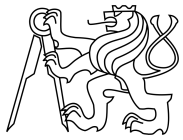
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Detekce a sledování pohybu osob v kamerových záznamech
Student:	Bc. Jan Kozák
Vedoucí:	doc. RNDr. Ing. Marcel Jiřina, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

Cílem práce je navrhnout a implementovat SW aplikaci, která umožní detekovat a následně sledovat osoby v kamerových záznamech z kamer, jejichž pohledy se mohou částečně překrývat. Výstupem aplikace jsou strukturované datové údaje o pohybu osob v prostoru a čase. Na základě dat budou následně stanoveny trajektorie jednotlivých osob a souhrnné statistiky.
1) Seznamte se s úlohou sledování osob v kamerových záznamech a proveďte rešerši stávajících metod, včetně vhodných SW knihoven.
2) Navrhněte vlastní robustní SW aplikaci s využitím dostupných SW knihoven, která umožní vyhodnocovat kamerové snímky a získané údaje převádět do podoby strukturovaných dat pro další zpracování.
3) Navrženou aplikaci implementujte ve vhodném programovém prostředí. Využijte knihovnu OpenCV a další SW z bodu 1.
4) Ověřte funkčnost aplikace na reálných datech.
5) Dosažené výsledky vyhodnoťte a diskutujte výhody a nevýhody zvoleného přístupu.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 16. února 2017

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Diplomová práce

Detekce a sledování pohybu osob v kamerových záznamech

Bc. Jan Kozák

Katedra softwarového inženýrství
Vedoucí práce: doc. RNDr. Ing. Marcel Jiřina, Ph.D.

9. ledna 2018

Poděkování

V první řadě děkuji svému vedoucímu práce doc. RNDr. Ing. Marcelu Jiřinovi, Ph.D. za cenné připomínky, konzultace a trpělivost. Také děkuji Ing. Jakubu Novákovi za poskytnutí hardwaru a softwaru k realizaci práce, a za rady s implementací. Dále děkuji Tomáši Chládovi, Ing. Dagmar Malé, Romaně Skořepové, Ivaně Dolejšové, Markétě Loučkové a Petře Stolejdové za pomoc s natáčením datasetů, s pomocí kterých vyhodnocuji úspěšnost implementovaného systému. V neposlední řadě děkuji své rodině za účinkování v datasetu z interiéru, a samozřejmě za trpělivost a podporu.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 9. ledna 2018 .....

Odkaz na tuto práci

Kozák, Jan. *Detekce a sledování pohybu osob v kamerových záznamech.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Analytická část této práce nabízí úvod do oblasti detekce a sledování osob. Nejčastěji využívané metody pracující s jednou a více statickými kamerami jsou kategorizovány a popsány. V části Návrh a Implementace je představena více-kamerová metoda detekce osob, která využívá geometrických vlastností scény k omezení vlivu částečných a úplných okluzí na úspěšnost detekce. Pomocí maďarského algoritmu jsou detekce přiřazovány k trajektoriím, jejichž stav je predikován Kalmanovým filtrem.

Klíčová slova detekce osob, sledování osob, více-kamerová detekce osob, separace popředí, projektivní geometrie, Kalmanův filtr, maďarský algoritmus

Abstract

The analytical part of this thesis offers an introduction to the field of person detection and tracking. The most commonly used methods utilizing single and multiple static cameras are categorized and described. The proposed tracking algorithm starts with a multi camera detection approach that uses the geometric properties of the scene to limit the effects of partial and complete occlusions. Hungarian algorithm is used to assign detections to the track positions estimated by the Kalman filter.

Keywords person detection, person tracking, multi camera person detection, background subtraction, projective geometry, Kalman filter, Hungarian algorithm

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Detekce a sledování osob v kamerových záznamech se zabývá využitím jedné či více kamer k rozpoznávání osob a sestavování dlouhodobých trajektorií pohybu. Potřeba automatického sledování se zvyšujícím se počtem kamer ve veřejném i soukromém prostoru stále roste. Například jen v Praze je rozmístěno přes 4500 kamer [1]. Je zřejmé, že k monitorování takové oblasti není možné použít lidský dozor. Kromě monitorování a ostrahy objektů nalézají metody detekce a sledování uplatnění například ve sportu, robotice a stále častěji také v oblasti samořiditelných vozidel.

Problém detekce a sledování osob je v ideálních podmínkách poměrně snadno řešitelný i s pomocí jediné kamery. Problém nastává, pokud se osoby překrývají – na scéně dochází k okluzím. Z toho důvodu jsou stále častěji zkoumány více-kamerové metody, které jsou schopné zaznamenat více informací o scéně.

Cílem mojí práce je tedy navrhnout algoritmus, který s využitím více kamer dokáže sledovat trajektorie několika osob.

První kapitola uvádí a diskutuje cíle práce. Ve druhé kapitole čtenáře krátce seznamuji s vybranými pracemi na téma detekce a sledování osob, které s mojí prací souvisí. Třetí kapitola představuje přehled nejběžnějších metod používaných zvlášť k detekci a ke sledování osob. Čtvrtá kapitola potom detailně popisuje metody, které jsem se rozhodl implementovat. Zvláštní důraz je věnován metodám adaptivní separace popředí. Pátá kapitola krátce shrnuje implementaci a v šesté kapitole popisuji, jakým způsobem jsem práci testoval.

Kapitola 1

Cíle práce

1.1 Zadání

1. Seznamte se s úlohou sledování osob v kamerových záznamech a proveďte rešerši stávajících metod, včetně vhodných SW knihoven.
2. Navrhněte vlastní robustní SW aplikaci s využitím dostupných SW knihoven, která umožní vyhodnocovat kamerové snímky a získané údaje převádět do podoby strukturovaných dat pro další zpracování.
3. Navrženou aplikaci implementujte ve vhodném programovém prostředí. Využijte knihovnu OpenCV a další SW z bodu 1.
4. Ověřte funkčnost aplikace na reálných datech.
5. Dosažené výsledky vyhodnoťte a diskutujte výhody a nevýhody zvoleného přístupu.

1.2 Diskuze zadání

Detekce a sledování osob v kamerových záznamech představuje se všemi možnostmi rozvržení scény, dostupnými algoritmy detekce a sledování velmi široké téma, obzvlášť z hlediska implementace. Proto jsme se s vedoucím práce domluvili, že aplikace, která bude výsledkem této práce, bude používat více kamer sledujících stejnou scénu. Nebude se snažit hledat korespondence mezi osobami, které scénu opustily a pak na ní zase přišly, a nebude nutně fungovat v reálném čase. Dále předpokládáme, že monitorovaná scéna je plně pod naší kontrolou, můžeme na ní provést libovolná měření, kamery rozmístit jakýmkoliv způsobem, nebo na scénu umístit objekty, pomocí nichž se provádí kalibrace kamer.

Kapitola 2

Související práce

V této kapitole uvádím stručný popis prací, které buď zmiňuji v kapitole 3, nebo ze kterých vycházím při návrhu aplikace. Jejich přístupy lze rozdělit na jedno-kamerové a více-kamerové.

2.1 Jedno-kamerové přístupy

Způsob fungování jedno-kamerových metod detekce je vysvětlen na známé práci P. Violy a M. Jonese [2], která se sice věnuje rozpoznávání obličejů, ale navržený postup lze zobecnit na detekci obecných objektů [3]. Vstupní obrázek je nejprve popsán Haarovými příznaky (*Haar-like features*), z nichž je pro zvýšení výkonu sestaven integrální obrázek. K detekci je použit klasifikátor AdaBoost natrénovaný na příslušný typ objektů.

N. Dalal a B. Triggs [4] popisují obrázek histogramem orientovaných gradientů (*HOG*). Příznaky poté klasifikují pomocí SVM.

Moderní metody jedno-kamerové detekce objekty klasifikují stále pomocí AdaBoostu nebo SVM, jako příznaky jsou ale nejčastěji použity Integral Channel Features [5], [6], [7] – kombinace několika sad příznaků, které jsou uložené ve formě integrálního obrázku.

G. Shu a kol. [8] se kromě detekce věnují i sledování. Detekci řeší pomocí HOG a SVM, které jsou pro efektivnější řešení okluzí natrénované k rozpoznávání částí lidského těla (*part-based model*). Korespondující detekce z jednotlivých snímků jsou spojovány v trajektorie. Míra korespondence závisí na podobnosti HOG příznaků, pozici a velikosti.

L. Zhu a kol. [9] používají více kamer, jejich pohledy se ale nepřekrývají. Osoby jsou detekovány separací popředí, sledování v pohledu jedné kamery je pak řešeno přiřazením Kalmanova filtru ke každé detekci. Stav Kalmanova filtru se skládá z pozice, rychlosti a velikosti detekce.

W. Nie a kol. v [10] sestavují krátkodobé stopy, které spojují do kompletních trajektorií hledáním korespondencí mezi grafy reprezentujícími rozmístění osob na posledním snímku předchozí stopy a na prvním snímku nové stopy.

2.2 Více-kamerové přístupy

Se vzrůstajícím počtem osob na scéně roste i náročnost přesného sledování osob, a to zejména v důsledku okluzí – částečných nebo úplných překrytí sledované osoby. Více-kamerové metody dokáží ze své podstaty zachytit více informací o scéně, nabízejí tedy robustnější řešení.

S. M. Khan a M. Shah [11] nejprve vypočítají homografie referenční roviny a rovnoběžných virtuálních rovin mezi jednotlivými pohledy. Na všechny vstupní snímky aplikují separaci popředí, pro každou rovinu zvlášť poté promítnou popředí příslušnými homografiemi do zvoleného referenčního pohledu. Tím na referenčním pohledu vzniknou bílá místa reprezentující osoby. Referenční pohledy z navazujících časových kroků seskupí do dávky, spojí jejich bílá místa a pomocí hledání maximálního toku v síti sestaví trajektorie pohybu osob. Dávky snímků sestavují tak, aby nedocházelo ke ztrátám trajektorie.

D. Arsić a kol. [12] detekují osoby podobným způsobem, jako [11], popředí jednotlivých pohledů ale místo do referenčního pohledu převádějí na půdorys scény. Sledování řeší pomocí Kalmanova filtru. Metoda D. Arsiće a ostatních má nejblíže k postupu navrženému v této práci.

M. Liem a D. Gavrila [13] používají separovaná popředí k vytvoření 3D rekonstrukce scény. Ke sledování je opět použit Kalmanův filtr.

2.3 Ostatní práce

Separace popředí představuje důležitou samostatnou úlohu. V této práci je použita metoda představená Z. Zivkovicem [14], která modeluje každý pixel scény několika normálními rozděleními. Zivkovic zobecňuje metodu Ch. Stauffera a W.E.L Grimsona [15], kde bylo nutné explicitně nastavit počet normálních rozdělení.

Oblast detekce a sledování osob zahrnuje rozsáhlou množinu různých problémů. Pro úplnost uvádím problémy a přístupy, které ze své podstaty sice souvisejí s detekcí a sledováním osob, ale které jsou natolik vzdálené od problémů řešených v této práci, že se jim nebudu dále věnovat.

V pracích řešících monitorování členitých oblastí a rozsáhlých prostor se také používá více kamer [9], jsou rozmístěny tak, aby pokryly co největší oblast. Tyto práce typicky vychází z existujících metod pro sledování jediné scény a snaží se je rozšířit tak, aby dokázaly sledovat několik scén současně. Hlavní úlohou takových systémů je re-identifikace osoby, která přejde z jedné scény do jiné.

Díky dostupnosti levných RGB-D kamer (například Microsoft Kinect) vzniká řada prací, které k detekci využívá kombinaci obrazových informací a hloubkové mapy [16].

Zajímavou aplikací sledování jsou systémy určené pro automatické monitorování pohybu osob s využitím pohyblivých PTZ (*pan tilt zoom*) kamer

[17]. Tyto systémy automaticky řídí natočení a zoom kamery podle polohy sledovaného objektu. PTZ kamery jsou také kombinovány se statickými, přehledovými kamerami [18]. Ty snímají celou scénu a vyhledávají potencionální objekty zájmu.

Dalším široce zkoumanou oblastí je stereoskopické rozložení kamer. To nachází využití zejména v robotice a stále častěji také v oblasti samořiditelných vozidel. Díky tomu lze předpokládat, že se využití stereoskopických kamer ještě rozšíří. Stereoskopické rozmístění kamer ale samozřejmě neposkytuje tolik informací o scéně, jako rozmístění s větší vzdáleností (úhlem) mezi kamerami.

Kapitola 3

Analýza

Problém detekce a sledování osob je nejčastěji řešen ve dvou samostatných krocích. Na vstupních videozáznamech jsou nejprve detekovány osoby. Jejich poloha a případné další požadované informace jsou poté předány sledovacímu algoritmu, jehož výstupem jsou trajektorie pohybu jednotlivých osob.

V této sekci uvádím neformální popis vybraných algoritmů, které se k řešení obou hlavních problémů používají.

3.1 Počet a umístění kamer

Počet a rozmístění kamer má zásadní vliv na výsledný výkon systému, volbu vhodných algoritmů a samozřejmě na cenu a náročnost nasazení navrženého systému.

V případě použití jediné kamery je situace jednoduchá. Kameru je třeba umístit tak, aby měla co nejlepší přehled o scéně. Pokud bude kamera příliš blízko, bude zabírat pouze část osob, což lze chápat jako okluze s okolním prostředím a povede k vynechání detekcí. Podle P. Dollára a kol. [19] pracují současné jedno-kamerové detektory nejlépe se snímky, ve kterých jsou osoby vyšší, než 100 pixelů. Je tedy vhodné zvolit odpovídající kombinaci rozlišení kamery a vzdálenosti od scény.

Hlavním důvodem použití více kamer je robustnější řešení okluzí. S rostoucím počtem kamer roste i pravděpodobnost, že v jednom z pohledů k okluzi nedochází, viz obrázek 3.1. Podle srovnání M. Liema a D. Gavrily [13] pracuje většina více-kamerových systémů s 2 – 5 kamerami.

V kombinaci s požadavky na umístění jediné kamery se většinou volí rozmístění kamer v rozích sledované scény tak, aby sousední kamery svíraly úhel 90° se středem v centru scény. Níže popsané metody však nevyžadují konkrétní rozmístění, to se tedy odvíjí převážně od možností pozorované scény.

3.2 Detekce

Algoritmy detekce řeší problém nalezení osob v pohledu kamery, případně pohledech několika kamer. Poměrně jednoduchá úloha se výrazně komplikuje, pokud na scéně dochází k okluzím (*occlusion*) – překryvům osob s okolním prostředím, případně překryvům osob navzájem.

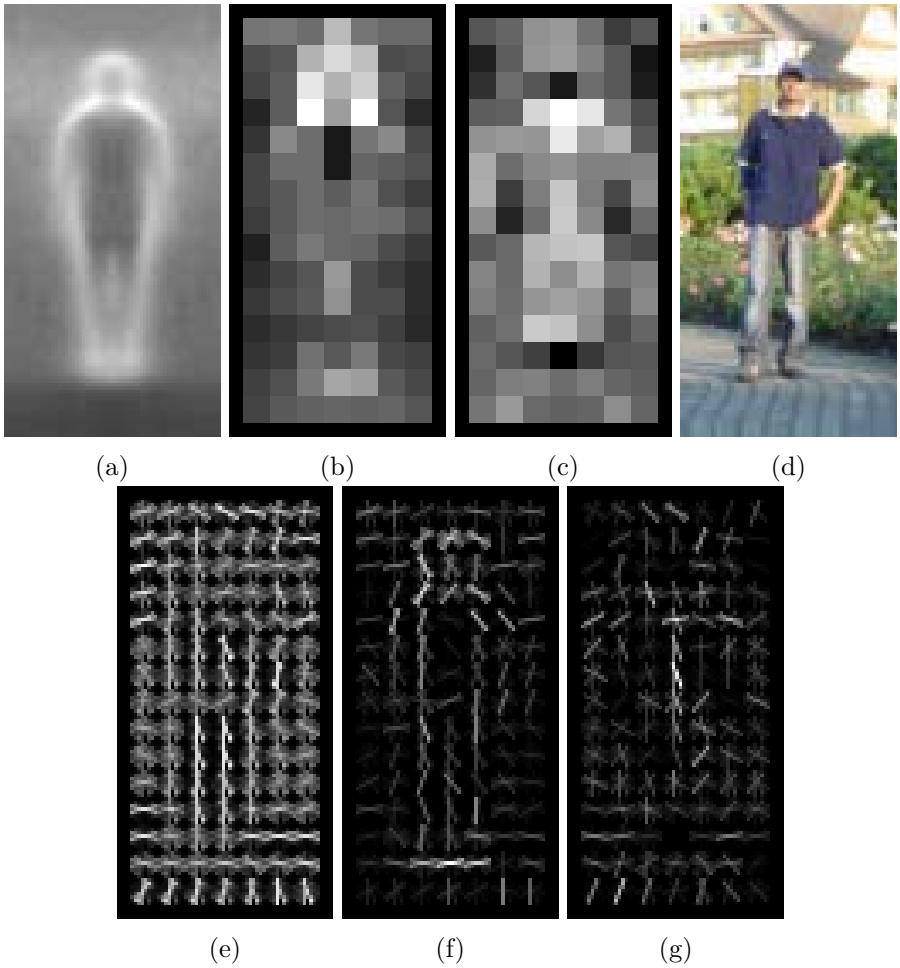
(a) (b)

Obrázek 3.1: Okluze z pohledu dvou kamer

Hlavní úlohou současného výzkumu je omezení vlivu okluzí při přijatelné míře falešných detekcí. Z obrázku 3.1 je patrné, že k okluzím nemusí docházet ve všech pohledech najednou. Toho se snaží využít více-kamerové metody. V situacích, ve kterých je nutné použít jedinou kameru, se používají pokročilé modely člověka, nebo se extrapolují výsledky sledování.

3.2.1 Detekce klasifikací příznaků

Detekce pomocí klasifikace příznaků (*features*) je v podstatě aplikací obecného rozpoznávání objektů na problém detekce osob. Spočívá v popisu snímku příznaky (*feature description*) a jejich klasifikaci. Existuje spousta druhů deskriptorů i klasifikátorů, princip metody ale zůstává stejný (viz zhodnocení state-of-the-art detektorů v práci P. Dollára a kol. [19]).



Obrázek 3.2: **Detekce pomocí HOG + SVM** (a) Průměrný gradientní obrázek z trénovací množiny. (b) Intenzita pixelů zobrazuje maximální kladnou (b), případně zápornou (c) váhu SVM v daném bloku. (d) Vstupní obrázek. (e) Odpovídající deskriptor R-HOG. (f, g) Deskriptor R-HOG vážený kladnými a zápornými váhami SVM. Obrázky převzaty z [4].

Vstupní obrázek je postupně zpracováván posuvným okénkem. Na každé vybrané oblasti detektor vypočítá příznaky, které tvoří vstup předem naučeného klasifikátoru. Ten rozhoduje, jestli tyto příznaky reprezentují člověka. Jelikož se osoby mohou nacházet v různých vzdálenostech od kamery, nemůžeme obecně předem určit jejich očekávanou velikost. Proto jsou při zpracování použita posuvná okénka různých velikostí. Na oblasti klasifikované jako osoby se často ještě používá technika potlačení duplicitních detekcí (*non-maximum suppression*), která zabraňuje vícenásobné detekci stejné osoby v sousedních oblastech.

K popisu obrázku se používají např. Haarovy příznaky [2] nebo histogram

Obrázek 3.3: Part-based model. Převzato z [20].

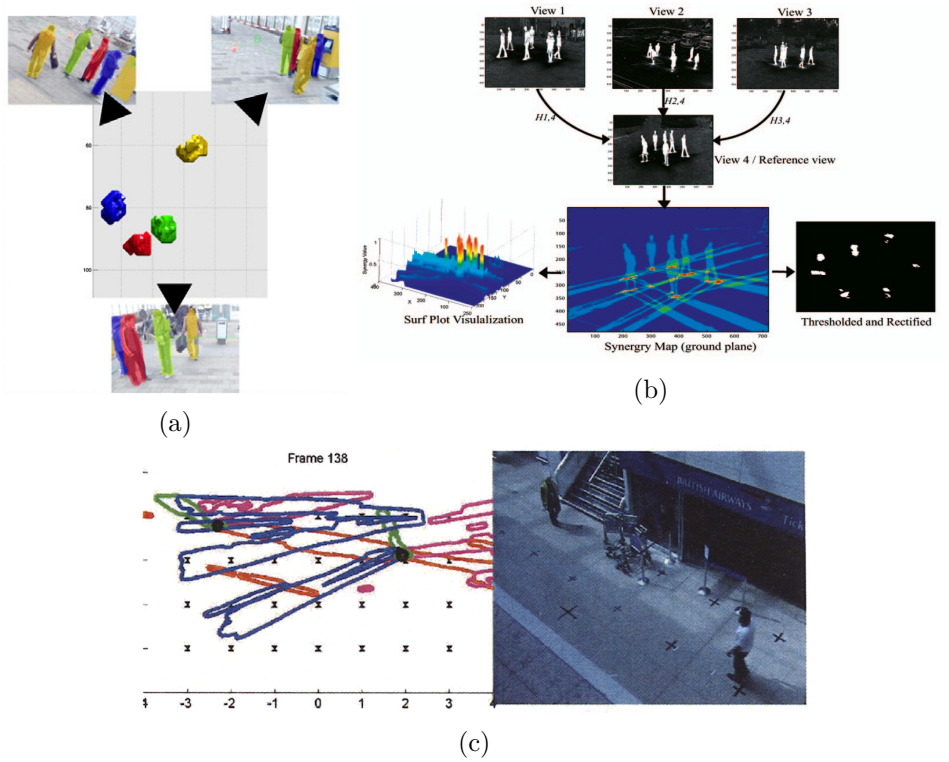
orientovaných gradientů [4]. Středem zájmu současného výzkumu je popis pomocí Integral Channel Features [5] a jeho variantami [7] [6]. Jde o vícedimenzionální deskriptor – pro vstupní obrázek je vypočítáno několik typů příznaků, které jsou uloženy jako samostatné kanály, v rámci zvýšení efektivity ve formě integrálního obrázku. P. Dollár v [5] používá jako kanály například původní obrázek převedený do barevného prostoru LUV nebo histogram a intenzitu gradientního obrázku.

Klasifikace je nejčastěji řešena pomocí AdaBoostu [2] nebo SVM [4] natrénovaných na pozitivních a negativních vzorcích příznaků použitého deskriptoru. Obrázek 3.2 (e) ukazuje všechny detekované příznaky, 3.2 (f) pak zobrazuje intenzitu příznaků upravenou váhami SVM.

Klasifikátory mohou být naučeny k detekci buď člověka jako celku, nebo hlavních částí lidského těla. Takzvaný part-based model reprezentuje osoby jako kolekci tělesných částí, které jsou na snímcích vyhledávány odděleně. Pokud se detekované části nacházejí v očekávaném počtu a přípustné pozici, je jejich kombinace označena za člověka. Za člověka je možné označit i určitou podmnožinu částí, díky tomu se part-based modely lépe vypořádávají s okluzemi. Vzájemná poloha tělesných částí navíc není pevně daná, což zvyšuje úspěšnost při detekci osob v nestandardních polohách. Zřejmou nevýhodou je delší výpočetní čas, protože tělesné části jsou detekovány odděleně. Model člověka složený z několika částí ukazuje obrázek 3.3.

Metody dostupné v roce 2012 porovnává P. Dollár a kol. v [19].

Výše uvedené přístupy se v současnosti používají téměř výhradně v jednokamerových systémech.



(a) (b) (c)

Obrázek 3.4: **Příklady detekce slučováním pohledů** (a) 3D rekonstrukce scény (b) Projekce do referenčního pohledu (c) Projekce do půdorysu scény. Obrázky převzaty z [13], [11], [12].

3.2.2 Detekce slučováním informací z několika pohledů

Detekce pomocí sloučení informací z několika pohledů spočívá v nalezení oblastí zájmu v každém pohledu a jejich sloučení do společné reprezentace, na níž jsou poté lokalizovány osoby. Jak název napovídá, tyto metody vyžadují více kamer.

Detekce klasifikací příznaků uvedená v předchozí sekci je v podstatě jedním ze způsobů hledání oblastí zájmu. Z praktického pohledu ale zamítá až příliš možných oblastí a měla by negativní dopad na výkon detekce. Proto se téměř výhradně používá tolerantnější a často výpočetně jednodušší separace popředí. Vzhledem k možným změnám osvětlení a statických objektů na scéně se používají adaptivní modely pozadí. Separace popředí představuje rozsáhlou oblast prací a přístupů, blíže se jí věnuji v sekci 4.3.1.

Metody slučování jsou naopak specifické pro každou práci. S. M. Khan a M. Shah [11] s pomocí homografií několika rovin mezi jednotlivými pohledy slučují popředí do referenčního pohledu. Obrázek 3.4 (b) ukazuje sloučení pomocí jedné roviny. Lokalizace je zde řešena současně se sledováním. D. Arsić

a kol. používají podobnou metodu, popředí jsou ale projektována do půdorysu scény. Průniky na půdorysu představují detekce, viz obrázek 3.4 (c). M. Liem a D. Gavrila z popředí rekonstruují 3D model scény, objekty jsou detekovány porovnáváním s predikcí pohybu osob detekovaných v předchozích časových krocích[1].

3.3 Sledování

Úlohou sledování je najít vztahy mezi detekcemi z jednotlivých časových kroků a sestavit z nich dlouhodobou trajektorii. Vstup sledovacího algoritmu tvoří kolekce detekcí z aktuálního časového kroku. Na rozdíl od detekce se sledovací algoritmy pro jedno-kamerové a více-kamerové metody nijak výrazně neliší. Rozdíly se nejčastěji objevují ve formě vstupních dat, kdy sledovací algoritmy některých více-kamerových metod provádějí současně i lokalizaci [11].

Existují dva základní přístupy ke sledování: rekurzivní a dávkové. Rekurzivní sledování spojuje detekce a trajektorie snímek po snímku, je vhodné pro systémy pracující v reálném čase. Dávkové sledování optimalizuje tvar trajektorií v dávce detekcí z několika časových kroků, za cenu vyšší výpočetní náročnosti je tedy schopné dosáhnout lepších výsledků.

3.3.1 Rekurzivní sledování

Rekurzivní přístup znamená, že osoby jsou sledovány pouze na základě informací z předchozích snímků. Typickými přístupy jsou hledání korespondencí mezi detekcemi (*tracking by detection*), použití Kalmanova filtru nebo částicového filtru. Zatímco první metoda spočívá ve spojování detekcí do trajektorií, obě později jmenované využívají rekurzivní bayesovský filtr – na základě předchozích stavů odhadují stav aktuální, který se poté porovnává s výstupem z detektoru. Kalmanův a částicový filtr jsou tedy schopny udržovat stopu osob, které nebyly v aktuálním snímku detekovány.

3.3.1.1 Hledání korespondencí mezi detekcemi

Jedná se o nejjednodušší způsob sledování, (*tracking by detection*). Detekce z aktuálních snímků jsou přiřazovány k detekcím z předchozích snímků, tak po-

[1] V případě více-kamerových metod je pojem 'snímek' nejednoznačný, 'časový krok' tedy používám pro všechny snímky nebo detekce pořízené ve stejném okamžiku.

stupně vznikají trajektorie [8]. Tento přístup vlastně formuluje úlohu sledování jako problém přiřazení. Základním řešením je přiřazení na základě vzdálenosti. Pro každou novou detekci je nalezena nejbližší trajektorie, ke které se přiřadí. Pokud se v blízkosti žádná trajektorie nenachází, je vytvořena nová. Přiřazení nalezená touto metodou však nemusí být optimální, což může vést k prohození identit (obrázek 3.5) nebo přiřazení falešné detekce. Řešením je použití maďarského algoritmu (*Hungarian algorithm*), který problém řeší v čase $O(n^3)$, kde $n = max(|detekce|, |trajektorie|)$, a který vždy nalezne optimální řešení.

(a) (b) (c)

Obrázek 3.5: Prohození identity

Přiřazení nemusí probíhat pouze na základě vzdálenosti. Podle informací, které detektor poskytuje, mohou být detekované osoby přiřazovány pomocí výšky, vzhledu, objemu, který na scéně zabírají, už zmiňované pozice, nebo kombinace všech uvedených parametrů. Některé metody pracují i s modelem sociálního chování [21]. Použitím kombinace parametrů vzniká více-dimenzionální problém přiřazení. Ten je možné buď linearizovat pomocí cenové funkce, nebo aproximovat vhodným algoritmem [22]. Zrychlení je možné dosáhnout tím, že v jednoznačných situacích bude přiřazení probíhat pouze na základě vzdálenosti a ostatní informace se použijí pouze v nejednoznačných případech.

3.3.1.2 Estimace aktuální pozice

Estimace, podobně, jako předchozí metoda, přiřazuje nové detekce ke stávajícím trajektoriím. Trajektorie ovšem nejsou reprezentovány detekcí z posledního časového kroku, jejich stav je modelován Kalmanovým nebo částicovým filtrem.

Po získání detekcí z aktuálního časového kroku je *predikován* stav každé trajektorie. Detekce jsou přiřazeny k predikcím pomocí algoritmů uvedených v sekci 3.3.1.1. Výsledky predikce jednotlivých filtrů jsou poté *korigovány* skutečným stavem přiřazené detekce.

Stav obou estimátorů je možné modelovat řadou parametrů. Těmi základními jsou pozice a rychlost osoby [12]. Dalším často sledovaným parametrem je například velikost odpovídajícího binárního blobu [9].

Hlavním rozdílem mezi Kalmanovým a částicovým filtrem je typ rozdělení, které oba estimátory dokážou odhadnout. Kalmanův filtr předpokládá normální rozdělení odhadované pravděpodobnostní funkce, oproti tomu částicový filtr dokáže odhadnout i multimodální pravděpodobnostní rozdělení (rozdělení s několika vrcholy). Volba vhodného estimátoru tedy závisí převážně na předpokládaném způsobu pohybu osob. Chodci se po ulici pohybují předvídatelně, za konkrétním cílem a bez náhlých změn směru. Takový případ nahrává použití Kalmanova filtru. Naopak ve sportovním utkání lze očekávat náhodný pohyb s častými změnami rychlosti i směru. Pro podobné případy je vhodný částicový filtr.

3.3.2 Dávkové sledování

Dávkové sledování zpracovává detekce z několika časových kroků najednou, hledá v nich nejpravděpodobnější trajektorie pohybu osob. Hlavní výhodou je možnost pohybu vpřed v čase – je tak snazší ověřit, jestli sledovaná osoba opustila scénu, nebo jestli došlo pouze k chybě detekce. Potenciálně tedy může dosáhnout lepších výsledků, než rekurzivní sledování.

Příkladem dávkového zpracování je metoda S. M. Khana a M. Shaha [11], která úlohu převádí na problém hledání maximálního toku v grafu. Detekce v dávce 15 snímků představují uzly propojené hranami. Hranám je přiřazena cena vypočítaná z prostorové a časové vzdálenosti uzlů, které spojují.

Počet časových kroků v dávce závisí na požadovaných vlastnostech sledovacího systému, vyšší hodnota vede k souvislejším trajektoriím, spolu s ní ale vzrůstá i výpočetní náročnost.

Kapitola 4

Návrh

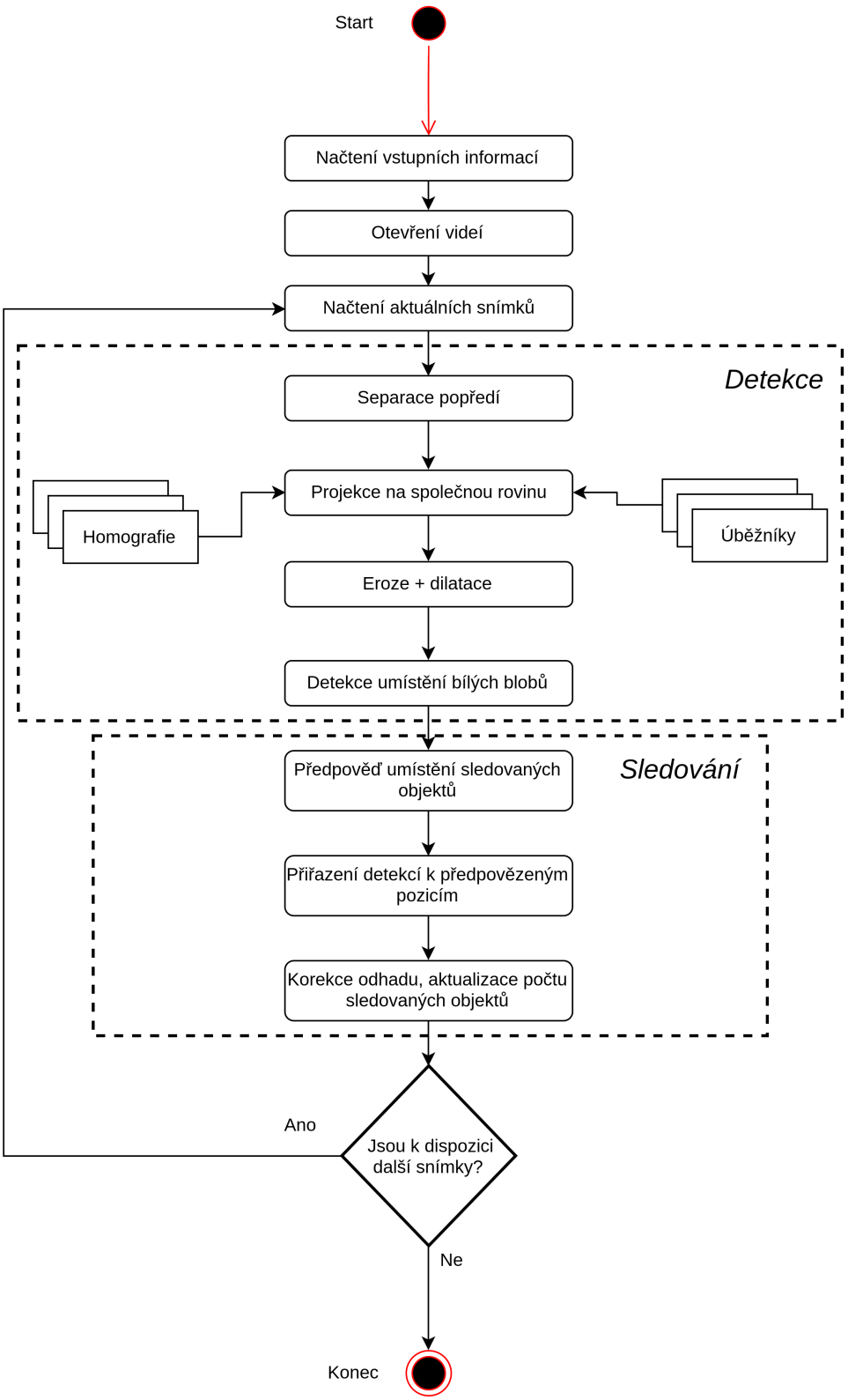
V této kapitole uvádím podrobný popis algoritmů, které používám ve své implementaci. Obrázek 4.1 ukazuje vývojový diagram navrhované metody.

Jedním z předpokladů je, že nad scénou máme plnou kontrolu. Volím tedy rozmístění kamer v rozích scény, které umožňuje získat o scéně maximum informací.

K detekci využívám upravený algoritmus představený v [11] a [12]. Jde o lokalizaci projekcí všech pohledů na referenční rovinu, případně do rovin s ní rovnoběžných. Metoda nevyžaduje synchronizaci ani kalibraci kamer, nepoužívá modely vzhledu ani velikosti. Pracuje pouze s geometrickými vlastnostmi scény, není proto citlivá na změny sledovaných osob. Jejím vstupem jsou popředí jednotlivých pohledů. Popředí separuji adaptivní metodou [14], která je součástí knihovny OpenCV.

Přiřazování detekcí k trajektoriím probíhá na základě vzdálenosti. Optimální přiřazení řeším vlastní implementací maďarského algoritmu.

Sledování řeším rekurzivním přístupem, konkrétně Kalmanovým filtrem. Jak je uvedeno v sekci 3.3.1, výhodou rekurzivního sledování je, že dokáže pracovat v reálném čase. Takový požadavek na systém sice neexistuje, věřím ale, že sledování v reálném čase nalezne širší uplatnění, než dávkové zpracování.



Obrázek 4.1: Vývojový diagram navržené aplikace

4.1 Načtení vstupu

Program ke svému běhu potřebuje poměrně obsáhlý seznam parametrů. Zadávání přes grafické rozhraní nebo příkazovou řádku při každém spuštění by bylo nepohodlné, proto jsou vstupní informace předávány pomocí souboru ve formátu YAML.

Každá scéna je popsána následujícími parametry:

ViewCount: Počet kamer

WorldHomographyCoords: Reálné souřadnice 4 bodů (x, y) pro výpočet homografie

WorldGroundPlaneCoords: Reálné souřadnice 4 bodů ohraničujících rovinu podlahy. Zadány relativně k souřadnicím prvního bodu homografie.

View$< i >$**HomographyCoords** 4 body pro výpočet homografie v obrazových souřadnicích pohledu i

View$< i >$**Filename** Cesta k souboru s videem pohledu i.

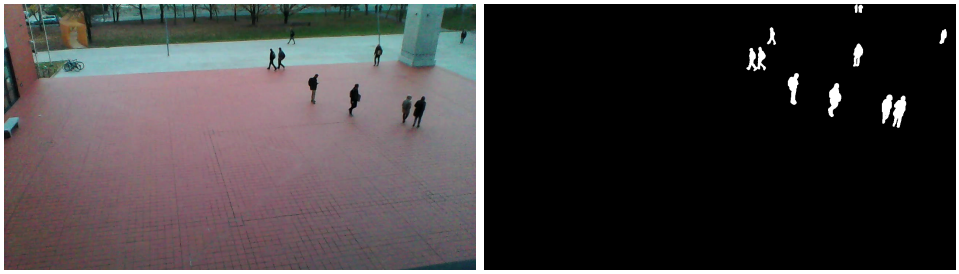
View$< i >$**DeviceId** OpenCV identifikátor kamery i. Nemůže být použit pro stejný pohled spolu s předchozím parametrem.

Příklad vstupního souboru `atrium.yml`:

```
%YAML:1.0

ViewCount: 3
WorldHomographyCoords: [0., 0., 800., 0., 800., 800.,
    0., 800.]
WorldGroundPlaneCoords: [0., 0., 800., 0., 800.,
   800., 0., 800.]
View0Filename: "/home/honza/Dokumenty/DIP/Datasets/
   Atrium/mjpeg/cam0.mkv"
View0HomographyCoords: [21., 451., 445., 552., 637.,
   352., 303., 298.]
View1Filename: "/home/honza/Dokumenty/DIP/Datasets/
   Atrium/mjpeg/cam1.mkv"
View1HomographyCoords: [630., 593., 1174., 526.,
   938., 319., 555., 346.]
View2Filename: "/home/honza/Dokumenty/DIP/Datasets/
   Atrium/mjpeg/cam2.mkv"
View2HomographyCoords: [1191., 530., 943., 312.,
   569., 331., 656., 576.]
```

Program se spouští pomocí



Obrázek 4.2: Zdrojový obrázek a ručně oddělené pozadí.

```
./personTracker atrium.yml
```

4.2 Kalibrace kamer

Navrhovaná metoda detekce toleruje mírné nepřesnosti v pozicích popředí jednotlivých snímků, chyby vznikající při předzpracování mají často větší vliv. Kamery, se kterými bude systém používán, neprodukují výrazné zkreslení, kalibrace kamer tedy nebyla implementována. V případě nasazení systému se širokoúhlými kamerami je možné systém jednoduše rozšířit o kalibraci využitím funkce `calibrateCamera()` z knihovny OpenCV.

4.3 Detekce

4.3.1 Separace popředí

Separace popředí (*background subtration*) je prvním krokem navrhovaného detekčního algoritmu. Je aplikována na každý pohled zvlášť, výstupem je binární obrázek – bílé pixely značí popředí, černé pozadí. Separací popředí dojde k oddělení pohybujících se objektů, potencionálních detekcí, od statického zbytku scény.

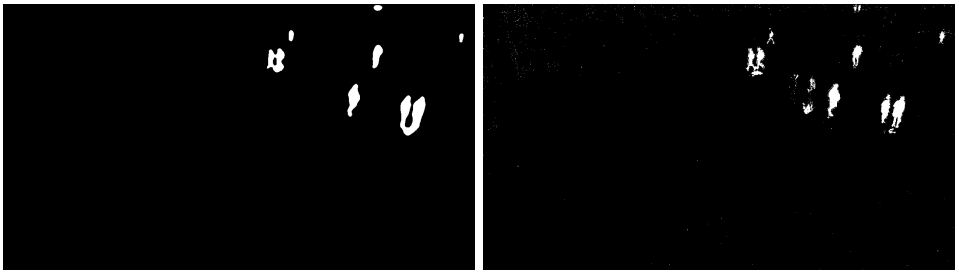
Na téma adaptivní separace popředí vznikla řada prací, vybrané algoritmy jsou implementovány v knihovně OpenCV, mnoho dalších poskytuje knihovna BGSLibrary [23].

Separace popředí bude zpracovávat každý pohled zvlášť, volba konkrétního algoritmu má tedy zásadní dopad na výkon celého sledovacího systému. Popředí detekovaná jednotlivými algoritmy se navíc výrazně liší. Nesprávná separace může zapříčinit falešnou detekci, nebo naopak ztrátu trajektorie sledované osoby. V tabulce 4.1 proto porovnávám algoritmy podle času běhu a počtu snímků za sekundu, odchylky proti manuálně vytvořené ideální separaci popředí 4.2 a podle skóre vypočteného z předchozích metrik.

Měření probíhalo na systému s parametry uvedenými v sekci 6.2.1. Výpočetní čas byl změřen pomocí std::chrono::high_resolution_clock na minu-

Algoritmus	Čas	FPS	Přesnost	Skóre
FrameDifference	12.83 s	140.21	10816	77
TwoPoints	17.50 s	102.83	10302	100
OpenCV::MOG2	20.80 s	86.53	10125	117
DPAdaptiveMedian	21.05 s	85.50	10584	123
StaticFrameDifference	12.09 s	148.85	21017	141
ViBe	23.63 s	76.16	10977	144
AdaptiveSelectiveBL	17.13 s	105.02	17056	162
WeightedMovingMean	29.77 s	60.45	11074	183
MixtureOfGaussianV2	34.30 s	52.47	10870	207
DPWrenGA	36.35 s	49.51	11074	223
DPMean	38.24 s	47.06	10816	229
VuMeter	38.41 s	46.85	11520	245
DPZivkovicAGMM	40.01 s	44.97	12238	272
AdaptiveBackgroundLearning	27.35 s	65.80	20348	309
SigmaDelta	52.60 s	34.21	12596	368
LBMixtureOfGaussians	56.84 s	31.66	12559	396
WeightedMovingVariance	64.26 s	28.00	11666	416
DPPratiMediod	80.26 s	22.42	10740	479
CodeBook	47.44 s	37.94	21003	553
MultiCue	28.53 s	63.07	36120	572
LBSimpleGaussian	37.05 s	48.57	33606	691
DPGrimsonGMM	129.09 s	13.94	12937	928
PixelBasedAdaptiveSegmenter	188.14 s	9.56	9202	962
T2FGMM_UM	170.46 s	10.55	10753	1019
T2FMRF_UM	180.62 s	9.96	10816	1085
IndependentMultimodal	127.23 s	14.14	16166	1143
LBAdaptiveSOM	137.91 s	13.05	15771	1208
T2FGMM_UV	207.68 s	8.66	10734	1239
DPEigenbackground	104.16 s	17.28	21525	1245
LBFuzzyGaussian	91.31 s	19.71	27118	1375
LBFuzzyAdaptiveSOM	175.51 s	10.25	14696	1433
T2FMRF_UV	285.84 s	6.29	10789	1715
FuzzyChoquetIntegral	316.55 s	5.68	10877	1914
FuzzySugenoIntegral	317.00 s	5.67	11345	2000
MultiLayer	521.71 s	3.45	9532	2762
LOBSTER	574.46 s	3.13	9866	3152
SuBSENSE	1002.95 s	1.79	9232	5157
PAWCS	2128.08 s	.84	9031	10751
DPTexture	1091.99 s	1.64	19498	11889
KDE	144.98 s	12.41	218980	17645
LBP_MRF	510.47 s	3.52	71662	20358
KNN	222.42 s	8.09	910784	112581

Tabulka 4.1: Porovnání výkonu algoritmů adaptivní separace popředí



Obrázek 4.3: **PAWCS a Zivkovic** Metoda PAWCS sice poskytuje dobré výsledky, její čas běhu je ale pro sledování v reálném čase neúnosný. Druhý obrázek byl zpracován metodou Z. Zivkovice.

tovém videu se snímkovou frekvencí 30 FPS v rozlišení 1280x720. Metrika *Přesnost* uvádí počet pixelů, ve kterých se popředí vypočítané daným algoritmem liší od ideálního popředí 4.2, nižší hodnoty jsou tedy lepší. Ze snímků byl aplikací eroze a dilatace odstraněn šum. Ten bude v pozdějších fázích detekce odfiltrován, nemá tedy smysl, aby ovlivňoval výběr algoritmu. Výsledné *Skóre* je vypočítáno následovně:

$$Skóre = \frac{Přesnost}{FPS}$$

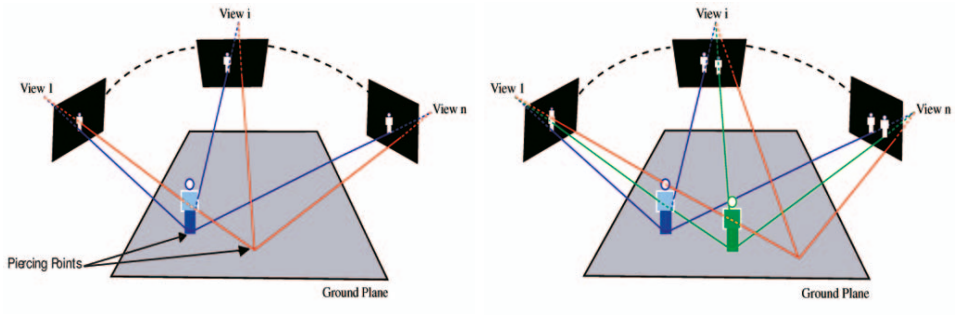
Metoda s nejvyšším skóre pouze počítá rozdíl mezi předchozím a aktuálním snímkem, není tedy dostatečně robustní. O druhé metodě jsem nenašel žádné informace, radši ji tedy nepoužívám. Na třetím místě skončila OpenCV implementace známé metody Z. Zivkovice [14]. Ta modeluje každý pixel množinou normálních rozdělení, jejichž počet a parametry jsou průběžně aktualizovány. Pokud pixel nového obrázku neodpovídá žádnému rozdělení, je označen za popředí.

Je zajímavé, že metody OpenCV::MOG2 a MixtureOfGaussians2, která vlastně pouze obaluje metodu OpenCV::MOG2, se liší nejen v času běhu, ale i v metrice přesnosti. To poukazuje na vady v implementaci knihovny BGSLibrary.

Na základě skóre, spolehlivosti implementace a kompatibility s obecným rozhraním OpenCV pro separátory popředí vybírám metodu Z. Zivkovice. Její implementaci poskytuje knihovna OpenCV ve třídě `BackgroundSubtractorMOG2`.

4.3.2 Řešení okluzí a podmínka obsazenosti indukovaná homografií

Podmínku obsazenosti indukovanou homografií (*homographic occupancy constraint*) zavádějí S. M. Khan a M. Shah v [11], jde o nutnou podmínku fungování jimi popsaných metod detekce.



(a) (b)

Obrázek 4.4: Podmínka obsazenosti indukovaná homografií. Obrázky převzaty z [11]

Tvrzení 1 *Pokud* $\exists P \in \mathbb{R}^3$ *ležící na rovině* π *a je součástí popředí* ψ*, potom pro každý pixel* p_i *odpovídající bodu* P *v pohledu* i *platí:*

- *pokud* ψ_i *je množina pixelů popředí pohledu* i*, pak* $p_i \in \psi_i$
- $\forall i, j, P = H_{i\to\pi}p_i = H_{j\to\pi}p_j$

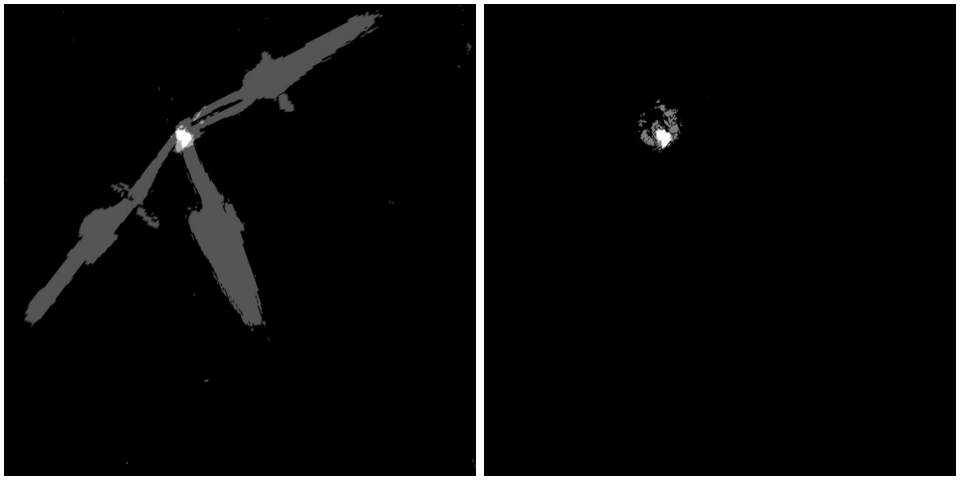
Tvrzení 1 v podstatě říká, že pokud bod popředí P leží na rovině π, pak ho příslušné homografie promítají do popředí v každém pohledu.

Tvrzení 2 (o obsazenosti indukované homografií) *Nechť* ψ_i *je množina pixelů popředí pohledu* i *a pixel* $p_i \in \psi_i$*. Potom bod* $P = H_{i\to\pi}p_i$ *je součástí popředí* ψ *právě tehdy, pokud* $\forall j p'_j = H^{-1}_{j\to\pi}H_{i\to\pi}p_i$ *a zároveň* $p'_j \in \psi_j$*.*

Tvrzení 2 je vlastně převráceným tvrzením 1. Bod P je součástí popředí ψ a leží na rovině π právě tehdy, pokud se po promítnutí odpovídající homografií zobrazí v každém pohledu i do pixelu, který je součástí popředí ψ_i. Tuto situaci ukazuje obrázek 4.4 (a). Obrazy modrého bodu jsou součástí popředí ve všech pohledech, modrý bod tedy představuje popředí a leží na rovině podlahy. Červený bod se sice v pohledu 1 promítne do popředí, ale jen proto, že ho překrývá osoba. V ostatních pohledech se promítá do pozadí, do popředí tedy nepatří.

Obrázek 4.4 (b) ukazuje, jakým způsobem jsou pomocí podmínky obsazenosti indukované homografií řešeny okluze. V pohledu 1 se modrá i zelená osoba překrývají. V ostatních pohledech je ale obě osoby možné rozlišit. Promítnutím všech pohledů na rovinu podlahy tedy vzniknou 2 izolovaná okolí, protože alespoň v jednom pohledu jsou obě popředí oddělená.

Autoři kombinují pohledy mírně odlišným způsobem, na rozdíl od zde uvedeného přístupu promítají obrázky všech kamer do zvoleného referenčního pohledu. To ale na platnost podmínky nemá vliv.



(a) (b)

Obrázek 4.5: **Projekce popředí tří pohledů na referenční rovinu** (a) Sloučení váženým součtem, (b) Sloučení průnikem.

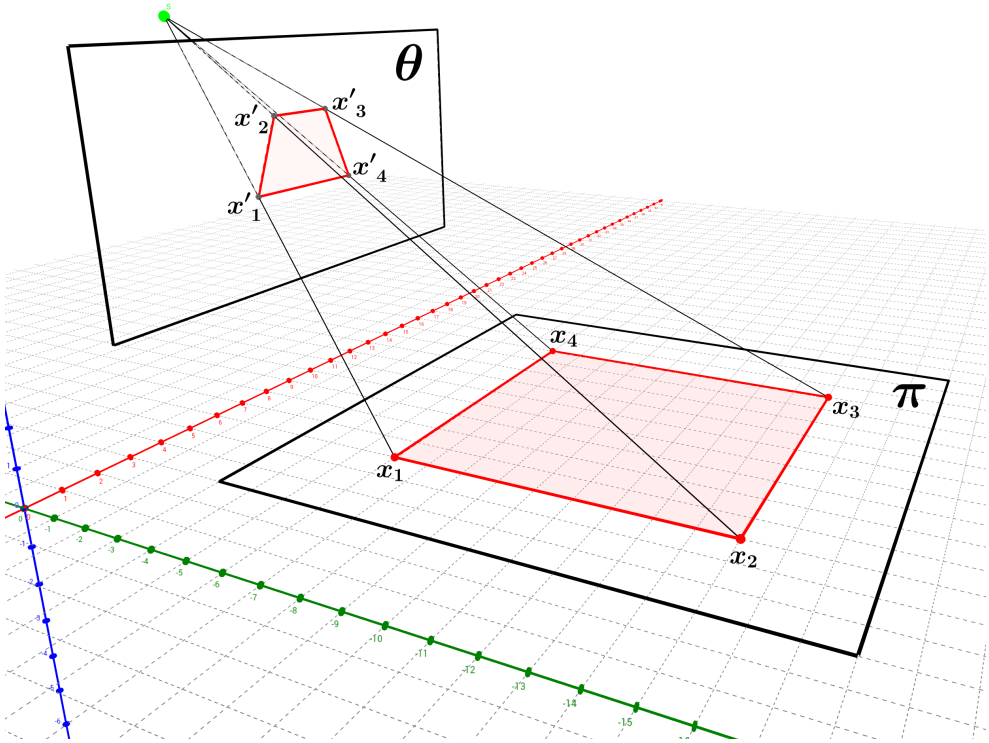
4.3.2.1 Modelování překážek a oblastí mimo záběr

Z tvrzení 2 vyplývá, že statická překážka v pohledu jediné z kamer, za kterou se mohou pohybovat osoby, znemožní detekci v celé překryté oblasti. Řešením je poskytovat ke každému pohledu masku, kde budou všechny překážky označeny a nebudou se tak v těchto oblastech podílet na detekci. Lze tak vyřešit i situaci, kdy se část scény nachází mimo dohled některé kamery. Stejné řešení volí M. Liem a D. Gavrila v [24].

4.3.3 Lokalizace na referenční rovině

Z tvrzení 2 vyplývá metoda lokalizace, která promítá popředí ψ_i jednotlivých pohledů i na referenční rovinu π. Za π se díky snadnému měření bodů pro výpočet homografie nejčastěji volí podlaha scény. Vstupem algoritmu jsou separovaná popředí ψ_i, výstupem binární obrázek ψ_π roviny π, který zobrazuje bílá místa tam, kde se π protíná s popředím ψ. Základem algoritmu je znalost homografií $H_{i\to\pi}$, které promítají pohled každé kamery na π. Projektivní transformace maticí $H_{i\to\pi}$ implementuje OpenCV funkcí `warpPerspective()`. Promítnutá popředí $\psi_i' = H_{i\to\pi}\psi_i$ jsou nakonec pixel po pixelu sloučena binární operací AND do ψ_π. Ke sloučení je opět použita funkce OpenCV, `bitand()`.

Pokud za referenční rovinu volíme podlahu, dojde sloučením popředí k detekci nohou. Ty jsou v porovnání se zbytkem lidského těla poměrně malé, navíc nepředstavují příliš stabilní cíle. Proto je vhodné výstupní obrázek dále kompenzovat, například metodami uvedenými v sekci 4.3.5.



Obrázek 4.6: Projekce reálné roviny na obrazovou rovinu

4.3.3.1 Projekce obrazu kamery na skutečnou rovinu

Naším cílem je najít projektivní transformaci z obrazu zachyceného kamerou do roviny ve skutečném světě. Obrazová i skutečná rovina jsou dvojrozměrné struktury, hledaná projektivní transformace je tedy homografií. Body roviny je možné vyjádřit dvousložkovým vektorem $[x, y]$. Protože se ale pohybujeme v projektivním prostoru $\mathbb{P}$, používáme homogenní souřadnice, které k původnímu vektoru přidávají ještě složku váhy, w. Platí, že $[x, y, w] = [k \cdot x, k \cdot y, k \cdot w]$ pro $k \neq 0$. Hledáme tedy matici homografie $H_{\theta->\pi} \in \mathbb{R}^{3,3}$, která zobrazuje body $[x', y', w'] \in \mathbb{P}^2$ obrazové roviny θ do bodů $[x, y, w] \in \mathbb{P}^2$ skutečné roviny π.

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{pmatrix} h_{0,0} & h_{0,1} & h_{0,2} \\ h_{1,0} & h_{1,1} & h_{1,2} \\ h_{2,0} & h_{2,1} & h_{2,2} \end{pmatrix} \cdot \begin{bmatrix} x' \\ y' \\ w' \end{bmatrix}$$

Matice $H_{\theta->\pi}$ má 8 stupňů volnosti [25], volíme $h_{2,2} = 1$. K výpočtu tedy použijeme 4 body na skutečné rovině π, $[x_i, y_i, 1]$, a jim odpovídající obrazy

na rovině θ, $[x'_i, y'_i, 1], 0 \leq i \leq 3$. Matici $H_{\theta->\pi}$ získáme vyřešením systému rovnic:

$$\begin{pmatrix} x_0 & y_0 & 1 & 0 & 0 & 0 & -x_0 * x'_0 & -y_0 * x'_0 \\ x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 * x'_1 & -y_1 * x'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 * x'_2 & -y_2 * x'_2 \\ x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3 * x'_3 & -y_3 * x'_3 \\ 0 & 0 & 0 & x_0 & y_0 & 1 & -x_0 * y'_0 & -y_0 * y'_0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 * y'_1 & -y_1 * y'_1 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 * y'_2 & -y_2 * y'_2 \\ 0 & 0 & 0 & x_3 & y_3 & 1 & -x_3 * y'_3 & -y_3 * y'_3 \end{pmatrix} \cdot \begin{bmatrix} h_{0,0} \\ h_{0,1} \\ h_{0,2} \\ h_{1,0} \\ h_{1,1} \\ h_{1,2} \\ h_{2,0} \\ h_{2,1} \end{bmatrix} = \begin{bmatrix} x'_0 \\ x'_1 \\ x'_2 \\ x'_3 \\ y'_0 \\ y'_1 \\ y'_2 \\ y'_3 \end{bmatrix}$$

Výpočet homografie $H_{\theta->\pi}$ je implementován v OpenCV funkcí `getPerspectiveTransform()`.

4.3.4 Lokalizace na virtuálních rovinách

Pokud mimo homografií $H_{i->\pi}$ známe pro každý pohled i i úběžník v_{z_i} ve směru kolmém na rovinu π, jsme schopni detekovat osoby nejen na referenční rovině, ale i ve virtuálních rovinách ϕ_j, které jsou s ní rovnoběžné. Metodou uvedenou v sekci 4.3.4.1 získáme pro každý pohled i a každou rovinu j homografie $H_{i\to\phi_j}$. Poté pro každou z rovin získáme průnik popředí ψ_{ϕ_j} způsobem ze sekce 4.3.3. Všechny ψ_{ϕ_j} jsou poté sečteny a prahovány funkcemi `add()` a `threshold()`. Výsledkem je tedy opět binární obrázek, jehož bílá místa představují detekce.

Výhodou lokalizace na více rovinách je, že ji lze aplikovat i na malých scénách, kde jednotlivé kamery nezabírají nohy všech subjektů. Díky sledování větších částí těla také dokáže detekovat vzdálenější osoby, než předchozí metoda. Při použití dostatečného počtu rovin je navíc možné určit i výšku sledovaných osob. Nevýhodou je zvýšená výpočetní náročnost, která roste s počtem použitých rovin, a citlivost na správné nastavení referenční homografie a úběžníku.

4.3.4.1 Projekce obrazu kamery na virtuální rovinu

Předpokládáme, že známe homografii $H_{\theta\to\pi}$ zobrazující obrazovou rovinu θ do skutečné roviny π a úběžník v_z obrazové roviny θ ve směru kolmém na rovinu π. Virtuální rovina ϕ je rovnoběžná s rovinou π a nachází se od ní ve vzdálenosti Z. Criminisi a kol. v [26] uvádí, jak z $H_{\pi\to\theta}$ získat $H_{\phi\to\theta}$.

$$H_{\phi\to\theta} = H_{\pi\to\theta} + [0 \quad 0 \quad \alpha \cdot Z \cdot v_z]$$

Inverzí uvedeného vztahu získáme homografii $H_{\theta\to\phi}$ zobrazující obrazovou rovinu θ na virtuální rovinu ϕ.

4.3.5 Morfologické operace

Slučování popředí neprodukuje exaktní výsledky. Nepřesnosti, zvláště pak nespojité oblasti reprezentující jediného člověka, mohou vést k falešným detekcím. Proto jsou na průnik popředí v referenční rovině aplikovány morfologické operace, jejichž úkolem je odstranit z něj šum a spojit chybně oddělené oblasti.

4.3.5.1 Eroze

Vstupem eroze je binární obrázek a *kernel* K, matice $K \in \{0,1\}^{n,m}$. Eroze postupně umístí střed kernelu K do každého pixelu p. Prvky kernelu rovné 1 definují jeho okolí. Odpovídající pixel p' výstupního obrázku je roven 1 (tedy označen jako bílý) právě tehdy, pokud je každý pixel z okolí p roven 1.

Obrázek 4.7 ukazuje aplikaci eroze s kernelem

1	1	1
1	1	1
1	1	1

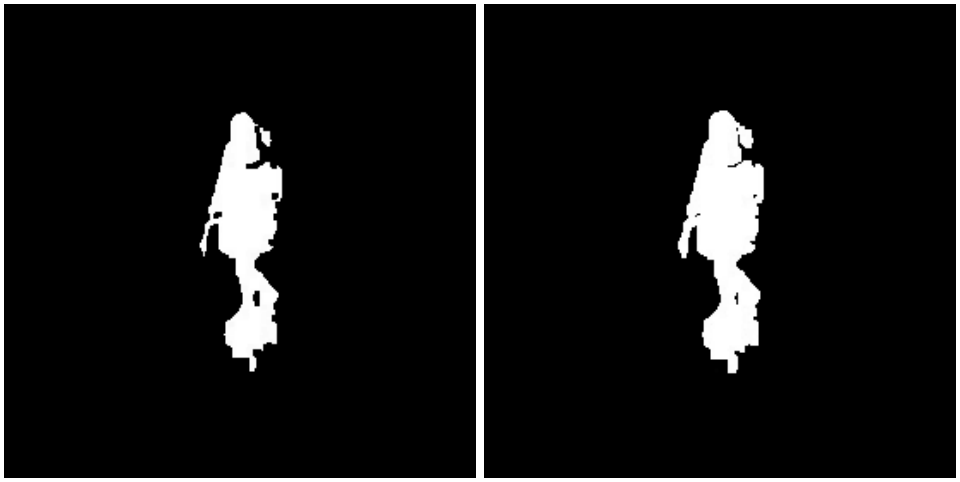
na výstup separace popředí. Šum z obrázku byl odstraněn, ale zároveň došlo ke zúžení osoby.

Obrázek 4.7: Eroze

4.3.5.2 Dilatace

Vstup i postup fungování dilatace jsou stejné, jako v případě eroze, liší se pouze operace, která je na pixel p a jeho okolí aplikována. Pixel p' výstupního obrázku je roven 1 právě tehdy, pokud se v okolí pixelu p nachází alespoň jeden pixel s hodnotou 1.

Na obrázek 4.8 byla aplikována dilatace se stejným kernelem, jako v části 4.3.5.1. Obrys postavy se rozšířil na původní velikost.



Obrázek 4.8: Dilatace

4.3.6 Hledání pozic

Výstupem algoritmů ze sekcí 4.3.3 a 4.3.4 je binární obrázek, na kterém jsou detekce vyznačené bílými oblastmi. Posledním krokem detekce je tedy najít v binárním obrázku ϕ_π, případně ϕ pozice těchto bílých míst. K tomu využívám funkci OpenCV `connectedComponentsWithStats()`, která nalezne všechny spojité oblasti a vrátí jejich souřadnice. Výstupem jsou i velikosti detekovaných souvislých komponent. Ty využívám k zamítnutí oblastí mimo očekávaný rozsah velikostí.

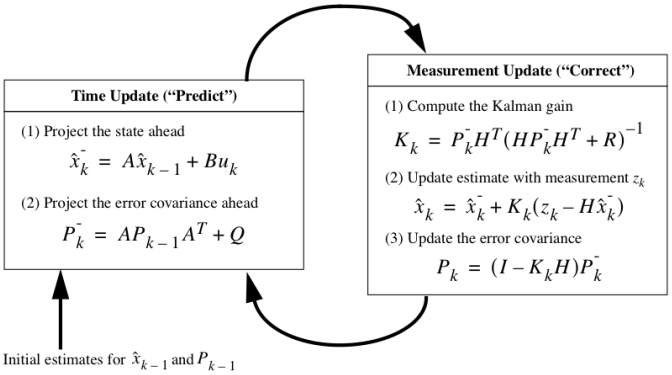
4.4 Sledování

4.4.1 Kalmanův filtr

Kalmanův filtr je algoritmus, který na základě údajů o předchozím stavu systému dokáže odhadnout stav aktuální. Pracuje ve dvou krocích, *predikci* a *korekci*, viz obrázek 4.9. V našem případě reprezentuje Kalmanův filtr trajektorii osoby. Vstupem Kalmanova filtru je v každém časovém kroku právě jedna detekce. Její pozice je v korekční části použita aktualizaci modelu predikce podle zjištěného skutečného stavu. Výstup predikce filtru je pak použit při přiřazování nových detekcí k trajektoriím.

Nejdůležitějším parametrem Kalmanova filtru je stavový vektor, který v našem případě reprezentuje stav sledované osoby. Prvky stavového vektoru $X_i = [x, y, x', y']^\mathsf{T}$ jsou pozice osoby (x, y) a rychlost jejího pohybu (x', y').

Dalším důležitým parametrem je přechodová matice F (na obrázku 4.9 matice A), která určuje, jakým způsobem se ze starého stavu odvodí stav



Obrázek 4.9: Cyklus Kalmanova filtru. Obrázek převzat z [27].

nový:

$$F = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Velikost přechodové matice jsou určeny dimenzionalitou stavu, její prvky pak závisí na konkrétních odhadovaných veličinách. Při predikci je nový stav vypočítán řešením

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \\ x'_{i+1} \\ y'_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_i \\ y_i \\ x'_i \\ y'_i \end{bmatrix} + \begin{bmatrix} w_i^0 \\ w_i^1 \\ w_i^2 \\ w_i^3 \end{bmatrix}$$

Vektor w_i představuje náhodně rozdělený procesní šum.

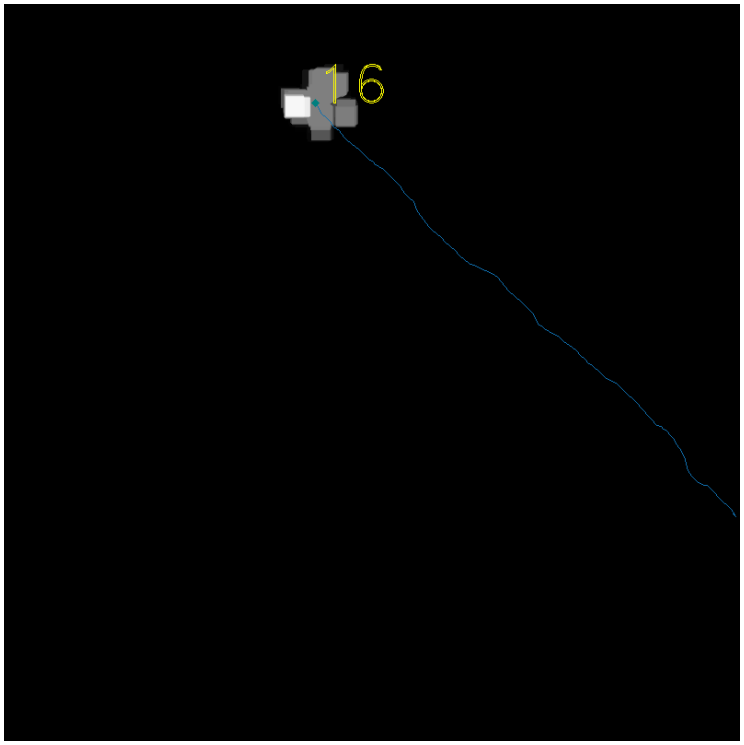
Posledním parametrem, který je nutné uvést, je matice H. Ta určuje vztah mezi měřením a predikovaným stavem systému. U detekcí měřím pouze pozici, tedy

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

Při detekci osoby, která nemůže být přiřazena k žádnému stávajícímu filtru, se vytvoří nový Kalmanův filtr. Jeho počáteční stav je nastaven na polohu nově detekované osoby. Kalmanův filtr je implementován v knihovně OpenCV třídou `KalmanFilter`.

4.4.2 Přiřazení detekcí k trajektoriím

Propojení pozic aktuálních detekcí a predikovaných pozic existujících trajektorií formuluji jako problém přiřazení. Řeším ho vlastní implementací maďarského algoritmu. Vstup algoritmu tvoří kolekce pozic predikcí p_i a de-



Obrázek 4.10: Trajektorie sledovaná Kalmanovým filtrem. Číslo a zelený kosočtverec ukazují aktuální polohu osoby na půdorysu, modrá stopa pak znázorňuje její pohyb.

tekcí d_j. Algoritmus nejprve sestaví matici vzdáleností M. Prvek $m_{i,j} = \sqrt{(p_i.x - d_j.x)^2 + (p_i.y - d_j.y)^2}$ představuje vzdálenost mezi p_i a d_j.

Algoritmus funguje následujícím způsobem:

1. Ve všech řádcích odečti od každého prvku nejnižší hodnotu v řádku.
2. Zopakuj 1. pro sloupce.
3. Pokryj všechny 0 minimálním počtem horizontálních nebo vertikálních čar.
4. Pokud je počet čar roven počtu řádků matice M, jdi na poslední krok.
5. Najdi minimální prvek nepokrytý čárou, odečti ho od všech nepokrytých prvků, přičti ke všem prvkům ležícím na průniku dvou čar.
6. Jdi na krok 3.
7. Pro každou řádku najdi a přiřaď sloupec, který ještě nebyl přiřazen a jejichž společný prvek je 0.

4.5 Omezení systému

Zvolený přístup přináší určitá inherentní omezení. První z nich vychází ze způsobu, jakým je separováno popředí. Pokud se sledovaná osoba přestane po dostatečně dlouhou dobu hýbat, bude modelována jako součást pozadí a nedojde k její detekci. Při rozšíření použití adaptivních metod separace pozadí je zajímavé, že se tomuto problému většina prací nevěnuje.

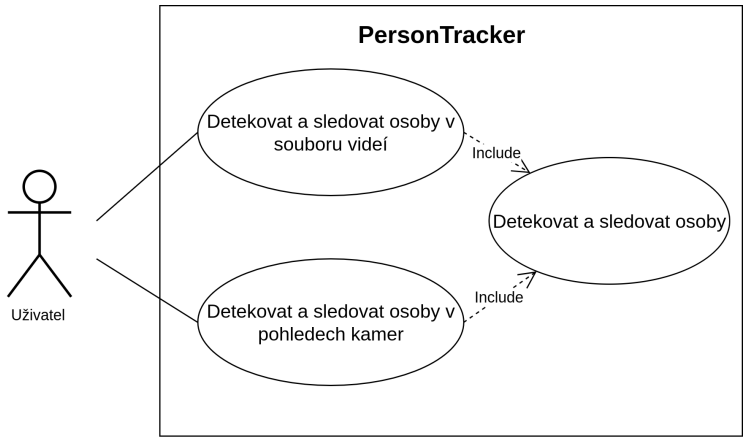
Další omezení platí zejména pro lokalizaci pomocí referenční roviny. Předpokládáme, že se osoby pohybují na jedné rovině. Takový předpoklad neplatí vždy, proto například ve víceúrovňových scénách nemusí být osoby mimo referenční rovinu rozpoznány. Řešením by bylo použít lokalizaci na více rovinách a příhodným způsobem nastavit počet a vzdálenost rovnoběžných rovin.

Poslední omezení vychází z použití Kalmanova filtru.

Kapitola 5

Implementace

5.1 Modelování případů užití



Obrázek 5.1: Případy užití

5.1.1 Detekovat a sledovat osoby v souboru videí

Uživatel zjistí počet vstupních videí a jejich názvy, informace zapíše do souboru definice scény podle vzoru v sekci 4.1. Dál pokračuje případem užití 5.1.3.

5.1.2 Detekovat a sledovat osoby v pohledech kamer

Uživatel zjistí počet připojených kamer a jejich ID, informace zapíše do souboru definice scény podle vzoru v sekci 4.1. Dál pokračuje případem užití 5.1.3.

5.1.3 Detekovat a sledovat osoby

Uživatel pomocí nástroje PointInViewSelector zjistí pozice 4 bodů v každé obrazové rovině, zjistí pozice bodů v reálných souřadnicích, informace zapíše do souboru definice scény podle vzoru v sekci 4.1.

Poté spustí sledovací program pomocí

```
./PersonTracker scene.yml
```

Program může pozastavit pomocí mezerníku, přeskakovat může šipkou doprava a program ukončí pomocí ESC.

5.2 Diagram tříd

Obrázek 5.2 ukazuje diagram tříd hlavní aplikace. Aplikace byla navržena tak, aby ji bylo snadné rozšířit o implementace jiných metod detekce a sledování. Za tím účelem poskytuje rozhraní Tracker a Detector.

5.2.1 Hlavní třída aplikace

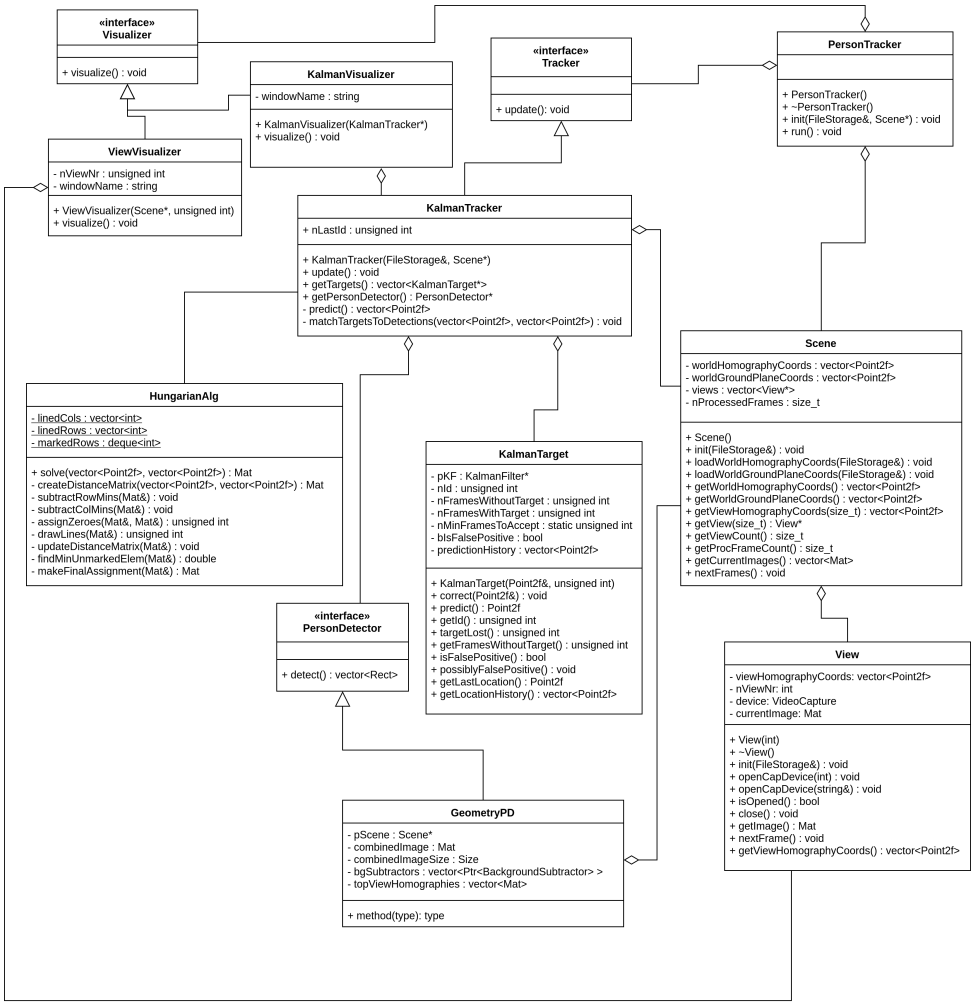
Třída PersonTracker obsahuje objekt reprezentující scénu a ukazatele na objekty Tracker a kolekci objektů Visualizer. Při inicializaci je inicializována scéna, jsou vytvořeny instance použitých Visualizerů a Trackerů. Běh aplikace řídí metoda `run()`.

5.2.2 Třídy reprezentující scénu

Třída View reprezentuje jeden pohled, ukládá si informace o bodech homografií a poskytuje metody k načítání snímků. Třída Scene obsahuje kolekci objektů View. Kromě nich ukládá informace o reálných souřadnicích homografie a poskytuje rozhraní k načtení kolekce všech aktuálních snímků.

5.2.3 Třídy Tracker

Tracker je abstraktní třída, rozhraní, které definuje metodu `update()`, pomocí které je aktualizována poloha sledovaných objektů. Třída KalmanTracker implementuje rozhraní Tracker. Udržuje si kolekci objektů KalmanTarget, každý z nich představuje jednu trajektorii. Třída KalmanTarget udržuje ID a historii pozic sledovaného cíle, pomocí třídy KalmanFilter implementované v OpenCV je schopna predikovat budoucí stav objektu. KalmanTracker získává detekce osob z objektu třídy implementující rozhraní Detector.



Obrázek 5.2: Diagram tříd hlavního programu

5.2.4 Třídy Detector

Detector je podobně jako Tracker rozhraní definující metodu `detect()`, která vrací detekce jako kolekci bounding boxů. GeometryPD implementuje metodu lokalizace na referenční rovině.

5.2.5 Třída HungarianAlg

HungarianAlg je statická utility class implementující maďarský algoritmus. Řeší problém přiřazení detekcí k trajektoriím.

5.2.6 Třídy Visualizer

Visualizer je opět rozhraní, definuje metodu `visualize()`. Jeho implementace, ViewVisualizer a KalmanVisualizer, jsou v konstruktoru parametrizovány objektem, jehož stav mají zobrazovat. ViewVisualizer zobrazuje obrázky zvoleného pohledu, KalmanVisualizer potom ukazuje trajektorie detekcí na referenční rovině.

5.3 Volba programovacího jazyka

Knihovna OpenCV je implementována ve třech jazycích, `C++`, Java a Python. Z hlediska Javy a Pythonu je OpenCV pouze wrapper pro implementaci v `C++`, ani jedna volba by tedy neměla ovlivnit její výkon. Implementuji ovšem řadu vlastních metod, z hlediska výkonu je tedy Python jako interpretovaný jazyk nevhodný. Z obou zbylých jazyků se na základě osobních preferencí rozhoduji pro `C++`.

5.4 Pomocné nástroje

5.4.1 PointInViewSelector

Jedním ze vstupních parametrů hlavního programu jsou obrazové souřadnice 4 bodů, kterými je definovaná homografie mezi obrazovou a skutečnou rovinou. PointInViewSelector je jednoduchá aplikace sloužící k jejich výběru a vypsání.

Kapitola 6

Vyhodnocení výsledků

6.1 Testovací data

Pro potřeby testování navrženého systému byly pořízeny tři datasety, jeden v interiéru a dva v exteriéru. Byly natáčeny nezkalibrovanými kamerami bez explicitní synchronizace, podmínky tedy odpovídají nasazení bez žádných zvláštních přizpůsobení kamerového systému. Pro potřeby výpočtu homografie byla na každé scéně změřena velikost objektu, který se nacházel v záběru všech kamer. Relativně k němu byla změřena velikost scény.

Jednou ze zkoumaných možností bylo pořízení dat pomocí 3D modelovacího softwaru. Ten dokáže v současnosti poskytovat dostatečně kvalitní data, aby na nich bylo možné testovat systém určený k reálnému nasazení. Výhodou tohoto přístupu je rozsáhlá možnost přizpůsobení generovaných dat. Navíc jsou známé exaktní parametry nastavení scény, kamer a v neposlední řadě poziční data pro každou modelovanou osobu, se kterými je možné implementované algoritmy porovnávat. Tento přístup však nebyl z důvodu veliké časové náročnosti využit.

6.1.1 Strahov

První dataset, *Strahov*, zobrazuje 6 minut pohybu chodců po centrálním parkovišti strahovských kolejí. Je natočen kamerami dvou mobilních telefonů, které byly umístěny v 5. patrech protilehlých budov, sledovaná oblast má rozměry 31,5 x 45 metrů. Rozlišení obou kamer je 1920x1080. Homografie je vypočítána z rozměrů nápisu (S)ilicon Hill. Natáčelo se v pozdním odpoledni za zataženého počasí. Osoby se v oblasti pohybují jednotlivě, s příjezdem autobusu se pak objevují větší shluky. Vzhledem k veliké ploše a zhoršeným světelným podmínkám dataset prověří především schopnosti detektoru a potažmo i přiřazovacího algoritmu.

Obrázek 6.1: Snímky z datasetu Strahov

6.1.2 Atrium

Druhý dataset, *Atrium*, byl natočen třemi webkamerami u vstupu do nové budovy ČVUT. Ukazuje 10 minut pohybu studentů po prostoru před vstupem do budovy. Hustota chodců ve vlnách stoupá a klesá podle toho, jak zrovna přijíždějí autobusy nebo končí přednášky. První 4 minuty stojí na scéně 2 postavy tak, že je většina adaptivních separátorů popředí označí za pozadí.

Obrázek 6.2: Snímky z datasetu Atrium

6.1.3 Domov

Třetí dataset zobrazuje pohyb 5 osob po průměrně veliké místnosti. Je natáčen třemi webkamerami, podobně, jako dataset Atrium. Osoby se překrývají výrazně častěji, než v předchozích datasetech, výhled je navíc překrytý překážkami. Náhodný pohyb a blízkost osob představují výzvu pro detekční i sledovací algoritmus.

Obrázek 6.3: Snímky z datasetu Domov

6.2 Hardware a software

6.2.1 Měřící systém

Kompilace a měření probíhaly na systému s následujícími parametry:

- **OS** Linux 4.14.10-1-ARCH #1 SMP PREEMPT x86_64 GNU/Linux
- **CPU** Intel(R) Core(TM) i7-7700HQ CPU @ 2.80GHz
- **RAM** 16GiB DDR4 2400 MHz
- **GPU** NVIDIA GeForce 1050 Ti
- **Kompilátor C++** gcc version 7.2.1 20171128 (GCC)

6.3 Měření času běhu

Měření času běhu probíhalo na první minutě datasetu Atrium. Zpracování 3 záznamů v rozlišení 1280x720 natáčenými 30 snímky za sekundu trvalo v průměru 128.6 sekund, systém v této konfiguraci tedy dokáže zpracovat přibližně 14 snímků za sekundu.

6.4 Náměty na zlepšení a budoucí práci

Před spuštěním programu je pro každou kameru nutné ručně vybrat body pro výpočet homografie. Očividným námětem na zlepšení je tedy její automatická detekce.

Některé navrhované metody nebyly z časových důvodů implementovány. Vzhledem k ověřeným výsledkům těchto metod by zajisté přispěly ke zlepšení kvality poskytovaných dat.

Detektor by při kombinaci informací z několika virtuálních rovin mohl každé rovině přisoudit váhu podle míry šumu tak, jak je popsáno v původním článku [11]. Jeho výkon by bylo dále možné vylepšit úpravou navrženou v [28], kde autoři přeformulovali problém detekce na více rovinách s využitím integrálního obrázku. Výkon z hlediska počtu zpracovaných snímků za sekundu by bylo dále možné vylepšit paralelizací zpracování jednotlivých pohledů.

Použití separace popředí k detekci přináší jisté nevýhody. Separace popředí detekuje pohybující se objekty, nerozlišuje ale jejich typ. Pokud se tedy na scéně pohybují například auta, budou detekovány a sledovány stejně, jako osoby. Jedním z možných řešení je nahrazení separace popředí detektorem osob založeným na klasifikaci příznaků, viz sekce 3.2.1. Takový přístup může pracovat v reálném čase [7].

Závěr

V prvních dvou kapitolách shrnuji současné nejpoužívanější metody detekce a sledování pomocí jedné nebo více kamer. Z popsaných metod jsem si vybral přístup, který kombinuje více-kamerovou detekci a sledování pomocí Kalmanova filtru a maďarského algoritmu. Volba konkrétních metod je zdůvodněna v kapitole 4, jejich nevýhody uvádím v sekcích 4.5 a 6.4. Další kapitola se krátce věnuje implementaci. Poslední kapitola popisuje testování. V rámci práce byly pořízeny 3 datasety s různou mírou obtížnosti zpracování. Z časových důvodů ovšem neproběhly testy na všech datasetech.

Jak uvádím v 6.4, některé metody nebyly implementovány. Objektový návrh aplikace ovšem umožňuje její snadné rozšíření implementací příslušných abstraktních tříd.

Literatura

[1] Paclíková, A.; Švec, P.: V ulicích největších měst přibudou kamery. Praha proinvestuje až miliardu. May 2017. Dostupné z: https://zpravy.idnes.cz/ulice-velka-mesta-kamery-miliardy-dll-/domaci.aspx?c=A170509_2323837_domaci_jav

[2] Viola, P.; Jones, M.: Robust Real-time Object Detection. In *International Journal of Computer Vision*, 2001.

[3] Lienhart, R.; Maydt, J.: An extended set of Haar-like features for rapid object detection. In *Proceedings. International Conference on Image Processing*, IEEE, doi:10.1109/icip.2002.1038171. Dostupné z: https://doi.org/10.1109/icip.2002.1038171

[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)*, IEEE, doi:10.1109/cvpr.2005.177. Dostupné z: https://doi.org/10.1109/cvpr.2005.177

[5] Dollar, P.; Tu, Z.; Perona, P.; aj.: Integral Channel Features. In *Proceedings of the British Machine Vision Conference*, BMVA Press, 2009, ISBN 1-901725-39-1, s. 91.1–91.11, doi:10.5244/C.23.91.

[6] Dollar, P.; Appel, R.; Belongie, S.; aj.: Fast Feature Pyramids for Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 36, č. 8, aug 2014: s. 1532–1545, doi:10.1109/tpami.2014.2300479. Dostupné z: https://doi.org/10.1109/tpami.2014.2300479

[7] Benenson, R.; Mathias, M.; Timofte, R.; aj.: Pedestrian detection at 100 frames per second. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, jun 2012, doi:10.1109/cvpr.2012.6248017. Dostupné z: https://doi.org/10.1109/cvpr.2012.6248017

[8] Shu, G.; Dehghan, A.; Oreifej, O.; aj.: Part-based multiple-person tracking with partial occlusion handling. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, jun 2012, doi:10.1109/cvpr.2012.6247879. Dostupné z: https://doi.org/10.1109/cvpr.2012.6247879

[9] Zhu, L.; Hwang, J.-N.; Cheng, H.-Y.: Tracking of multiple objects across multiple cameras with overlapping and non-overlapping views. In *2009 IEEE International Symposium on Circuits and Systems*, IEEE, may 2009, doi:10.1109/iscas.2009.5117941. Dostupné z: https://doi.org/10.1109/iscas.2009.5117941

[10] Nie, W.; Liu, A.; Su, Y.; aj.: Single/cross-camera multiple-person tracking by graph matching. *Neurocomputing*, ročník 139, sep 2014: s. 220–232, doi:10.1016/j.neucom.2014.02.040. Dostupné z: https://doi.org/10.1016/j.neucom.2014.02.040

[11] Khan, S.; Shah, M.: Tracking Multiple Occluding People by Localizing on Multiple Scene Planes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 31, č. 3, mar 2009: s. 505–519, doi:10.1109/tpami.2008.102. Dostupné z: https://doi.org/10.1109/tpami.2008.102

[12] Arsic, D.; Hristov, E.; Lehment, N.; aj.: Applying multi layer homography for multi camera person tracking. In *2008 Second ACM/IEEE International Conference on Distributed Smart Cameras*, IEEE, sep 2008, doi:10.1109/icdsc.2008.4635731. Dostupné z: https://doi.org/10.1109/icdsc.2008.4635731

[13] Liem, M. C.; Gavrila, D. M.: Joint multi-person detection and tracking from overlapping cameras. *Computer Vision and Image Understanding*, ročník 128, nov 2014: s. 36–50, doi:10.1016/j.cviu.2014.06.003. Dostupné z: https://doi.org/10.1016/j.cviu.2014.06.003

[14] Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, IEEE, 2004, doi:10.1109/icpr.2004.1333992. Dostupné z: https://doi.org/10.1109/icpr.2004.1333992

[15] Stauffer, C.; Grimson, W.: Adaptive background mixture models for real-time tracking. In *Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, IEEE Comput. Soc, doi:10.1109/cvpr.1999.784637. Dostupné z: https://doi.org/10.1109/cvpr.1999.784637

[16] Munaro, M.; Basso, F.; Menegatti, E.: OpenPTrack: Open source multi-camera calibration and people tracking for RGB-D camera networks.

Robotics and Autonomous Systems, ročník 75, jan 2016: s. 525–538, doi:10.1016/j.robot.2015.10.004. Dostupné z: https://doi.org/10.1016/j.robot.2015.10.004

[17] Kumar, P.; Dick, A.; Sheng, T. S.: Real Time Target Tracking with Pan Tilt Zoom Camera. In *2009 Digital Image Computing: Techniques and Applications*, IEEE, 2009, doi:10.1109/dicta.2009.84. Dostupné z: https://doi.org/10.1109/dicta.2009.84

[18] Lu, Y.; Payandeh, S.: Cooperative hybrid multi-camera tracking for people surveillance. In *2008 Canadian Conference on Electrical and Computer Engineering*, IEEE, may 2008, doi:10.1109/ccece.2008.4564764. Dostupné z: https://doi.org/10.1109/ccece.2008.4564764

[19] Dollar, P.; Wojek, C.; Schiele, B.; aj.: Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, ročník 34, č. 4, apr 2012: s. 743–761, doi:10.1109/tpami.2011.155. Dostupné z: https://doi.org/10.1109/tpami.2011.155

[20] Solano, A.: Human pose estimation using OpenPose with TensorFlow (Part 1). Oct 2017. Dostupné z: https://arvrjourney.com/human-pose-estimation-using-openpose-with-tensorflow-part-1-7dd4ca5c8027

[21] Leal-Taixe, L.; Pons-Moll, G.; Rosenhahn, B.: Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, IEEE, nov 2011, doi:10.1109/iccvw.2011.6130233. Dostupné z: https://doi.org/10.1109/iccvw.2011.6130233

[22] Byeon, M.; Oh, S.; Kim, K.; aj.: Efficient Spatio-Temporal Data Association Using Multidimensional Assignment in Multi-Camera Multi-Target Tracking. In *Proceedings of the British Machine Vision Conference (BMVC)*, editace M. W. J. Xianghua Xie; G. K. L. Tam, BMVA Press, September 2015, ISBN 1-901725-53-7, s. 68.1–68.12, doi:10.5244/C.29.68. Dostupné z: https://dx.doi.org/10.5244/C.29.68

[23] Sobral, A.: BGSLibrary: An OpenCV C++ Background Subtraction Library. In *IX Workshop de Visão Computacional (WVC'2013)*, Rio de Janeiro, Brazil, Jun 2013. Dostupné z: https://github.com/andrewssobral/bgslibrary

[24] Liem, M. C.; Gavrila, D. M.: *A Comparative Study on Multi-person Tracking Using Overlapping Cameras*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ISBN 978-3-642-39402-7, s. 203–212, doi:10.1007/978-

3-642-39402-7_21. Dostupné z: https://doi.org/10.1007/978-3-642-39402-7_21

[25] Szeliski, R.: *Computer Vision*. Springer London, 2011, doi:10.1007/978-1-84882-935-0. Dostupné z: https://doi.org/10.1007/978-1-84882-935-0

[26] Criminisi, A.; Reid, I.; Zisserman, A.: Single view metrology. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, IEEE, 1999, doi:10.1109/iccv.1999.791253. Dostupné z: https://doi.org/10.1109/iccv.1999.791253

[27] Can't recognize math symbol - bold "minus"above variable. Dostupné z: https://tex.stackexchange.com/questions/171400/cant-recognize-math-symbol-bold-minus-above-variable

[28] Yildiz, A.; Akgul, Y. S.: A Fast Method for Tracking People with Multiple Cameras. In *Trends and Topics in Computer Vision*, Springer Berlin Heidelberg, 2012, s. 128–138, doi:10.1007/978-3-642-35749-7_10. Dostupné z: https://doi.org/10.1007/978-3-642-35749-7_10

Příloha **A**

Seznam použitých zkratek

HOG Histogram of Oriented Gradients

SVM Support Vector Machine

SIFT Scale-invariant feature transform

PTZ kamera Pan Tilt Zoom kamera

Příloha **B**

Obsah přiloženého USB disku

- `bin` ..... adresář se spustitelnou formou implementace
- `Datasets` ..... adresář obsahující natočené datasety
  - `Atrium`
  - `Domov`
  - `Strahov`
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `DP_Kozák_Jan.pdf` ..... text práce ve formátu PDF
  - `DP_Kozák_Jan.ps` ..... text práce ve formátu PS