHodu



- ROC křivky detektorů za normálních podmínek (prostor uvnitř budovy osvětlený zářivkou)
- úspěšnost:
  - HOG - 81%
  - HOD - 90%
  - Combohod 92%
- možné využití - statistická aplikace, která sbírá informace o pohybu osob před výlohou obchodu

Daniela Johanová, xjohan03@stud.fit.vutbr.cz, FIT VUT v Brně 2013