VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

ODHAD VÝŠKY OSOB ZAZNAMENANÝCH NA VIDEU

ESTIMATE THE HEIGHT OF A PERSONS FROM THE VIDEO DATA

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **VILÉM JELEN**
AUTHOR

VEDOUCÍ PRÁCE **Ing. TOMÁŠ GOLDMANN**
SUPERVISOR

BRNO 2017

Zadání bakalářské práce/19345/2016/xjelen09

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav inteligentních systémů Akademický rok 2016/2017

Zadání bakalářské práce

Řešitel: **Jelen Vilém**
Obor: Informační technologie
Téma: **Odhad výšky osob zaznamenaných na videu**
Estimate the Height of a Persons from the Video Data
Kategorie: Zpracování obrazu

Pokyny:

1. Seznamte se s problematikou sledování hlídaného prostoru pomocí bezpečnostních kamer. Seznamte se s algoritmy pro detekci objektů v obraze.
2. Proveďte zhodnocení algoritmů z hlediska využitelnosti pro odhadovaní výšky objektů. Na základě získaných informací vyberte jeden algoritmus, který je vhodný pro detekci osob v obraze
3. Navrhněte algoritmus, kterým lze odhadnout výšku osob v obraze. Jako vstupní parametry algoritmu použijte pozici kamery, úhel záběru, vzdálenost k jednotlivým objektům od kamery nebo referenční body.
4. Algoritmus implementujte v libovolném programovacím jazyku a vytvořte jednoduché uživatelské rozhraní s důrazem na intuitivní ovládání.
5. Proveďte a vyhodnoťte experimenty na několika videích z různých prostředí. Zaměřte se na analýzu odchylky výšky od skutečného stavu.

Literatura:

- Davies E. R.: Computer and Machine *Vision, Fourth Edition: Theory, Algorithms, Pra*cticalities. Academic Press, 2012.
- Sonka M., Hlavac V., and Boyle R. *Image processing, analysis, and machine vision*. Cengage Learning, 2014.
- Sefidgari B.L. *Human Body Detection and Safety Care System for a Flying Robot*. Computer Science, 2013.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1 až 3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Goldmann Tomáš, Ing.**, UITS FIT VUT
Datum zadání: 1. listopadu 2016
Datum odevzdání: 17. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav inteligentních systémů
612 66 Brno, Božetěchova 2

L.S.

doc. Dr. Ing. Petr Hanáček
vedoucí ústavu

Abstrakt

Cílem této práce je vytvořit aplikaci detekující osoby a provádějící odhad jejich výšky z videozáznamu. Ve stěžejní části práce jsem představil bezpečnostní kamery a metody zpracování obrazu vhodné pro detekci a klasifikaci objektů. Na základě porovnání těchto metod jsem realizoval detekci osob pomocí histogramů orientovaných gradientů. Jako vstupní parametry navrženého algoritmu odhadu výšky jsem použil pozici a vlastnosti kamery a referenční objekty nebo body. Vytvořené řešení poskytuje při dostupnosti všech parametrů odhad s průměrnou chybou $\sim 1\%$ až 15%.

Abstract

The aim of this bachelor thesis is to create an application that can detect pedestrians and make an estimate of their height from the video data. In the main part of the thesis I introduced security cameras and image processing methods suitable for detecting and classifying objects. Based on the comparison of these methods, I realized the pedestrian detection using Histograms of Oriented Gradients. As the input parameters of the proposed height estimation algorithm, I used the position and properties of the camera and reference objects or points. The solution provides an estimate with an average error of $\sim 1\%$ to 15%, when all parameters are available.

Klíčová slova

CCTV, zpracování obrazu, detekce osob, odhad výšky osob, HOG, OpenCV

Keywords

CCTV, image processing, human detection, human height estimation, HOG, OpenCV

Citace

JELEN, Vilém. *Odhad výšky osob zaznamenaných na videu.* Brno, 2017. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Tomáš Goldmann

Odhad výšky osob zaznamenaných na videu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Tomáše Goldmanna. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Vilém Jelen
17. května 2017

Poděkování

Tímto děkuji mému vedoucímu Ing. Tomáši Goldmannovi za cenné rady a vedení při řešení této práce.

Obsah

Kapitola 1

Úvod

Při vyšetřování trestného činu je výška osoby jednou z informací užitečných k dopadení pachatele. Dotazováním svědků se můžeme dozvědět přibližnou výšku osoby, ale již pravěcí lovci používali přesnější způsob odhadováním velikosti kořisti z jejích stop. Dnes máme k dispozici databáze s údaji o korelaci výšky osob s velikostí stop [12]. Realitou je však špatné zanechávání stop ve městském prostředí. Proto se na tuto možnost nelze spolehnout a je nutno hledat další přístupy.

K zajištění bezpečnosti a ušetření nákladů na zabezpečení jsou dnes běžně instalovány bezpečnostní kamery. Vhodným příkladem je Londýn, kde sledovací systémy pokrývají většinu města, proto je v dnešní době často svědkem činu kamera [14]. V tomto případě jsou již možnosti určení výšky pachatele rozsáhlejší a dosahují větší přesnosti. Příkladem používaného odhadu výšky je využití kalibrační klece. Tato klec je složena z částí (nejčastěji trubic) známé délky, proto když je postavena v místě, kde se nacházel pachatel, stačí pro získání jeho výšky spočítat odpovídající části klece. Dalším přístupem je změření velikosti objektu, který se nachází ve stejné vzdálenosti od kamery jako pachatel a následné přepočítání poměrů jejich velikosti. Nevýhodou těchto postupů je nutnost osobního měření a s tím související doprava pracovníků a jejich náčiní na místo. Pokud je potřeba změřit výšku více osob z videozáznamu, může být měření časově a finančně náročné.

Ideálním řešením tohoto problému by byl systém autonomně zobrazující v reálném čase výšku osob procházejících před bezpečnostní kamerou s chybou maximálně jednoho nebo dvou centimetrů. Využití tohoto systému nemusí být čistě zaměřeno na vyšetřování trestní činnosti, mohl by být využitelný také k statistickému sledování výšky obyvatel. Cílem této práce je vytvořit program schopný s použitím běžné bezpečnostní kamery a několika pomocných měření odhadnout výšku osoby a experimentálně ověřit přesnost odhadu. Důležité je použití běžné kamery, které jsou široce rozšířeny a bude možné program používat se současnou infrastrukturou. Cílem je udělat první krok k ideálnímu autonomnímu systému. V případě dosažení dobré přesnosti odhadu bude použitím programu možné zjednodušit práci při vyšetřování a v důsledku toho ušetřit finanční a časové prostředky.

Při tvorbě programu pracujícím s videozáznamem bezpečnostní kamery jsou užitečné znalosti o sledování bezpečnostními kamerami. V rámci první kapitoly jsou proto popsány informace o technologii bezpečnostních kamer a okolnostech jejich použití. Od různých metod sledování a historie bezpečnostních kamer až po vlastnosti kamer. Následující část této kapitoly je již přímo zaměřena na rozpoznávání objektů. V této části se dozvíte jaké metody a algoritmy se používají pro rozpoznávání objektů, především pak pro rozpoznávání osob. Z hlediska odhadu výšky je důležitá přesnost detekce lidské postavy, proto jsou algoritmy podle tohoto hlediska porovnány, zmíněny jsou zde také některé nové robustní přístupy,

například neuronové sítě. Kapitola je zakončena popisem principů metod jako je odečítání pozadí, které lze při rozpoznávání objektů využít k optimalizaci a vylepšení detekce. Následuje kapitola popisující návrh aplikace, kde je kromě jiného popsáno, proč je nutné výstup detektoru upravit, jak je lze jednoduše odhadnout výšku osoby a jak bylo navrženo uživatelské rozhraní aplikace. Čtvrtá kapitola přibližuje použité nástroje, implementaci odhadu výšky a uživatelského rozhraní, zakončena je analýzou experimentů s aplikací. Závěrem jsou zmíněny případná vylepšení aplikace a práce je celkově vyhodnocena.

Kapitola 2

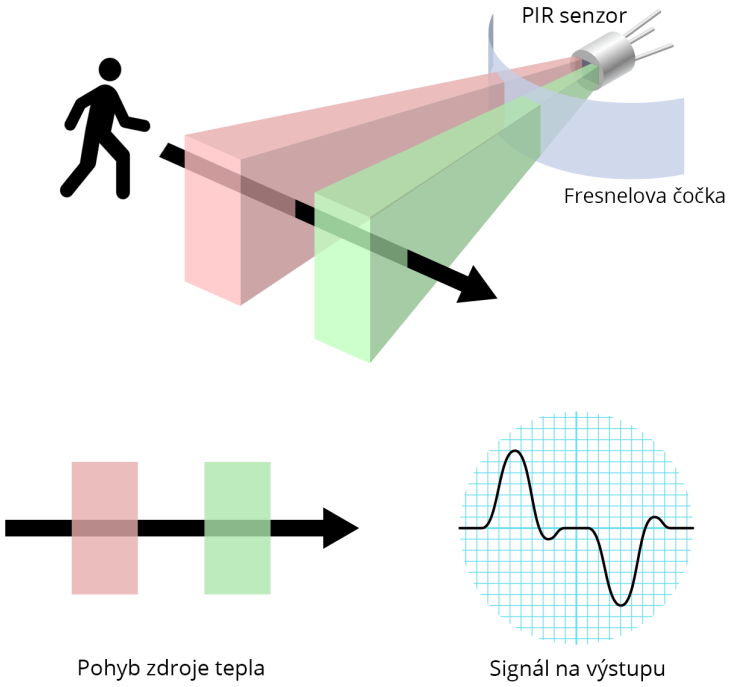
Sledování a detekce osob

Sledování osob je nejčastěji nasazováno z důvodu zajištění bezpečnosti, ať už sledovaných nebo jejich okolí. Spočívá ve sběru informací o aktivitách dané osoby ze všech dostupných zdrojů. Dříve byl počet těchto zdrojů menší než dnes a rozšířeným způsobem bylo nasazení pracovníka s úkolem špionáže. Dnes lze vybírat z velkého množství: satelitní snímky, GPS (Global Positioning System), RFID (Radio Frequency Identification), štěnice (ukryté mikrofony, kamery a jiná sledovací elektronika), implantování mikročipů, pohybové PIR senzory (pasivní infračervený senzor), sledování e-mailů, sledování pomocí dronů, biometrické údaje, analýza sociálních sítí, obecně sledování aktivity na počítačích a internetu, sledování telefonů a v neposlední řadě použití bezpečnostních kamer.

Ke sledování pomocí GPS se používá malé zařízení „*tracker*“, které nosí sledovaná osoba s sebou. Ten obsahuje modul GPS k získávání aktuálních souřadnic, následně mohou být data uložena do paměti zařízení, a běžně jsou navíc informace rovnou odesílány ke sledujícímu pomocí GSM/GPRS modemu v podobě SMS nebo IP paketů. Kromě špionáže, využití při sportování nebo monitorování kvůli ochraně, je tento typ sledování často používán pro kontrolu dodržování domácího vězení nebo jiných omezení volného pohybu, které jsou sledované osobě nařízeny zákonem.

Zajímavá je i detekce pohybu pomocí PIR (pasivní infračervený) senzoru, který reaguje na změny teploty v záběru (obr. 2.1). Senzor je postavený z pyroelektrických materiálů, ty produkují elektrickou energii při vystavení teplu (neboli infračervenému záření). Například při vstupu osoby do záběru senzoru dojde ke zvýšení napětí a potom při odchodu dojde k jeho snížení, vlastně je detekován neobvyklý puls napětí. Ke správnému zamíření IR záření na senzor je použita Fresnelova čočka, která také umožňuje rozdělit IR záření z různých částí prostoru na odpovídající části senzoru. PIR senzor reaguje vždy změnu na IR záření, nerozeznává detekovaný objekt, například nerozezná psa od člověka. Tato vlastnost je podstatnou nevýhodou při použití ke sledování osob. Nejčastěji jsou PIR senzory využívány k automatickému rozsvícení světel. Dalším použitím je detekce nežádoucího pohybu například k zabezpečení budovy [9].

V této kapitole je podrobněji popsán způsob sledování bezpečnostními kamerami a následně detekce a klasifikace objektů v obraze, zejména osob.



Obrázek 2.1: Princip funkce PIR senzoru [9].

2.1 Bezpečnostní kamery

Cílem této části je seznámit se s bezpečnostními kamerami. Od začátku používání prošly videokamery prudkým vývojem, jejich historie je stručně popsána v sekci 2.1.1. Různé typy používaných kamer jsou stručně popsány v části 2.1.2. Nové technologie přinesly lepší typy senzorů ke snímání obrazu, dva nejzajímavější jsou popsány v části 2.1.3. Následuje část zaměřená na vlastnosti a parametry kamer. Nakonec je zmíněn vliv prostředí na kvalitu záznamu je v části 2.1.4.

2.1.1 Historie

První záznamy použití CCTV (closed-circuit television) kamer popisují sledování startů raket V2 v Německu roku 1942, konkrétně použití kamer společnosti Siemens v Peenemünde. Ke konci druhé světové války používali Američané podobné kamery pro sledování testů atomových bomb. Sledování veřejnosti má počátek v 60. letech 20. století, kdy se začínaly používat trubicové kamery [13]. V tomto období byly nainstalovány první bezpečnostní kamery v Anglii na Trafalgar Square k zajištění bezpečnosti královské rodiny při jejich návštěvě Londýna. O rok později byly Londýnským dopravcem pro zlepšení bezpečnosti přidány kamery kolem železniční sítě. Během 60. let byla většina kamer vlastněná vládou a používaná pro policejní sledování, až ke konci desetiletí byly kamery nasazovány soukromě (například ke hlídání obchodu), zejména díky nižší ceně a zvyšující se spolehlivosti. První domácí sytém zabezpečení s kamerou byl uveden Marií Van Britten Brown roku 1969. Rané verze tohoto systému se skládaly z čtyřech kukátek ve dveřích, mezi kterými se pohybovala kamera. Záběry byly zobrazeny na televizi a navíc bylo možné komunikovat s osobou před dveřmi pomocí mikrofonů. Zámek dveří byl také ovládán dálkově.

V 70. letech byl díky počátku používání videorekordéru nahrazen živý přenos obrazu sledováním záznamů na videokazetách. Roku 1974 bylo CCTV poprvé použito k monitorování dopravy na hlavních silnicích z a do Londýna. O rok později byly kamery nasazeny na sledování davu při fotbalových zápasech. V této době poskytovaly bezpečnostní kamery stále velmi zrnitý černobílý obraz, byly pořád drahé a při slabém osvětlení byl zachycený obraz často nepoužitelný. Další nevýhody spočívaly ve videokazetách, bylo nutné je pravidelně vyměňovat nebo přepisovat starý záznam novým, jejich skladování bylo náročné a občas byl záznam zničen opotřebením kazety.

Podstatným pokrokem v technologii bylo vynalezení *solid-state* videokamery (neobsahovala pohyblivé prvky) v 80. letech [13]. Během 90. let 20. století video technologie postupně přešly od analogové k digitální technologii zobrazování. Na začátku 90. let byla většina nově instalovaných kamer již typu *solid-state* a používala CCD (Charge Coupled Device) senzor, ten vyřešil problém se zachycením obrazu při nízkém osvětlení. Nové typy senzorů byly kromě CCD také MOS (Metal Oxide Semiconductor) a CMOS (Complimentary MOS). Rozdíly mezi těmito senzory jsou popsány v sekci 2.1.3. Uvedení multiplexování umožnilo zobrazení výstupu z několika kamer na jednom displeji, time-lapse videa a záznam videa pouze při detekci pohybu. Díky použití menšího množství videokazet byl zmírněn problém s jejich skladováním a nízkou kapacitou záznamu.

Mezi prvními cenově dostupnými kamerami byly chůva kamery v roce 1992, ty jsou navržené ke sledování malých dětí nebo jejich chův. Po teroristickém útoku na World Trade Center v Americe roku 1993 byly CCTV kamery instalovány na veřejných prostranstvích, považovaných za ohrožené dalším útokem. Na konci desetiletí proběhla druhá vlna popularity CCTV kamer, důvodem byla nová digitální technologie. S kamerou používající digitální záznam (DVR) bylo možné nahrát přibližně měsíc dlouhý záznam a uložit jej na pevném disku, navíc bylo možné video upravovat a tím usnadnit identifikaci osob.

Po teroristických útocích roku 2001 se kamery staly nedílnou součástí zabezpečení. V předešlých desetiletích byly CCTV kamery používány převážně v 60. letech k sledování dopravy, kontrolování davu a dodržování zákonů v 70. a 80. letech, a ke komerčnímu použití v 90. letech 20. století. V posledním desetiletí se zlepšováním technologií mají kamery běžně integrovaný software pro rozpoznávání obličejů a další automatizované funkce. S růstem Internetu jsou běžně používány IP (Internet protokol) kamery, které přenášejí data přes Internet a díky tomu lze k nahrávanému obrazu přistupovat i z chytrých mobilů a dalších přenosných zařízení. [23][13]

2.1.2 Typy kamer

Podle způsobu zpracování obrazu se kamery dělí na analogové, digitální a digitálně-analogové. Analogové kamery byly první používané s jednodušší elektronikou a stále obvyklým využitím. Tyto kamery neposkytují žádné přídavné funkce, pouze posílají obraz k zobrazení na monitoru nebo do DVR (Digital Video Recorder). Snímková frekvence je obvykle 24 nebo 25 FPS. Nevýhodou analogových kamer je ztráta kvality obrazu při převodu mezi analogovým a digitálním signálem. Jejich hlavní výhodou byla v minulosti větší citlivost na intenzitu světla, protože používají CCD senzor (viz. sekce 2.1.3). Nejnovější pokroky v CMOS technologii však umožňují dosahovat podobné citlivosti i u digitálních kamer, které používají CMOS senzory (viz. 2.1.3).

Digitální kamery se dnes používají převážně ve formě IP (Internet Protocol) kamery. Tento typ kamery přenáší informace přes síť (intranet nebo internet) a proto má také vlastní IP adresu. Díky technologii PoE (Power over Ethernet) mohou být tyto kamery napájeny

přes datový kabel. Mají také vyšší výpočetní výkon, proto často poskytují přídavné funkce, například detekci pohybu v obraze nebo inteligentní analýzu obrazu. Jejich snímková frekvence je také vyšší než u analogových kamer, běžně se pohybuje od 6 do 60 FPS. Velkou výhodou IP kamer je možnost sledovat záběr kamery například z mobilního telefonu. Tyto kamery poskytují vyšší rozlišení (běžně 1.3 nebo 2Mpx) než analogové (0.4 Mpx).

Digitálně-analogové kamery jsou kombinací obou přístupů. Díky tomu je možné ukládat obrazovou informaci digitálně a zároveň využít výhod analogového senzoru, například jeho vyšší citlivosti. Zachycení obrazu je zde rozděleno do dvou částí: analogovou je senzor kamery a digitální částí je její operační paměť. Záznam probíhá periodickým ukládáním informace ze senzoru do paměti pro jednotlivé pixely. Ke vzorkování je použit převodník analogového signálu na digitální. Získané data jsou zpracovány do jediného obrázku. Pro tyto kamery je preferovaný CMOS senzor, zejména kvůli menším rozměrům [2].

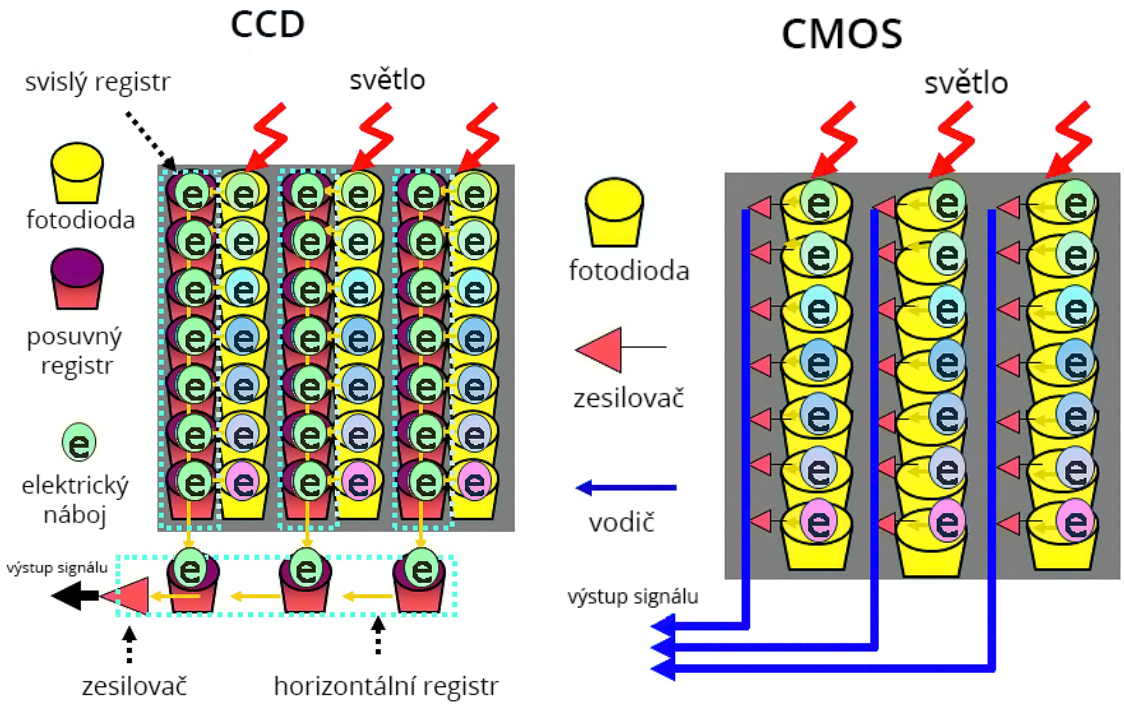
Z konstrukčního hlediska jsou nejpoužívanější standardní kamery ve tvaru hranolu s vystupujícím objektivem. Ten bývá často výměnný. Pro jejich nenápadný vzhled jsou používány dome kamery, které mají kopulovitý tvar. Běžně jsou umísťovány na stropy nebo stěny, například na strop tramvaje. Dalším typem jsou miniaturní kamery pro skryté sledování. Termokamery snímají záření v infračerveném spektru a jejich využití je například pro hledaní úniků tepla nebo sledování osob za velmi špatných světelných podmínek. [13]

Obrázek 2.2: Ukázka některých používaných typů kamer [13].

2.1.3 Snímače obrazu

Senzory ve videokamerách se dělí do dvou skupin podle fyzikální podstaty způsobu zachycení informace z fotonů. První jsou založeny na foto-emisním principu, druhé na fotovoltaice. Fotovoltaika je založena na využití energie fotonu, který nárazem do elektronu jej přesune z valenčního pásu v atomu do pásu vodivosti, takový elektron se označuje jako excitovaný a je zdrojem elektrického napětí v polovodiči. Velikost vznikajícího elektrického proudu je přímo úměrná počtu dopadajících fotonů [20]. Technologie CCD a CMOS patří do druhé skupiny (fungující na principech fotovoltaiky). Tyto senzory se široce rozšířily postupně s vývojem polovodičů a dnes je většina kamer vybavena senzorem CCD nebo CMOS. Dříve byly k zachycení obrazu využívány také vidicon trubice, které jsou založeny na foto-emisním principu.

Základní rozdíl ve funkci těchto technologií je v umístění AD převodníku (analogový signál na digitální). Nabití pixelu u CCD je přesouváno k jednomu výstupnímu bodu, kde je konvertováno na napětí a výstupem čipu je souvislý analogový signál, který je potom dle potřeby převeden na digitální. V CMOS senzoru je konverze na digitální signál prováděna pro každý pixel zvlášť (každý má vlastní AD převodník) a běžnou součástí senzoru jsou také zesilovače a obvody pro odstranění šumu [20]. Komplikovanější návrh pole pixelů omezuje přístup světla a proto CMOS čipy vyžadují lepší světelné podmínky než CCD. Na obrázku 2.3 níže jsou zobrazeny části těchto čipů v řetězci za sebou: po průchodu čočkou je světlo usměrněno do snímače čipu, tomu předchází filtr barev, který na pixel propustí pouze světlo o frekvenci, které má snímat. V CCD je vznikající napětí na fotodiodě posouváno ze všech pixelů k jednomu výstupu, kde je zesíleno. U CMOS má každý pixel vlastní výstup a napětí je zesilováno přímo u fotodiody. Výstup pole pixelů na CMOS čipu je proto v bitech.



Obrázek 2.3: Ilustrace funkce CCD a CMOS senzorů [19].

Vzhledem k rozdílné konstrukci čipů se obě technologie liší vlastnostmi. Jejich vývoj započal na konci 60. a začátku 70. let dvacátého století, kvůli náročnější technologii výroby pole pixelů pro CMOS, byl tento čip s kvalitou obrazu oproti CCD pozadu. S pokrokem v technologii CMOS je dnes kvalita obrazu srovnatelná. Cena výroby CMOS čipů je většinou větší než u CCD. Komplexnější senzor CMOS má za následek větší šum oproti CCD. Výhodou CMOS je nižší spotřeba a menší velikost čipu. [15]

2.1.4 Vlastnosti kamer

Podstatnou částí kamery je její čočka inspirována funkcí čočky v lidském oku, koncentruje odražené světlo ze scény do senzoru kamery. Obecně platí, že čím větší je průměr čočky, tím více světla se dostane na senzor kamery a vzniká kvalitnější obraz. Většina kamer používá čočky s pevnou ohniskovou vzdáleností (Fixed Focal Length), které mají konstantní

úhel zorného pole [13]. Kromě FFL čočky lze použít přibližovací čočky s nastavitelnou ohniskovou vzdáleností. Speciální je například panoramatická čočka zobrazující 360° kolem kamery, výsledkem je obrázek ve tvaru kruhu. Kamera může také mít více čoček na jeden senzor, kdy každá čočka směruje světlo pouze na část senzoru a je tak možno do jednoho videozáznamu zaznamenávat více úhlů pohledu zároveň.

Bezpečnostní kamery často používají čočky s nastavitelnou clonou [13]. Clona je důležitá pro zajištění optimálního množství světla pro senzor kamery. Může být manuálně nastavitelná nebo automatická, kdy se otevírá nebo uzavírá podle intenzity světla.

Rozlišení kamery určuje počet pixelů tvořících obraz. Většinou je udáváno ve tvaru $x \times y$, kde x udává počet horizontálních pixelů a y počet vertikálních pixelů. Rozlišení je důležitým parametrem při detekci nebo identifikaci osob, vysoké rozlišení umožňuje rozpoznání na větší vzdálenost a za horších podmínek. U analogových kamer je rozlišení uváděno v počtu televizních řádků.

Senzor kamery ovlivňuje výslednou kvalitu a barvy obrazu (např. monochromatický senzor), zejména za horších světelných podmínek. Ale také velikost kamery - CCD senzory jsou větší než CMOS. Proto například u mobilních zařízení jsou používány CMOS senzory. Má také velký vliv na spotřebu kamery, více o senzorech je v sekci 2.1.3.

Citlivost kamery na světlo je měřena v jednotkách Lux. Udává minimální osvětlení při kterém kamera dokáže snímat obraz. Pro výrazné navýšení světelné citlivosti kamery je používána technika OCL (on-chip lens), při které je senzor kamery vybaven mikroskopickou čočkou pro každý pixel, čímž je světlo lépe koncentrováno na světločivá místa senzoru.

Počet snímků za sekundu (Frames Per Second, dále FPS) udává rychlost snímkování kamery. Pro lidské oko se videozáznam začíná jevit jako spojitý od 15 FPS. Běžně kamery poskytují video s 25 FPS. V některých případech stačí pro plnění úkolu i 3 FPS, nízká snímková frekvence je užitečná pro snížení množství přenášených dat, zejména při použití kamer s velmi vysokým rozlišením. [13]

Vliv osvětlení sledované scény

Snímaný obraz je kromě parametrů kamery ovlivněn i okolním prostředím, zejména jeho osvětlením. Scéna může být osvětlena přírodním, umělým zdrojem světla nebo kombinací obou. Při použití černobílých kamer nezáleží na zdroji světla, na rozdíl od barevných kamer. Ty potřebují osvětlení se všemi barvami rovnoměrně rozloženými, jinak nemá výsledný obraz dostatečnou kvalitu.

Během dne se vlastnosti přirozeného osvětlení neustále mění v závislosti na čase a počasí. Aby kamera poskytovala vhodný obraz při nestálém osvětlení, je důležité aby byla vybavena automatickou čočkou. Pro kamery v prostředí s konstantním osvětlením (například uvnitř budov) je dostačující čočka s pevnou clonou.

Jako umělé osvětlení jsou používány například tyto zdroje: tungsten, xenon, tungsten-halogen, metal-arc, rtuť, sodík, IR lampy a LED IR pole [13]. Lampy založené na tungstenu produkují světlo s nejlépe vyváženými barvami a jsou proto nejlepší pro barevné kamery. Pole infračervených LED bývají přidávána přímo na kamery nebo blízko nich a využívají se k přisvětlení u monochromatických kamer [5].

Kvalita výsledného obrazu závisí zejména na intenzitě osvětlení, kontrastu objektů vzhledem k pozadí scény, rozmanitosti pozadí scény (může způsobovat „maskování“ objektům v popředí) a pohybu objektů případně jejich rychlosti [13].

2.2 Detekce a rozpoznávání objektů

I ty nejjednodušší úkoly pro strojové vidění potřebují pomoc s rozpoznáváním objektů, proto je důležité se nejprve seznámit s vybranými metodami detekce a klasifikace objektů. K tomuto úkolu je většinou používáno rozpoznávání vzorů. Aby bylo možné je rozpoznat, je nutné nejdříve algoritmus naučit jak vypadají a jaké mají vlastnosti nebo je určitým způsobem definovat. K učení nebo definování je třeba znalostí, těmi a také jejich reprezentací se zabývá studium umělé inteligence. Strojové vidění značně využívá poznatků tohoto oboru a vzájemně se doplňují, například lepším rozpoznáváním objektů roboty. Hlavními způsoby reprezentace znalostí v AI jsou formální gramatiky, predikátová logika, pravidla výroby a sémantické sítě. Méně běžné ale stejně důležité jsou rysy (features) a deskriptory. Typicky nestačí pouze jeden deskriptor a proto jich bývá více spojováno do vektorů rysů (feature vectors). [20]

Objekt je fyzickou jednotkou, běžně reprezentovanou ve zpracování obrazu jako určitá oblast obrázku. Objekty lze rozdělit do tříd podle jejich společných vlastností, jinými slovy je klasifikovat. Rozpoznávání objektů spočívá v přiřazování tříd objektům a tuto práci vykonává klasifikátor. Jako je například SVM (Support Vector Machine, viz. 2.2.2). Klasifikaci předchází výše zmíněný popis znalostí o objektu, v této fázi je třeba vytvořit formální popis, existují různé přístupy, například počítání gradientů nebo zaměření se na texturu objektu. Tato kapitola slouží k seznámení s vybranými metodami použitými k rozpoznávání objektů. V sekci 2.2.4 jsou porovnány běžně používané metody detekce osob a také je provedeno zhodnocení jejich využití za účelem odhadu výšky osob (viz. 2.2.4).

2.2.1 Detektor Viola-Jones

Detektor byl uveden v roce 2001 Paulem Violou a Michaelem Jonesem [21]. Je zaměřený na rychlou detekci objektů s přesností podobnou ostatním dostupným metodám. Hlavní motivací byla detekce obličejů osob v reálném čase. Aby byla detekce obličeje zjednodušena a přitom nedošlo k velkému omezení praktického použití, lze obličej detekovat jen pokud je celý a otočený přímo na kameru.

Pro detekci výrazných znaků lidského obličeje jsou použity Haarovy příznaky. Důvodem je nižší náročnost výpočtu oproti jiným metodám a vzájemná podobnost obličejů. Například oblast kolem očí je tmavší než líce a můstek nosu je světlejší než oči, ukázka na obrázku 2.4.

Obrázek 2.4: Příklad dvou Haarových příznaků lidského obličeje. Oblast kolem očí je tmavší než líce (druhý zleva) a můstek nosu je světlejší než oči (vpravo).

Obrázek je reprezentován integrálním obrazem (integral image), který umožňuje vyhodnocení příznaků v konstantním čase a tím pomáhá k rychlejšímu zpracování. V této

reprezentaci je každý bod vypočítaný podle vztahu:

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y') \tag{2.1}$$

kde x, y jsou souřadnice pozice v integrálním obraze ii, jejíž hodnota je součtem pixelů umístěných nahoře, vlevo a včetně na souřadnicích x', y' obrázku i.

Protože možných příznaků na detekční okno o velikosti 24x24 pixelů je až 180 tisíc [21], bylo by neefektivní vyhodnocovat je všechny. Proto tato metoda používá k výběru nejvhodnějších příznaků algoritmus AdaBoost. Ten sestavuje lepší klasifikátory skládáním slabých příznaků nebo dalších klasifikátorů. Původním použitím algoritmu AdaBoost je zlepšení účinnosti klasifikování jednoduchým učícím se algoritmem (např. perceptronem). Ten bývá označován termínem „weak learner“ (slabý student), důvodem jsou slabší výsledky klasifikátoru před podpořením algoritmem AdaBoost. Pro zlepšení výsledků klasifikátor zpracuje sadu obrázků vybranou pro učení. Po klasifikaci jsou obrázky ohodnoceny s důrazem na chybné označení předešlým klasifikátorem. Výsledkem je silný klasifikátor, který vnikne váhovou kombinací slabých omezenou prahovou hodnotou. Slabý klasifikátor $h_j(x)$, kde x je klasifikovaná část obrázku, je definován příznakem f_j, prahem θ_j a paritou p_j takto [21]:

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{jinak} \end{cases} \tag{2.2}$$

Použitý postup pro výběr příznaků a vytvoření klasifikátorů u algoritmu Viola-Jones lze popsat následovně. Vstupem je sada obrázků k učení $(x_1, y_1), \ldots, (x_n, y_n)$, kde $y_i = 0, 1$ značí negativní nebo pozitivní obrázek. Počáteční váhy obrázků jsou nastaveny jako $\omega_{1,i} = \frac{1}{2m}$ pro negativní a $\omega_{1,i} = \frac{1}{2l}$ pro pozitivní obrázky, kde m značí počet negativních a l počet pozitivních obrázků. Pro každý cyklus učení jsou normalizovány váhy a pro každý příznak je trénován slabý klasifikátor, ke kterému je vypočtena chyba detekce. Vybrán je klasifikátor s nejmenší chybou, poté jsou aktualizovány váhy. Výsledný silný klasifikátor je definován takto:

$$h_j(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{jinak} \end{cases} \tag{2.3}$$

kde $\alpha_t = \log \frac{1-\epsilon_t}{\epsilon_t}$ (ϵ_t značí velikost chyby klasifikátoru h_t) [21].

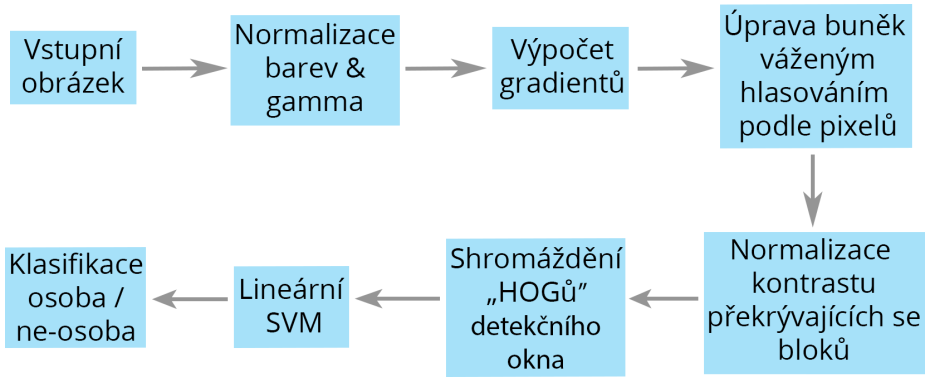
Optimalizací klasifikátorů je jejich seřazení za sebe v kaskádě. Jako první je použit klasifikátor, který vyřadí většinu detekčních oken obrázku s malou náročností na výpočetní výkon. Tento jednoduchý klasifikátor skládající se z dvou příznaků lidského obličeje umožňuje detekovat až 100% obličejů správně a až 40% falešných detekcí. Pokud je detekován obličej, předává se informace dalšímu klasifikátoru v pořadí, který vyřadí případnou falešnou detekci nebo obrázek označí za obličej a dále jej zpracovává další klasifikátor. Obecně jsou klasifikátory seřazeny od nejméně výpočetně náročného k nejvíce náročnému. Celá kaskáda má 38 částí a dohromady zahrnuje více jak 6000 příznaků. Experimentálně bylo zjištěno, že v průměru jsou obličeje detekovány s vyhodnocením prvních deseti částí kaskády na jedno detekční okno [21].

Hlavními výhodami této metody je její efektivita a rychlost, nezávislost detektoru na velikosti a umístění obličeje v obrázku (stačí upravit velikost příznaků), využitelnost stejného postupu pro detekci jiných objektů.

Nevýhodami je využitelnost pouze pro obličeje zobrazené zpříma (detektor velmi špatně pracuje s obličejem otočeným o 45° a více mimo přímý pohled) a citlivost na světelné podmínky (způsobeno použitím příznaků) [21].

2.2.2 Detekce pomocí histogramů orientovaných gradientů

Tento algoritmus je založený na histogramech orientovaných gradientů (HOG), byl navržen k detekci lidské siluety. Ke klasifikaci objektu používá trénovaný lineární SVM (Support Vector Machine, viz. 2.2.2). Přehled postupu práce metody HOG je na obrázku 2.5 a architektury na obr. 2.6. Základní myšlenkou je možnost dobře charakterizovat vzhled a tvar objektu podle směru změn intenzity nebo směru hran v obraze, zjednodušeně řečeno podle jeho tvaru.



Obrázek 2.5: Přehled schématu výběru charakteristických rysů a detekce objektů metodou HOG.

Prakticky je to implementováno rozdělením obrázku na menší části (buňky), kde každá buňka obsahuje 1-D histogram směru změn nebo natočení hran pro pixely v buňce. Nejprve jsou vypočítány horizontálně a vertikálně orientované středové gradienty bez vyhlazování ($\sigma = 0$) pro které byly experimentálně ověřeny jako nejvhodnější následující 1-D masky ve směru x a y:

$$D_x = \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} \qquad D_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \tag{2.4}$$

Dalším krokem je výpočet orientace gradientu (vektoru), kde g_x a g_y jsou jeho složky, přičemž $g_x = ID_x$ a $g_y = ID_y$. I reprezentuje obraz a D masku v daném směru:

$$\theta = \arctan(\frac{g_y}{g_x}) \tag{2.5}$$

A výpočet velikosti gradientu:

$$g = \sqrt{g_x^2 + g_y^2} \tag{2.6}$$

Použití větší masky nebo vyhlazování (např. $\sigma = 2$) výrazně snižuje rychlost algoritmu. Při použití barevného vstupu jsou vypočítány gradienty pro jednotlivé barvy (definované barevným prostorem RGB nebo LAB[1]) odděleně a vybrán nejvhodnější jako vektor gradientu daného pixelu. Úhly v histogramu mohou být od 0° do 360° (bez znaménka), pro detekci lidské siluety je vhodnější rozložení úhlů od 0° do 180° se znaménkem. Pro lepší výkon je znaménko u detekce osob ignorováno, pestrost barev oblečení osob a pozadí scény způsobují, že znaménko kontrastu většinou nepřináší užitečnou informaci.

[1]Barevný prostor ve kterém je barva určena světelností (L-lightness), osou A (od azurové k purpurové barvě) a osou B (od modré ke žluté barvě).

Aby byla detekce odolnější vůči rozdílům osvětlení nebo stínění v obrázku, je vhodné před použitím buněk normalizovat jejich kontrast. Je to provedeno seskupením buněk do bloků, které jsou následně normalizovány. Dalal a Triggs zkoumali čtyři rozdílné metody normalizace:

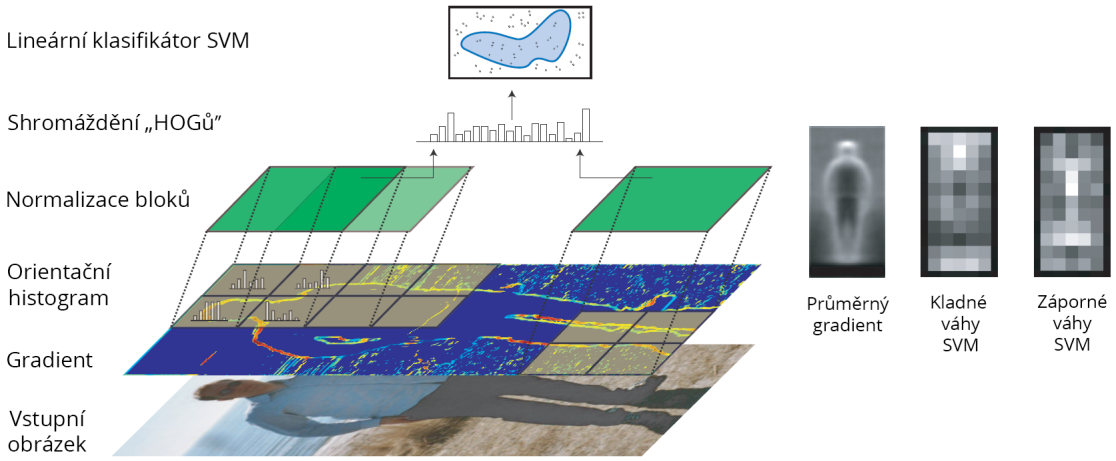
$$\text{L2-norm:} \quad f = \frac{v}{\sqrt{\|v\|_2^2 + \epsilon^2}} \tag{2.7}$$

$$\text{L1-norm:} \quad f = \frac{v}{\|v\|_1 + \epsilon} \tag{2.8}$$

$$\text{L1-sqrt:} \quad f = \sqrt{\frac{v}{\|v\|_1 + \epsilon}} \tag{2.9}$$

kde v je vektor obsahující histogramy daného bloku, $||v||_k$ je jeho k-norm pro $k = 1, 2$ a ϵ je malá konstanta. Čtvrtá metoda L2-hys spočívá v normalizaci pomocí L2-norm, oříznutí hodnot v na 0.2 a opakování normalizace. Jejich experimenty ukázaly, že L2-hys, L2-norm a L1-sqrt podávají téměř stejné výkony, jen L1-norm byla méně spolehlivá metoda.

Vhodná velikost bloku je 3x3 buněk. Částečné překrývání bloků navzájem, vylepšuje kvalitu normalizace a zvyšuje výkon detektoru. Podle typu bloku, který může být kruhový nebo čtvercový, rozdělují autoři [4] bloky na R-HOG a C-HOG. R-HOG bloky jsou podobné



Obrázek 2.6: Přehled architektury HOG s lineárním SVM. Vpravo je ukázán průměrný gradient obrázku společně s pozitivními a negativními váhami SVM v jeho částech [8].

SIFT (Scale Invariant Feature Transform) popisu vlastností. Nastavením parametrů lze měnit vlastnosti detektoru, parametry jsou ς, η, β. Kde $\varsigma \times \varsigma$ je mřížka rozdělující obraz do buněk o velikosti $\eta \times \eta$ pixelů, které obsahují β směrů vektorů. Pro detekci osob je nejúčinější velikost bloků 3x3 buněk o velikosti 6x6 pixelů [4]. C-HOG bloky mají dva typy uspořádání buněk. První má uprostřed jednu celou buňku, u druhého je centrální buňka rozdělena podle úhlů krajových buněk. Protože oba typy měly při experimentování podobný výkon, byl použit strukturálně jednodušší typ. Finální implementace používá R-HOG bloky, pro jejich lepší vlastnosti.

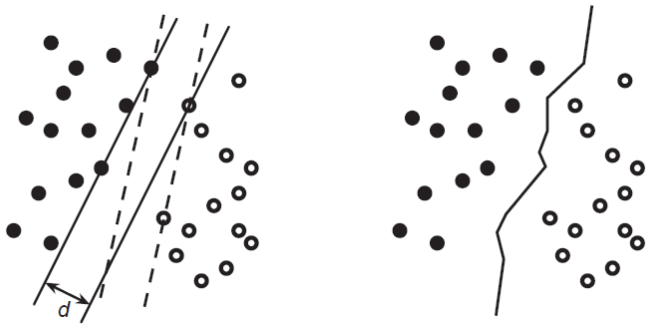
Detekční okno je velké 64×128 s rezervou o velikosti přibližně 16 pixelů kolem osoby na všechny strany. Menší rezerva získaná zmenšením detekčního okna nebo zvětšení osoby, snižuje výkon detektoru. Při rozpoznávání touto metodou je detekční okno postupně po-

sunováno po obrazu a pro každou pozici jsou vypočítány buňky a bloky uvnitř a následně provedena klasifikace lineárním SVM.

Klasifikátor SVM

Support Vector Machine je metoda statistického rozpoznávání vzorů, používaná od 90. let minulého století [5]. Základní princip je blízký lineárnímu rozdělení příznaků do dvou prostorů, viz. obrázek 2.7a. Cílem je nalezení páru rovnoběžných nadrovin, které vedou k oddělení s největší vzdáleností skupin bodů reprezentujících rozdílné vlastnosti a tím také zajistit malý počet chybně rozpoznaných objektů. Na obrázku 2.7a je příklad dvojice nadrovin (zobrazené přerušovanými čárami), které mají vzdálenost rozdělení d menší než u druhého páru, proto má tato dvojice menší ochranu proti chybám a není ideálním řešením. Každý pár rovnoběžných nadrovin je charakterizován množinami bodů vlastností, které rozděluje, tzv. podpůrnými vektory (support vectors). V prostoru na obr. 2.7a jsou roviny plně definovány třemi podpůrnými vektory, ačkoli tento počet platí pouze pro 2D prostory vlastností. Pro N dimenzí platí, že maximální počet podpůrných vektorů popisujících daný prostor je $N+1$. Z toho plyne důležitá pojistka proti zahlcení vektory, protože při jakémkoli počtu bodů vlastností existujících v prostoru, je maximální počet vektorů popisující daný prostor omezený.

Pro porovnání ukazuje obr. 2.7b situaci, která by nastala použitím metody nejbližšího souseda pro jednotlivé body. V tomto případě by byla ochrana proti chybám lepší, důvodem je optimalizace každé pozice oddělující roviny tak, aby blízké body různého typu dělila největší možná vzdálenost. Nicméně, toto vylepšení přesnosti je za cenu o dost většího počtu definujících vzorů. Avšak, jak je naznačeno výše, velká výhoda SVM pochází z využití nejmenšího možného počtu definujících vzorů (podpůrných vektorů). Nevýhodou je, že tato základní metoda funguje pouze v případech, kdy je množina dat lineárně rozdělitelná.



(a) (b)

Obrázek 2.7: Princip SVM. Vlevo (a) ukazuje dvě množiny lineárně oddělitelných bodů vlastností, kde dvě rovnoběžné nadroviny mají maximální možné rozdělení d. Přerušovanou čárou jsou naznačeny alternativní nadroviny. Vpravo (b) je zobrazeno ideální řešení, množiny jsou rozděleny podle jednotlivých bodů metodou nejbližšího souseda (nearest neighbor). [5]

Tento problém lze odstranit transformací trénovacích a testovacích dat do prostoru vyšší dimenze, ve které jsou data lineárně oddělitelná. Faktem je, že tento přístup často také odstraní hlavní výhodu SVM a vede k přeplnění daty a horšímu využití SVM pro obecné případy. Avšak, pokud není použitá transformace lineární, může mít výsledný prostor pří-

znaků nižší počet dimenzí než s lineární transformací a výhoda SVM nebude ztracena. Pro tento přístup je důležité stanovit základní omezení pro lineární oddělitelnost, aby bylo možné jednoznačně a efektivně určit, která data jsou oddělitelná a která nejsou. Bylo zjištěno, že je užitečné přidat do optimalizačních rovnic pomocné „*slack*“ proměnné s_i, které reprezentují o kolik lze překročit omezení oddělitelnosti. Je toho dosaženo přidáním cenové podmínky $C \sum_i s_i$ do běžné chybové funkce. Parametr C je nastavitelný a slouží k regulaci SVM, jeho hodnota je korigována podle výsledků klasifikátoru na množině trénovacích dat. [5]

2.2.3 Lokální binární vzory

Local Binary Pattern (LBP) je jednoduchý a zároveň velmi efektivní popis textury objektu. Je založený na značkování pixelů obrázku prahováním okolí každého pixelu a výsledek reprezentuje jako binární číslo. Vzhledem k jeho účinnosti a malé výpočetní náročnosti se LBP stal velmi populární metodou, která je využívána pro různé účely. Lze jej považovat za přístup sjednocující tradičně protichůdné statistické a strukturální modely analýzy textur. Asi nejdůležitější vlastností LBP je jeho robustnost vůči změnám způsobených měnícím se osvětlením. Další výhodou je nenáročnost výpočtu, díky tomu jej lze použít pro aplikace pracující v reálném čase.

Základní myšlenkou LBP je, že 2D texturu povrchu lze definovat dvěma doplňujícími se mírami: lokálními prostorovými vzory a kontrastem ve stupních šedi. Originální LBP vytvořený roku 1996 značkuje jednotlivé pixely obrázku pomocí prahování 3×3 okolím, přičemž je daný pixel uprostřed a výsledkem je binární číslo. Histogram těchto $2^8 = 256$ různých značek je možné potom použít jako deskriptor textury. Tento operátor společně s lokálním měřením kontrastu při automatické segmentaci textur podává velmi dobré výkony.

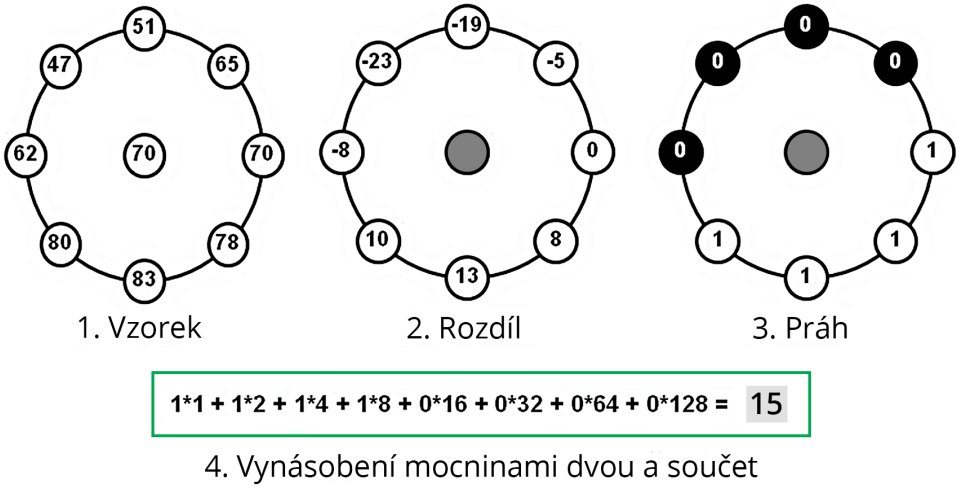
Roku 2002 byl LBP rozšířen o schopnost použití okolí s různou velikostí. Použitím kruhového okolí a bilineární interpolací hodnot pixelů s desetinnými souřadnicemi je možné mít okolí s libovolným poloměrem a počtem pixelů. Jako doplněk měření používá odchylku stupňů šedi od okolí pixelu. Hodnota LBP kódu pixelu se souřadnicemi na obrázku (x_c, y_c) je určena (příklad na obr. 2.8):

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^P \qquad s(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{jinak} \end{cases} \tag{2.10}$$

kde P znamená počet vzorkovaných bodů kruhového okolí s poloměrem R, g_p je hodnota bodu kruhového okolí a g_c je hodnota bodu na souřadnicích x_c, y_c a $s(x)$ je funkce prahování.

Dalším rozšířením originální verze je přidání možnosti definovat *uniformní vzory*, které lze použít ke zkrácení délky deskriptoru a implementaci jednoduchého deskriptoru nezávislého na rotaci. Toto rozšíření bylo inspirováno faktem, že některé vzory v texturách jsou běžnější než ostatní. Local binary pattern je nazýván uniformním, když binární vzor obsahuje maximálně dva bitové přechody z 0 na 1 a opačně. Při výpočtech LBP značek má každý takový vzor unikátní značku , ostatní (neuniformní) vzory jsou označeny jednou společnou značkou. Například okolí $(8, R)$ má celkem 256 vzorů, z toho je 58 uniformních a použito je 59 rozdílných označení. Po vypočítání obrázku s LBP označeními $f_l(x, y)$, je histogram definován následovně:

$$H_i = \sum_{x,y} I\{f_l(x, y) = i\} \qquad i = 0, \ldots, n-1 \tag{2.11}$$



Obrázek 2.8: Výpočet kruhového LBP [17].

kde n je počet rozdílných označení vytvořených LBP a $I\{A\}$ je 1, pokud je A true, jinak je 0. Pokud jsou porovnávány histogramy s rozdílnou velikostí, je nutné nejdříve provést normalizaci histogramů:

$$N_i = \frac{H_i}{\sum_{j=0}^{n-1} H_j} \tag{2.12}$$

Tuto metodu je vhodné kombinovat s rozpoznáváním pomocí tvaru, výsledný detektor je robustnější a proto přesnější [22]. Použitím algoritmu AdaBoost lze zvednout přesnost detekce objektů o 5% oproti samotnému LBP, tato metoda je označována jako BLB - Boosted Local Binaries. [18] [17]

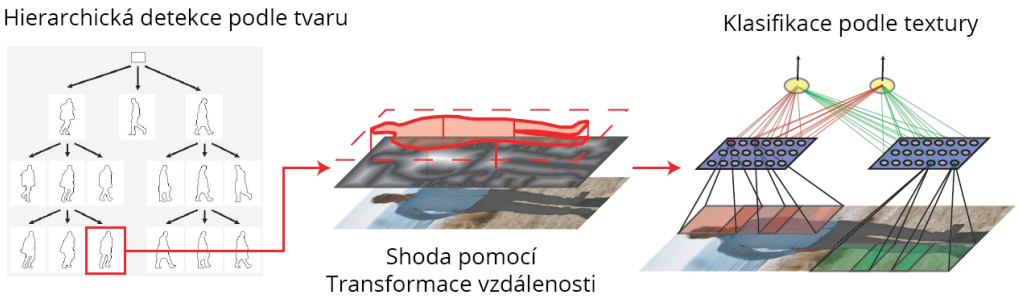
2.2.4 Shrnutí a porovnání detektorů

Pro odhad výšky je důležité, aby detektor správně umístil *bounding box* (ohraničení osoby), ideálně na pixel přesně. Dalším důležitým faktorem je nízký počet falešných detekcí, pokud by systém běžel automatizovaně, falešné detekce by znehodnocovaly výstup. V následující části jsou popsány nové pokroky, zjištění a porovnání často používaných metod detekce osob na základě uvedených článků.

Pokroky a nová zjištění

Aktuální možnosti detekce chodců jsou důkladně popsány ve článku *Survey of Pedestrian Detection for Advanced Driver Assistance Systems* [11], ačkoli se autoři zaměřili na využití detekce pro asistování řízení vozidel, je tento článek užitečný i pro detekci za účelem odhadu výšky. Ve článku je kladen důraz na fakt, že chodci mají značně rozdílnou velikost, držení těla, styl chůze, oblečení, přenášené věci a další. Snímané scény jsou často vyplněny různými objekty, které komplikují detekci. Kvůli tomu může být silueta chodce částečně zakrytá, v částech scény může být špatný kontrast a obecně scény bývají také dost rozmanité. Podstatná je i poznámka autorů, že detekce osob podle siluety je velmi rozšířená, navzdory tomu že je třeba ji doplnit o další krok založený na vzhledu objektu. Není to však míněno jako nevýhoda této metody, ale ukázka toho, že bez spojování metod nebude výsledný detektor obstojný vzhledem k rozmanitosti osob i sledované scény. Zajímavostí je,

že Kalmanův filtr je pořád zdaleka nejpoužívanější metodou ke sledování pohybu, zejména protože chodci se většinou pohybují podél chodníků, pěších zón nebo přecházejí cestu a nebývá stálý, ale při vyhýbání překážkám a ostatním chodcům je často chaotický. Geronimo a kolektiv také zdůrazňují nutnost použitelnosti detektoru během různého počasí a času, k tomu dodávají, že technologie snímkování s NIR (Near Infrared, blízké infračervené záření) poskytuje téměř stejně kvalitní snímky v noci jako za denního světla, proto je možné použít stejné algoritmy detekce. Tolik to však neplatí pro termální (vzdálené infračervené záření, FIR) snímky, které jsou běžně označovány jako „*noční vidění*“. Termokamery mají své využití pro detekci s odlišitelnou teplotou, například rozpoznání chodců mezi vozidly, ale jsou nevhodné pro detekci dopravních značek nebo různých objektů v pozadí. V takových případech musí být termální kamery doplněny o běžné ve dne a v noci o NIR kamery, proto jsou obecně zbytečným přepychem [11][5].



Obrázek 2.9: Princip kombinované klasifikace na základě tvaru a textury osob [8].

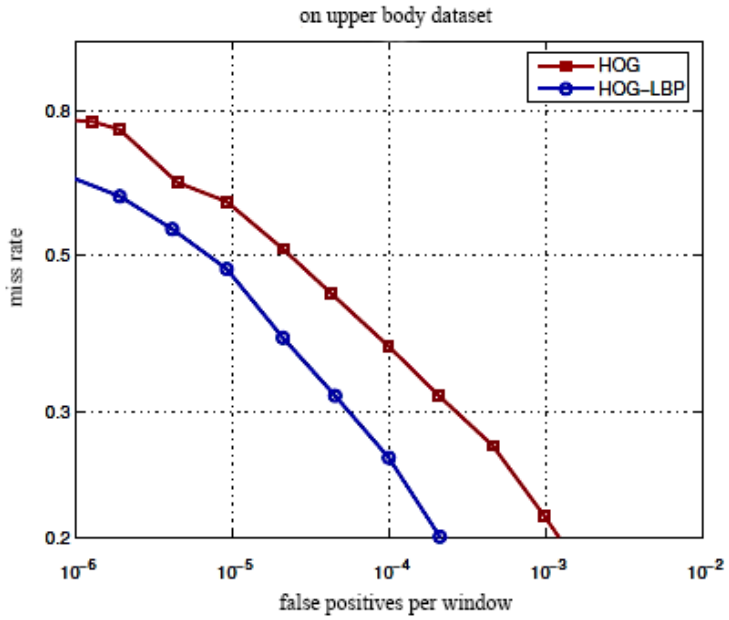
Ve článku *Multi-cue pedestrian detection and tracking from a moving vehicle* autoři Gavrila a Munder popisují vícestupňový systém detekce osob, který po rozsáhlém testování v městském prostředí považují za aktuálně (2007) nejlepší. Jejich detektor se skládá ze čtyřech hlavních modulů: generátor ROI (oblastí zájmu) používající stereo vizi, detekce podle tvaru objektu, klasifikace podle textur a verifikace pomocí stereo vize, navíc je celý systém doplňován sledovacím modulem. Částečně je úspěch tohoto systému založený na nové architektuře kombinující klasifikaci podle siluety a textury objektu. Podstatným zjištěním autorů je, že tento přístup podává lepší výkony než samotná klasifikace podle textur [10][5].

Stereo vize přináší kromě výhod také podstatné komplikace a v případě detektoru osob z větší vzdálenosti se ji často nevyplatí použít, navíc většina kamer ve městech stereo nepodporuje. Ve článku *Monocular pedestrian detection: Survey and experiments* autoři provedli experimentální porovnání metod detekce osob využívajících mono vizi. Nejdůležitějšími porovnávanými metodami byly: (1) kaskády Haarových vlnek, (2) neuronové sítě s lokálními poli vnímání, (3) HOG se SVM klasifikátorem a (4) kombinovaný přístup detekce podle siluet a textur. Čtvrtá metoda byla následně z porovnání vyřazena, protože její hlavní výhodou je rychlost zpracování a ta nebyla autory považována za relevantní. Jako nejvýkonnější metoda z výzkumu vyšla kombinace HOG se SVM, jejíž výsledky byly lepší než vlnky a neuronové sítě. Konkrétně s citlivostí 70% (procento detekovaných osob z celkového počtu) byl výskyt falešných detekcí respektive $0.045, 0.38$ a 0.86, reprezentující značnou redukci falešných detekcí oproti ostatním. Podobně při citlivosti 60%, byl HOG značně přesnější, zejména vůči neuronovým sítím. Tyto výsledky platí pro středně veliké obrázky osob cca 48×96 pixelů, při nižším rozlišení $\sim 18 \times 36$ pixelů měly nejlepší výsledky Haarovy vlnky. [8][5]

Zajímavé jsou i možnosti detekce výšky osob v noci, autoři článku *Pedestrian detection in infrared images based on local shape features* prováděly testy na obrázcích pořízených pomocí IR záření (s vlnovou délkou $7-14\,\mu m$). Jejich zjištěním bylo, že IR snímky jsou téměř podobné obrázkům za denního světla, proto je na ně možné aplikovat stejné algoritmy a není třeba vynalézat nové pro noční použití. Zejména detekci pomocí HOG se SVM (viz. 2.2.2 a 2.2.2) lze upravit pro zpracovávání IR snímků a obdobně také metody používající *boosting* (např. Viola-Jones, viz. 2.2.1). Důvodem jsou velmi podobné siluety na IR a normálních snímcích ve dne [25][5].

Zhodnocení metod detekce

Na základě zjištění autorů článků [8] a [11] je ve většině situací nejlepší metodou detekce HOG se SVM klasifikátorem. Metoda Viola-Jones může být v konkrétních situacích přesnější (méně falešných detekcí), ale obecně při detekci lidské siluety podává horší výsledky [8]. Kombinováním detekce podle textur s detekcí podle tvaru lze dosáhnout ještě lepších výsledků [10], příklad vylepšení HOG kombinací s LBP je popsán ve článku *An HOG-LBP human detector with partial occlusion handling*, kde autoři popisují zlepšení přesnosti kombinovaného detektoru o $\sim 20\%$ (při 10^{-4} falešných detekcí na snímek) oproti samostatnému HOG (obr. 2.10) [22]. Obecně je vhodné detektor doplnit algoritmem sledování, zlepšení výkonu detektoru je demonstrováno ve článku *Multi-cue pedestrian detection and tracking from a moving vehicle*, kde je uveden kombinovaný systém detekce podporovaný sledováním [10]. Metoda HOG je také s úpravami aplikovatelná na noční snímky [25]. Vzhledem k výše uvedeným výhodám, rozšiřitelnosti HOG a experimentálním ověřením článků [8] a [11], považuji tuto metodu za nejlepší pro detekci osob za účelem odhadu jejich výšky.



Obrázek 2.10: Porovnání výkonu detektorů HOG-LBP a HOG při detekci horní části těla [22].

2.3 Detekce siluety osoby

Následující sekce popisuje metody, které se často používají pro vylepšení, asistenci detektoru nebo dokonce samotné detekci (v případě modelování pozadí). Zejména jsou užitečné při odhadu výšky, pro který je podstatné přesně označit siluetu osoby. Substrakce pozadí detekuje (zvýrazní) pohybující se objekty, nevýhodou je omezená použitelnost pro některé scény. Kvůli tomuto omezení je vhodné použít detekce hran, například Cannyho algoritmu, který v této oblasti patří mezi nejvíce rozšířené metody. Jeho nevýhodou je nerozeznávání příslušnosti objektu k pozadí nebo popředí, pokud se sledovaná osoba pohybuje v rozmanité scéně, v důsledku hran pozadí je pak přesné ohraničení osoby nedosažitelné.

2.3.1 Substrakce pozadí

Hlavním přínosem metody substrakce pozadí je lepší detekce sledovaných objektů, zejména protože tyto objekty bývají součástí popředí snímku a po odečtení pozadí dojde k jejich zvýraznění.

Cílem modelování pozadí je vytvořit idealizovaný obrázek pozadí, který lze odečíst od libovolného snímku k získání popředí nebo sledovaného objektu. Nejjednodušší strategií jak toho dosáhnout je vzít snímek o kterém je známo, že neobsahuje sledované objekty (např. osoby) a použít jej jako model pozadí. Pro snížení šumu je užitečné vytvořit model pozadí z více takových snímků vypočítáním jejich průměru. Tato strategie má však dva problémy. Jak poznat, že na snímku opravdu nejsou žádné cíle, aby snímek reprezentoval pravé pozadí? A jak se vypořádat s situací běžnou venku, kdy se osvětlení scény mění s časem a podle aktuálního počasí? Druhý problém lze vyřešit používáním nedávno pořízených snímků, ale pokud bude model tvořen touto cestou, ztíží se tím řešení prvního problému. Možným kompromisem je vzít průměr většího množství nedávných snímků pozadí pořízených během intervalu Δt bez ohledu na objekty popředí. Pokud nejsou objekty v popředí časté, bude většina snímků vhodná a výsledný odhad modelu pozadí bude kvalitní. Samozřejmě při výskytu cílů bude občas jejich negativní vliv na výsledný model viditelný. Proto je vhodné model vylepšit prodloužením intervalu Δt a tím omezit výskyt prvního problému. Nebo interval Δt zkrátit a tím minimalizovat druhý problém. Kompromisu se však nelze vyhnout, částečně je to kvůli přímému „průměrování“ snímků. Tuto metodu lze vylepšit aplikací mediánového filtru na hodnotu $\mathbf{I}$ (intenzita a barva) každého pixelu přes celou sekvenci snímků během Δt. Použití mediánu je vhodné kvůli vyřazení krajních hodnot pixelů.

Při modelování pozadí je však třeba řešit i obtížnější problémy. Běžně se totiž pohybují nejenom objekty popředí (např. auta), ale i objekty pozadí. Zejména stíny se mění s časem a počasím, vegetace se pohybuje pod nápory větru s různou frekvencí závislou na síle větru a například velikosti listů. Pohyb vegetace proto způsobuje nežádoucí oscilaci hodnot pixelů pozadí. V těchto případech lze z předešlých trénovacích snímků získat pravděpodobnost s jakou pixel patří k pozadí nebo k novému objektu popředí.

Modely vytvořené z více složek pixelu (např. kombinace intenzity a barvy) jsou označovány jako smíšené modely. V praxi rozložení složek pixelu připomíná Gaussovo rozložení. Proto jsou další používané typy označovány jako Gaussovy smíšené modely a směsi Gaussových rozložení. Ačkoli většinou pixel obsahuje jen jednu složku a často jsou tři složky maximem, zpočátku není známý jejich počet. Proto je třeba analyzovat každý pixel jednotlivě, k tomu je používán algoritmus maximalizace očekávání (expectation maximization). Protože je tento výpočet náročný, běžně se používá pouze pro inicializaci tvorby modelu

Obrázek 2.11: Extrakce siluet osob substrakcí pozadí algoritmem ViBe [6].

pozadí a aktualizace těchto dat je prováděna efektivnějšími technikami, díky tomu může substrakce pozadí pracovat v reálném čase [5].

Gaussovy smíšené modely však nefungují při velmi vysokých frekvencích změn pozadí. K překonání tohoto problému autoři článku *Non-parametric model for background subtraction* [7] využili přístupu bez parametrů. Jejich metoda spočívá v aplikování vyhlazovací (typicky Gaussovy) funkce na každý pixel. Výpočet je proveden nad N vzorky s hodnotou $\mathbf{I}$ pro snímky pořízené během intervalu Δt před aktuálním časem t. Hlavní výhodou tohoto přístupu je jeho schopnost se rychle přizpůsobovat skokům intenzity a zároveň získat lokální odchylky jednotlivých pixelů. Další výhoda spočívá v tom, že je založen na pravděpodobnostech a zároveň nepotřebuje používat algoritmus maximalizace očekávání. Díky tomu pracuje velmi efektivně v reálném čase. Navíc jsou s touto metodou objekty v popředí detekovány s velkou citlivostí a nízkým počtem falešných detekcí.

K dosažení těchto výsledků je s barvou pixelu počítáno ve třech samostatných barevných kanálech a každý kanál má vlastní šířku pásma. Společně s použitím Gaussovy funkce je odhad pravděpodobnosti příslušnosti k pozadí pro bod s hodnotou $\mathbf{I}$ vypočítán následovně:

$$P(\mathbf{I}) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=1}^{C}\frac{1}{(2\pi\sigma_j^2)^{1/2}}e^{-(I_j-I_{j,i})^2/2\sigma_j^2} \tag{2.13}$$

kde i iteruje přes N vzorků z intervalu Δt a j iteruje přes C kanálů barev, σ je směrodatná odchylka a $I_j, I_{j,i}$ jsou hodnoty barvy j pro vzorek i. Tato funkce není výpočetně náročná, lze ji však ještě více zrychlit použitím vyhledávací tabulky s předem vypočítanými hodnotami jádra Gaussovy funkce.

Aby byl omezen efekt měnících se stínů, používá tato metoda souřadnice *chromaticity*. Vzhledem k tomu, že tyto souřadnice jsou nezávislé na množství osvětlení a stíny mohou být považovány za špatně osvětlené pozadí, budou stíny většinou zahrnuty do modelu pozadí. A o to méně je šance, že popředí bude obsahovat stíny po odečtení pozadí. Souřadnice *chromaticity* r, g, b jsou odvozeny z běžných R, G, B souřadnic rovnicemi $r = R/(R+G+B)$ a obdobně pro ostatní s tím, že platí $r + g + b = 1$ [7].

Faktem je, že stíny mohou být docela problémové. Stíny nejen zkreslují tvar objektů popředí po substrakci pozadí, ale také mohou spojit oddělené objekty popředí a tím ještě více zkreslit výsledek. Tento problém je možné řešit jejich odstraněním, například autoři článku *Shadow removal with blob-based morphological reconstruction for error correction* uvedli metodu využívající morfologie [24].

Celkově patří důvody selhání substrakce pozadí do dvou kategorií: (1) *Problém nehybného pozadí*, kvůli kterému není tvar objektů definován dostatečně přesně. (2) *Problém pomíjivého pozadí*, kdy nelze dostatečně rychle naleznout počátek a konec objektů popředí.

Pokud nemá model pozadí adekvátní přesnost nebo reaktivitu, povede substrakce pozadí k detekci falešných objektů. Stíny mají navíc tendenci tyto problémy spojovat.

2.3.2 Cannyho detekce hran

Pro správný odhad výšky je důležité co nejpřesněji označit siluetu osoby, pokud nelze z poskytnutých záběrů vytvořit kvalitní model pozadí (viz. 2.3.1), je vhodné k tomu použít alespoň algoritmus detekce hran. Již od roku 1986 patří mezi nejpoužívanější metody Cannyho detekce hran [3]. Mezi její výhody patří přesná specifikace šířky pásma v prostoru, ve které funguje. Také tato metoda vyřadila nepotřebné prahové hodnoty (používané u jiných algoritmů) bez ovlivnění detekce tenkých hran a zároveň zajišťuje jejich maximální propojení, aby výsledné detekované struktury byly co nejvíce kompletní. Celkem Cannyho detekce hran zpracovává obraz v následujících krocích:

1. Filtr prostorové frekvence (spatial frequency) s dolní propustí
2. Aplikování diferenčních masek prvního řádu
3. Potlačení nemaximálních (*non maximum suppression*) hodnot zahrnující subpixelovou interpolaci intenzity pixelů
4. Hysterezní prahování

Filtrování dolní propustí v zásadě musí být realizováno Gaussovými konvolučními operátory, u kterých je předem známa standardní odchylka σ. Poté je nutné aplikovat diferenční masky prvního řádu, k tomuto účelu je lze použít Sobelův filtr. V tomto případě lze za masku Sobelova filtru považovat konvoluci ($\otimes$) základního typu masky $\begin{bmatrix}-1 & 1\end{bmatrix}$ s vyhlazovací maskou $\begin{bmatrix}1 & 1\end{bmatrix}$. Derivace Sobelova filtru podle x tedy je:

$$\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix} = \begin{bmatrix}1\\ 2\\ 1\end{bmatrix}\begin{bmatrix}-1 & 0 & 1\end{bmatrix} \tag{2.14}$$

kde

$$\begin{bmatrix}1 & 2 & 1\end{bmatrix} = \begin{bmatrix}1 & 1\end{bmatrix} \otimes \begin{bmatrix}1 & 1\end{bmatrix} \tag{2.15}$$

a

$$\begin{bmatrix}-1 & 0 & 1\end{bmatrix} = \begin{bmatrix}-1 & 1\end{bmatrix} \otimes \begin{bmatrix}1 & 1\end{bmatrix} \tag{2.16}$$

Tyto rovnice ukazují, že samotný Sobelův filtr obsahuje podstatné množství filtrování s dolní propustí a díky tomu může být filtrování v prvním kroku rozumně omezeno. Důležité je nezapomenout, že filtrování s dolní propustí může být realizováno vyhlazovací maskou podobného typu jako B na obr. 2.12. Zajímavá je také podobnost této masky úplné 2-D Gaussově masce A zobrazené na obr. 2.12. Pro přehlednost nebyly matice na obrázku 2.12 normalizovány faktorem 1/16. Gaussova matice mimo zobrazenou část klesá až k nule a její integrál je 18.128, oproti integrálu matice B, který je 16. Proto lze říci, že jádro B aproximuje Gaussovo s rozdílem v rámci $\sim 13\%$. Skutečná standardní odchylka vyhlazovacího jádra je 0.707 a pro Gaussovo 0.849.

Na řadě je třetí krok, *non maximum suppression*. K tomuto účelu je třeba určit směr normály hrany s použitím následující rovnice:

$$\theta = \arctan\frac{g_y}{g_x} \tag{2.17}$$

$$A = \begin{bmatrix} 0.000 & 0.000 & 0.004 & 0.008 & 0.004 & 0.000 & 0.000 \\ 0.000 & 0.016 & 0.125 & 0.250 & 0.125 & 0.016 & 0.000 \\ 0.004 & 0.125 & 1.000 & 2.000 & 1.000 & 0.125 & 0.004 \\ 0.008 & 0.250 & 2.000 & 4.000 & 2.000 & 0.250 & 0.008 \\ 0.004 & 0.125 & 1.000 & 2.000 & 1.000 & 0.125 & 0.004 \\ 0.000 & 0.016 & 0.125 & 0.250 & 0.125 & 0.016 & 0.000 \\ 0.000 & 0.000 & 0.004 & 0.008 & 0.004 & 0.000 & 0.000 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Obrázek 2.12: Matice A obsahuje přesné hodnoty Gaussova vyhlazovacího jádra, které je ve svém středu nejblíže 3×3 vyhlazovacímu jádru B [5].

kde g_y a g_x jsou komponenty vektoru gradientu hrany a jeho velikost lze vypočítat:

$$g = (g_x^2 + g_y^2)^{1/2} \tag{2.18}$$

Pro snížení náročnosti výpočtu je běžnou praxí používat tuto aproximaci:

$$g = |g_x| + |g_y| \tag{2.19}$$

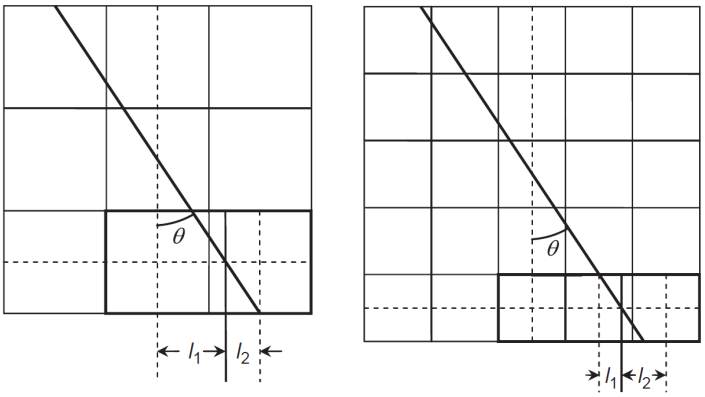
Po určení směru normály po ní podélně prochází algoritmus a určuje, zda je aktuální pozice lokálním maximem. Pokud není, je aktuální místo hrany odstraněno a zachovány jsou jen body podél normály hrany, které jsou lokálním maximem. Protože v každém směru může být lokálním maximem pouze jeden bod, budou hrany ztenčeny až na šířku jednoho pixelu. Vzniká zde malý problém, většinou totiž směr normály hrany neprochází středem sousedních pixelů a Cannyho metoda odhaduje intenzity podél normály interpolací. V okolí 3×3 je to provedeno snadno, protože normála hrany se musí nacházet mezi jedním párem pixelů (obr. 2.13a). Ve větším okolí může být interpolace prováděna mezí více páry pixelů. Například v okolí 5×5 je třeba určit, který pár je relevantní (obr. 2.13b) a také vybrat vhodný vzorec interpolace. Nicméně není třeba používat větší okolí než 3×3, protože to již obsahuje všechny důležité informace. A navíc díky vyhlazení v prvním kroku, bude dopad na přesnost velmi malý. Pokud však bude signál obsahovat impulzní šum, tak může dojít k seriozní chybě, ale filtrování s dolní propustí v žádném případě nezaručuje eliminaci impulzního šumu, proto použití menšího okolí pro non maximum suppression nepřináší podstatné ztráty. Tyto okolnosti je nutné pečlivě ověřit pro konkrétní vstupní data a šum, který obsahují. Obrázek 2.13 ukazuje vzdálenosti l_1 a l_2, které je nutné určit. Intenzita pixelů podél normály hrany je definována vážením korespondujících intenzit pixelů I_1, I_2 s *obráceným* poměrem ke vzdálenostem l_1, l_2:

$$I = \frac{l_2 I_1 + l_1 I_2}{l_1 + l_2} = (1 - l_1) I_1 + l_1 I_2 \tag{2.20}$$

kde vzdálenost l_1 je rovna tangens směru normály hrany [[?]]:

$$l_1 = \tan \theta \tag{2.21}$$

Posledním krokem je hysterezní prahování. Do této chvíle prováděly všechny operace co nejvíce úprav bez použití prahování, v této části se mu už nelze vyhnout. Účelem aplikace dvou hysterezních prahů je omezení poškození, které může způsobit první práh a opravit jej dalším. To znamená vybrat vyšší práh pro zajištění detekování *spolehlivých* hran a poté zvolit další body s velkou pravděpodobností příslušnosti k hraně, tyto body sousedí s body



(a) (b)

Obrázek 2.13: Cannyho interpolace pixelů. (a) Interpolace mezi dvěma označenými pixely vpravo dole v okolí 3×3. (b) Interpolace v 5×5 okolí, zde existují dvě možnosti interpolace mezi páry sousedících pixelů, vzdálenosti jsou naznačeny pro pár vpravo [5].

spolehlivých hran. Pro výběr spodního prahu existuje jednoduché pravidlo, jeho velikost by měla být přibližně polovinou vyššího prahu. Opět je to však obecné pravidlo a je lepší nastavit velikost prahů přesně podle vstupních dat [5]. Na obrázku 2.14 je ukázka výstupu Cannyho detekce hran.

Obrázek 2.14: Ukázka aplikace Cannyho detekce hran s hodnotou spodního prahu 41 a hodnotou vrchního 189. Použita byla OpenCV funkce `Canny`.

Kapitola 3

Návrh aplikace

Navrhovaný program je v podstatě videopřehrávač s detektorem osob a výpočtem jejich výšky. Od uživatele je očekáváno jako vstup video, vložení parametrů videokamery a označení referenčních objektů (tzv. etalonů) ve videu, společně s vložením informací o jejich vzdálenosti od kamery a výšce. Uživatelské rozhraní proto musí poskytovat možnosti vhodně vložit tyto údaje. Každý snímek videa je zpracováván několika kroky. Nejdříve je dekódován a načten do paměti aplikace, následuje detekování osob, protože samotný detektor neoznačí přesně siluetu osoby a občas jsou dokonce výstupem falešné detekce, je dalším krokem přesné označení a minimalizování falešných detekcí. K tomu je vhodné použít model pozadí (2.3.1) a odečíst jej od snímku, výsledkem jsou přibližně přesně ohraničeny siluety osob. Tento přístup však nelze použít, pokud video neobsahuje dostatečný počet snímků pozadí (vysvětlení v 2.3.1). Proto je vhodné pokusit se detekcí hran rozpoznat hranice siluety, jedním z algoritmů detekce hran je Canny 2.3.2. Jakmile je upraven výstup detektoru, je dalším krokem zjištění vzdálenosti osoby a následně výpočet její výšky, zde jsou k tomuto účelu využity uživatelem vložené referenční objekty. Vypočítaný odhad je zobrazen u detekované osoby.

První část kapitoly pojednává o detekci osob. Další o úpravě výstupu detektoru, proč je jej třeba upravovat a jaké zde jsou úskalí. Následuje návrh výpočtů vzdálenosti a výšky, jsou zde různé možnosti odhadu, ale obecně platí, že propracovanější přístup podává přesnější výsledky. Nakonec je popsán návrh uživatelského rozhraní a jeho inspirace v existujících řešeních.

3.1 Detekce osob

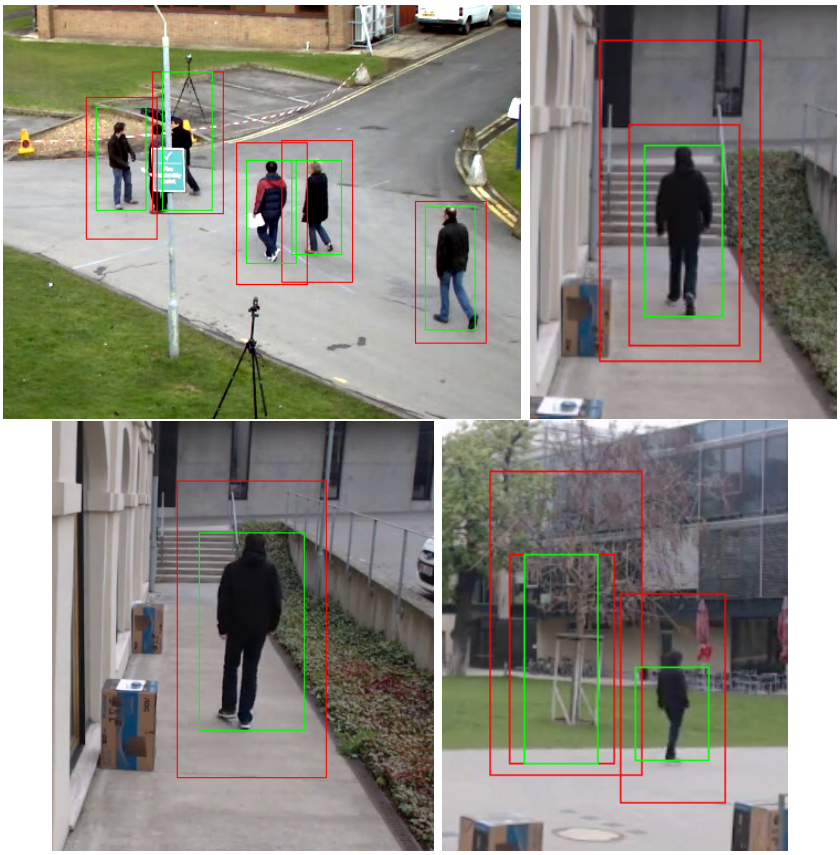
K detekci osob je vybrán HOG (sekce 2.2.2) s klasifikátorem SVM (2.2.2). Důvod výběru této kombinace je podrobněji popsaný v části 2.2.4. Stručně je HOG nejpřesnější (nejméně falešných detekcí) mezi běžnými detektory osob. Tento detektor reprezentuje celého člověka jedním vektorem vlastností a pro detekci používá posuvné okno 64×128. Pro každou pozici detekčního okna je vypočítán HOG deskriptor, následně jej SVM označí jako „člověk“ nebo „ne-člověk“.

Pro dosažení nejlepších výsledků klasifikace je vhodné SVM natrénovat na co největší množině dat, například na setu INRIA, který použili autoři HOG [4]. Alternativou je již natrénovaný SVM, v knihovně OpenCV jsou na výběr *defaultPeopleDetector* a *DaimlerPeopleDetector*. Volba mezi nimi závisí na konkrétním videu (vstupních datech detektoru), druhý zmiňovaný trénovali v laboratořích Daimleru na jiné množině snímků s menším roz-

lišením. Výsledkem trénování na takových snímcích je menší počet prováděných výpočtů nad detekčním oknem a díky tomu rychlejší klasifikace, v některých případech tento přístup však může vést k většímu počtu chybných detekcí.

3.2 Úprava výstupu detektoru

Výstupem HOG detektoru je obdélník ohraničující osobu (*bounding box*), který je však přibližně o 16 pixelů do všech stran větší než silueta osoby. Kromě správných výstupů se občas objeví falešné detekce. Z těchto důvodů je nutné nalezené *bounding boxy* upravit. Jako jednoduché řešení se nabízí odečíst od označené výšky v pixelech přibližnou hodnotu přesahu *bounding boxu*, avšak samozřejmě s větší nepřesností měření výšky v pixelech se chyba přenáší dále do výpočtu a silně ovlivňuje výsledný odhad výšky. Proto je vhodné začít s metodami, u kterých je větší pravděpodobnost správného označení siluety. Na obrázku 3.1 jsou ukázány výsledky úprav výstupu detektoru.



Obrázek 3.1: Úprava bounding boxů, které jsou výstupem detektoru. Zeleně je výsledek úprav, červeně původní. Na prvním a posledním obrázku je vidět jak hrana pozadí ovlivnila výsledný bounding box.

Metoda odečítání pozadí (viz. 2.3.1) označí pohybující se objekty v obraze velmi přesně, pokud bude použita pro místa, ve kterých byly detekovány osoby, je možné změnit velikost *bounding boxů*, aby se rovnala pohybujícímu objektu v dané oblasti snímku. Kromě toho je vhodná i k odstranění falešných detekcí, a to jednoduchým způsobem: *bounding boxy* bez nalezeného pohybu budou odstraněny. Nevýhodou této metody je nemožnost jejího

použití, pokud videozáznam neobsahuje dostatečný počet snímků pozadí (v tomto případě bez chodců).

Dalším přístupem je detekce hran v oblasti *bounding boxu*. Vhodným algoritmem je Cannyho detektor hran, princip jeho funkce je popsán v sekci 2.3.2. Protože nejsou rozeznávány rozdíly mezi hranami popředí a pozadí snímku, hlavní nevýhodou je možnost méně přesného ohraničení siluety než při odečtení 32 pixelů, tento případ se nejvíce projeví při detekci hran v pozadí kolem osoby nebo při částečném splynutí osoby s pozadím. Navíc u této metody není možné spolehlivé odstranění falešných detekcí a je třeba eliminovat je alternativně. Například kontrolou velikosti *bounding boxu* relativně k ostatním ve videozáznamu a příliš veliké nebo velmi malé odstranit.

Posledním problémem výstupu HOG jsou překrývající se *bounding boxy*. Autoři detektoru doporučují k jejich odstranění algoritmus Mean-Shift [4], který je v OpenCV přímo zabudovaný do metody detektoru (`cv::HOGDescriptor::detectMultiScale`). Opět však výsledky závisí na vstupních datech a proto je často lepší použít algoritmus non-maximum suppression. Ten porovnává plochu jednotlivých *bounding boxů* a podle nastavitelného prahu (s běžnou hodnotou 0.3) a souřadnic *boxů* rozhodne, které je vhodné odstranit.

3.3 Vzdálenost a výška osoby

Obrázek 3.2: Porovnání odhadu s využitím ohniskové vzdálenosti (230 cm), velikosti senzoru, výšky kamery a referenčních bodů (vlevo) s odhadem vpravo pomocí výšky kamery a referenčních objektů (192 cm). Skutečná výška osoby je 191 cm.

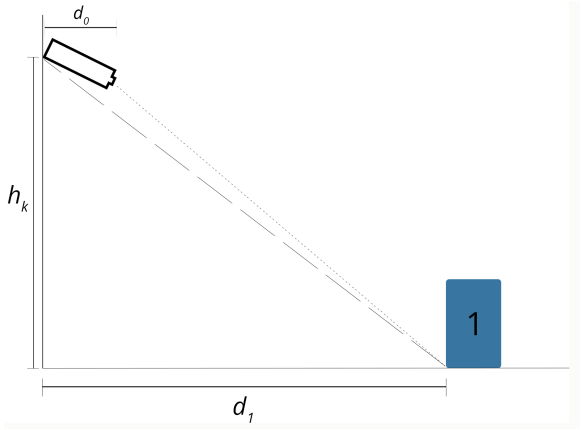
Výpočtu výšky lze dosáhnout více způsoby, jedním z jednodušších je porovnání objektu s jiným objektem známé výšky v obraze, například dopravní značkou nebo patníkem, které jsou ve stejné vzdálenosti od kamery jako detekovaný objekt. Většinou však bude osoba stát v jiné vzdálenosti. Zajímavým přístupem je počítat výšku v pixelech na obličej, takto je možné získat výšku osoby bez znalosti parametrů kamery. Průměrný obličej dospělého je široký 15.5 centimetrů [1, str. 72]. Z toho jde získat poměr cm/pixel v dané vzdálenosti a následně hrubý odhad výšky osoby podle počtu pixelů mezi vrchním a spodním bodem hranice siluety. Tento přístup je možný díky daleko menším variacím mezi šířkou obličeje než jaké jsou mezi výškou osob. Na obrázku 3.4 je zobrazená výška odhadnutá pomocí této metody. Nevýhodou je nutnost rozpoznatelnosti obličeje, například tento postup nelze

uplatnit na osoby otočené zády. Další možná metoda využívá znalostí optiky čočky kamery. Proto pro odhad výšky potřebuje následující parametry: výšku, ohniskovou vzdálenost a velikost senzoru kamery. Výšku objektu v záběru lze vypočítat podle následujícího vztahu:

$$h_o = \frac{d_o \times h_{op} \times h_{senzor}}{f \times res_h} \tag{3.1}$$

kde h_o je výška osoby, d_o vzdálenost osoby od kamery, h_{op} výška osoby v pixelech, h_{senzor} výška senzoru kamery, f ohnisková vzdálenost a res_h velikost vertikálního rozlišení kamery v pixelech.

Neznámé jsou výška osoby a její vzdálenost od kamery. Umístěním referenčních bodů do scény s jejich vzdáleností od kamery, je umožněno vypočítat vzdálenost osoby od kamery a následně její výšku. Referenční body mají známou vzdálenost od umístění kamery, ale pro výpočet je třeba vzdálenosti od čočky. Tu lze jednoduše dopočítat, pokud je zanedbána vzdálenost čočky a její rozdílná výška od umístění kamery, Pythagorovou větou: $d_c = \sqrt{h_k^2 + d_1^2}$, kde d_1 je vzdálenost objektu, h_1 výška umístění kamery a d_c vzdálenost od čočky, jak je znázorněno na obrázku 3.3. Získaná vzdálenost od čočky je dále přepočítána



Obrázek 3.3: Výpočet vzdálenosti referenčního objektu od čočky kamery. Vypočítaná vzdálenost od čočky je zobrazená přerušovanou čarou, tečkovaně je skutečná vzdálenost.

pomocí horizontálního úhlu záběru na vzdálenost od čočky na vertikální ose obrazu. Nejprve je nutné vypočítat úhel β pod kterým je daný bod zabraný:

$$\beta = \alpha \times \frac{w_1}{w_2} \tag{3.2}$$

kde α je horizontální úhel záběru, w_1 je šířka obrazu v pixelech a w_2 je vzdálenost x-souřadnice bodu od osy obrazu. Následně je pomocí úhlu β vypočítána vzdálenost bodu od čočky kamery na ose:

$$d = d_c \times \cos\beta \tag{3.3}$$

Poté je vypočítána závislost vzdálenosti na ose od čočky vzhledem k y-souřadnici v obrázku pomocí proložení přímkou:

$$y = A \times x + B \tag{3.4}$$

kde pro první přímku y představuje y-souřadnici bodu v obrázku a x vzdálenost od čočky na ose, A a B jsou parametry určující směr přímky. Vzdálenost osoby je získána obdobně,

úhel záběru je $\beta = \alpha \times \frac{w_2}{w_1}$, kde w_2 je vzdálenost x-souřadnice středu *bounding boxu*. A vzdálenost osoby je dopočítána $d = \frac{d_c}{\cos \beta}$, kde d_c je vzdálenost osoby na ose získaná dosazením do přímky vypočítané lineární regresí z referenčních bodů (rovnice 3.4). Jakmile je známa vzdálenost, je výška vypočítána rovnicí 3.1. Tato metoda by teoreticky měla fungovat správně, ale vzhledem k nepřesnostem u většiny vstupních parametrů výpočtu (například ohnisková vzdálenost často není výrobcem zveřejněná přesně) a praktickému ověření při implementaci, je tato metoda velmi nepřesná, viz. obrázek 3.2.

Obrázek 3.4: Odhad výšky (141 cm) pouze pomocí detekce obličeje a znalosti průměrné šířky obličeje (15.5 cm). Skutečná výška je 191 cm.

Na obrázku 3.2 je vpravo ukázán výsledek další navrhované metody. Ta využívá referenčních objektů, podle kterých je vypočítána druhá závislost: vzdálenost na ose od čočky na poměr metr/pixel v dané vzdálenosti. Odhadovaná výška je potom dopočítána dosazením do rovnice přímky 3.1, kde x je vzdálenost na ose od čočky a y je poměr metr/px. Tato metoda je přesná zejména v případech, kdy se osoba nachází ve vzdálenosti mezi referenčními objekty. Nevýhodou je nepřesnost vstupních parametrů vzdálenosti a výšky referenčních objektů a výšky kamery, tato nepřesnost se poté projeví ve výsledku odhadu.

Výše uvedené metody odhadu vzdálenosti a výšky osoby neberou v potaz deformaci obrazu perspektivním pohledem. Tu lze korigovat aplikací 3×3 transformační matice M, která je vypočítána následovně:

$$\begin{bmatrix} t_i x'_i \\ t_i y'_i \\ t_i \end{bmatrix} = M \times \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \tag{3.5}$$

kde (x_i, y_i) jsou souřadnice vstupních bodů, které vybere uživatel a (x'_i, y'_i) jsou souřadnice výstupních bodů minimálního obdélníku ohraničujícího vstupní body, $i = 0, 1, 2, 3$. Pomocí transformační matice M jsou vypočítány nové pozice bodů v obraze:

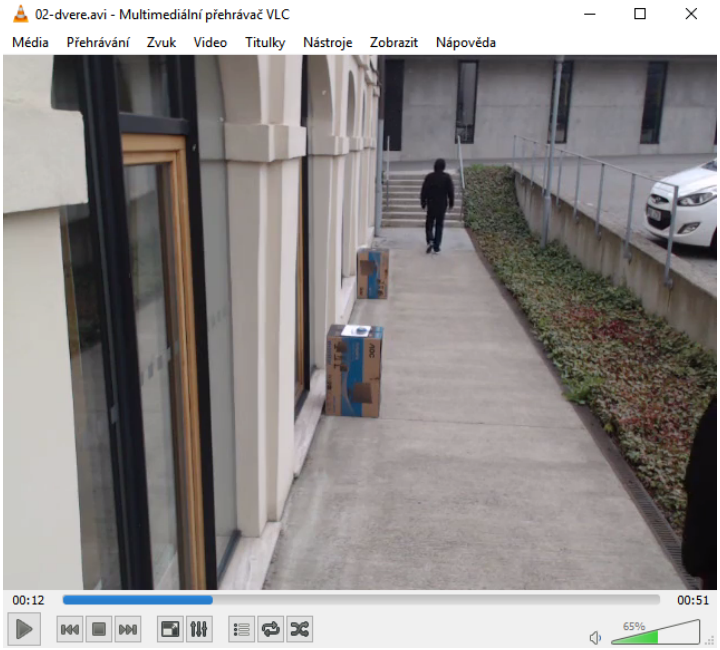
$$I(x, y) = oldI\left(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}}\right) \tag{3.6}$$

kde $I(x, y)$ je nový bod obrázku a $oldI(x, y)$ starý bod se souřadnicemi x, y. Ideálním výsledkem je obrázek bez deformace perspektivou.

3.4 Uživatelské rozhraní

Hlavním úkolem uživatelského rozhraní je poskytnout pohodlnou a ideálně rychlejší možnost ovládání aplikace. Požadovanými schopnostmi jsou načítání a přehrávání videa, vkládání parametrů videokamery, označení referenčních objektů a zadání jejich vzdálenosti a výšky. Příkladem přehrávače je populární program VLC zobrazený na obrázku 3.5, podobností však není mnoho. Například pro tento program není příliš důvodů pro přehrávání zvuku, zejména kvůli většině bezpečnostních kamer, které zvuk nenahrávají nebo je velmi špatné kvality. Také podpora změny rychlosti přehrávání není důležitá. Přehrávači aplikace pro odhad výšky stačí základní funkce: přehrát, pozastavit, zastavit a přesun v čase videa. Většina akcí v aplikaci má přiřazeny klávesové zkratky a také jsou dostupné pomocí ikon nebo položek v menu. Náčrt návrhu vzhledu a rozložení prvků UI je v příloze na obrázku B.1.

Vkládání parametrů videokamery je vyřešeno postranním panelem, který je v základu skrytý a je zobrazitelný klávesovou zkratkou nebo kliknutím na ikonu. Panel obsahuje pouze formulář s popisky, ukládání vložených údajů se provede automaticky po dokončení editace. Intuitivním prostředkem k vložení referenčních objektů je počítačová myš, různé kombinace tlačítek lze použít pro přidávání ref. bodů, objektů nebo jejich odstranění. Po kliknutí na požadované místo ve videu se zobrazí vedle kurzoru jednoduchý dialog pro zadání vzdálenost od kamery a výšky objektu. Odstranění proběhne ihned po kliknutí pravým tlačítkem na bod, nevyžaduje potvrzení. Lepší zaměření myší umožňuje křížový kurzor při „najetí“ na video, který sice pomůže zaměřit správný pixel, ale ideálním řešením by byla možnost přiblížení až na viditelnou mřížku pixelů, až potom je zaměření přesné. Pro vykreslení bounding boxů a odhadnuté výšky je zvolena zelená barva, jelikož je dobře vidět a zároveň je běžně používanou barvou k označení detekovaného objektu. Referenční objekty jsou vykresleny červeně, protože je to také kontrastní barva. Ikony použité v aplikaci jsou open source nebo vlastní výroby.



Obrázek 3.5: Uživatelské rozhraní přehrávače VLC.

Kapitola 4

Implementace

Úkolem bylo implementovat navržený algoritmus odhadu výšky a jednoduché uživatelské rozhraní (viz. návrh v kapitole 3). Implementační jazyk byl volitelný, při navrhování byl vybrán jazyk C++, díky jeho rozsáhlým možnostem a existenci vhodných nástrojů. Aplikace byla implementována kompletně od začátku, se zaměřením na použití v operačním systému Windows, avšak díky použitým multiplatformním nástrojům jsou možností i linuxové systémy.

4.1 Nástroje

Aplikace byla vyvíjena a testována pod operačním systémem Windows 10 pro 64 bitovou architekturu. Použitým vývojovým prostředím byl Qt Creator 4.2.1. A použitými knihovnami Qt 5.8 společně s OpenCV (Open Source Computer Vision Library) 3.2. Pro úpravu a vytváření ikon uživatelského rozhraní byl použit Photoshop.

OpenCV

Open source knihovna zaměřená na počítačové vidění a strojové učení [16]. Obsahuje více jak 2500 optimalizovaných algoritmů, které zahrnují jak klasické tak současné techniky pro zpracovávání obrazu a strojové učení. Tyto algoritmy lze například použít k detekci a rozpoznání obličeje, identifikaci objektů, klasifikace činností osob ve videu, sledování pohybu kamery, sledování pohybujících se objektů, extrakce 3D modelů z obrazu, vytváření 3D mraků ze stereo kamer, spojování obrázků pro vytvoření snímku celé scény s vysokým rozlišením, nalezení podobných obrázků v databázi (Google Images), odstranění „červených očí“ ze snímků, sledování pohybu očí, rozpoznání scény a vložení značek pro překrytí rozšířenou realitou, a další. Tato knihovna je rozsáhle používána ve firmách, výzkumných skupinách a vládních orgánech.

Praktické příklady použití OpenCV počínají od spojování obrázků streetview dohromady, detekování narušení hlídaného prostoru v Izraeli, monitorování těžebních strojů v Číně, pomoc při navigaci a přenášení objektů roboty firmy Willow Garage (VUT vlastní jejich robota PR2), detekce topících se lidí, provozování interaktivního umění, až po rychlou detekci obličejů v Japonsku.

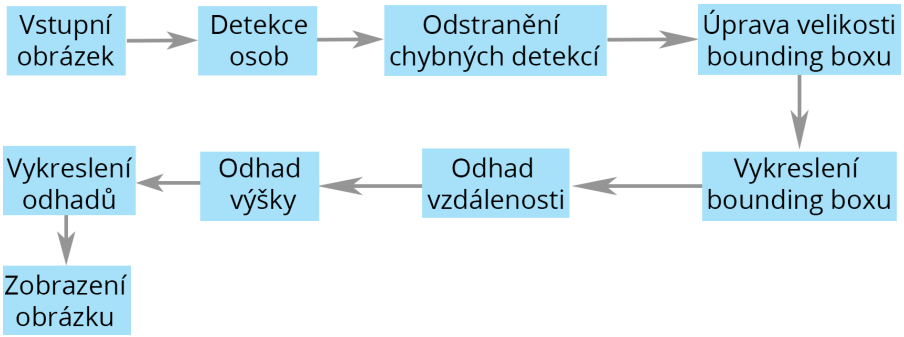
OpenCV je podporuje operační systémy Windows, Linux, Android, Mac OS a má rozhraní pro jazyky C++, C, Python, Java a MATLAB. Nativně je napsáno v jazyce C++ [16].

Qt

Multiplatformní framework pro vývoj aplikací na desktop, vestavěné systémy a mobilní zařízení. Mezi podporované platformy patří Linux, OS X, Windows, VxWorks, QNQ, Android, iOs, BlackBerry a Sailfish OS. Qt je objektově orientované a je napsáno v jazyce C++, který však rozšiřuje pomocí preprocessoru MOC (Meta-Object Compiler) o další funkce jako jsou signály a sloty objektů. Před kompilací analyzuje MOC zdrojové soubory napsané rozšířeným C++ a vygeneruje z nich standardní C++ zdroje. Díky tomu může být ke kompilaci použit jakýkoli standardní C++ kompilátor. Například Clang, GCC, ICC, MinGW a MSVC, který byl používán při implementaci aplikace odhadu výšky. Kromě komerčních licencí je Qt dostupné pod různými verzemi GPL a LGPL.

Společně s Qt je dodáváno IDE (integrované vývojové prostředí) Qt Creator. Podporuje operační systémy Linux, OS X a Windows. Jeho hlavní výhodou je provázání s Qt, díky tomu poskytuje inteligentní doplňování kódu, zvýraznění syntaxe a zejména integrovanou nápovědu Qt. Kromě toho obsahuje i debugger a podporu verzovacích systémů (například Git).

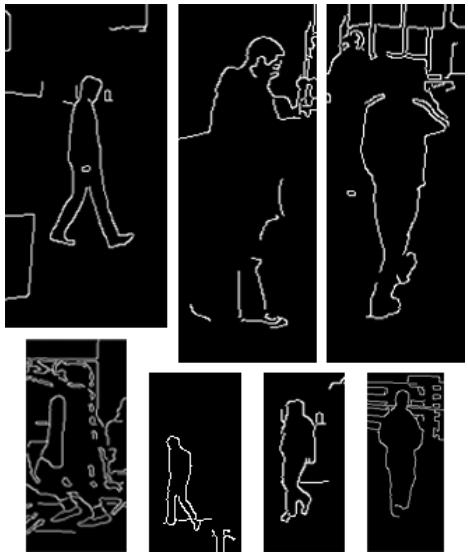
4.2 Zpracování obrazu



Obrázek 4.1: Schéma implementovaného postupu zpracování obrazu.

Postup programu při odhadu výšky je na obrázku 4.1. Většina funkcí provádějících zpracování obrazu je ve třídě `QGLdisplay`, která také obstarává vykreslení aktuálního snímku pomocí OpenGL. Zpracovávané snímky jsou do ní posílány třídou `MainWindow`, která je dekóduje z otevřeného videozáznamu. V této třídě je také implementován videopřehrávač, který pomocí časovače řídí proces dekódování snímků (přehrávání). V `QGLdisplay` je hlavní metodou `Read`, ta je volána funkcí `MainWindow::ReadNextFrame`, její podrobnější popis je v sekci 4.3. Metoda `Read` nejprve na snímek aplikuje OpenCV implementaci HOG s klasifikátorem SVM `detectMultiScale`. Klasifikace je prováděna pomocí `defaultPeopleDetector`. Výstupem detekce je `std::vector` obdélníků (`cv::Rect`), každý obdélník (bounding box) má přiřazenou váhu, která představuje „jistotu“ správné detekce a jeho důležitost. Tato váha je následně zohledněna ve funkci `NMS`, která algoritmem *non maximum suppression* odstraní překrývající se nebo nadprůměrně velké bounding boxy.

Další krok zajišťuje funkce `resizeRectsForEstimation`. Tím je upravení výšky bounding boxů, tak aby co nejlépe ohraničovaly siluetu osoby, šířka je vždy pevně zmenšena o 30%. V cyklu je postupně provedena detekce hran funkcí `Canny` (viz. 2.3.2), nalezené hrany jsou spojeny `findCountours` a zakresleny `drawContours` bílou barvou. Výsledek aplikace



Obrázek 4.2: Ukázka nalezených hran v bounding boxech, podle kterých je upravena výška.

těchto funkcí je na obrázku 4.2. Podle prvních nalezených vrchních a spodních pixelů je zmenšena výška bounding boxu (použita je funkce `findNonZero` - hledá bílé body).

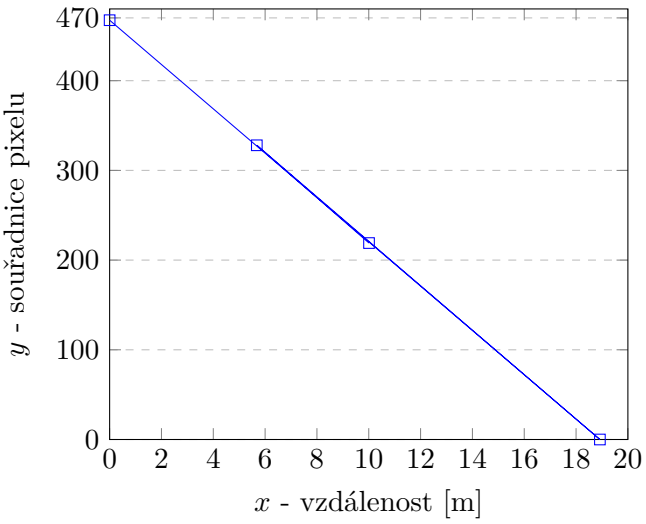
Po úpravě bounding boxů následuje odhad vzdálenosti a výšky funkcí `estimateHeight`. Tato funkce vybere jednu z metod odhadu výšky podle dostupných parametrů a priority metody. Pokud jsou dostupné referenční objekty, výška kamery a horizontální úhel obrazu nebo velikost senzoru, je odhad proveden na základě dvou přímek vypočítaných lineární regresí nad daty referenčních objektů (funkce `fitLinearLine`). První přímka (funkce `setDistanceLine`) vypočítá parametry přímky reprezentující vztah souřadnice y a reálné vzdálenosti objektu na ose obrazu od čočky kamery, příklad je na obrázku 4.3. Výpočet vzdálenosti na ose je proveden nejprve získáním vzdálenosti od čočky funkcí `getDistanceFromLens` (viz. obrázek 3.3). Poté je přepočtena na vzdálenost na ose pomocí `getViewAngleDistance`. Druhá přímka (`setHeightLine`) reprezentuje vztah vzdálenosti na ose a výšky v metrech na pixel. Dosazením parametrů bounding boxu do těchto přímek je ve funkci `estimateHeight` vypočítán odhad vzdálenosti a výšky osoby.

Pokud jsou dostupné referenční body, ohnisková vzdálenost a velikost senzoru nebo horizontální úhel obrazu je odhad vzdálenosti počítán stejně jako u ref. obj., ale odhad výšky je vypočítán podle rovnice 3.1. Poslední možností je odhad podle šířky obličeje, kdy nejsou potřeba vstupní parametry, stačí pouze *bounding box* obličeje.

Pokud je třeba odstranit deformaci perspektivou může uživatel stisknutím Shift + levé tlačítko myši označit body pro transformaci perspektivy. Po označení čtyřech bodů je provedena transformace funkcí `doPerspektiveTransform`. Ve které je získán minimální obdélník ohraničující vstupní body pomocí funkce `boundingRect`. Dalším krokem je vypočítání transformační matice funkcí `getPerspectiveTransform` (viz. rovnice 3.5). Matice je na obrázek aplikována funkcí `warpPerspective` (viz. rovnice 3.6).

4.3 Uživatelské rozhraní

Vzhled uživatelského rozhraní v akci je na ukázán na obrázku 4.5. Aplikace má standardní menu `QMenuBar`, které je možné skrýt klávesou `ALT`, pod ním se nachází lišta `QToolBar` s iko-

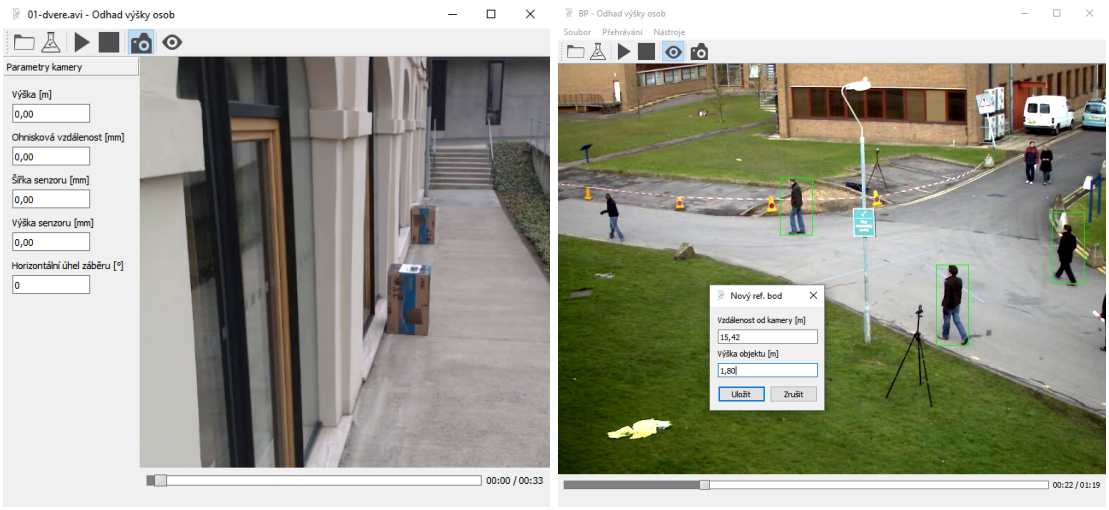


Obrázek 4.3: Příklad závislosti pozice referenčních objektů v obraze na jejich vzdálenosti od čočky. Souřadný systém OpenCV má nulový bod obrázku v levém horním rohu, proto je přímka klesající.

nami často prováděných akcí. Tento toolbar je možné přesunout pod video nebo jej vyjmout a umístit kamkoli na obrazovku. Centrální částí rozhraní je widget třídy `QGLdisplay`, ten zajišťuje většinu výpočetních operací a také vykreslení snímku pomocí OpenGL. Součástí tohoto widgetu je metoda `mousePressEvent`, ta umožňuje intuitivní přidání referenčních objektů nebo bodů a jejich odstranění. První stisk levého tlačítka myši slouží k označení prvního bodu ref. objektu a druhým stiskem dojde k zobrazení dialogu `AddRefDialog` pro zadání vzdálenosti a výšky objektu, potvrzením je přidán ref. objekt. Referenční body a objekty jsou vykresleny červeně funkcemi `drawPoint` a `drawLine`, jejich vzdálenost (u spodního bodu) a výška (mezi body objektu) je zobrazená v metrech funkcí `drawText`. Jmenované kreslicí funkce zobrazují přes obrázek jako overlay. Přidání pouze referenčních bodů je možné stiskem `CTRL+LMB` (levého tlačítka myši), poté se zobrazí dialogové okno vyžadující vyplnění vzdálenosti bodu. Odstranění přidaných objektů a bodů lze provést kliknutím pravým tlačítkem myši.

Upravený výstup detektoru je zobrazen v podobě zeleného obdélníku vykresleného funkcí `rectangle`, pokud jsou do aplikace vloženy povinné parametry pro odhad (výška kamery a referenční objekty), je odhadnutá vzdálenost v metrech zobrazena u pravého dolního rohu obdélníku a odhadovaná výška u levého vrchního rohu, pomocí funkce `putText`, která stejně jako `rectangle` kreslí přímo do snímku. Zadávání parametrů využitých v odhadu je ukázáno na obrázku 4.4. Aktuální čas přehrávaného videa je zobrazen pomocí `QLabel` a je aktualizován pro každý snímek. Video lze přetáčet widgetem `QSlider`. Ve status baru `QStatusBar` je vypsána rychlost videa získaná z jeho metadat a aktuální doba zpracování jednoho snímku v milisekundách.

Implementovaný videopřehrávač načte otevřené video do `VideoCapture`, následně z metadat videa získá informace o počtu snímků, rychlosti videa a jeho rozlišení. Podle rozlišení videa nastaví pevnou velikost okna funkcí `setWindowSize`. Podle rychlosti `CAP_PROP_FPS` a celkového počtu snímků `CAP_PROP_FRAME_COUNT` inicializuje časovač `QTimer` a zobrazení aktuálního času videa. Ovládání přehrávání funguje na spouštění/vypínání časovače přehrávače, klávesovou zkratkou pro přehrání a pozastavení je mezerník, pro zastavení videa



Obrázek 4.4: Vlevo ukázka nastavení parametrů kamery a vpravo dialog přidání nového referenčního bodu.

slouží klávesa S. Časovač periodicky spouští funkci `ReadNextFrame` ve které je následující snímek načten, dekódován a poslán ke zpracování funkcí `Read`. Doba trvání tohoto procesu je měřena pomocí `high_resolution_clock`.

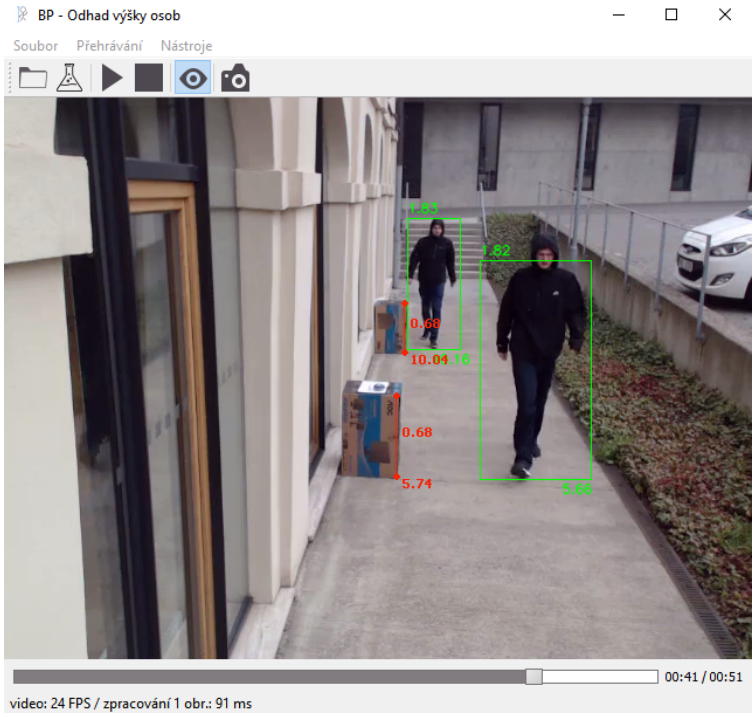
4.4 Experimenty a analýza výsledků

Experimentováno bylo s parametry detektoru, konkrétně s funkcí `detectMultiScale`. Existují tři různé stavy nalezení osoby: správná detekce (true pozitive - TP), chybná detekce (false positive - FP) a nedetekováno (false negative - FN). Vyhodnocení úspěšnosti bylo provedeno pro čtyři rozdílné scény (na obr. 4.7). Rychlost a přesnost detekce nejvíce ovlivňuje parametr `scale`, který určuje zvětšení obrázku, na tabulce 4.1 je vidět, že nejvýhodnější bude hodnota 1.15, která je kompromisem mezi rychlostí a přesností. Vybrána byla scéna 4, protože je natočena z pohledu umístění bezpečnostní kamery a nachází se v ní dostatečně osob v různých situacích. Hodnocen byl každý 30. snímek. Při tomto nastavení je detektor rychlejší než s nižšími hodnotami a zároveň poměr TP, FP a FN je přibližně stejný. Zajímavým zjištěním je větší počet vnořených detekcí při vyšším `scale`.

Scéna 4 - scale	1,05	1,1	1,25	1,45
TP	104	103	97	95
FP	2	0	0	0
FN	45	46	52	54
doba zpracování	570 ms	350 ms	230 ms	200 ms

Tabulka 4.1: Porovnání vlivu parametru `scale` funkce `detectMultiScale` na počet správných detekcí a dobu zpracování jednoho snímku.

Vyhodnocení úspěšnosti úpravy bounding boxu, aby co nejlépe ohraničoval siluetu je obtížné. Protože úprava pracuje na základě detekce hran a často do ní zasahují hrany pozadí. Kvůli tomu velmi záleží na podmínkách, struktuře pozadí a vzhledu osoby. Výsledkem je



Obrázek 4.5: Ukázka vzhledu uživatelského rozhraní při zapnuté detekci a odhadu výšky, červeně jsou referenční objekty. Červená a zelená čísla jsou v metrech. Dole ve stavové liště je zobrazena doba zpracování jednoho snímku a původní rychlost videa.

přesné ohraničení při „hladkém“ pozadí a velmi nepřesné při detekci hran pozadí, kdy bounding box zůstává cca o 30 cm větší než silueta osoby.

Za účelem experimentálního ověření přesnosti odhadu bylo natočeno několik videí kamerou Logitech HD Pro C920 na obrázku 4.6. Natočeny byly scény 1 a 2 na obrázku 4.7. První ověřuje přesnost při malém ovlivnění perspektivou, ve druhé scéně je více viditelná deformace perspektivou. V tabulce 4.2 jsou výsledky odhadů výšky, protože při chůzi se výška člověka periodicky mění a také například při pohledu zezadu je bounding box ovlivněn chůzí: při části kroku s jednou nohou vzadu je noha cca o 30 cm blíže ke kameře než celý člověk. Je vhodné průměrovat odhad výšky, jelikož se tím přibližně odstraní chyby způsobené chůzí. Při použití referenčních objektů je vhodnější váhový průměr, protože přímá úměra podle které je odhad prováděn, platí s různou přesností závislou na pozici osoby vůči referenčním objektům. Při odhadu výšky byla mírně upravena perspektiva obrazu funkcí `doPerspektiveTransform`, aby byl odhad méně zkreslen perspektivním pohledem kamery.

Obrázek 4.6: Kamera Logitech HD Pro C920, která byla použita pro natočení experimentálních videí.

	Odhadovaná výška v cm				
Video č.	Min.	Max.	Průměr	Vážený průměr	Rozdíl v. prům. od skutečné
1	158	212	191,29	193,21	2,21
2	167	214	195,00	196,06	5,06
3	198	215	206,17	205,88	14,88

Tabulka 4.2: Porovnání rozdílu odhadu od skutečné výšky 191 cm. První dvě videa jsou z pohledu scény 1 a poslední je z pohledu scény 2. Vstupními daty byly bounding boxy ohraničující dostatečně siluetu. Vážený průměr byl počítán podle vzájemné polohy osoby a referenčních objektů a to s těmito vahami: před - 0.4, za - 0.6, mezi - 0.8, u - 1.0.

Hlavním cílem práce bylo určit přesnost odhadu výšky bez použití kalibrace kamery a přesných znalostí parametrů kamery. Z provedených experimentů vyplývá, že tento přístup očekávaně není tolik přesný jako metody využívající kalibraci, ale i bez kalibrace lze učinit použitelný odhad. Průměrná chyba odhadu je cca 5 cm, viz. tabulky 4.2 a v příloze C.1. Odchylka odhadu výšky v procentech je vypočítána vzorcem:

$$\left| \frac{odhadVysky}{realnaVyska} - 1 \right| \times 100 \tag{4.1}$$

Měření vzdáleností referenčních objektů bylo provedeno s přesností cca 5 - 10 cm, to samozřejmě ovlivní přesnost odhadu, ale pokud by byla aplikace používána v reálných podmínkách, lze takovou přesnost měření vzdáleností očekávat.

Obrázek 4.7: Ukázky možných scén. Experimenty odhadu výšky byly provedeny na scéně č. 1 a 2.

Kapitola 5

Závěr

Pro splnění cíle práce bylo třeba nastudovat bezpečnostní kamery a jejich vlastnosti. Dalším krokem bylo studium metod rozpoznávání objektů za účelem vyhodnocení možností detekce osob a výběru detektoru, který bude implementován. Protože výstup detektoru je třeba před odhadem výšky upravit, byly nastudovány i vybrané metody zpracování obrazu, jako je například detekce hran. Na základě získaných znalostí a omezení zadáním byl navržen algoritmus odhadu výšky osob. Navržený postup detekce, úprav výstupu detektoru a odhadu výšky byl implementován v programovacím jazyce C++. Součástí návrhu a implementace bylo i uživatelské rozhraní.

Prvním bodem zadání bylo seznámení se s problematikou sledování bezpečnostními kamerami a s algoritmy pro detekci objektů v obraze. Tento bod byl splněn v kapitole 2, konkrétně podkapitola 2.1 se zabývá bezpečnostními kamerami a okolnostmi jejich použití. Algoritmy pro detekci objektů v obraze se zabývá podkapitola 2.2, ve které jsou popsány běžně používané metody a algoritmy užitečné odhadu výšky. Dalším bodem zadání bylo zhodnocení algoritmů z hlediska využitelnosti pro odhadování výšky a následný výběr vhodného algoritmu pro detekci osob v obraze. Zhodnocení a výběr jsou popsány v sekci 2.2.4. Třetí bod zadání požadoval návrh algoritmu odhadu výšky, ten popsán v kapitole 3, konkrétně v sekci 3.3. Implementaci navrženého algoritmu a uživatelského rozhraní, podle čtvrtého bodu zadání, popisuje kapitola 4. Poslední bod zadání je zpracován v části 4.4, popisující provedené experimenty a jejich vyhodnocení se zaměřením na odchylku odhadu od skutečnosti.

Účelem bylo vytvoření algoritmu nevyužívajícího kalibrace kamery ani dalších pokročilých parametrů, ale používajícího pouze referenční objekty nebo body a základní informace o kameře. A ověření přesnosti takového odhadu. Z dosažených výsledků je možné konstatovat, že úkolu bylo dosaženo, ale tento algoritmus je méně přesný než algoritmy používající kalibraci kamery a další metody. Důvodem je zanedbání několika skutečností, jako je deformace čočkou a perspektivní deformace obrazu. Výsledná odchylka se průměrně pohybuje od 2 cm do 15 cm, podle prostředí a jeho podmínek.

Implementovanou aplikaci je možné použít při sledování osob k odhadu jejich výšky, a usnadnit tak dopadení pachatele trestného činu nebo nalezení hledané osoby. Využití je však omezeno přítomností vhodných podmínek, jinak je odhad spíše orientační.

Možným pokračováním práce je vylepšení rychlosti detektoru a přesnosti označení osob sledováním pohybu, díky tomu by nebylo nutné skenovat detektorem každý snímek a bylo by možné korigovat odchylky způsobené chůzí.

Literatura

[1] Army, U. S. G.: *Human Engineering Design Data Digest*. Createspace Independent Pub, 2013, ISBN 9781493603978.

[2] Berezin, V.: Active pixel sensor with mixed analog and digital signal integration. Online, Listopad 21 2006, [cit. 18. 12. 2016].
URL https://www.google.com/patents/US7139025

[3] Canny, J.: A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, , č. 6, 1986: s. 679–698.

[4] Dalal, N.; Triggs, B.: Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, ročník 1, IEEE, 2005, s. 886–893.

[5] Davies, E. R.: *Computer and machine vision*. Waltham, Mass.: Elsevier, Čtvrté vydání, 2012, ISBN 9780123869081.

[6] Droogenbroeck, M. V.; Barnich, O.: ViBe - a powerful technique for background detection and subtraction in video sequences. Online, 2014, [cit. 11. 1. 2017].
URL http://www.telecom.ulg.ac.be/research/vibe/

[7] Elgammal, A.; Harwood, D.; Davis, L.: Non-parametric model for background subtraction. *Computer Vision—ECCV 2000*, 2000: s. 751–767.

[8] Enzweiler, M.; Gavrila, D. M.: Monocular pedestrian detection: Survey and experiments. *IEEE transactions on pattern analysis and machine intelligence*, ročník 31, č. 12, 2009: s. 2179–2195.

[9] Fried, L.: PIR Motion Sensor. Online, 2014, [cit. 20. 4. 2017].
URL https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/

[10] Gavrila, D. M.; Munder, S.: Multi-cue pedestrian detection and tracking from a moving vehicle. *International journal of computer vision*, ročník 73, č. 1, 2007: s. 41–59.

[11] Geronimo, D.; Lopez, A. M.; Sappa, A. D.; aj.: Survey of Pedestrian Detection for Advanced Driver Assistance Systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 32, č. 7, July 2010: s. 1239–1258, ISSN 0162-8828, doi:10.1109/TPAMI.2009.122.

[12] Giles, E.; Vallandigham, P. H.: Height estimation from foot and shoeprint length. *Journal of Forensic Science*, ročník 36, č. 4, 1991: s. 1134–1151.

[13] Kruegle, H.: *CCTV Surveillance*. Elsevier Butterworth Heinemann, druhé vydání, 2006, ISBN 0750677686.

[14] Lewis, P.: You're being watched: there's one CCTV camera for every 32 people in UK. Online, 2011, [cit. 10. 5. 2017].
URL https://www.theguardian.com/uk/2011/mar/02/cctv-cameras-watching-surveillance

[15] Litwiller, D.: CMOS vs. CCD: Maturing Technologies, Maturing Markets. *Photonics Spectra*, ročník 39, č. 8, 2005: s. 54–61.

[16] OpenCV, T.: About OpenCV. Online, 2016, [cit. 2. 5. 2017].
URL http://opencv.org/about.html

[17] Pietikäinen, M.: Local binary patterns. *Scholarpedia*, ročník 5, č. 3, 2010: str. 9775.

[18] Ren, H.; Li, Z.-N.: Object detection using boosted local binaries. *Pattern Recognition*, ročník 60, 2016: s. 793 – 801, ISSN 0031-3203.

[19] Rubinstein, J.: CCD vs CMOS. Online, 2013, [cit. 9. 2. 2017].
URL
http://www.digitalbolex.com/wp-content/uploads/2013/06/CCD-VS-CMOS.jpg

[20] Sonka, M.; Hlavac, V.; Boyle, R.: *Image Processing, Analysis, and Machine Vision*. CENGAGE LEARNING, Čtvrté vydání, 2014, ISBN 1133593607.

[21] Viola, P.; Jones, M.: Rapid object detection using a boosted cascade of simple features. 2001.

[22] Wang, X.; Han, T. X.; Yan, S.: An HOG-LBP human detector with partial occlusion handling. In *2009 IEEE 12th International Conference on Computer Vision*, IEEE, 2009, s. 32–39.

[23] Woodhouse, A.: History of CCTV. Online, 2016, [cit. 7. 4. 2017].
URL http://www.maplin.co.uk/history-of-cctv

[24] Xu, L.-Q.; Landabaso, J. L.; Pardàs, M.: Shadow removal with blob-based morphological reconstruction for error correction. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on*, ročník 2, IEEE, 2005, s. ii–729.

[25] Zhang, L.; Wu, B.; Nevatia, R.: Pedestrian detection in infrared images based on local shape features. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, IEEE, 2007, s. 1–8.

Příloha A

Obsah CD

src Zdrojové kódy programu.

bin Přeložené spustitelné soubory.

video Ukázková videa pro vyzkoušení aplikace.

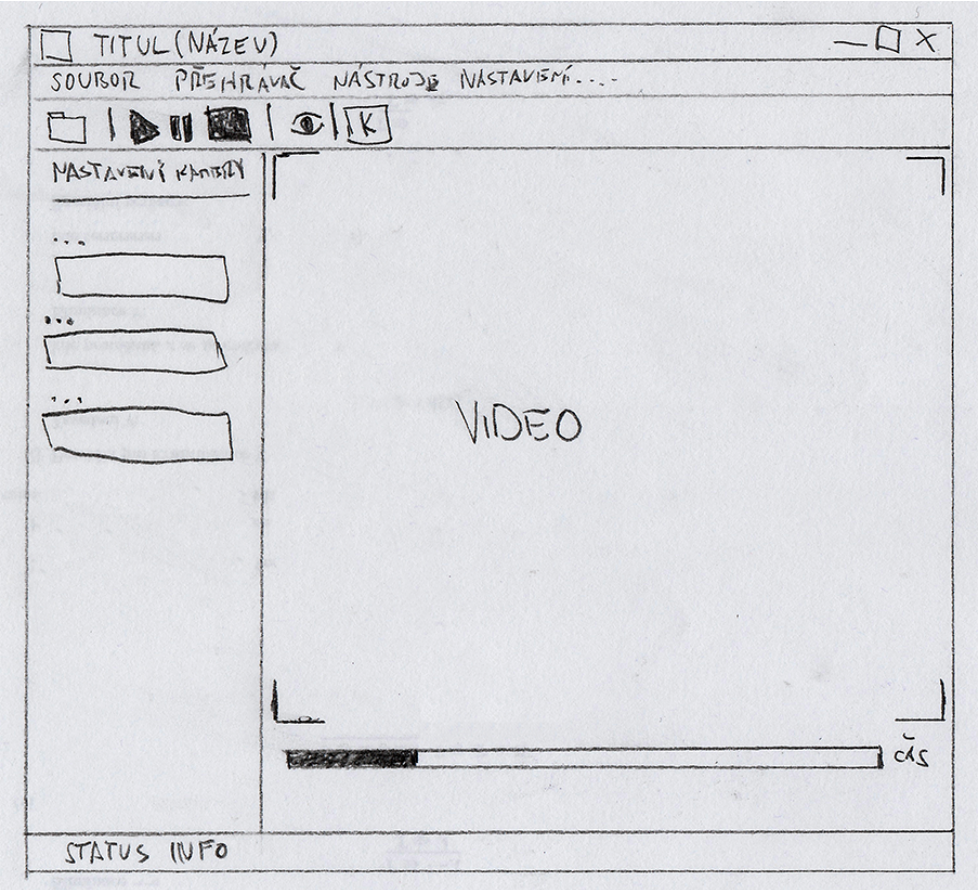
report Tento text ve formátu PDF.

src Zdrojové texty v LaTeXu a použité obrázky

Příloha B

Obrázky

Obrázek B.1: Návrh uživatelského rozhraní.



Příloha C

Tabulky

Tabulka C.1: Naměřené hodnoty odhadů výšky a vzdálenosti v prvním videu pro vybrané správně označené siluety.

Odhad vzdálenosti [m]	Odhad výšky [m]	Váha	Chyba odhadu výšky [%]
4,10	1,67	0,40	12,57
4,87	1,82	0,40	4,71
5,20	1,84	0,40	3,66
5,64	1,92	1,00	0,52
5,68	1,96	1,00	2,62
6,13	2,06	0,80	7,85
6,49	2,00	0,80	4,71
7,22	2,12	0,80	10,99
7,46	2,08	0,80	8,9
7,54	2,01	0,80	5,24
8,15	2,06	0,80	7,85
8,68	2,09	0,80	9,42
9,04	2,05	0,80	7,33
9,08	2,03	0,80	6,28
9,45	2,02	0,80	5,76
9,77	1,99	0,80	4,19
9,81	1,97	0,80	3,14
10,05	1,93	1,00	1,05
10,17	1,89	0,60	1,05
10,25	1,89	0,60	1,05
10,30	1,88	0,60	1,57
10,38	1,89	0,60	1,05
10,54	1,87	0,60	2,09
10,54	1,91	0,60	0
10,58	1,88	0,60	1,57
10,98	1,79	0,60	6,28
11,15	1,72	0,60	9,95
11,19	1,75	0,60	8,38
11,55	1,62	0,60	15,18