VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE HLEDANÝCH OSOB VE VIDEU

DETECTION OF WANTED PEOPLE IN VIDEO

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **DAVID BAŽOUT**
AUTHOR

VEDOUCÍ PRÁCE **Ing. VÍTĚZSLAV BERAN, Ph.D.**
SUPERVISOR

BRNO 2019

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2018/2019

Zadání bakalářské práce

Student: **Bažout David**
Program: Informační technologie
Název: **Detekce hledaných osob ve videu**
Detection of Wanted People in Video
Kategorie: Zpracování obrazu

Zadání:

1. Seznamte se s metodami zpracování obrazu a videa zaměřenými na detekci objektů a extrakci obrazových příznaků.
2. Navrhněte systém, který bude detekovat vybrané osoby podle obličeje ve videu.
3. Připravte vhodnou datovou sadu a její anotaci tak, aby obsahovala různé relevantní situace.
4. Implementujte navržený systém pomocí dostupných knihoven.
5. Vyhodnoťte systém na připravené sadě dat.
6. Prezentujte klíčové vlastnosti řešení formou plakátu a krátkého videa.

Literatura:

- M. Sonka, V. Hlaváč, R. Boyle. *Image Processing, Analysis, and Machine Vision*, CL-Engineering, ISBN-13: 978-0495082521, 2007.
- Gary R. Bradski, Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*, ISBN 10: 0-596-51613-4, September 2008.

Pro udělení zápočtu za první semestr je požadováno:

- Body 1, 2, 3 a částečně bod 4.

Podrobné závazné pokyny pro vypracování práce viz http://www.fit.vutbr.cz/info/szz/

Vedoucí práce: **Beran Vítězslav, Ing., Ph.D.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2018
Datum odevzdání: 15. května 2019
Datum schválení: 1. listopadu 2018

Abstrakt

Cílem této práce je vytvořit nástroj umožňující vyhledávání podezřelých osob ve videozáznamu pocházejícího z dohledových kamer. Hledané osoby jsou systému určeny pomocí několika fotografií obličeje. Výstup tvoří informace o výskytu hledaných osob na konkrétních snímcích. Úloha je řešena rozdělením problému na detekci tváře a její následnou identifikaci. Experimenty s existujícími přístupy na vhodných datových sadách zajišťují relevantní porovnání úspěšnosti metod za různých podmínek. Výstupy testů umožňují vybrat vhodné metody a jejich optimální nastavení pro tuto konkrétní úlohu. Práce se zabývá i návrhem vhodné architektury, průzkumem existujících knihoven implementujících zkoumané metody a dalšími způsoby optimalizace výpočtu. Výsledkem je implementace uživatelské aplikace splňující zadané parametry. Její funkce byla otestována na vlastní datové sadě věrně napodobující podmínky reálného provozu.

Abstract

The aim of this work is to create a software tool for searching of wanted people in video recordings from surveillance cameras. Wanted people are identified to the system using multiple facial photos. The output consists of information on the occurrence of wanted persons in specific frames. The problem consists of face detection and its subsequent identification task. Experiments with existing approaches on appropriate datasets provide relevant comparisons of method performance under different conditions. Appropriate methods and their optimal settings for this particular task are chosen according to the results of the experiments. The thesis also deals with the design of suitable architecture, research of existing libraries implementing the tested methods and other ways of optimizing the calculation. The result is the implementation of a user application that meets the specified parameters. The application's functionality has been tested on the own dataset simulating real-world conditions.

Klíčová slova

Detekce tváře, Rozpoznávání tváře, Sledovací algoritmy, ChokePoint, Head Pose Image Dataset, OpenCV, Dlib, Haar, LBP, CNN, HOG

Keywords

Face detection, Face recognition, Tracking algorithm, ChokePoint, Head Pose Image Dataset, OpenCV, Dlib, Haar, LBP, CNN, HOG

Citace

BAŽOUT, David. *Detekce hledaných osob ve videu.* Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Vítězslav Beran, Ph.D.

Detekce hledaných osob ve videu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Vítězslava Berana Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
David Bažout
13. května 2019

Poděkování

Touto cestou bych chtěl poděkovat panu Ing. Vítězslavu Beranovi Ph.D. za jeho podnětné a konstruktivní rady, které mi výrazně pomohly při tvorbě této práce.

Obsah

Kapitola 1

Úvod

Někdy i vteřiny mohou při hledání nebezpečných osob znamenat rozdíl mezi životem a smrtí civilistů. Rychlé a efektivní vyhledávání v množství kamerových záznamů nelze provádět manuálně. Moderní výpočetní technika a pokročilé technologie počítačového vidění nabízejí prostředky pro sledování budov, objektů nebo veřejných prostranství v reálném čase. Podobný systém by měl napomáhat dobré věci, ale jeho nasazení může mít pro společnost v případě napojení na širokou síť pouličních kamer za následek výrazné narušení soukromí. Je nutné zajistit, aby podobná situace v budoucnu nenastala.

Cílem této práce je vytvoření nástroje, který umožňuje efektivní vyhledávání osob podle obličeje ve videu z kamerového systému. Hledané osoby jsou systému zadávány pomocí sady fotografií jejich obličeje. Video může pocházet z IP kamery v reálném čase, nebo je zde možnost načítání ze souboru. Systém je schopen určit, na kterých snímcích videa a ve které části se hledaná osoba nachází. Je nutné dosahovat únosné výpočetní náročnosti, aby mohla být data zpracovávána i na přenosném počítači v reálném čase.

Nedílnou součástí práce je návrh a realizace testování existujících přístupů, jehož cílem je objektivně porovnat úspěšnost metod detekce i rozpoznávání tváře pomocí vhodné statistické metriky a na základě výsledků určit nejvhodnější metody pro tuto konkrétní úlohu. U metod lze často měnit množství jejich parametrů. Změny parametrů metody mohou ovlivňovat i jejich výpočetní náročnost. Důležité je porozumět způsobu interního zpracování obrazových dat a algoritmu, který metoda používá. Díky této znalosti je možné provádět další optimalizace systému. Výběr vhodné datové sady je klíčem k dosažení relevantních výsledků. V současné době je pro účely testování metod detekce i identifikace k dispozici množství datových sad. Práce zahrnuje jejich podrobný průzkum a porovnání jejich dat s daty z reálného nasazení.

Výsledky testů existujících přístupů poskytují dostatek informací o úspěšnosti metod v nejrůznějších situacích a vlivu změn parametrů na jejich chování. Získaná data umožňují zvolit metodu, která nejvíce odpovídá požadavkům na chování aplikace, a určují i její optimální nastavení. Kromě základních metod pro detekci a identifikaci tváří je nutné zvážit architekturu celé aplikace a algoritmus, jakým budou snímky vstupní videosekvence zpracovávány. Nabízí se i využití dalších komponent, které mohou napomoci k lepším výsledkům. Metody detekce jsou často doplňovány o sledovací algoritmy, které dokáží eliminovat jejich chybné výsledky.

Fungování výsledné implementace je ověřeno testy na vlastní datové sadě. Vlastní datová sada věrně napodobuje podmínky reálného nasazení a umožňuje zhodnotit, zda se podařilo dosáhnout očekávaných výsledků.

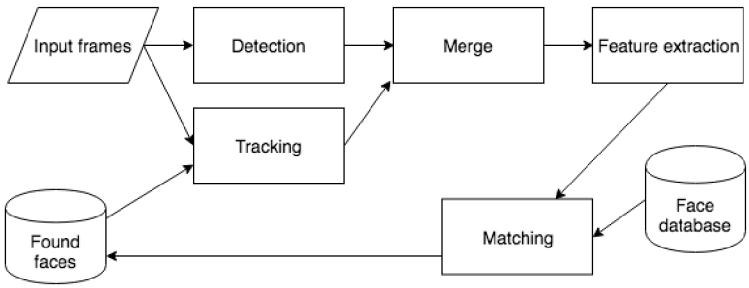
Kapitola 2

Systémy pro detekci a identifikaci osob

Zkoumání podoby lidské tváře je jedním z nejběžnějších způsobů, pomocí kterého dokáží lidé rozpoznávat osoby z jejich okolí. Systémy automatizovaného rozpoznávání tváře mají využití v mnoha oblastech. Jejich výhoda oproti ostatním biometrickým systémům spočívá v bezkontaktnosti celého procesu a zkoumaná osoba se nemusí o ničem dozvědět. Široké možnosti uplatnění vedly ve spojení s nárůstem výkonu výpočetní techniky k prudkému rozvoji algoritmů počítačového vidění realizujících tuto úlohu [20].

2.1 Architektura systému

Proces rozpoznávání tváře lze rozdělit na úlohu detekce tváře ve snímcích videa a její následnou identifikaci. Tyto klíčové úlohy bývají obvykle doplněny o další komponenty napomáhající zlepšit výsledky celého systému. V minulosti bylo vytvořeno množství podobných systémů. Obvykle se všechny systémy skládají z komponent schématicky zobrazených na obrázku 2.1. Odlišností je způsob, jakým jsou realizovány jednotlivé podúlohy [19]. Množství vědeckých pracovníků po celém světě přichází se stále efektivnějšími a úspěšnějšími metodami detekce i rozpoznávání tváře. Metody využívané v systémech pro rozpoznávání tváře jsou v praxi vystaveny mnoha nepříznivým vlivům a jsou zde velké rozdíly v jejich úspěšnosti. Tyto faktory je nutné identifikovat a pro výslednou implementaci využít metod dosahujících nejvyšší robustnosti.



Obrázek 2.1: Schéma logických komponent systému pro rozpoznávání tváře.

Zpracování videosekvence probíhá obvykle po jednotlivých snímcích. Prvním krokem je provedení detekce tváří nacházejících se na snímku. Úkolem detektoru je definovat souřadnice oblastí ve snímku s hodnotou specifikující míru pravděpodobnosti výskytu tváře na daném místě [20]. Je nutné zajistit, aby v nejmenší možné míře docházelo k situaci, kdy systém tvář nedetekuje, za tolerance občasného výskytu falešných detekcí. Současně je nutné optimalizovat výpočetní náročnost tohoto kroku, protože výpočet musí být proveden pro každý zpracovaný snímek.

Využití sledovacího algoritmu [19] není pro funkci systému bezpodmínečně nutné. Motivací nasazení sledovacích algoritmů může být optimalizace výpočetní náročnosti detekce. Sledování rovněž napomáhá identifikovat a odstranit její chyby. Úlohou sledovacího algoritmu je určit polohu objektu v novém snímku porovnáním s jeho polohou v posloupnosti navazujících snímků videozáznamu. Pomocí sledovacích algoritmů lze rovněž zjistit, zda se na následujícím snímku nachází stejný objekt z předcházejícího snímku s posunutím, nebo zda se jedná o zcela nový objekt. Problematika detekce a sledování pohybu je podrobněji popsána v kapitole 3.

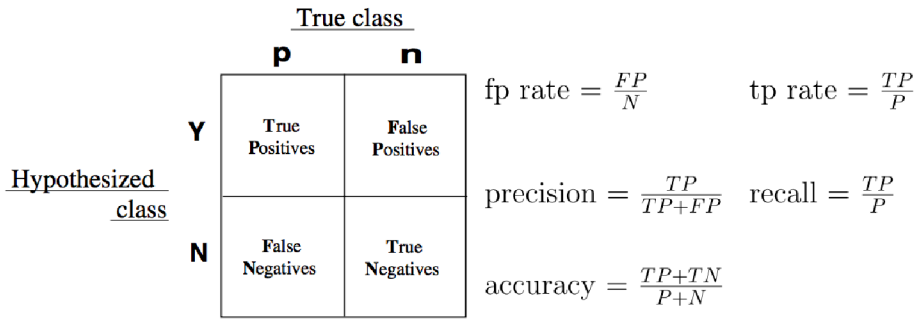
Mezi výstupní data detektoru a extraktor příznakového vektoru je obvykle zasazen krok, ve kterém jsou prováděny detekce klíčových bodů tváře. Klíčové body definují její natočení a na základě těchto informací je provedeno její geometrické vycentrování. Před extrakcí příznakového vektoru může být ještě provedena optimalizace rozlišení na úroveň vyhovující metodě extrakce nebo ořezání okrajů detekce [19]. Příznakový vektor představuje strojový popis detekované tváře. Je to množina skalárních hodnot, které popisují určité vlastnosti zkoumaného objektu. Pokud by byl příznakový vektor vytvářen člověkem, položky by mohly znamenat například barvu vlasů, barvu očí, účes, tvar nosu, vrásky nebo další vlastnosti.

Pro tento účel se velice často využívají speciálně navržené hluboké konvoluční neuronové sítě. Jejich vstup představuje normalizovaný obrázek tváře, na jehož základě dokáže neuronová síť vytvořit jeho strojový popis [20]. Neuronové sítě vytvořené za účelem extrakce příznakového vektoru ze snímku tváře byly natrénovány způsobem, který během procesu trénování přiřadil strojový význam položkám příznakového vektoru. Problematika normalizace a extrakce příznakového vektoru je podrobněji popsána v kapitole 4.

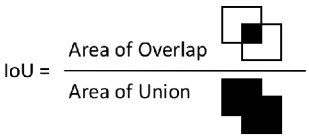
Identifikace osoby se zakládá na porovnání příznakového vektoru osoby neznámé identity s databází příznakových vektorů známých tváří [20]. Metrik pro porovnání vzdálenosti příznakových vektorů existuje více, ale v oblasti rozpoznávání tváře se obvykle pracuje s výpočtem euklidovské vzdálenosti. O shodě dvou příznakových vektorů rozhoduje jejich vzdálenost. Pro správnou funkci systému je klíčové zvolit vhodnou prahovou vzdálenost pro shodu. Specifikace opět požaduje zachytit hledanou osobu ve videu i za cenu falešných upozornění.

2.2 Vhodné metriky porovnání metod

Statistické vyhodnocení úspěšnosti komponent celého systému je klíčem k analýze jejich chování a následnému výběru nejvhodnější metody. Před samotným testováním je vhodné zamyslet se nad volbou vhodných metrik, které umožňují získat dostatek potřebných informací o testovaných metodách. ROC křivka [6] je jedna z nejpoužívanějších metrik pro hodnocení a optimalizaci binárního klasifikátoru. Binární klasifikátor [6] při detekci tváře rozhoduje o tom, zda se na dané souřadnici vyskytuje nebo nevyskytuje lidská tvář. V případě úlohy rozpoznávání klasifikátor rozhoduje o tom, zda je na dvou fotografiích obličeje stejný člověk, nebo zda se jedná o různé osoby.



Obrázek 2.2: Rozdělení množin binární klasifikace. (převzato z [6])



Obrázek 2.3: Výpočet hodnoty IoU(Intersection over Union).

Při binární klasifikaci je možné rozdělit výsledky testu do 4 množin. Podle toho, zda u daného vzorku očekáváme pozitivní (positives) nebo negativní (negatives) výsledek, můžeme množinu vzorků rozdělit na 2 části. Vzniklé množiny lze ještě rozdělit podle toho, zda klasifikátor odpověděl podle očekávání (true positives/negatives), nebo zda došlo při klasifikaci k chybě (false positives/negatives). Rozdělení vzorků do těchto množin vyjadřuje matice záměn na obrázku 2.2. ROC křivka vyjadřuje vztah mezi senzitivitou (true positive rate) a specificitou (true negative rate). Senzitivita je poměr správných a pozitivních výsledků ke všem pozitivním vzorkům. Specificita je poměr správně negativních výsledků ke všem negativním vzorkům. [6]

Většina klasifikátorů vrací kromě binární odpovědi ještě hodnotu vyjadřující míru jistoty ve správnost daného výsledku. Bod ROC křivky je vždy vztažený k této hodnotě. Různým nastavením prahu lze získat body ROC křivky a křivku graficky zobrazit. Při úloze detekce tváře lze ROC křivku vytvořit na základě prahové hodnoty IoU (Intersection over union) [12]. Z obrázku 2.3 je patrné, že se jedná o poměr průniku plochy detekce a anotace se součtem plochy detekce a anotace. Precision - recall křivka je pouhou variantou ROC křivky. Hodnota precision vyjadřuje poměr správně detekovaných výsledků ke všem detekcím. Hodnota recall vyjadřuje poměr správně detekovaných vzorků ke všem vzorkům, které měly být detekovány. Úspěšnost metody je tím vyšší, čím je vyšší obsah plochy pod ROC křivkou. Nastavováním různých prahových hodnot se snažíme určit vhodný kompromis mezi množstvím falešně negativních výsledků a falešně pozitivních výsledků. [6]

Metodu lze posuzovat podle plochy pod ROC křivkou. Tato hodnota bývá označována AUC (area under curve). Metody s hodnotou AUC blížící se 1 dosahují vynikajících výsledků. Naopak pokud metoda dosahuje hodnoty AUC rovné 0.5, není její úspěšnost lepší než hod mincí. [20]

2.3 Výběr vhodných datových sad

Funkce metod detekce i identifikace je negativně ovlivňována mnoha nepříznivými vlivy. Mezi tyto vlivy patří různé rozlišení fotografií, barevná hloubka fotografie nebo nízká kvalita osvětlení tváře. Dalším negativním faktorem jsou různá natočení obličeje, zakryté části, sluneční brýle nebo pokrývky hlavy. U metod lze specifikovat množství parametrů, které mají vliv na úspěšnost, ale i výpočetní náročnost. Cílem testování je získat podrobné informace o chování existujících přístupů a zjistit, jaké metody a ve které konkrétní konfiguraci plní nejvhodněji zadanou úlohu. K tomu je nezbytné provést průzkum existujících datových sad a pro účely testování zvolit datovou sadu věrně napodobující reálné podmínky.

Datové sady pro detekci a rozpoznávání

Mezi nejznámější datové sady z oblasti detekce tváře patří datová sada *FDDB (Face Detection Data Set and Benchmark)* [12]. Zajímavostí této datové sady je odlišnost v anotaci plochy, na které se nachází obličej. Tato plocha je určena pomocí elipsy namísto čtvercového ohraničení. To umožňuje provádět přesnější analýzu a porovnávání detektorů. Vyhodnocení pomocí ROC křivky probíhá na základě prahové hodnoty IoU (Intersection over Union). Datová sada sestává z 2845 fotografií s 5171 anotovanými tvářemi.

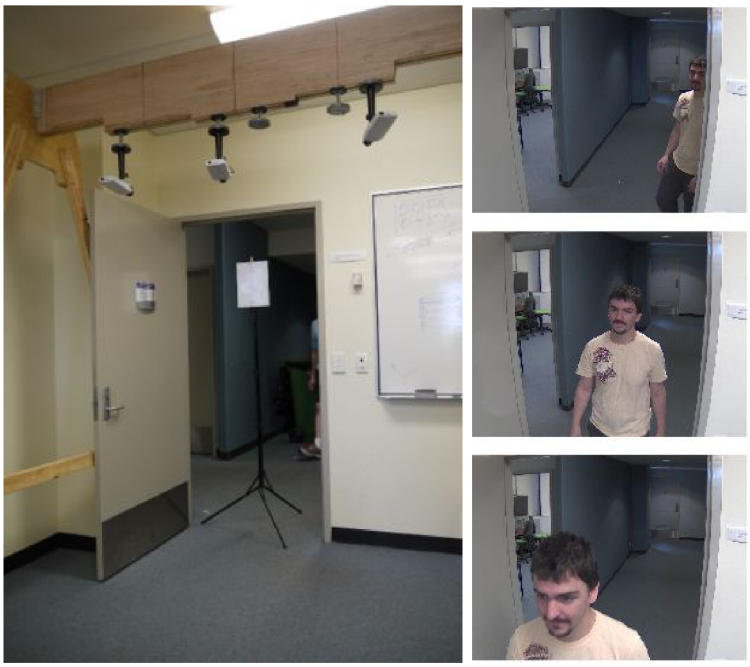
Jednou z největších datových sad určených pro testy detekce tváře je datová sada *Wider face* [24]. Při jejím vytváření byl kladen důraz na skladbu odpovídající mnoha reálným situacím. Obsahuje bohaté anotace zahrnující popis stínů, natočení tváře, ohraničující obdélníky a kategorie situací, ze kterých daný snímek pochází. Skládá se z 32 203 obrázků s celkovým výskytem 393 703 anotovaných tváří.

Známou datovou sadou z oblasti rozpoznávání tváře je datová sada *LFW (Labeled Faces in the Wild)* [11]. Tato datová sada sestává z 13000 anotovaných fotografií z internetových zdrojů. Celkově se na ní vyskytuje 1680 různých osob. Fotografie této datové sady jsou v příliš dobrém rozlišení a kvalitě oproti datům, pro které je aplikace určena.

Vhodná datová sada pro účely testování by měla být podobná datům z reálného provozu. Měly by se zde vyskytovat náročné světelné podmínky a nízká kvalita záznamů nejlépe z dohledových kamer. Osoby na záznamech by měly být různých národností a jejich oblečení by se mělo skládat i z prvků, které zakrývají části obličeje. Menší datová sada *ChokePoint* [23] v porovnání s ostatními datovými sadami pravděpodobně nejlépe odpovídá podmínkám reálného nasazení systému a z tohoto důvodu jsem ji využil k porovnání metod detekce i identifikace.

Tato datová sada byla vytvořena pro experimenty v oblasti reidentifikace osob za podmínek odpovídajících reálným dohledovým systémům. Vyskytují se zde snímky s náročnými světelnými podmínkami, různým natočením tváří a ostrostí obrazu. Je tvořena 48-mi sekvencemi videozáznamů, které byly pořizovány 3-mi kamerami umístěnými nad dveřmi v odlišných úhlech. V datové sadě se vyskytuje celkem 25 různých osob (19 mužů a 6 žen) v první části a 29 různých osob (23 mužů a 6 žen) v části druhé. Snímky byly pořízeny v rozlišení 800x600px s počtem 30 snímků za vteřinu. Na prvních 100 snímcích každé sekvence je pouze prázdné pozadí. Celkový počet snímků sekvence je 100 000 a na 64 000 snímcích se vyskytuje obličej. Osoby přicházejí ke dveřím z různých úhlů a procházejí směrem dovnitř. Data se podobají záznamům z interního kamerového systému. Anotace nese pro každou osobu souřadnice levého a pravého oka a její identifikátor. Ukázka dat je na obrázku 2.4.

Datová sada je určena primárně pro úlohu rozpoznávání, ale její anotace lze s drobnými odchylkami aplikovat i pro úlohu detekce tváře. Anotace této datové sady nesou pro každý



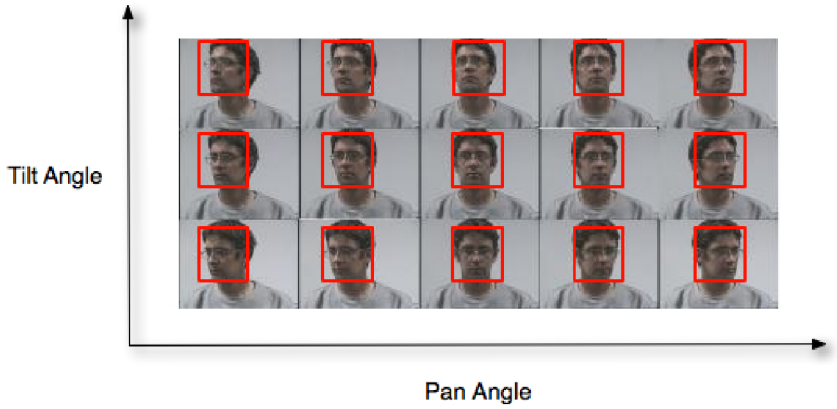
Obrázek 2.4: Ukázka dat z datové sady ChokePoint. (převzato z [23])

snímek informace o pozici levého a pravého oka osob ze záznamu. Výsledkem metod detektorů tváře je obdélník ohraničující oblast detekce. S ohledem na anotace datové sady *ChokePoint* [23] lze testovat, zda se souřadnice očí nacházejí uvnitř obdélníku detekce. Problémem je, že anotace jsou dostupné pouze pro osoby, které procházejí dveřmi. Na snímcích dochází ale i k výskytu jiných osob, což negativně ovlivňuje reprezentativnost výsledků testování. Problém jsem vyřešil poloautomatickým určením inkriminovaných snímků a jejich vyřazení z testovacích dat.

Datové sady zkoumající vliv natočení tváře

V praxi se lze běžně setkat se situací, kdy osoby ve videozáznamu nesledují objektiv kamery a jejich tvář s objektivem svírá určitý úhel. Natočení tváře výrazně ovlivňuje úspěšnosti metod detekce i identifikace. V případě, kdy je použita neinvariantní metoda detekce s invariantním identifikátorem, odolnost identifikátoru oproti natočení tváře zůstává nevyužita. Stejná situace vzniká i v případě opačném. Metody detekce i identifikace byly z tohoto důvodu podrobeny testování, jehož výsledkem je relevantní porovnání jejich robustnosti oproti tomuto jevu.

Srovnání lze provést na datových sadách vytvořených speciálně pro tento problém. Datových sad s anotacemi, které nesou informace o souřadnicích, identifikátoru a úhlu natočení tváře, je dostupných několik. Při testování jsem využil datovou sadu *Head Pose Image Dataset* [7], která se skládá z 2790 fotografií 15 různých osob. Ke každé osobě existují 2 série fotografií po 93 snímcích. Varianty natočení pokrývají horizontální i vertikální úhly v rozpětí od -90° do 90° po krocích 15° pro vertikální úhel a 30° pro horizontální úhel (obrázek 2.5). Série fotografií stejné osoby se odlišují různým účesem, oblečením nebo brýlemi. Anotace nesou informace o souřadnicích tváře, identifikátoru osoby, číslu série, pořadovém číslu fotografie a horizontálním a vertikálním úhlu natočení.



Obrázek 2.5: Ukázka dat z datové sady Head Pose Image Dataset (převzato z [7])

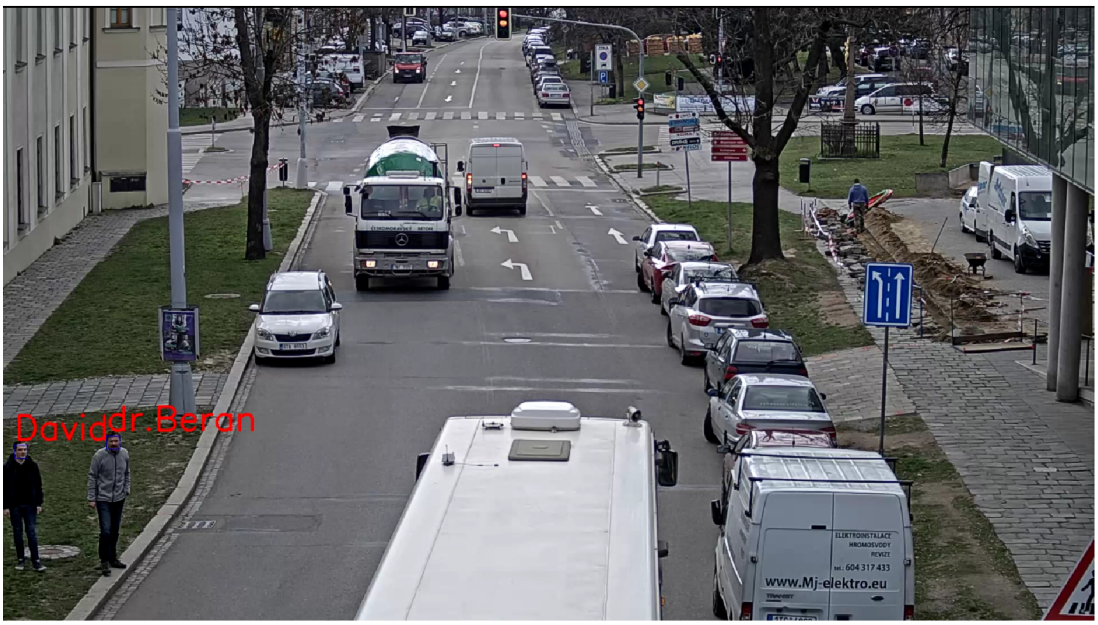
K relevantnímu testování metod detekce bylo nutné datovou sadu *Head Pose Image Dataset* [7] podrobněji popsanou v předchozí kapitole upravit. Úprava datové sady spočívala v doplnění 200 fotografií bez výskytu obličejů, aby zde byl větší prostor pro falešné detekce. Doplňující fotografie byly získány z datové sady *ChokePoint* [23]. Na několika prvních vteřinách těchto záznamů se vyskytují pouze prázdné záběry vnitřních prostor.

Vlastní datová sada

Účelem vytváření vlastní datové sady je ověření funkcionality aplikace na datech odpovídajícím reálnému nasazení. Datová sada by měla pomoci odhalit, zdali se podařilo splnit požadavky ze zadání. Před jejím pořízením je důležité zamyslet se nad podobou nahrávaných dat.

Každá hledaná osoba by měla být specifikována několika fotografiemi, nejlépe s rozmanitým stářím. Videozáznamy monitorovaného prostoru by měly být nahrávány ve Full-HD rozlišení. Toto rozlišení lze běžně získat z kamerového systému a pro další experimenty je zde možnost jeho umělého snížení. Pro testovací účely je vhodné, aby byly záznamy pořízeny za různých světelných podmínek. Hledaná osoba by měla pod kamerou procházet v odlišných úhlech z důvodu testování robustnosti oproti natočení. Je vhodné, aby hledaná osoba procházela pod kamerou v různých vzdálenostech od objektivu. Vzhledem k relevantní analýze je důležitý výskyt neznámých osob na záznamu.

Celkově byly pořízeny dva záznamy o délce 7 a 8 minut. Záznamy se liší v pohledu na odlišnou stranu ulice. Vyskytují se na nich 2 známé osoby a větší množství neznámých osob. Osoby realizují několik různých průchodů odpovídajících příchodu po různé straně chodníku nebo přecházení ulice v různé vzdálenosti od objektivu kamery. Odlišné úrovně osvětlení jsou způsobeny pozicí slunce. Některé z průchodů jsou narušeny větvemi stromů. Ukázka pořízených dat je na obrázku 2.6. K pořízené datové sadě byla vytvořena jednoduchá anotace, která poskytuje informace o identifikátoru známé osoby a o časovém úseku, kdy se tato osoba nacházela na záznamu.



Obrázek 2.6: Ukázka dat z vlastní datové sady.

Kapitola 3

Detekce a sledování tváře ve videu

Detekce tváře je jedna z důležitých úloh v oblasti počítačového vidění a zpracování obrazových dat. V posledních letech zažívá rychlého rozvoje z důvodu nárůstu výpočetního výkonu počítačů a dalších zařízení. Vzhledem k stále sofistikovanějším algoritmům detekce obličeje je možné spustit podobný software i na obyčejném mobilním zařízení. Detekce obličeje může být integrována do fotoaparátu a umožňuje tím automatické zaostřování fotografií podle nalezených obličejů na snímku. Moderní počítače nebo telefony umožňují odemknutí a přístup k citlivým datům na základě fotografie uživatele. V tomto případě se jedná o kamerový systém, který umožňuje automatické prohledávání videozáznamů a pomáhá policii dopadnout hledanou osobu.

Řešením celé úlohy je označit na fotografii místo, na kterém se nachází obličej. Na první pohled by se mohlo zdát, že se jedná o poměrně jednoduchou úlohu. Pro počítač je fotografie jenom množství digitálních hodnot s určitým významem a určit pozici tváře pouze na základě těchto hodnot je velice složité.

3.1 Algoritmus Viola-Jones

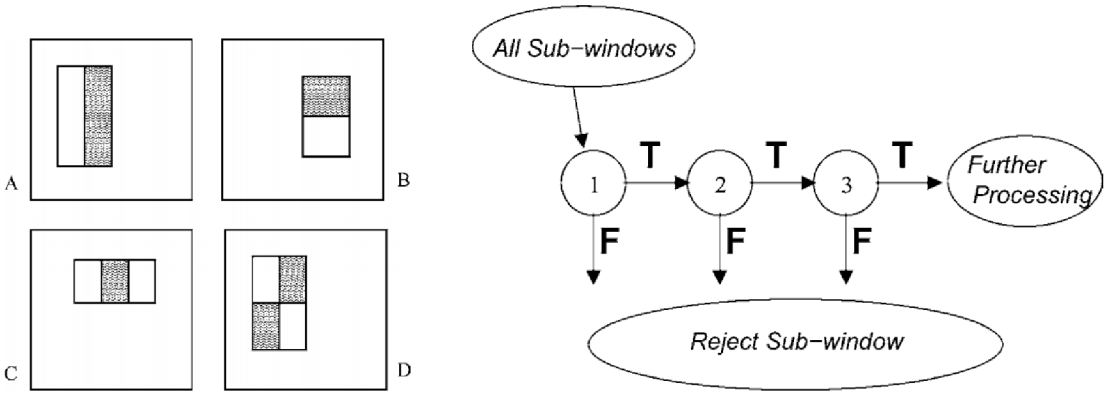
Pravděpodobně první metoda detekce obličeje ve videozáznamu umožňující zpracování v reálném čase byla publikována v roce 2001 Paulem Violou a Michaelem Jonesem[1].

Základem detektoru je extrakce tzv. Haarových příznaků [22] zobrazených na obrázku 3.1. Haarovy příznaky si můžeme představit jako dva nebo čtyři obdélníky. Tyto obdélníky mohou být bílé nebo černé, mohou se po fotografii libovolně pohybovat a mohou mít libovolnou velikost. Haarův příznak je číselná hodnota získaná z obrázku ve stupních šedé jako rozdíl součtu hodnot pixelů pod bílým obdélníkem a součtu hodnot pixelů pod černým obdélníkem. I pro obrázek s relativně malým rozlišením lze touto cestou extrahovat množství příznaků.

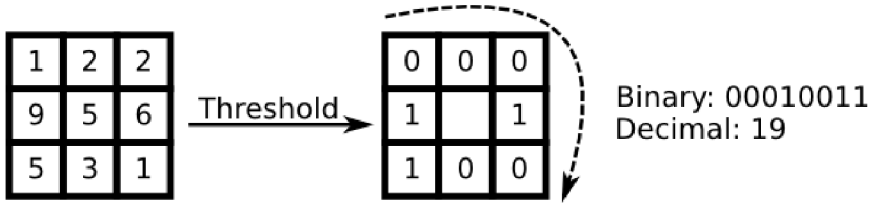
Pro zlepšení výpočetní efektivity tohoto algoritmu bylo využito převodu fotografie na strukturu integral image [25]. Výsledkem tohoto procesu je matice hodnot pixelů získaná z matice předchozí tak, že hodnota libovolného pixelu v nové matici je rovna součtu všech pixelů vlevo nebo nahoře v původní matici. Nově vzniklá matice integral image umožňuje výrazně nižší počet aritmetických operací při extrakci Haarových příznaků.

Další inovací je využití upraveného algoritmu AdaBoost [25]. Detektor využívá rozhodovacího stromu (obrázek 3.1) s aplikací série testů pomocí binárních klasifikátorů tvář/bez tváře. Tyto klasifikátory seskupují příznaky do několika skupin. V prvních skupinách jsou

[1] https://en.wikipedia.org/wiki/Viola%E2%80%93Jones_object_detection_framework



Obrázek 3.1: Ukázka Haarových příznaků a grafické znázornění rozhodovacích stromů. (převzato z [22])



Obrázek 3.2: Ukázka výpočtu příznaku LBP. (převzato z [4])

testy na příznaky spíše obecnější. Testy na skupiny jsou aplikovány postupně a pokud libovolný z nich selže, obrázek je vyřazen a nepokračuje se dále. Cílem je zkombinovat výsledky z většího množství slabých klasifikátorů a dosáhnout tím lepších výsledků.

3.2 Další metody založené na extrakci příznaků

Následující metody detekce tváře jsou založeny na podobném principu. Z inkriminované oblasti ve fotografii provádějí extrakci příznakového vektoru. Extrahovaný příznakový vektor je vstupem pro binární klasifikátor rozhodující o výskytu tváře na daných souřadnicích.

LBP detektor

Local binary pattern [1] je dalším typem příznaku používaného pro klasifikaci ve strojovém vidění. Při výpočtu příznakového vektoru typu LBP v pokročilé podobě v kombinaci s histogramy podobně jako u příznaků HOG [5] je obrázek procházen po buňkách o velikosti 3x3 pixely. Algoritmus pracuje s obrázky ve stupních šedi. Každá buňka je reprezentována 8-bitovým číslem, které bylo získáno porovnáváním hodnoty středového pixelu s hodnotami pixelů sousedních. Výpočet je znázorněný na obrázku 3.2. Pokud byla hodnota pixelu uprostřed buňky větší než u pixelu sousedního, v binární reprezentaci je tento stav reprezentován hodnotou 1. V opačném případě je to hodnota 0. Níže uvedená rovnice popisuje tento výpočet formálně.

$$LBP(x_c, y_c) = \sum_{p=0}^{P-1} 2^P s(i_p - i_c)$$

Hodnoty x_c a y_c představují souřadnice středového pixelu s velikostí intenzity i_c, sousední pixel je vyjádřen intenzitou i_n a znaménková funkce s je definovaná předpisem níže.

$$s(x) = \begin{cases} 1 & pokud \quad x \geq 0 \\ 0 & jinak \end{cases}$$

Příznakový vektor je tvořen normalizovaným histogramem výskytu 8-bitových čísel vypočtených pro každou buňku. Z toho vyplývá, že bude dosahovat 256 dimenzí (8-bitové číslo může nabývat 256 různých hodnot). Příznakový vektor typu LBP je později klasifikován pomocí SVM klasifikátoru [25] (nebo jiného algoritmu strojového učení) a tím je určeno, zda se na obrázku nachází obličej.

HOG detektor

Koncept metody HOG (Histogram of oriented gradients) [5] byl poprvé popsán již v roce 1986 Robertem K. McConnelem[2]. Většího rozšíření dosáhla až v roce 2005, kdy byla tato metoda využita k detekci chodců ve videu a prezentována na konferenci Computer Vision and Pattern Recognition (CVPR).

Základní myšlenkou je využití lokální intenzity světelného gradientu na obrázku v odstínech šedé k popisu objektů. Světelný gradient je vektor, který vyjadřuje změnu intenzity nebo barvy pixelu. Prvním krokem při výpočtu HOG příznaku je určení vodorovných a svislých gradientů aplikací Sobelova operátoru na matici s intenzitami pixelů. Z těchto hodnot lze získat údaje o velikosti a směru vektoru gradientu dosazením do následujícího vzorce. Hodnoty g_x a g_y představují velikost vodorovného a svislého gradientu na daném pixelu.

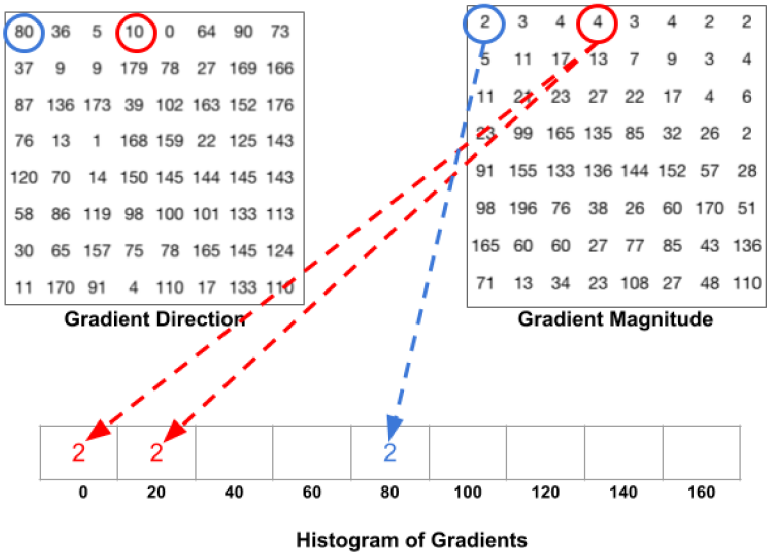
$$\sqrt{g_x^2 + g_y^2}$$

$$\phi = arctan \frac{g_y}{g_x}$$

Úhly vektorů gradientu jsou pouze v rozmezí 0 - 180 (bezznaménkový gradient) namísto 0 - 360 (znaménkový gradient). V praxi se ukázalo, že bezznaménková varianta dosahuje lepších výsledků. Výsledkem tohoto kroku je histogram s devíti položkami, které odpovídají úhlům 0, 20, 40, ... 160. Pro každý vektor gradientu je podle jeho úhlu určeno odpovídající pole histogramu a je k němu přičtena hodnota jeho délky (obrázek 3.3). Fotografie je při prohledávání rozdělena na určitý počet buněk. Pro každou buňku je proveden výpočet histogramu a výsledný příznakový vektor je konkatenací histogramů vypočtených pro každou buňku odděleně.

Při vyhledávání obličeje na fotografii metodou HOG je fotografie rozdělena na menší části. Pro každou z těchto částí je extrahován příznakový vektor typu HOG a pomocí klasifikátoru je určeno, zda se na fotografii nachází obličej. V praxi jsem se setkal s kombinací extraktoru HOG příznaků a SVM [25] (Support vector machine) klasifikátoru.

[2] https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients



Obrázek 3.3: Výpočet příznaku typu HOG. (převzato z webu[3])

Support vector machine

Klasifikace je ve strojovém učení druh problému, jehož cílem je přiřadit novému vzorku kategorii na základě trénovacích vzorků, jejichž kategorie je známa. Support vector machines [25] je algoritmus strojového učení realizující klasifikaci.

Tento přístup umožňuje oddělit množiny vzorků dvou různých tříd nalezením vhodné nadroviny, která rozděluje prostor příznaků tak, že trénovací data odlišných tříd jsou touto nadrovinou oddělena (obrázek 3.4). Nadrovina dimenze n je jakýkoliv její podprostor s dimenzí n-1. Příznaky jsou označovány jako tzv. support vectors a definují diskriminační funkci. Cílem procesu učení je najít diskriminační funkci, která obě třídy rozděluje rovnoměrně, aby se v okolí diskriminační funkce nacházel největší možný prostor bez výskytu trénovacích dat.

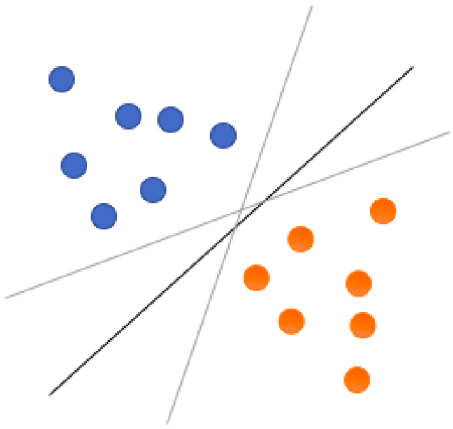
Tento algoritmus byl původně vytvořen pro lineární klasifikaci do dvou tříd. Nadrovinu lze v tomto případě popsat obecnou rovnicí přímky. Součet nejkratší vzdálenosti bodů na každé straně přímky představuje šířku hraničního pásma. Cílem trénování je nalézt diskriminační funkci takovou, aby bylo hraniční pásmo největší možné. Toho je dosaženo aplikací Lagrangeovy funkce.

Pokud řešíme lineárně neseparovatelnou úlohu, je nutné využít jádrové transformace prostoru příznaků dat do prostoru transformovaných příznaků vyšší dimenze. Tato tzv. kernel transformation umožňuje převést původně lineárně neseparovatelnou úlohu na lineárně separovatelnou.

3.3 Neuronové sítě

V současné době jsou dosahovány nejlepší výsledky v oblasti detekce obličeje aplikací neuronových sítí a existuje množství detektorů obličejů na nich založených [12]. Při své práci jsem se věnoval detektoru publikovaném v dokumentu Joint Face Detection and Alignment

[3] https://www.learnopencv.com/histogram-of-oriented-gradients/



Obrázek 3.4: Ukázka lineárního oddělení prvků dvou množin.

using Multi-task Cascaded Convolutional Networks [9] a detektoru tváří implementovaném v knihovně *Dlib* [14].

CNN detektor

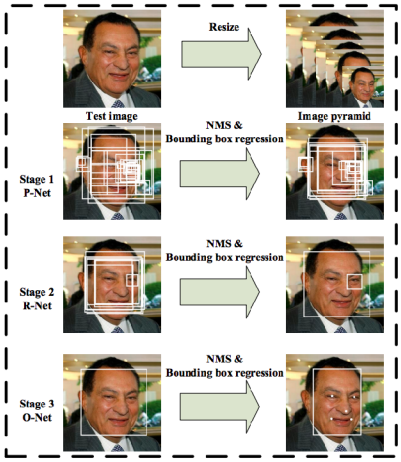
Knihovna *Dlib* [14] obsahuje předpřipravený natrénovaný model a další nástroje pro detekci obličeje v obrázku. Architektura neuronové sítě z této knihovny umožňuje procházet obrázek a pro každou jeho část zjišťuje, zda se na ní nachází hledaný objekt. V implementaci z knihovny *Dlib* je místo pohyblivého okna využíváno algoritmu MMOD (Max-Margin Object Detection) [15], který umožňuje plynulé zpracování vstupních dat bez rozdělení na menší okna. Výstupem je jediné číslo, jehož hodnota určuje úroveň pravděpodobnosti nalezení shody na daném místě v obrázku.

MTCNN detektor

Autoři výše zmíněného dokumentu [9] spojují proces rozpoznávání klíčových bodů obličeje a jeho detekci v jednu úlohu. Úloha je řešena pomocí konvoluční neuronové sítě s využitím metody multi-task learning. Multi-task learning představuje způsob současného řešení více úloh za využití společných částí výpočtu. Díky tomu lze dosáhnout lepších výsledků a účinnosti trénování neuronové sítě.

Dříve zmiňovaný algoritmus detekce Viola-Jones využívá Haarových příznaků a algoritmu AdaBoost pro trénování kaskádových klasifikátorů. Umožňuje zpracování dat v reálném čase, ale nedosahuje vždy uspokojivých výsledků při všech úlohách detekce. Konvoluční neuronové sítě dosahují velice dobrých výsledků v úlohách detekce obličeje, ale z důvodu jejich složité struktury je zpracování pomocí nich výpočetně náročné.

Neuronová síť byla autory rozdělena na 3 části (obrázek 3.5). V první části jsou v obrázku rychle vyhledávána okna, ve kterých je vysoká pravděpodobnost výskytu obličeje, pomocí malé a výpočetně nenáročné neuronové sítě. V další části jsou detekovaná okna dále zkoumána neuronovou sítí se složitější strukturou a jejich množství je výrazně zredukováno. V poslední části je pomocí neuronové sítě s nejsložitější strukturou určena konečná poloha obličeje a 5 landmark bodů [9].



Obrázek 3.5: Diagram znázorňující činnost MTCNN detektoru. (Převzato z [9])

3.4 Algoritmy sledování pohybu

Sledování pohybu objektu je proces určování polohy sledovaného objektu na základě posloupnosti snímků videozáznamu. V oblasti počítačového vidění se jedná o široký problém s velkým množstvím algoritmů jeho řešení. Sledovací algoritmy [25] u objektů pracují s jejich pohybovým a vzhledovým modelem. Pohybový model obsahuje informace o aktuální poloze objektu a o směru pohybu modelu. Z těchto údajů lze vypočítat přibližnou polohu na dalším snímku. Vzhledový model nese kódovanou informaci o vzhledu sledovaného objektu. Kombinací údajů z pohybového a vzhledového modelu lze docílit větší přesnosti výpočtu aktualizované polohy. Na novém snímku se nejprve identifikuje předpokládaná pozice objektu podle pohybového modelu. V blízkém okolí této pozice je porovnáván vzhled oblasti se vzhledovým modelem objektu a na základě toho jsou určeny přesnější souřadnice. Porovnání vzhledového modelu s částí snímku se provádí pomocí klasifikátoru, který byl natrénován pomocí dřívějších výskytů objektu. Algoritmy lze rozdělit do několika skupin podle počtu sledovaných objektů nebo způsobu interního zpracování[4].

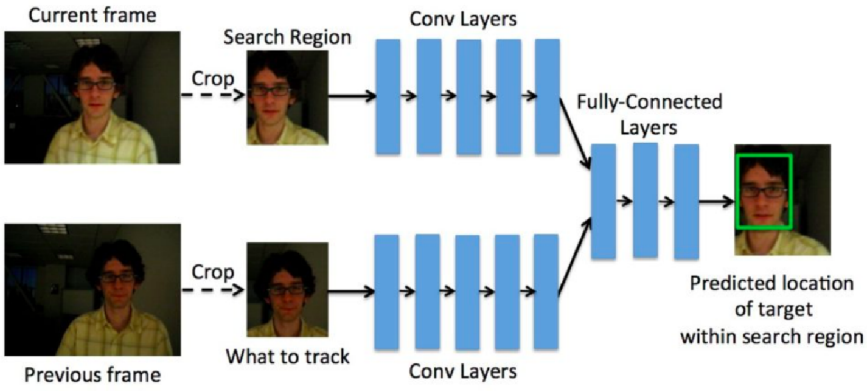
MOSSE tracker

Knihovna *Dlib* obsahuje velice dobrou implementaci MOSSE trackeru z dokumentu Visual Object Tracking using Adaptive Correlation Filters [3]. Tento tracker využívá koleračních filtrů založených na výpočtu Minimum Output Sum of Squared Error (MOSSE) a aplikace Fourierovy transformace. Tracker je schopen inicializace pouze na základě jediného snímku sledovaného objektu. Je velice dobře robustní oproti vlivům změn osvětlení, měřítka sledovaného objektu a dalších geometrických deformací. Jeho nespornou výhodou je rychlost zpracování přesahující 600 FPS a velice přesné výsledky.

GOTURN tracker

Při průzkumu metod sledování pohybu mě dále zaujala implementace GOTURN trackeru [10] z knihovny *OpenCV* [4]. Tracker je založený na použití neuronové sítě. Neuronová síť očekává na vstupu oříznutý předcházející a aktuální snímek sledovaného objektu.

[4]https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/



Obrázek 3.6: Zjednodušené schéma architektury neuronové sítě GOTURN trackeru. (převzato z [10])

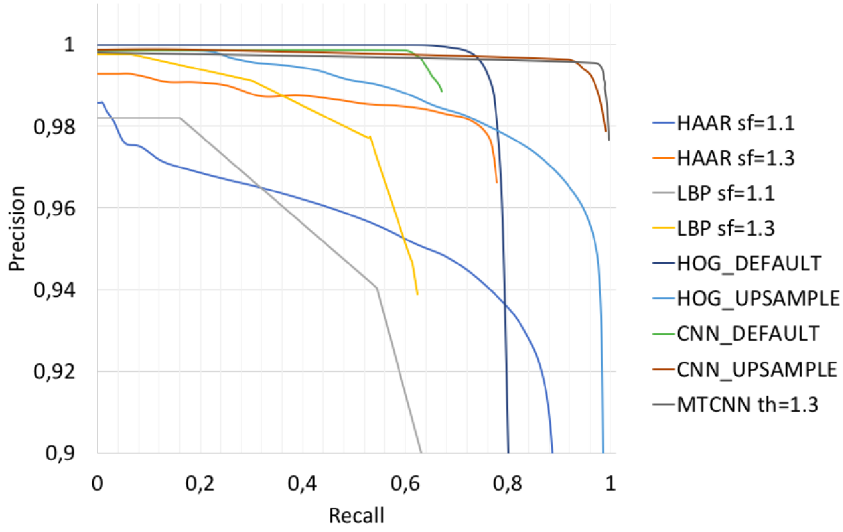
Na předcházejícím snímku je označen sledovaný objekt a neuronové síti je předáván i se svým okolím. Z aktuálního snímku je oříznuta stejná oblast jako ze snímku předcházejícího. Cílem je najít ohraničení objektu na aktuálním snímku. Výsledkem poslední vrstvy jsou 4 čísla, která představují souřadnice hranic objektu. Na obrázku 3.6 je zobrazeno zjednodušené schéma architektury neuronové sítě, kterou tracker využívá.

BOOSTING tracker

Za zmínku stojí ještě BOOSTING tracker [8], který interně využívá algoritmu AdaBoost [25]. AdaBoost je algoritmus strojového učení, který dokáže pomocí kombinace výsledků z více klasifikátorů získat přesnější výsledek. Klasifikátor je trénován za provozu na základě pozitivních a negativních příkladů sledovaného objektu. Na začátku jsou uživatelem (nebo jiným způsobem) definovány hranice sledovaného objektu a jsou považovány za jeho pozitivní příklad. Ostatní části snímku jsou považovány za příklad negativní. Při předložení dalšího snímku klasifikátor prochází okolí předchozí polohy a hledá pozici s největší pravděpodobností shody. Z dalšího snímku jsou dostupné nové pozitivní a negativní příklady objektu. Algoritmus je už spíše zastaralý a využívají se jiné.

3.5 Vyhodnocení testů a porovnání

Funkci metod detekce zmíněných výše jsem otestoval na datové sadě *ChokePoint* [23]. Výstupem testů jsou precision - recall křivky popisující jejich chování v různých konfiguracích. Porovnání úspěšnosti je souhrnně zobrazeno v grafu na obrázku 3.7. Z obrázku je zřejmé, že metody založené na konvolučních neuronových sítích [14] [9] dosahují výrazně lepších výsledků oproti ostatním přístupům. V případě tolerance vyššího množství falešných detekcí lze dosáhnout vynikajícího pokrytí ještě s detektorem HOG [5]. Bližší podrobnosti k jednotlivým metodám a výsledky testů většího množství jejich konfigurací jsou k dohledání v dalším textu.



Obrázek 3.7: Porovnání precision - recall křivek všech testovaných metod.

Viola - Jones detektor

U implementace algoritmu prohledávání z knihovny *OpenCV* [4] lze nastavit několik parametrů. Parametr `minNeighbours` definuje minimální počet výskytu detekce na stejných souřadnicích, aby byla tato detekce považována za relevantní. Pomocí parametru `scaleFactor` lze nastavovat rozdíl v rozlišení sousedních snímků ve scale pyramid. Parametry `minSize` a `maxSize` určují minimální a maximální velikost hledané detekce. Nastavení posledních 3 parametrů ovlivňuje výpočetní náročnost a všechny parametry ovlivňují chování detekce.

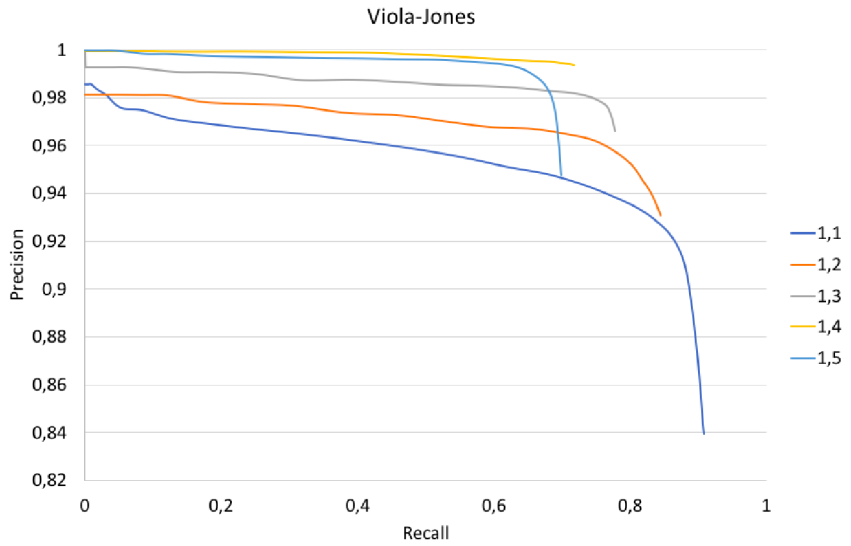
Při testování jsem využíval implementace detektoru Viola-Jones z knihovny *OpenCV*. Jednalo se o předtrénovaný kaskádový klasifikátor realizovaný funkcí `cv::CascadeClassifier::detectMultiScale`. Podle dokumentu [16] byl klasifikátor trénován na datové sadě s 5000 pozitivními a 3000 negativními snímky. Přesný název použité datové sady zde ale uveden není. Viola-Jones detektor se vyznačuje poměrně vysokou přesností detekce a nízkým počtem falešně pozitivních výsledků. Zároveň je poměrně citlivý na stíny a kvalitu osvětlení.

Vlivem různých konfigurací se počet zpracovaných snímků pohybuje v rozmezí 0,8 - 2,7 FPS na 1 jádru referenčního stroje MacBook Air 2017 při zpracování videa v rozlišení Full HD. Pro porovnání úspěšnosti této metody detekce v různých konfiguracích parametru `scaleFactor` byly sestaveny precision - recall křivky na obrázku 3.8.

LBP detektor

Detektor LBP je rovněž implementovaný v knihovně *OpenCV* [4] pomocí předtrénovaného klasifikátoru, který je realizovaný funkcí `cv::CascadeClassifier::detectMultiScale`. Parametry funkce jsou stejné jako v případě detektoru Viola-Jones. Z dokumentace lze zjistit, že byl trénovaný na datové sadě s 3000 pozitivními a 1500 negativními snímky. Další podrobnosti o datech nebo názvu datové sady nejsou dostupné.

Detektor je výpočetně velice nenáročný a lze u něj pozorovat malou citlivost na lokální změny osvětlení. K hlavním nevýhodám patří vyšší počet falešně pozitivních výsledků a celková nižší úspěšnost. Rychlost zpracování se na 1 jádru referenčního stroje MacBook Air 2017 u videa v rozlišení Full HD pohybovala v rozpětí 1,0 - 4,4 FPS podle zvolených para-



Obrázek 3.8: Precision - recall křivky pro různá nastavení parametru scale factor metody Viola-Jones.

metrů. Porovnání vlivu různých nastavení na úspěšnost metody vyjadřují precision - recall křivky na obrázku 3.9.

HOG detektor

Implementace detektoru z knihovny *Dlib* [14] využívá extrakce příznaků typu HOG v kombinaci s SVM klasifikátorem a interně obrázek prochází pomocí scale pyramid a posuvného okna. Detektor dosahuje nejlepších výsledků v porovnání s klasickými metodami. Oproti metodám, které využívají neuronových sítí, zaostává v detekci tváří s vysokým úhlem natočení. Parametr `upsample` umožňuje měnit minimální velikost detekcí s dopadem na výpočetní náročnost.

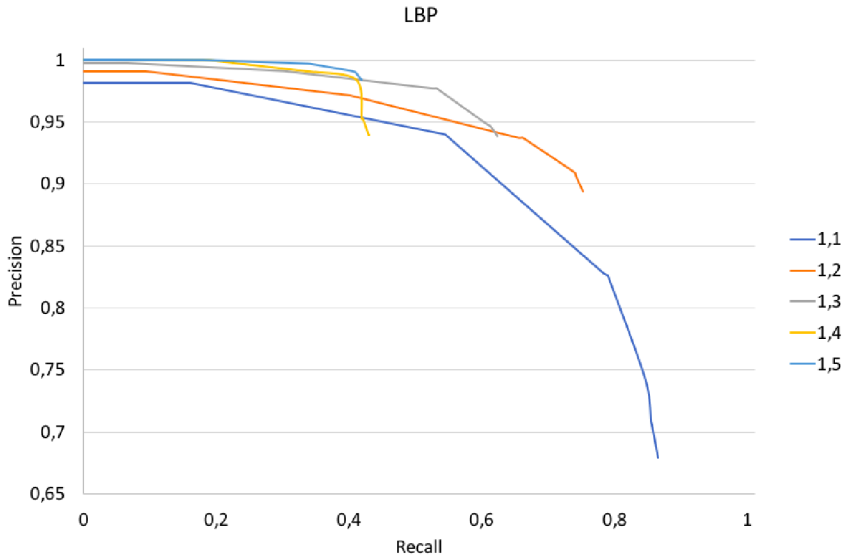
Rychlost zpracování se na 1 jádru referenčního stroje MacBook Air 2017 u videa v rozlišení Full HD pohybovala v rozpětí 0,8 - 2,7 FPS podle zvoleného parametru upsample. Chování metody vyjadřují precision - recall křivky na obrázku 3.10.

CNN detektor

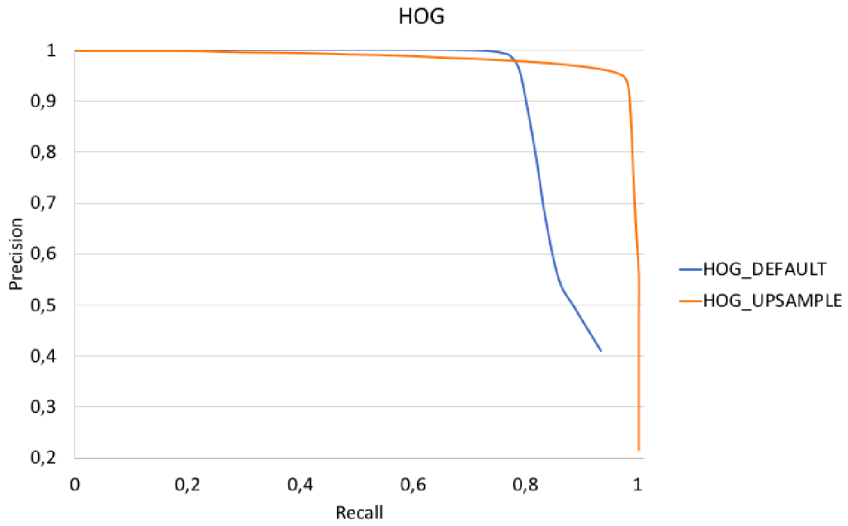
Implementace detektoru z knihovny *Dlib* umožňuje měnit stejně jako u metody HOG parametr `upsample` s dopadem na minimální velikost detekcí a výpočetní náročnost. Rychlost zpracování na 1 jádru referenčního stroje MacBook Air 2017 bez dedikované grafické karty, jejíž využití implementace z knihovny *Dlib* umožňuje, u videa v rozlišení Full HD dosahovala pouze 0,16 FPS. Bez možnosti akcelerace výpočtu na grafické kartě nelze tuto metodu uplatnit pro zpracování dat v reálném čase. Chování metody vyjadřují precision - recall křivky na obrázku 3.11.

MTCNN detektor

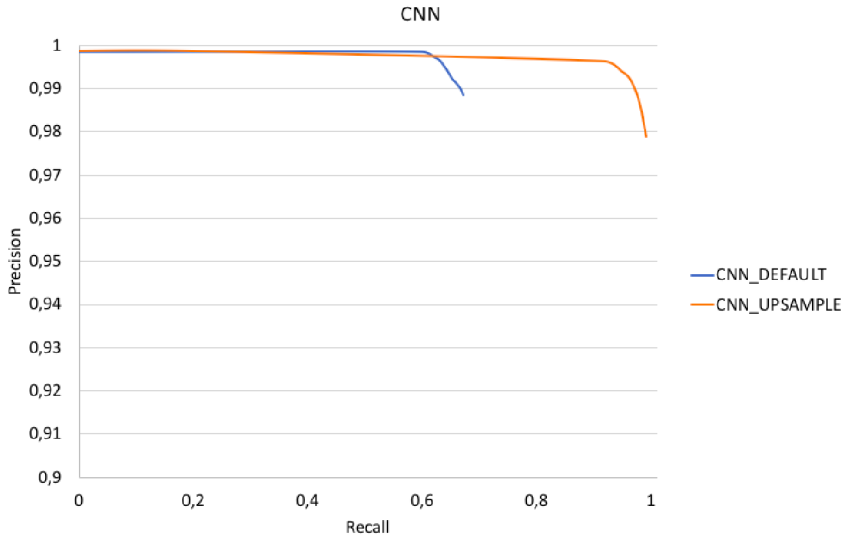
K praktickému nasazení této metody detekce lze využít volně dostupnou implementaci v jazyce Python využívající frameworku Caffe, nebo fakultní implementaci v jazyce C++, kte-



Obrázek 3.9: Precision - recall křivky pro různá nastavení parametru scale factor metody LBP.



Obrázek 3.10: Precision - recall křivky pro různá nastavení parametru upsample metody HOG.



Obrázek 3.11: Precision - recall křivky pro různá nastavení parametru upsample metody CNN.

rou do projektu poskytl její autor Ing. Michal Hradiš, Ph.D.. U fakultní implementace lze nastavovat parametr `thresholdScale`, `minSize` a `threadCount`. Parametr `thresholdScale` umožňuje zlepšit kvalitu detekce s následkem zhoršení výpočetní náročnosti. Parametr `minSize` je vhodné zvolit podle schopností metody identifikace, která detekci následuje.

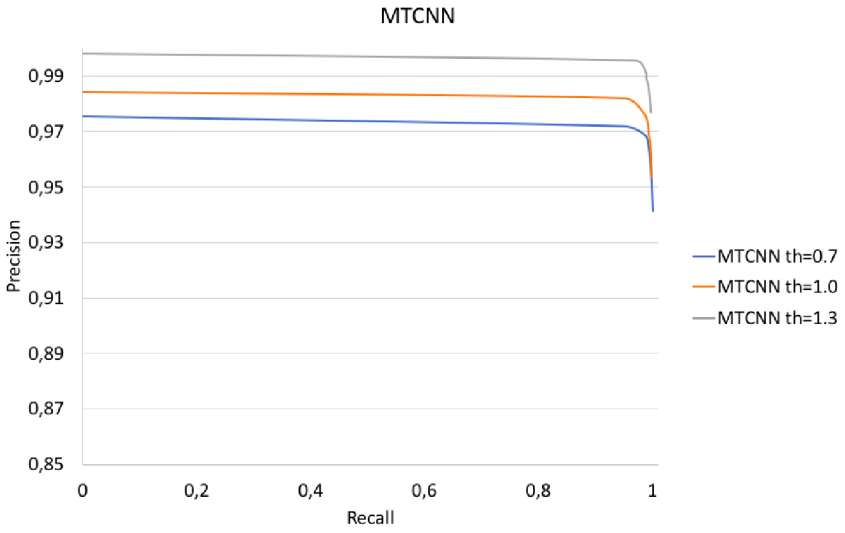
Detektor dosahuje na 1 jádru referenčního stroje MacBook Air 2017 u videa v rozlišení Full HD za využití výhradně prostředků CPU vynikající rychlosti 10 - 13 FPS. Na testovací datové sadě *ChokePoint* se nedopouští téměř žádných chyb. Další podrobnosti o chování metody vyjadřují precision - recall křivky na obrázku 3.12.

Invariance metod detekce k natočení

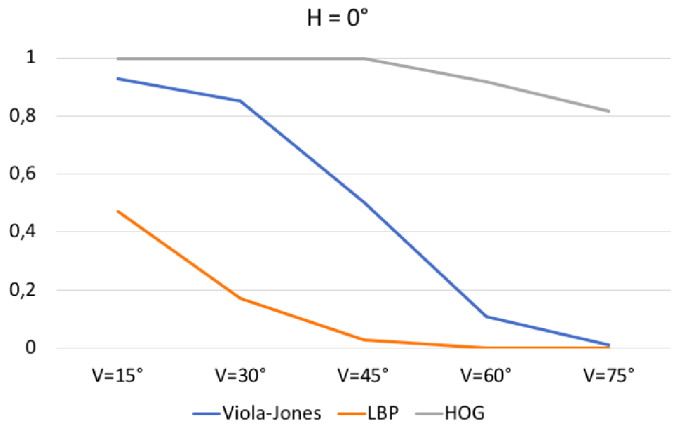
Odolnost metod oproti natočení tváří je jedna z klíčových vlastností rozhodující o možnostech jejich nasazení v praxi. Robustnost metod k tomuto jevu jsem otestoval na datové sadě *Head Pose Image Dataset* [7] doplněné o snímky bez výskytu tváře pocházející z datové sady *ChokePoint* [23]. Tím vznikl větší prostor pro výskyt falešných detekcí. Pro vhodnou reprezentaci výsledků jsem nejdříve ke každé metodě sestavil precision - recall křivky pro různé úhly natočení tváře s využitím pevné hodnoty IoU ≥ 0.5 [12]. Informace o robustnosti metod je ale mnohem lépe čitelná z grafů na obrázcích 3.13 a 3.14. Hodnoty v grafech popisují vliv natočení tváře na plochu pod precision - recall křivkou testované metody. Výsledky testů byly rozděleny vzhledem k velkým rozdílům mezi metodami založenými na neuronových sítích a ostatními přístupy do dvou obrázků odlišujících se hodnotou horizontálního úhlu natočení.

Výpočetní náročnost

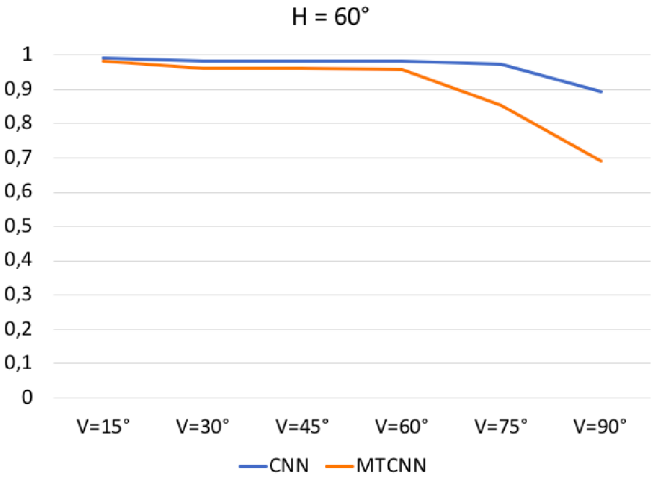
Důležitou vlastností testovaných metod je i jejich výpočetní náročnost kvůli podmínce na zpracování dat v reálném čase. Rychlost zpracování snímků odpovídá průměrné hodnotě pro vstupní data o rozlišení Full-HD na 1 ze 4 virualizovaných jader procesoru Intel-i5@1.8GHz. Z výsledků na obrázku 3.15 je patrné, že je pro všechny testované metody klíčová minimální



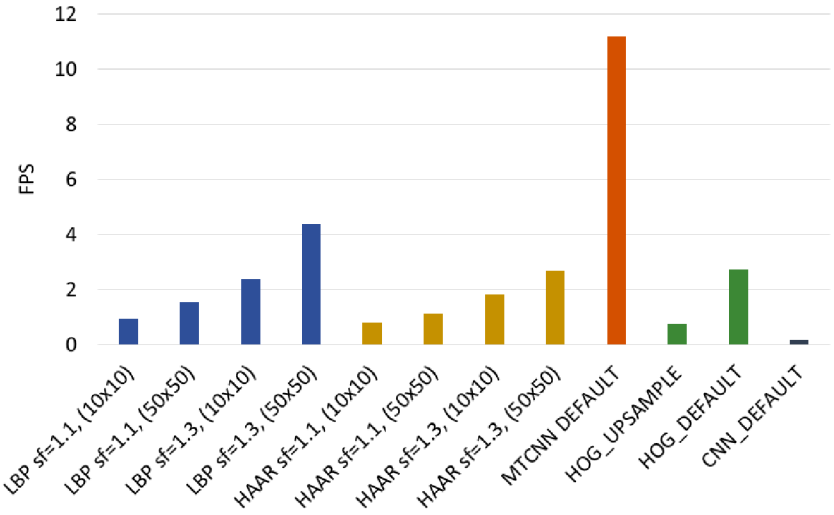
Obrázek 3.12: Precision - recall křivky pro různá nastavení parametru thresholdScale metody MTCNN.



Obrázek 3.13: Testy invariance klasických detektorů.



Obrázek 3.14: Testy invariance detektorů založených na neuronových sítích.



Obrázek 3.15: Výkonnostní srovnání metod detekce.

velikost detekovaných tváří. Metody Viola-Jones a LBP byly testovány pro minimální velikost 10x10px a 50x50px, výchozí nastavení MTCNN detektoru odpovídá 60x60px, výchozí nastavení metod HOG a CNN odpovídá nastavení velikosti 80x80px a varianta metody HOG_UPSAMPLE provádí detekce do velikosti 40x40px. Pro optimalizaci výpočetních nároků je nutné zvážit minimální velikost detekce potřebnou k extrakci relevantního příznakového vektoru.

Kapitola 4

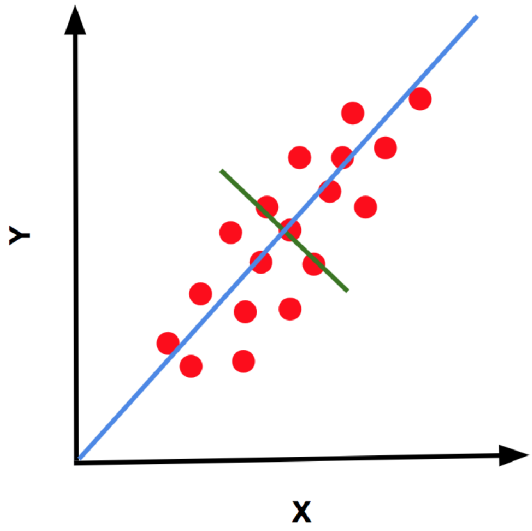
Rozpoznávání tváře

Cílem procesu rozpoznávání tváře je přiřadit neznámé detekci identitu porovnáním s fotografiemi tváří z databáze. Výsledky rozpoznávání lze významným způsobem vylepšit vycentrováním a normalizací detekovaných tváří. Metody rozpoznávání přiřazují detekovaným tvářím příznakový vektor. Porovnáním příznaku s příznaky známých osob lze získat identitu neznámé detekce. Existují klasické přístupy extrahující příznakový vektor aplikací matematické operace na vstupní data, ale pravděpodobně nejlepších výsledků lze v současné době dosáhnout využitím neuronových sítí.

4.1 Metody normalizace

Detekce klíčových bodů tváře (facial landmark detection) je proces, jehož cílem je na ohraničené oblasti tváře detekovat souřadnice očí, obočí, nosu, rtů a dalších částí tváře. Znalost souřadnic těchto oblastí umožňuje určit natočení tváře, provést její vycentrování a napomoci tím zlepšení výsledků rozpoznávání. Tato úloha bývá nejčastěji řešena trénováním neuronové sítě na speciálních datových sadách. *Labeled Face Parts in the Wild (LFPW) Dataset* [2] je známou datovou sadou zaměřenou na detekci klíčových bodů tváře. Skládá se ze 1432 obrázků pocházejících z internetových zdrojů. Na každém obrázku se nachází tvář s anotacemi popisujícími 29 klíčových bodů znázorněných na obrázku 4.1.

Obrázek 4.1: Ukázka anotací datové sady LFPW. (převzato z [2])



Obrázek 4.2: Průběh určování dimenze s největší variancí. (převzato z [4])

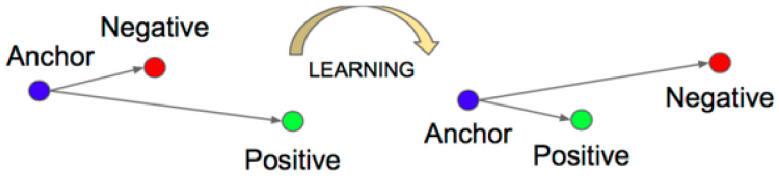
Řešení normalizace implementované v knihovně *Dlib* [14] vychází z článku One Millisecond Face Alignment with an Ensemble of Regression Trees [13] publikovaného v roce 2014. Implementace poskytuje srovnatelnou rychlost výpočtu a úspěšnost detekce s údaji uvedenými v článku. Rozdílem je využití modelu s 68 klíčovými body namísto 194. Novější verze implementace detektoru klíčových bodů pracuje pouze s 5 klíčovými body a je založena na neuronové síti. Výhodou oproti původní verzi je vyšší rychlost a nižší požadavky na rozlišení snímků tváře. Celý proces normalizace spočívá v ořezání a vycentrování detekované tváře a úprava jejího rozlišení na hodnoty vyhovující architektuře vstupní vrstvy neuronové sítě extraktoru.

4.2 Metody Eigenfaces a Fisherfaces

Problémem reprezentace tváře na fotografii je její vysoká dimenzionalita. Principal Component Analysis [25] je statistická metoda umožňující transformaci reprezentace snímku tváře do nižší dimenze. PCA lze považovat za lineární transformaci, která umožňuje reprezentaci dat v novém systému souřadném, se zachováním dimenzí nesoucích nejpodstatnější informace. PCA může být považována i za algoritmus ztrátové komprese se zachováním těch vlastností sady dat, které se vyznačují největší variancí (obrázek 4.2). Výsledkem aplikace PCA na obrázek jsou příznaky typu Eigenfaces [21]. Tyto příznaky nabízí způsob reprezentace informací o tváři z obrázku.

Za vhodnější metodu pro extrakci příznakového vektoru je považována metoda Fisherfaces [25]. Příznaky typu Fisherfaces vznikají aplikací transformace Linear Discriminant Analysis [25] na data vstupního obrázku. Příznaky typu Eigenfaces i Fisherfaces lze rovněž využít při úloze rozpoznávání pohlaví.

Knihovna *OpenCV* [4] obsahuje sadu nástrojů, které umožňují provést extrakci těchto příznaků a na trénovacích datech vytvořit klasifikátor umožňující rozpoznávání pohlaví. V současnosti jsou ale považovány za zastaralé a obvykle jsou tyto úlohy řešeny pomocí neuronových sítí [25].



Obrázek 4.3: Obecný trénovací algoritmus využívající hodnotící funkce triplet loss. (převzato z webu[2])

4.3 Neuronové sítě

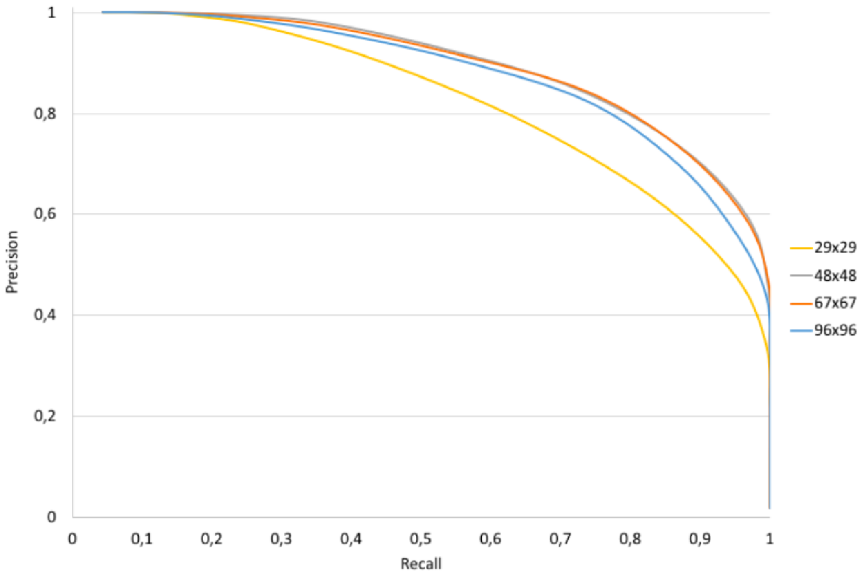
V oblasti extrakce příznakového vektoru tváře pro účely identifikace jsou v současnosti ve většině případů využívány různě navržené architektury neuronových sítí natrénovaných na vhodné datové sadě. Obecný trénovací algoritmus vyjádřený na obrázku 4.3 nejdříve vezme dva obrázky stejného člověka a jeden obrázek jiného člověka. Snaží se upravit váhy spojů mezi neurony s podmínkou, aby byly příznakové vektory stejné osoby blíže než příznakové vektory rozdílných osob. Toho lze dosáhnout aplikací hodnotící funkce triplet loss [25]. Tato funkce se snaží minimalizovat vzdálenost mezi referenčním příkladem a pozitivním příkladem a maximalizovat vzdálenost mezi referenčním příkladem a negativním příkladem. Porovnávání příznakových vektorů spočívá v jednoduchém výpočtu jejich euklidovské vzdálenosti.

Jednou z nejlepších volně dostupných metod pro extrakci příznakového vektoru je model neuronové sítě typu ResNet-34 implementovaný v knihovně *Dlib* [14]. Konkrétně se jedná o modifikaci residuální neuronové sítě z dokumentu [9] s nižším počtem vrstev a počtem filtrů pro každou vrstvu snížených na polovinu. Tato síť byla natrénována na datové sadě [14], která byla vytvořena spojením datových sad *The face scrub* [17], *VGG dataset* [18] a dalším množstvím fotografií tváří z internetu. Celkový počet fotografií tváře vzniklé datové sady dosahuje počtu 3 milionů. Vstupní vrstva neuronové sítě je navržena pro fotografie tváře normalizované na rozlišení 150x150px. Výsledkem extrakce je příznakový vektor o 128 dimenzích. Úspěšnost metody lze zlepšit realizací jitter operací pro vytvoření dalších vzorků vstupních fotografií a následným zprůměrováním příznakových vektorů. Výsledný model dosahuje podle dokumentace knihovny *Dlib* [14] při použití jitter operací úspěšnosti 99,38% na známé srovnávací datové sadě *LFW (Labeled Faces in Wild)* [11].

4.4 Vyhodnocení metod rozpoznávání

K otestování vlastností metody rozpoznávání z knihovny *Dlib* [14] jsem využil datovou sadu *ChokePoint* [23]. Tato datová sada velice věrně napodobuje podmínky reálného nasazení systému a pro účely testování identifikace ji lze použít bez nutnosti dalších úprav. Podle technické zprávy k datové sadě *Labeled Faces in the Wild* [11] nedosahují metody rozpoznávání Eigenfaces [21] a FisherFaces [25] v porovnání s metodami založenými na neuronových sítích příliš dobrých výsledků a z tohoto důvodu nemá smysl vyhodnocovat jejich úspěšnost. Vyhodnocení probíhalo podle doporučeného testovacího protokolu použité datové sady a jeho výsledkem je precision - recall křivka na obrázku 4.4. Získaná data lze rovněž využít k určení optimální prahové hodnoty pro shodu mezi příznaky ve finální implementaci.

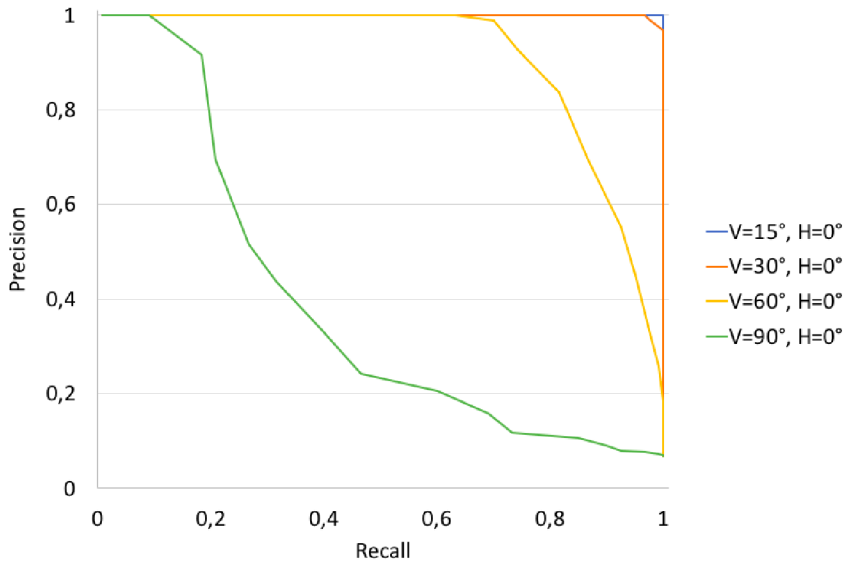
[2]https://towardsdatascience.com/lossless-triplet-loss-7e932f990b24



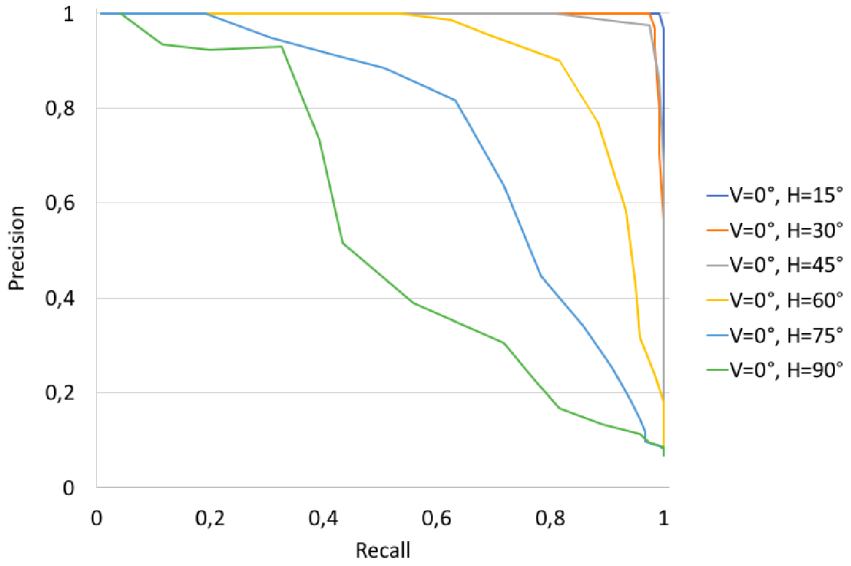
Obrázek 4.4: Výsledky testování metody z knihovny Dlib.

Implementace metody využívá knihoven OpenBLAS (Basic Linear Algebra Subprograms) a LAPACK (Linear Algebra Package) pro optimalizaci některých výpočtů. Je zde i podpora pro akceleraci výpočtu na grafické kartě pomocí rozhraní CUDA [14]. Výpočet příznakového vektoru na referenčním stroji MacBook Air 2017 vyžaduje bez akcelerace na grafické kartě zhruba 220ms procesorového času. Většina prostředků je spotřebována na samotnou extrakci, preprocessing ovlivňuje výpočet minimálně.

Metoda identifikace byla rovněž otestována z pohledu robustnosti oproti natočení tváře. Datovou sadu *Head Pose Image Dataset* [7] jsem využil k sestavení precision - recall křivek [20] pro různé úhly natočení tváře. Příznakové vektory získané testovanou metodou z fotografií osob s přímým pohledem do kamery byly porovnávány s příznaky tváří o určitém natočení a na základě různých hodnot prahové vzdálenosti příznaků byly sestaveny křivky na obrázcích na obrázcích 4.5 a 4.6. Výsledky napovídají prudkému poklesu úspěšnosti v případě úhlu natočení tváře větším než 45°.



Obrázek 4.5: Invariance metody z knihovny Dlib vzhledem k horizontálnímu natočení.



Obrázek 4.6: Invariance metody z knihovny Dlib vzhledem k vertikálnímu natočení.

Kapitola 5

Implementace experimentálních nástrojů a uživatelské aplikace

V předchozích kapitolách byl provedený důkladný průzkum existujících metod detekce i identifikace a jejich podrobné testování z více kritérií. Pro tyto účely bylo nutné implementovat několik experimentálních nástrojů realizujících veškerá vyhodnocení. Abych se blíže seznámil s problematikou implementace systému pro rozpoznávání tváře, realizoval jsem nejprve prototyp uživatelské aplikace v programovacím jazyce Python. Veškeré znalosti získané z výsledků experimentů a implementace prototypu jsem později reflektoval do implementace finální verze uživatelské aplikace v jazyce C++ a na vlastní datové sadě jsem ověřil, zda se podařilo dosáhnout očekávaných výsledků.

5.1 Experimentální nástroje pro testy

Veškeré experimentální nástroje jsem vytvořil v programovacím jazyce Python z důvodu rychlé implementace a poměrně jednoduchého uvedení potřebných knihoven s algoritmy počítačového vidění do provozu. Veškeré závislosti bylo možné nainstalovat pomocí terminálového nástroje `pip`.

Před implementací samotných nástrojů jsem vytvořil sadu funkcí, které umožňují procházet snímky datových sad a analyzovat jejich anotace. Funkce `frame_paths(sequence_name)` vrací datový typ `list` s množinou cest k obrázkům videosekvence datové sady *ChokePoint* s pojmenováním `sequence_name`. Pomocí funkce `load_ref_face(filename)` lze získat z XML souboru dodávaného k datové sadě informace o souřadnicích očí osoby na snímku definovaném pomocí `filename`. Zpracování XML dat mi usnadnila knihovna `xml2dict`. Funkce `load_identity(filename)` získává z anotací namísto souřadnic očí identifikátor osoby na snímku.

Anotace datové sady *Head Pose Image Dataset* jsou zakódovány z části do názvu souboru jednotlivých snímků. Funkce `person_info(filename)` vrací identifikátor osoby a úhly natočení jejího obličeje pro snímek s názvem `filename`. Ke každému snímku zároveň existuje stejnojmenný soubor s příponou `txt`, ve kterém jsou uvedeny informace o souřadnicích tváře na snímku. K těm se lze dostat voláním funkce `load_detection(filename)`.

Testovací skript realizující vyhodnocení metod detekce nejprve detekuje tváře ve snímcích datové sady *ChokePoint* [23] a jejich souřadnice společně s hodnotou skóre ukládá do souboru. Uložené hodnoty jsou později porovnávány s anotacemi ve formátu XML a pro různé prahové hodnoty skóre jsou vypočítány body precision - recall křivek. Následuje

uložení do souboru ve formátu CSV. Ten lze otevřít v tabulkovém editoru a výsledky vizualizovat.

Implementace testovacího skriptu pro vyhodnocení metody rozpoznávání z knihovny *Dlib* [14] odpovídá doporučenému testovacímu protokolu z dokumentace datové sady *ChokePoint* [23]. Protokol popisuje využití fotografií ořezaných obličejů osob z datové sady k vytvoření všech možných kombinací jejich dvojic. Je tedy nutné vypočítat příznakový vektor pro všechny fotografie tváře a vytvořit matici, kde jsou pro každou dvojici určeny její identifikátory a euklidovská vzdálenost. Porovnání identifikátorů a hodnot euklidovské vzdálenosti vede k výpočtu bodů precision - recall křivek. Fotografie ořezaných tváří jsou k datové sadě dodávány v normalizovaném rozlišení 96x96px. Realizoval jsem i experimenty s vlivem sníženého rozlišení na úspěšnost metody.

Dalším úkolem bylo implementovat testy pro vyhodnocení vlivu natočení tváře na výsledky detekce a identifikace. Pro veškeré fotografie upravené datové sady *Head Pose Image Dataset* [7] byla provedena detekce, jejíž výstupem byla matice s hodnotou vertikálního úhlu natočení, horizontálního úhlu natočení, hodnota IoU a skóre. Hodnota IoU byla určena porovnáním detekce s anotací. Výstupem skriptu byl opět CSV soubor s body precision - recall křivky. Výpočet plochy pod křivkou jsou prováděl už v tabulkovém editoru.

Skript pro vyhodnocení metody rozpoznávání z knihovny *Dlib* [14] nejprve vypočítal pro fotografie neupravené datové sady *Head Pose Image Dataset* [7] matici s hodnotou vertikálního úhlu natočení, horizontálního úhlu natočení, identifikátor osoby a extrahovaný příznakový vektor. Pro řádky matice reprezentující tváře s přímým pohledem byl proveden kartézský součin s položkami matice zkoumaného úhlu natočení. Porovnání identifikátorů osob a euklidovské vzdálenosti jejich příznaků umožnilo vygenerovat precision - recall křivku do CSV souboru.

5.2 Implementace prototypu

Pro získání prvních zkušeností se systémem na rozpoznávání tváře jsem se rozhodl vytvořit jednoduchou demo aplikaci implementovanou v jazyce Python. Programy vytvořené v tomto programovacím jazyce nedosahují vysoké efektivity z pohledu využití výpočetních zdrojů, ale je zde možnost rychlé a jednoduché implementace prototypu, který lze později přepsat do cílového jazyka.

Pomocí parametrů příkazové řádky je aplikaci definována složka s fotografiemi hledaných osob, cesta k videozáznamu a složka pro ukládání výstupních dat. Výstupní data představuje videozáznam s vizualizací identifikace osob pomocí ohraničujícího rámečku s identifikátorem a textový soubor s uloženými detekcemi. Každá detekce je určena číslem snímku, souřadnicemi a příznakovým vektorem.

Po spuštění probíhá extrakce příznakových vektorů hledaných osob, jejichž fotografie jsou v zadané složce. Identifikátor osoby a její příznakový vektor jsou uloženy jako atributy třídy `Face`. Mezi atributy této třídy patří ještě souřadnice detekce. Pro všechny atributy jsou implementovány gettery a settery. Třída `Face` dále implementuje metodu `distance(other_face)` vracející euklidovskou vzdálenost s tváří předanou parametrem a metodu `update_position(new_frame)`, která aktualizuje atribut souřadnic výsledkem trackovacího algoritmu na základě nového snímku videa.

Po inicializaci začíná procházení videosekvence po snímcích. Pomocí detektoru je provedena detekce tváří na aktuálním snímku. Aktuálně sledované tváře z předchozích snímků jsou drženy v proměnné typu `list` a jejich poloha je pravidelně aktualizována s každým snímkem. Souřadnice detekcí jsou porovnány s již sledovanými tvářemi a zjišťuje se, které

tváře jsou na snímku nové. Pro nové detekce je vytvořena instance třídy `Face` nesoucí příznakový vektor, souřadnice a inicializovaný tracker. Instance je uložena mezi ostatní sledované tváře. Identita sledovaných tváří je určena porovnáním s databází příznakových vektorů. V dalších krocích jsou pouze generována výstupní data do souboru.

Aplikace využívá implementace Viola-Jones detektoru z knihovny *OpenCV* [4]. Detektor je použit pouze se základním nastavením `scaleFactor=1.3`. Extrakci příznakových vektorů a normalizaci detekcí zajišťují modely neuronových sítí z knihovny *Dlib* [14]. Prahová euklidovská vzdálenost pro shodu příznaků byla zvolena podle dokumentace knihovny. Knihovna *Dlib* rovněž poskytuje implementaci korelačního trackeru. Obě zmiňované knihovny lze nainstalovat pomocí terminálové aplikace `pip`.

Implementace jednoduché demo aplikace mi umožnila první praktické seznámení s danou problematikou. Naučil jsem se používat klíčové metody knihoven *OpenCV* a *Dlib* a mohl jsem sledovat vliv úprav parametrů na výstup programu. Získané znalosti jsem dále využíval při implementaci výsledné aplikace v jazyce C++.

5.3 Implementace uživatelské aplikace

Po prvotním seznámení s implementací systému pro reidentifikaci osob v jazyce Python a podrobném testování klíčových parametrů a chování metod lze z dostupných údajů provést implementaci aplikace v jazyce C++.

Popis požadavků na aplikaci

Obecně je kvalita řešení problému vždy pevně spojena s přesnou specifikací požadavků, které definují hranice mezi úspěšným a nedostatečným řešením. Při vývoji systému je velice důležité mít od začátku přesný popis požadavků, které musí uživatelská aplikace splňovat.

Výsledný systém by měl mít podobu terminálové aplikace implementované v jazyce C++. Vstupem pro aplikaci bude IP adresa kamery zadávaná parametrem příkazové řádky. Aplikace by měla být schopna komunikovat po Wi-Fi nebo ethernetovém připojení s běžně používanými IP kamerami s maximálním rozlišením 1080p. Druhou možností je zadání názvu souboru s videosekvencí. Aplikace by měla být schopna pracovat se všemi běžnými formáty video souborů.

Druhým vstupním parametrem je cesta ke složce obsahující fotografie hledaných osob. Fotografie mohou být ve všech běžně používaných formátech. Jejich rozlišení může být od několika desítek pixelů po několik megapixelů. Název fotografie bez přípony systému specifikuje identifikátor osoby. Na fotografii by se měla vždy vyskytovat jediná osoba. V případě výskytu většího množství osob může systém využít fotografii k identifikaci libovolné osoby ze snímku. Další parametry slouží k určení cesty k předtrénovaným modelům klasifikátorů nebo neuronových sítí. V případě nesprávně zadané cesty aplikace vypisuje chybovou hlášku a ukončí svou činnost vrácením příslušného chybového kódu.

Výstupem aplikace je množina hodnot specifikující každou detekci na snímku videa. Množina hodnot specifikující detekce nese informace o časové známce snímku, souřadnicích, které specifikují ohraničení detekce, identifikátor osoby v případě, že se podařilo identifikovat shodu s databází, a příznakový vektor extrahovaný z detekce. Tyto hodnoty jsou ukládány do souboru a lze zapnout i jejich vizualizaci.

Aplikace by měla být schopna zpracování dat v reálném čase, což odpovídá alespoň hodnotě 5 FPS. Nižší hodnota FPS je opodstatněna nepříliš rychlými změnami ve videu. Aplikace by měla být připravena na využití v terénu a měla by si vystačit s hardwarovou

konfigurací výkonnějšího notebooku. Lze uvažovat minimální procesor Intel-i5 a možnost využití dedikované grafické karty.

Dříve popsané jádro systému by mělo být integrovatelné do modulu realizujícího komunikaci se vzdáleným serverem pro ukládání vypočtených hodnot. Komunikační rozhraní je implementováno fakultní výzkumnou skupinou za účelem propojení více modulů realizujících úlohy počítačového vidění.

Volba metod a jejich konfigurace

Výsledky testování metod detekce ukazují, že detektor MTCNN dosahuje vysokého pokrytí s nízkým počtem falešných detekcí. Jeho implementace zahrnuje i velice rychlý korelační tracker. Možnosti použití detektoru CNN implementovaného v knihovně *Dlib* [14] značně omezují jeho vysoké výpočetní nároky a nutnost akcelerace výpočtu na grafické kartě. Z tohoto důvodu jsem se rozhodl použít fakultní implementaci detektoru MTCNN s přihlédnutím na dostačující invarianci oproti natočení tváře v porovnání s robustností příznaku. S ohledem na potřeby extrakce příznakového vektoru není vhodné pracovat s detekcemi menšími než 30 - 35px. Vysoká úspěšnost detekce nevyžaduje volit parametr `thresholdScale` vyšší než `1.0`.

Metody Eigenfaces a Fisherfaces byly zmíněny spíše z historických důvodů a nelze očekávat, že by mohly představovat konkurenci pro metody založené na neuronových sítích. Z důvodu vysoké invariance oproti natočení tváře a dalším negativním vlivům jsem pro implementaci aplikace zvolil volně dostupnou implementaci extraktoru příznakových vektorů z knihovny *Dlib* [14]. Pro lepší využití výpočetních zdrojů a rozložení zátěže je vhodné přenést výpočet extraktoru na GPU. Výsledky testování napovídají, že vhodná prahová vzdálenost příznakových vektorů se pohybuje v rozpětí `0.52` až `0.58`.

Implementační detaily

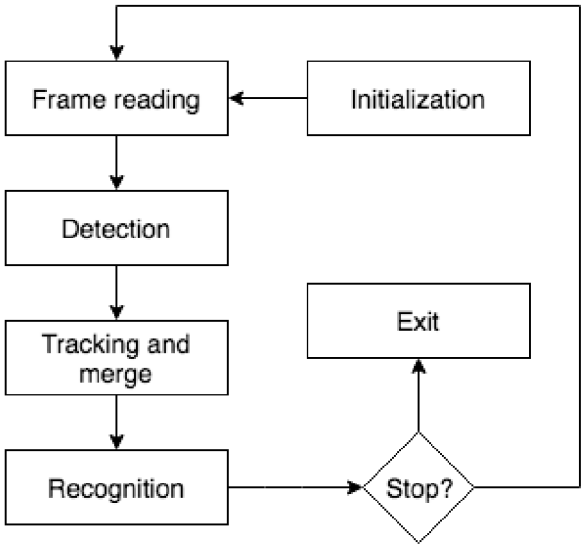
Uživatelskou aplikaci jsem z důvodu lepších možností optimalizace výpočtu implementoval v jazyce C++. Algoritmus zpracování dat v uživatelské aplikaci je zobrazen ve schématu na obrázku 5.1. Logické operace ze schématu zpracování dat odpovídají modulům sdružujícím významově podobné funkce, do kterých jsem zdrojový kód pro lepší přehlednost rozdělil.

Modul `initial.h` se skládá z funkcí inicializujících systém po spuštění. Funkce `load_database()` slouží k prohledání složky s fotografiemi hledaných osob a vrací datový typ `std::vector<Detection>`. Struktura `Detection` reprezentuje abstrakci detekované tváře. Nese informace o souřadnicích detekce, skóre, identifikátoru a příznakovém vektoru. Pomocí funkce `initialize_app()` lze provést načítání předtrénovaných modelů ze souboru. Zpracování parametrů příkazové řádky zajišťuje nástroj `cv::CommandLineParser`.

Modul `detector.h` je tvořen funkcí `detect_faces_cnn()` pro detekování tváří na snímku, funkcí `track_faces()` pro aktualizaci souřadnic dřívějších detekcí a jejich spojení s nově detekovanými tvářemi a sadou funkcí pro převody formátů detekcí.

Další modul `recognition.h` sdružuje funkce pro identifikaci tváře. Součástí modulu je funkce `extract_feature_vector()` sloužící pro extrakci příznakového vektoru detekované tváře a funkce `match_with_database()` zajišťující porovnávání příznaků detekcí s příznaky z databáze a přiřazení příslušných identifikátorů.

Načítání snímků videosekvence je zajištěno pomocí objektu `cv::VideoCapture` a po celou dobu pracuje modul s abstrakcí snímků typu `cv::Mat` z knihovny *OpenCV*. Všechny funkce pro práci se snímky jsou sdruženy v modulu `frames.h`.



Obrázek 5.1: Algoritmus zpracování dat v uživatelské aplikaci.

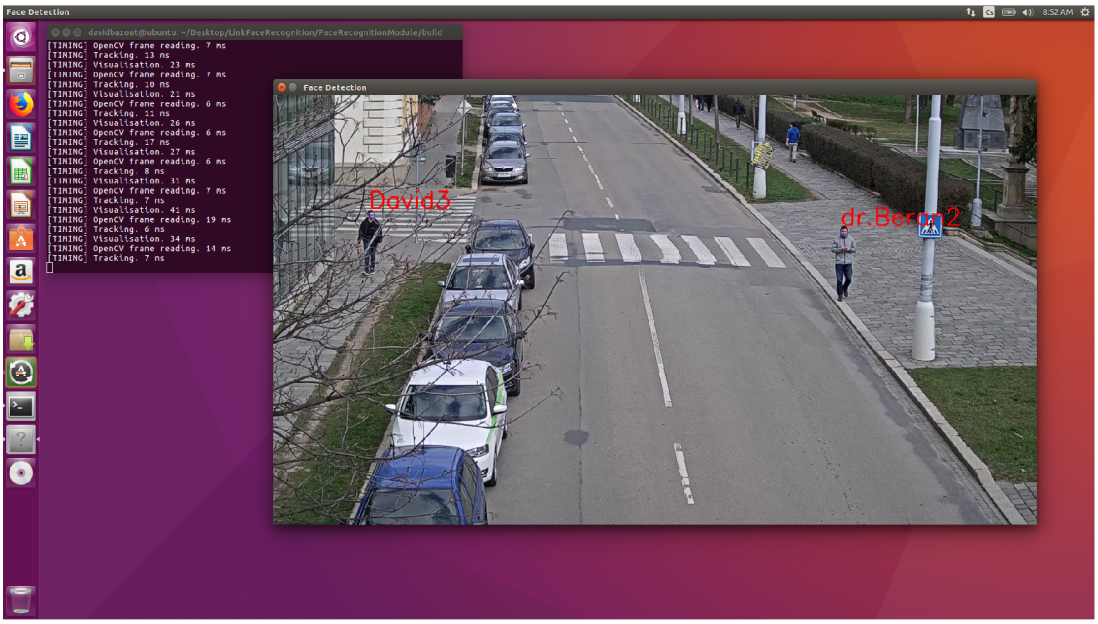
Třída `FaceRecogModule` definovaná v modulu `module.h` zajišťuje integraci systému s fakultní implementací komunikačního rozhraní. Zmiňované komunikační rozhraní umožňuje pracovat i se speciálními datovými typy `cv::Rect` a `cv::Point`. Celý modul je spušten voláním metody `FaceRecogModule::run()`, která nejdříve volá inicializační funkce modulu `initial.h` a poté provádí pro každý načtený snímek příslušné zpracování pomocí modulů `detector.h` a `recognition.h`.

Doplňující přepínače modulu umožňují spustit odesílání výsledků na vzdálený server, vizualizaci probíhající detekce, ukládání vizualizace do video souboru nebo ukládání metadat o detekcích do textového souboru. Je zde i možnost specifikovat detekci nebo rozpoznávání pouze pro každý n-tý snímek nebo upravit parametry metody detekce a identifikace. Snímek obrazovky se spuštěnou aplikací je na obrázku 5.2.

Uživatelskou aplikaci jsem vzhledem k programátorské přívětivosti vyvíjel na operačním systému Ubuntu 16. Aby byla zajištěna možnost spuštění na ostatních platformách, pro uživatelskou aplikaci a veškeré její závislosti jsem vytvořil Docker image, který lze bez problému spustit na Mac OS i OS Windows. Vedlejším produktem celého procesu je terminálový skript, který po spuštění nainstaluje na OS Ubuntu veškeré závislosti aplikace a provede její kompilaci a instalaci.

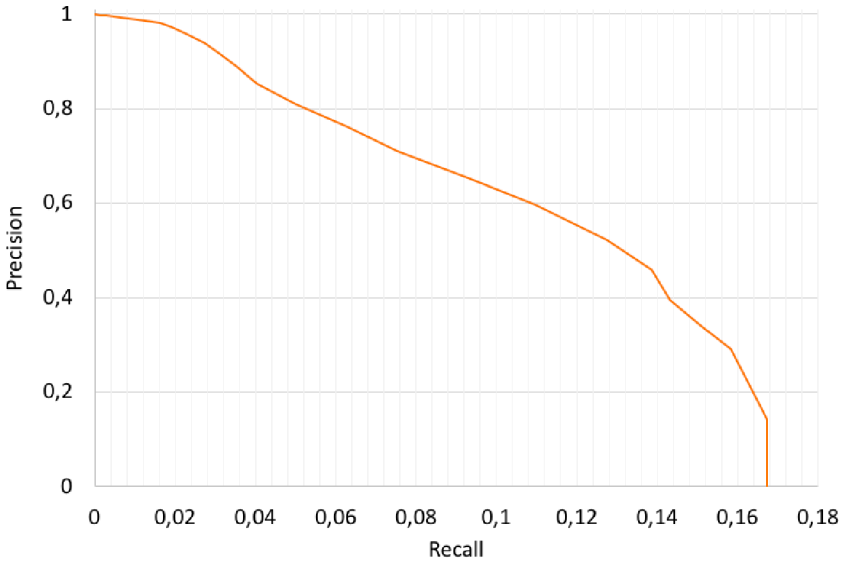
5.4 Vyhodnocení na vlastní datové sadě

Aby bylo možné ověřit, zda výsledná aplikace splňuje dříve stanovené požadavky, vytvořil jsem vlastní datovou sadu a aplikaci podrobil dalším testům. Vyhodnocení úspěšnosti bylo realizováno opět pomocí precision - recall křivek. Neméně důležitým výstupem je výpočetní náročnost aplikace a informace o potřebném rozlišení a umístění IP kamery pro snímání určitého prostoru. Výsledek vyhodnocení je na obrázku 5.3.



Obrázek 5.2: Ukázka výstupu z uživatelské aplikace.

Při zpracování reálných dat probíhala detekce s velice dobrými výsledky. Nebylo problémem detekovat tváře o rozměrech pouhých 10x10px a to i za výrazného odklonu od objektivu kamery. Z předchozího testování na datové sadě *ChokePoint* vyplývá, že rozpoznávání není vhodné pro tváře s nižším rozlišením než 30x30px. Na snímaném videu tato velikost odpovídá zhruba vzdálenosti 60 metrů. Poměrně nízká hodnota recall je způsobena jednoduchou anotací. Anotace nese informaci o výskytu osoby na snímku ale nezaručuje viditelnost tváře. Pouhý 10 vteřinový průchod osoby poskytuje při dosaženém zpracování 8 FPS celkem 80 snímků tváře v různých vzdálenostech od objektivu, což umožňuje dosahovat rozumných výsledků i pro nižší pokrytí. Z tohoto důvodu je vhodné zvolit prahovou euklidovskou vzdálenost spíše vyšší a nezatěžovat obsluhu příliš vysokým množstvím falešných upozornění.



Obrázek 5.3: Precision - recall křivka pro vlastní datovou sadu.

Kapitola 6

Závěr

Cílem této práce bylo prozkoumat existující přístupy metod detekce a identifikace, navrhnout a realizovat experimenty poskytující jejich relevantní porovnání a načerpané znalosti promítnout do implementace uživatelské aplikace.

Bližší průzkum literatury o dohledových systémech umožnil identifikovat vlastnosti, které mají zásadní dopad na úspěšnost existujících přístupů. Výběr datových sad pro realizaci experimentů byl zaměřen na průzkum chování metod z hlediska invariance k natočení tváře, vlivu nepříznivého osvětlení, nízkého rozlišení vstupních dat nebo jejich špatné kvality, a zkoumána byla i výpočetní náročnost veškerých metod. Realizace experimentů spočívala v implementaci několika testovacích skriptů, jejichž výsledky posloužily k výběru metod vhodných pro implementaci.

Tvorbu uživatelské aplikace předcházel návrh jednoduššího prototypu, který mi poskytl bližší seznámení s implementací algoritmů počítačového vidění s využitím volně dostupných knihoven. Vyladěný prototyp byl transformován do optimalizované finální implementace.

Funkčnost uživatelské aplikace byla ověřena na vlastní datové sadě napodobující data z reálného provozu. Podařilo se dosáhnout uspokojivé rychlosti zpracování 8 FPS na videu v rozlišení Full-HD s nepříliš výkonným procesorem Intel-i5@1.8GHz. Aplikace byla schopna rozpoznávat osoby až do vzdálenosti 60 metrů od objektivu kamery.

Úloha detekce a identifikace je v současné době řešena s nejlepšími výsledky pomocí neuronových sítí. Vývojem složitějších architektur neuronových sítí, výkonnějších počítačů pro trénování jejich modelů a vytvářením větších a náročnějších datových sad bude pravděpodobně dosahováno stále působivějších výsledků.

Literatura

[1] Ahonen, T.; Hadid, A.; Pietikainen, M.: Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, , č. 12, 2006: s. 2037–2041.

[2] Belhumeur, P. N.; Jacobs, D. W.; Kriegman, D. J.; aj.: Localizing parts of faces using a consensus of exemplars. *IEEE transactions on pattern analysis and machine intelligence*, ročník 35, č. 12, 2013: s. 2930–2940.

[3] Bolme, D. S.; Beveridge, J. R.; Draper, B. A.; aj.: Visual object tracking using adaptive correlation filters. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE, 2010, s. 2544–2550.

[4] Bradski, G. R.: *Learning OpenCV*. Computer Programming. Robotics, Sebastopol: O'Reilly, 2008, ISBN 978-0-596-51613-0.

[5] Dalal, N.; Triggs, B.: Histograms of Oriented Gradients for Human Detection. In *In CVPR*, 2005, s. 886–893.

[6] Fawcett, T.: An introduction to ROC analysis. *Pattern recognition letters*, ročník 27, č. 8, 2006: s. 861–874.

[7] Gourier, N.; Hall, D.; Crowley, J. L.: Estimating Face Orientation from Robust Detection OF SALIENT FACIAL STRUCTURES. In *FG NET WORKSHOP ON VISUAL OBSERVATION OF DEICTIC GESTURES*, 2004.

[8] Grabner, H.; Grabner, M.; Bischof, H.: Real-time tracking via on-line boosting. In *Bmvc*, ročník 1, 2006, str. 6.

[9] He, K.; Zhang, X.; Ren, S.; aj.: Deep Residual Learning for Image Recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[10] Held, D.; Thrun, S.; Savarese, S.: Learning to track at 100 fps with deep regression networks. In *European Conference on Computer Vision*, Springer, 2016, s. 749–765.

[11] Huang, G. B.; Ramesh, M.; Berg, T.; aj.: Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. Technická Zpráva 07-49, University of Massachusetts, Amherst, October 2007.

[12] Jain, V.; Learned-Miller, E.: FDDB: A Benchmark for Face Detection in Unconstrained Settings. Technická Zpráva UM-CS-2010-009, University of Massachusetts, Amherst, 2010.

[13] Kazemi, V.; Sullivan, J.: One Millisecond Face Alignment with an Ensemble of Regression Trees. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[14] King, D. E.: Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research*, ročník 10, 2009: s. 1755–1758.

[15] King, D. E.: Max-Margin Object Detection. *CoRR*, ročník abs/1502.00046, 2015.

[16] Lienhart, R.; Kuranov, A.; Pisarevsky, V.: Empirical analysis of detection cascades of boosted classifiers for rapid object detection. In *Joint Pattern Recognition Symposium*, Springer, 2003, s. 297–304.

[17] Ng, H.-W.; Winkler, S.: A data-driven approach to cleaning large face datasets. In *2014 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2014, s. 343–347.

[18] Parkhi, O. M.; Vedaldi, A.; Zisserman, A.: Deep Face Recognition. In *British Machine Vision Conference*, 2015.

[19] Ramchandra, A.; Kumar, R.: Overview of face recognition system challenges. *International Journal of Scientific & Technology Research*, ročník 2, č. 8, 2013: s. 234–236.

[20] Sundaram, M.; Mani, A.: Face Recognition: Demystification of Multifarious Aspect in Evaluation Metrics. Jul 2016.

[21] Turk, M.; Pentland, A.: Eigenfaces for recognition. *Journal of cognitive neuroscience*, ročník 3, č. 1: s. 71–86.

[22] Viola, P.; Jones, M. J.: Robust real-time face detection. *International journal of computer vision*, ročník 57, č. 2, 2004: s. 137–154.

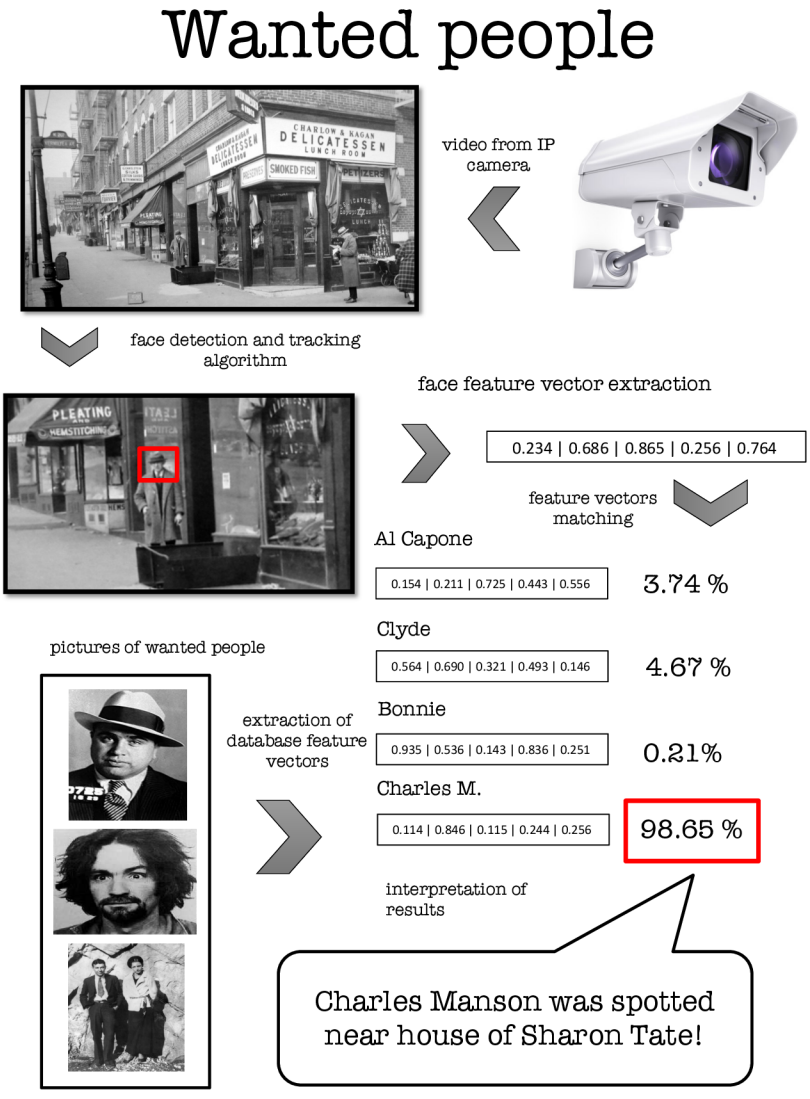
[23] Wong, Y.; Chen, S.; Mau, S.; aj.: Patch-based Probabilistic Image Quality Assessment for Face Selection and Improved Video-based Face Recognition. In *IEEE Biometrics Workshop, Computer Vision and Pattern Recognition (CVPR) Workshops*, IEEE, June 2011, s. 81–88.

[24] Yang, S.; Luo, P.; Loy, C. C.; aj.: WIDER FACE: A Face Detection Benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[25] Šonka, M.: *Image processing, analysis, and machine vision.* Toronto: Thomson, třetí vydání, 2008, ISBN 978-0-495-08252-1.

Příloha A

Plakát



Příloha B

Obsah přiloženého paměťového média

- **Dataset** - Záznamy vlastní datové sady s příkladným výstupem uživatelské aplikace.
- **Thesis** - Zdrojové soubory pro kompilaci bakalářské práce.
- **TestScripts** - Zdrojové soubory experimentálních nástrojů.
- **FaceRecognitionModule** - Zdrojové soubory uživatelské aplikace.
- **technicka_zprava.pdf** - Bakalářská práce ve formátu PDF.
- **demo.tar** - Docker Image pro jednoduché spuštění aplikace.
- **README** - Stručný návod ke spuštění uživatelské aplikace v prostředí Docker.