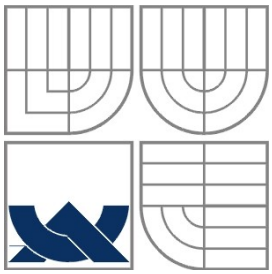
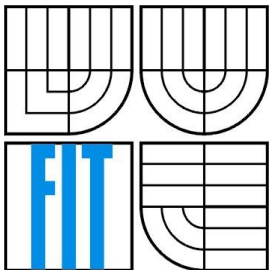


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

DETEKCE OBLIČEJŮ V OBRAZE, NEZÁVISLE NA NATOČENÍ

FACE DETECTION, INVARIANT TO ROTATION

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR
VÁCLAV BUREŠ

VEDOUCÍ PRÁCE
SUPERVISOR
ING. VÍTĚZSLAV BERAN

BRNO 2010

Abstrakt

Práce se věnuje problematice detekce typově stejných objektů (konkrétně obličejů) v obraze. Následuje rozšíření popsaných metod o detekci objektů v různých natočeních. Naleznete zde stručný přehled využitelných metod, jako je například Logical Binary Patterns, Histogram Of Gradients, Eigen Faces a blíže popsanou metodu AdaBoost. Následuje stručný přehled volně dostupných datasetů a popis jejich vybraných vlastností. Ke konci práce jsou popsány experimenty s využitím algoritmu AdaBoost a jejich vyhodnocení.

Klíčová slova

Detekce obličejů, počítačové vidění, AdaBoost, EigenFaces, Histogram Of Gradients, Logical Binary Patterns, OpenCV, dataset

Abstract

This bachelor thesis focuses on the detection of type uniform objects (concretely faces) in an image. Furthermore the thesis concentrates on the detection of objects in various rotations. The thesis covers a brief overview of methods available, such as Logical Binary Patterns, Histogram Of Gradients, Eigen Faces and more closely specified AdaBoost. Next, freely available datasets are presented, with a descripiton of their chosen characteristics. At the end of the thesis, experiments using AdaBoost algorythm and their evaluation are described.

Keywords

Face detection, computer vision, AdaBoost, EigenFaces, Histogram Of Gradients, Logical Binary Patterns, OpenCV, dataset

Citace

Václav Bureš: Detekce obličejů v obraze, nezávisle na natočení. bakalářská práce, Brno, FIT VUT v Brně, 2010

Detekce obličejů v obraze, nezávisle na natočení

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vítězslava Berana. Další informace mi poskytli Ing. Michal Hradiš a Ing. Michal Španěl.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Václav Bureš
17.5.2010

Poděkování

Zde bych rád poděkoval vedoucímu práce za jeho cenné rady, myšlenky a za vedení správným směrem. Poděkování patří také autorům OpenCV, OpenCV Wrapperu a všem ostatním, kteří mi svým přispěním pomohli realizovat tuto práci.

Obsah

1 Úvod

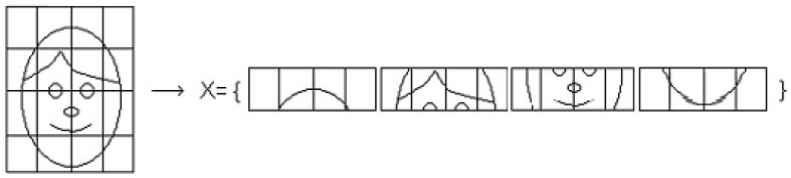
Práce, která se čtenáři dostává do rukou, se věnuje metodám řešení vyhledávání různě natočených obličejů v obraze. Pojem různě natočeného obličeje lze v trojrozměrném prostoru chápat několika způsoby, jak popisuje kapitola Detekce různě natočených obličejů. Já se v práci zaměřím na natočení v horizontální rovině. Na začátku práce je třeba prozkoumat dostupné metody detekce určitého typu objektů v obraze a následně jeden z těchto způsobů zvolit a podrobněji nastudovat. Po prostudování zvolené metody je třeba navrhnout postup, jak detekční metodu rozšířit, aby detekovala tentýž objekt v různých natočeních. Při návrhu rozšíření metody je třeba vzít v úvahu doposud použité a vyzkoušené postupy, které byly využity k dosažení podobného cíle, případně se poučit z uvedených úspěchů a neúspěchů. Vybranou metodu s navrženými rozšířeními je třeba uvažovat jako ucelený blok, který má být určitým způsobem implementován a následně vyzkoušen. K dosažení tohoto cíle bude vhodné využít již nějakou připravenou knihovnu pro práci s počítačovou grafikou, která má již implementovány potřebné funkce. Závěrem práce jsou navržené postupy vyhodnoceny a porovnány.

2 Přehled metod detekce obličejů

V následující kapitole je čtenář obeznámen s několika vybranými základními metodami, které demonstrují některé z možných přístupů k problému počítačové detekce lidských obličejů v obraze. Tato problematika spadá do oboru počítačové grafiky, která má na řešení obdobných problémů v dnešní době širokou řadu metod. Jednotlivé metody se mezi sebou liší složitostí implementace, názorností, rychlostí detekce a dalšími parametry. Některé vybrané metody jsou dále krátce představeny.

2.1 Eigen faces

Tato metoda je založena na principu PCA (Principal Component Analysis), což je technika redukce dimenze příznakového prostoru. Porovnávaný obraz je považován za vektor hodnot jasu. Z třírozměrného původního obrazu (šířka, výška, barva) získáme vektor poskládáním jednotlivých řádků obrazu za sebe, tento vektor pak má délku danou součinem výšky a šířky obrazu v pixelech. Princip převodu obrázku na vektor je naznačen na **obrázku 1**.



Obrázek 1: Převod obrazu na vektor, převzato z [1]

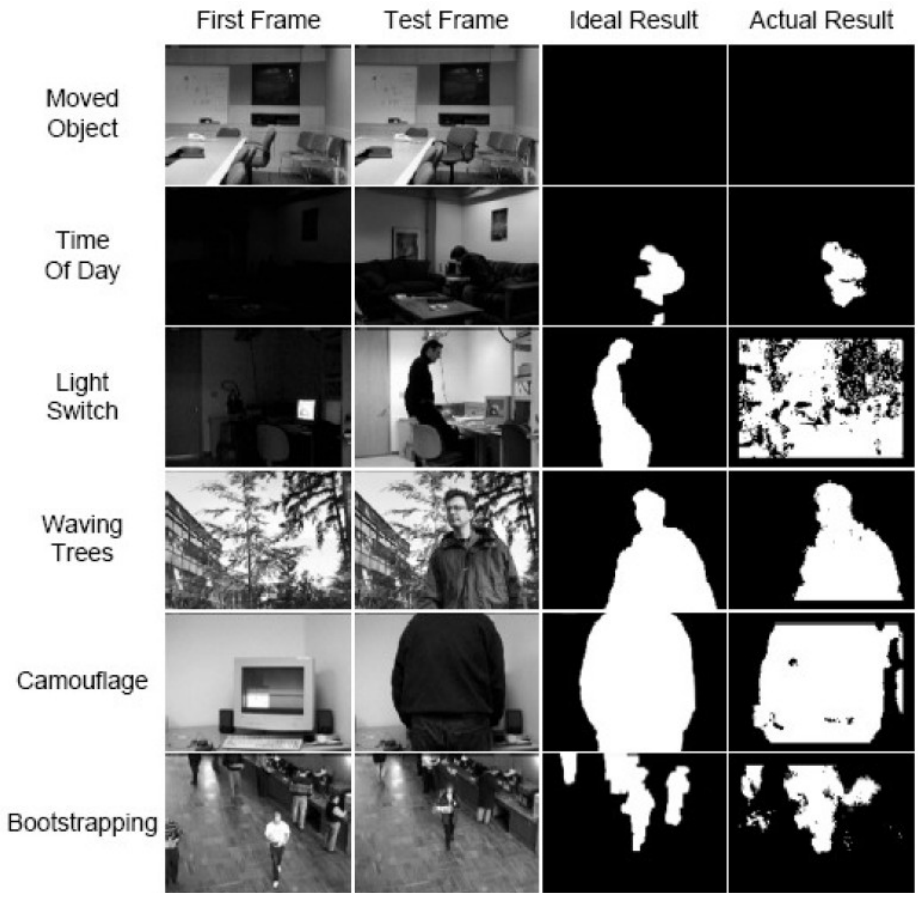
Technika PCA se snaží tento vektor na základě matematických principů redukovat tak, aby byla jeho délka výrazně kratší. Tím se získá promítnutí konkrétního obrazu jako bod ve vícerozměrném prostoru. Protože jsou si však jednotlivé tváře jistým způsobem podobné, promítají se body různých obličejů v tomto prostoru zhruba do stejné oblasti. Díky tomu je možno testované obrazy rozdělit do tříd a rozpoznávat typ objektu v obraze. Také je možné vytvořit knihovnu otisků (redukovaných podob jednotlivých vzorů) a po výpočtu otisku testovaného obrazu hledat v knihovně nejbližší podobný otisk. Tímto způsobem lze na tomto principu zkonstruovat algoritmus schopný například identifikovat jednotlivé osoby.

Obrázek 2: Různě redukovaný příznakový prostor, převzato z [2]

Na **obrázku 2** je patrný rozdíl obrazů zrekonstruovaných z různě zredukovaného příznakového prostoru. Vzorky, které jsou zredukované na menší počet rozměrů se ve výsledném porovnávání rychleji zpracovávají a jejich uložení zabere menší blok dat, je však také třeba brát v úvahu, že takové se promítají do menšího prostoru a proto následné rozlišování typu objektu případně identity musí být jemnější (a tedy vznikají větší nároky na přesnost).

2.2 Local Binary Patterns

Metoda LBP je jedna z metod založených přímo na práci s pixely obrazu. Pro vstupní obraz je nejdříve vypočtena LBP podoba obrazu a následně se pracuje s jejím histogramem. Tato metoda je velice výhodná z hlediska své nezávislosti na jasu celého snímku. Pro každý pixel scény je z původního obrazu vypočtena jeho LBP hodnota. Každá LBP hodnota je spočítána z hodnot pixelu v jeho osmi-okolí. Jednotlivé hodnoty v okolí jsou prahovány hodnotou právě vypočítávaného pixelu. Následně jsou tyto hodnoty (0 nebo 1) vynásobeny řadou čísel, která se získají jako 2^x. Takto získané hodnoty jsou sečteny a tím je dána výsledná LBP hodnota pixelu. Tímto způsobem jsou přepočítány veškeré hodnoty pixelů pro celý obraz. Z LBP obrazu je potom vypočítán LBP histogram, z jehož změn lze usuzovat změny v obraze. Díky tomu, že hodnota každého pixelu v LBP obraze je závislá na svém okolí, je výsledek jen velice málo citlivý na změny osvětlení celé scény. Pokud totiž bude osvětlení stejné scény měněno (například vlivem mraků, rozsvěcení a podobně), hodnoty LBP celého obrazu (a tedy i charakteristika LBP histogramu) by se měly měnit velice málo, ideálně vůbec. To je výhodou oproti klasickému porovnávání histogramů, kde se tyto změny projevují více. Z toho vyplývá výhodnost použití tohoto algoritmu na detekci změn v obraze. Pro jeden referenční snímek je vypočítán LBP histogram a ten je následně porovnáván s histogramy testovaných snímků. Pokud se charakteristika výrazně liší, lze odvodit změnu ve snímku. Na **obrázku 3** je ukázka z práce [4], která je dílem výzkumníků metody LBP.



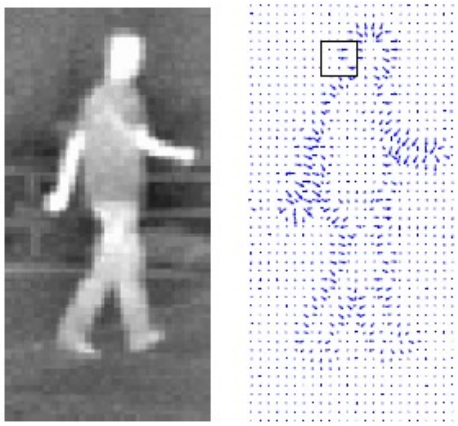
Obrázek 3: Ukázka metody LBP, převzato z [4]

Uvedenou funkci algoritmu by bylo možné s výhodou adaptovat na detekci obličejů. Bylo by možné například z velké sady vzorů vypočítat průměrnou podobu obličeje. Z této šablony by se potom vypočítal LBP histogram a ten se používal pro vyhledávání. Testovaný snímek by se procházel po částech (rozdělil by se na jednotlivé úseky), pro které by se vypočítával LBP histogram. Vzorový a spočítaný histogram by se poté porovnával. V případě podobných charakteristik histogramů by se dalo usuzovat nalezení obličeje. Tato metoda by měla velkou výhodu nezávislosti na osvětlení a barvě pleti. Nevýhodou by však byla špatná možnost vytvoření detektoru nezávislého na natočení obličeje, protože tím se jedná o jiný vyhledávaný vzor. Na úrovni práce s pixely, na které je tato metoda založena, by bylo nutné sestrojit více porovnávacích vzorů a jednotlivě porovnávat části obrazu s různými vzory.

2.3 Histogram of gradients

Histogram Of Gradients je metoda založená na porovnávání orientací hran. Celý obraz je nejprve

zpracován jednoduchými filtry pro hledání hran. Pro nalezení horizontálních hran se uplatní filtr (-1 0 1) a pro nalezení vertikálních $(-1\ 0\ 1)^T$. Po uplatnění těchto filtrů získáme obraz se zvýrazněnými obrysy jednotlivých barevných ploch. Obraz se následně rozdělí na buňky a pro každou buňku je spočítán vektor, který udává směr průchodu hrany buňkou a významnost hrany. V úvahu je brána velikost rozdílu sousedních hodnot, tedy "velikost hrany" a tyto hodnoty ovlivňují vliv jednotlivých částí buňky na výsledný vektor. Následně jsou buňky sloučeny do bloků, což jsou skupiny buněk, jejichž hodnoty je mezi sebou třeba normalizovat. Z těchto hodnot je pak možno vytvořit popisný vektor. Na **obrázku 4** je ukázka vstupní a výstupní podoby snímku zpracovaného metodou HOG.

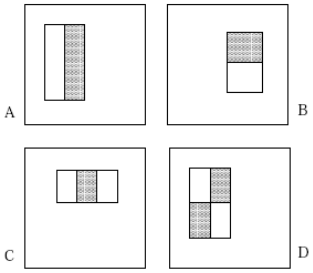


Obrázek 4: Vlevo snímek z termokamery, vpravo zpracováno algoritmem HOG, převzato z [5]

2.4 AdaBoost

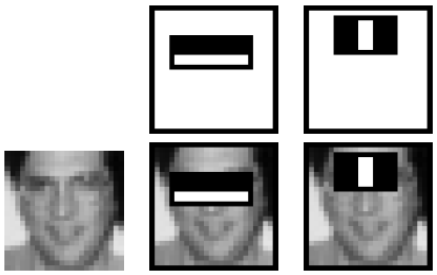
Algoritmus AdaBoost je obecně metodou strojového učení, která vhodným způsobem kombinuje slabé klasifikátory, a dává tak vzniknout jednomu silnému. Slabý klasifikátor je takový, který je relativně rychlý ve své práci, avšak má na určení výsledku relativně malý vliv. Teoreticky by bylo možné sestavit takový klasifikátor, který by pracoval s dobrou přesností, nicméně by byl složitý (musel by obsáhnout velké množství porovnávaných vlastností), což by způsobilo jeho nízkou rychlost. AdaBoost je algoritmus představující kompromis mezi uvedenými přístupy, kdy se snaží využít rychlosti jednotlivých slabých klasifikátorů a zároveň dosáhnout vysoké přesnosti tím, že kombinuje výsledky jednotlivých slabých klasifikátorů.

AdaBoost v počítačové grafice v oblasti detekce objektů představili Paul Viola a Michael Jones ve své práci [6]. Využívá se zde Haarových příznaků, což jsou příznaky v obraze představující vztah kontrastu jednotlivých ploch. V detekovaném obraze se při trénování klasifikátorů hledají významné kontrastní rozdíly ploch, které pak detektor využívá jako masku a porovnává je s prohledávaným vzorkem. Příznaky mají různé formy, jak demonstruje **obrázek 5**.



Obrázek 5: Různé formy příznaků aplikovatelných
na obrazy, převzato z [6]

Jednotlivé vzory se potom při trénování klasifikátoru mapují na různé oblasti obrazu, jak je znázorněno na **obrázku 6**:

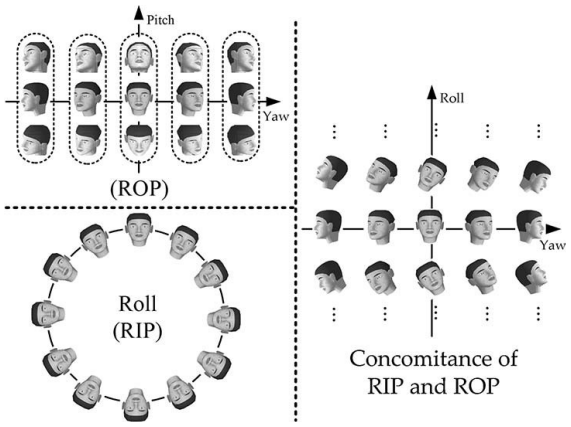


Obrázek 6: Přikládání příznakové masky na obraz, převzato z [6]

Z takto vytvořených slabých klasifikátorů je následně sestaven pomocí AdaBoostu jeden silný klasifikátor, který uvažuje významovou váhu jednotlivých slabých klasifikátorů (váhy jsou určeny při trénování). Podrobný postup výpočtů a tvorby těchto klasifikátorů nalezneme například v [7]. AdaBoostu je předložena sada slabých klasifikátorů a sada testovacích snímků, o kterých je známo, zda obsahují či neobsahují hledaný objekt. AdaBost je učící algoritmus, který postupně příznaky testuje nad vzorky a nastavením váhového parametru jednotlivých slabých klasifikátorů mění jejich vliv na celkový výsledek budovaného silného klasifikátoru. V cyklu jsou postupně procházeny jednotlivé slabé klasifikátory a jednotlivé (pozitivní a negativní) vzorky a váhy jsou upravovány tak, aby výsledný silný klasifikátor označil v co nejvíce případech správně pozitivní výskyt a v co nejméně případech označil za výskyt pozici, na které se detekovaný objekt nenachází.

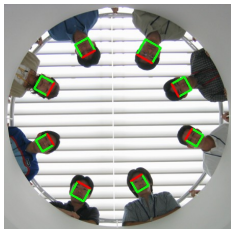
2.5 Detekce různě natočených obličejů

Problémem detekce obličejů v různých náklonech a natočeních se již zabývalo několik prací. Jednotlivé práce se liší konkrétním přístupem k detekci obličejů. Možnosti, jakým způsobem může být pohled na obličej (obecně jakýkoli objekt) natočen představuje **obrázek 7**.



Obrázek 7: Možnosti rotace obličeje, převzato z [8]

Některé práce se zabývají pouze obličejem otočeným v rovině obrazu, jak naznačuje **obrázek 8**:


Obrázek 8: Natočení v rovině obrazu, převzato z [9]

V práci, ze které ukázka pochází, bylo cílem vyřešit detekci obličeje jakkoli natočeného v rovině obrazu, překonat problémy detekce obličejů, z nichž jsou vidět jen části a při dodržení tohoto zadání dosáhnout co nejkratšího času detekce.

Poněkud komplexnější pohled na reálné snímky pořízené ve všedním životě přináší například práce [10], která již uvažuje jak rotaci ve smyslu dříve uvedeném, tak i náklon v obou dalších osách. Princip metody uvedené v této práci spočívá v několika krocích. Nejdříve jsou v prohledávaném vstupním obraze detekovány regiony barvy lidské kůže, z nichž jsou pro další postup vybrány pouze ty, které splňují dané předpoklady pro to, aby se mohlo jednat o obličej. V dalším kroku je metodou LBP o jednotlivých kandidátech rozhodnuto, zda se jedná skutečné o obličeje.

Jiný přístup uvádí [11]. Detekci různě natočených objektů lze pojmout jako detekci různých objektů, a tudíž využít paralelního běhu několika klasifikátorů natrénovaných na různě natočené podoby objektu. Uvedená práce vylepšuje tento postup o uspořádání klasifikátorů do různých struktur (stromů, pyramid). Zamýšleno je zlepšit výkonnostní vlastnosti algoritmů.

3 Návrh řešení

Tato kapitola shrnuje návrh postupu řešení práce, objasňuje některá rozhodnutí. Jsou zde popsány základní principy využité při realizaci řešení.

3.1 Využití dostupných prostředků

Implementovat přímo některou z dříve představených metod by bylo pouze zopakováním práce, kterou již vykonal někdo jiný. Proto jsem se rozhodl ve své práci využít již hotovou knihovnu pro práci s počítačovou grafikou, zaměřil jsem se na prozkoumání jejích možností a využil jsem dostupných funkcionalit k dosažení cíle a experimentoval. Zvolena byla knihovna OpenCV (viz [12]), která již nabízí například implementaci algoritmu AdaBoost a to včetně trénovacího mechanismu, testů výkonnosti a API pro vlastní práci. Tato knihovna byla zvolena z hlediska jednoduché dostupnosti, je volně k použití a také je na internetu dostupné množství příkladů a dokumentace.

Splnit cíl práce, a tedy navrhnout a implementovat detektor schopný detekovat různě natočené obličeje, lze rozložit na dva hlavní úkoly. Prvním z nich je seznámit se s vybranou knihovnou, problematikou detekce pomocí zvolené metody a následně implementovat jednoduchý detektor obličejů v konkrétním natočení. Po splnění tohoto úkolu je možné přistoupit k řešení druhé části, která představuje rozšíření detektoru tak, aby detekoval obličeje v různých natočeních.

3.2 Třídy objektů a detekce

Detekovat obličeje v obraze lze různými metodami. Dnes je v oblasti počítačového vidění k dispozici již mnoho předpřipravených nástrojů použitelných pro tento cíl. Některé využitelné metody byly popsány v kapitole Přehled metod detekce obličejů. V této kapitole budou představeny dva teoretické postupy, jak uvedené metody rozšířit o detekci nezávislou na natočení obličeje.

Metody určené k detekci objektů v obraze jsou konstruovány tak, aby měly při klasifikaci konkrétního vzorku jistou toleranci. Je to proto, že obvykle konstruujeme detektor, který natrénujeme nad určitými daty, ale chceme, aby byl univerzální. Pod touto univerzálností si můžeme představit jistou toleranci klasifikátoru při ohodnocování konkrétního vzorku. Kdyby tato tolerance

neexistovala, detektor by správně určil pouze ty vzorky, které byly použity při testování. Protože ale chceme detekovat jistou třídu (typ) objektů, nesmí být klasifikátor vázán pouze na konkrétní přesné rysy jednotlivých vzorků, ale musí do něho z trénovací sady být extrahována jen taková podmnožina příznaků, které obecně popisují tento typ objektu.

Díky této vlastnosti není třeba pro každou jednotlivou možnost (například každý jeden stupeň natočení objektu) sestavovat přesný klasifikátor, ale je možné z celé škály natočení objektu vybrat jen několik zástupců, kteří budou každý reprezentovat svoji třídu. Díky určité podobnosti mezi jednotlivými podobami objektu z pohledu v podobných úhlech můžeme tedy sestavit pouze jeden klasifikátor zastupující celou třídu pro určitý rozsah natočení objektu. Získáme tedy N tříd, pro které platí, že N je výrazně menší než M, kde M je celkový počet možných natočení objektu.

Protože vytvoření klasifikátoru pro třídu objektů nad trénovací sadou obvykle stojí jisté prostředky, může být výhodné (z hlediska doby vývoje a následně i doby detekce) počet těchto tříd minimalizovat. Protože můžeme (v případě symetrických objektů, což obličej je) celý prostor vzorků rozdělit na dvě stejné skupiny, které jsou k sobě symetrické, lze při vývoji uvažovat pouze jednu polovinu celého prostoru a zabývat se konstrukcí detektoru pouze v rozmezí 0 až 90 stupňů. Detektor funkční nad celým prostorem pak získáme buď rozšířením detektoru o další třídy objektů, nebo prostým zrcadlením testovacího obrazu a druhým pokusem o detekci stejným systémem jako pro obraz původní.

Pokud budeme uvažovat systém tříd objektů výše uvedený, můžeme navrhnout několik algoritmů, které budou na základě těchto tříd klasifikovat daný objekt v různých natočeních. V následujícím textu budou uvedeny postupy, které byly použity při testování. Tyto obecné postupy jsou aplikovatelné na metody představené v kapitole Přehled metod detekce obličejů (AdaBoost, EigenFaces, Local Binary Patterns) i když pro účely této práce byla využita pouze implementace pro AdaBoost.

Detekce vzorku v obraze probíhá dle následujícího pseudokódu:

```
velikost_okna = velikost_obrazu;
while (velikost_okna >= minimální_velikost_detekovaného_objektu)
{
	for (x = 0;x + velikost_okna.x <= velikost obrazu.x; x += krok_posunu_okna)
	{
		for (y = 0;y + velikost_okna.y <= velikost obrazu.y; y += krok_posunu_okna)
		{
			klasifikační_výřez = výřez(data_obrazu, x, y, velikost_okna);
			klasifikační_výřez = uprav_velikost(klasifikační_výřez, nominální_velikost_klasifikátoru);
			klasifikuj(klasifikační_výřez);
		}
	}
	velikost_okna /= zmenšovací_poměr;
}
```

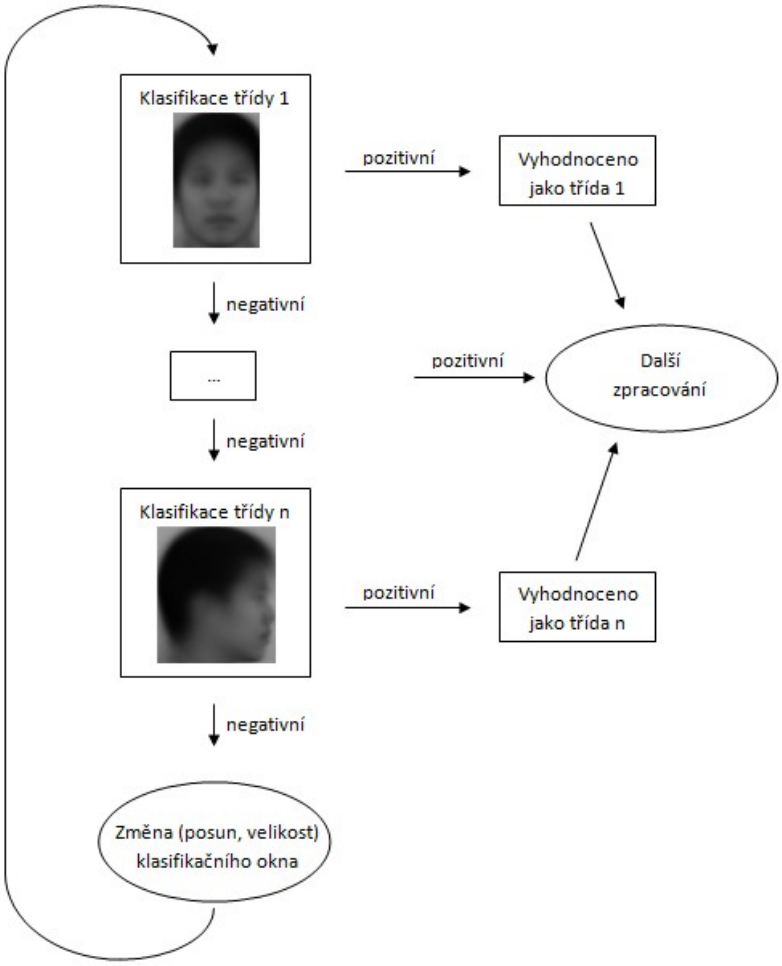
Pseudokód 1: Algoritmus detekce

Nad obrazem se tedy pohybuje pomyslné plovoucí okno, které představuje klasifikovanou oblast. Okno má nejdříve velikost celého obrazu a po dokončení detekce ve všech možných umístění (krok posunu okna nemusí být roven jednomu pixelu) okna nad testovaným obrazem se toto okno zmenší a cyklus se opakuje tak dlouho, doku není dosaženo limitu minimální velikosti okna.

Dále se už budeme klasifikací zabývat pouze jako spuštěním klasifikačního procesu (tedy aplikací klasifikátoru) nad vybraným výřezem obrazu zmenšeným na odpovídající velikost a výstupní informací klasifikátoru. Výstupní informací klasifikátoru v takovém případě je již pouze verdikt říkající zda daný vstupní vzorek odpovídá klasifikačním podmínkám či nikoli.

3.3 Postupná klasifikace

První možností rozšíření detektoru o detekci různě otočeného objektu (tedy různých tříd) je triviální postupné spuštění různých klasifikátorů nad obrazem. Tato metoda je výhodná svojí jednoduchostí použití. Nevýhodou metody je její pomalost. Pokud bude trvat například detekce v obraze jedním klasifikátorem (uvažujeme jistým způsobem průměrný obraz a průměrně složitý klasifikátor, který pracuje v průměrném čase) 1 sekundu a budeme chtít rozpoznávat 2 třídy objektů, získáme celkovou dobu detekce 2 sekundy. Je tedy zřejmé, že čas celkové detekce je přímo úměrný počtu rozpoznávaných tříd. Výstupem tohoto postupu je informace o tom, který klasifikátor na kterých pozicích označil výskyty. Výstupní informací tedy může být nejen informace o tom, na které pozici se nachází obličej, ale také informace o tom, jakým směrem je tento obličej přibližně natočen. Situaci znázorňuje následující diagram na **obrázku 9**:

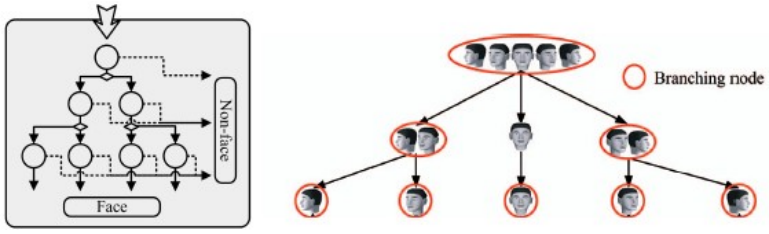


Obrázek 9: Diagram postupné klasifikace

3.4 Strom klasifikátorů

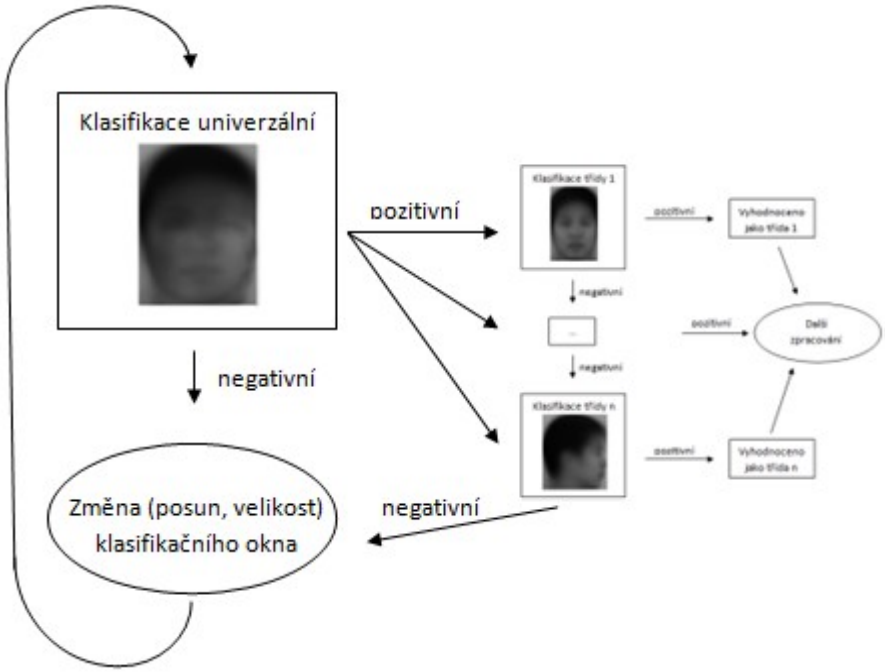
Druhá možnost rozšíření detekčního systému na rozpoznávání více tříd vychází z výše uvedeného postupu. Hlavní myšlenkou vylepšení (a tedy zkrácení potřebného detekčního času) je fakt, že celému obrazu se věnují všechny detektory pro všechny třídy natočení. Nabízí se tedy myšlenka minimalizace tohoto faktoru použitím univerzálního klasifikátoru. Čtenář si jej může představit jako hrubé síto, kterým propadne většina míst, na kterých se detekovaný objekt nenachází v žádné podobě. Zbude jen podmnožina potenciálních výskytů objektu, která je poměrně menší (dle hrubosti síta) než celé univerzum možných výskytů v obraze. Na takto vyfiltrované menší množství potenciálních výskytů je již možno pohlížet jako na nalezený obličej, pokud nezáleží na informaci o konkrétním natočení. Pokud je tato informace důležitá, lze zde postupně aplikovat již jemnější síta (klasifikátory určené pro jednotlivé třídy). Obdobný postup byl navržen v článku [8], který byl

částečnou inspirací této práce. **Obrázek 10** ukazuje strom klasifikátorů. Ve vrchní části je klasifikátor univerzální, směrem dolů se zvětšuje počet klasifikátorů specializovanějších.



Obrázek 10: Stromová struktura klasifikátorů, převzato z [8]

Na obrázcích vidíme obdobnou myšlenku v práci [8], kde jsou klasifikátory uspořádány do stromové struktury. Pokud je kořenem větve stromu vzorek vyřazen, není dále prozkoumáván. Pokud je označen jako pozitivní nález, posunuje se ve stromu k dalším dceřiným uzlům. Pokud tímto způsobem vzorek doputuje až na list stromu a tímto je taktéž označen jako pozitivní, je předpoklad, že se jedná o objekt daného typu. Diagram této metody je na **obrázku 11**.



Obrázek 11: Strom klasifikátorů

Tato metoda je však silně závislá na vlastnostech univerzálních uzlů (tedy takových, které nejsou listy). V případě využití kaskády AdaBoostu tak, jak je implementována v OpenCV se jedná hlavně o použitá trénovací data, maximální false alarm jednotlivých silných klasifikátorů a jejich počet. Je třeba najít vhodný kompromis mezi situací, kdy univerzální klasifikátor, coby nelistový uzel stromu, vyřazuje skutečné pozitivní výskyty, které by byly použití samotných listů stromu odhaleny,

a situací, kdy naopak univerzální klasifikátory vyvolávají přílišné množství false alarmů, což způsobuje zbytečné spouštění dceřiných klasifikátorů a to představuje nadbytečnou spotřebu strojového času a ztrácí se tím výhoda této modifikace. Právě nalezení tohoto kompromisu, tedy natrénování univerzálního klasifikátoru, bude hlavním úkolem experimentu.

4 Datasety

Na internetu je dnes dostupné množství volně použitelných datasetů. Existují dokonce databáze těchto datasetů, například [13]. Každý dataset má specifické vlastnosti, jenž ho předurčují jako výhodnější pro nějaké konkrétní využití, některé vlastnosti datasety zase omezují. V následujícím oddíle budou představeny některé datasety a předvedeny jejich odlišné vlastnosti. Některé datasety obsahují anotaci, což je obvykle textový soubor (výjimečně ve formátu XML), který popisuje pro jednotlivé fotografie umístění objektů. Většina datasetů je však pouze sadou fotografií a je již na experimentátorovi, aby sadu upravil tak, aby vyhovovala potřebným podmínkám (například udělal anotaci sám, případně sadu přepracoval vytvořením výřezů obsahujících pouze detekované objekty).

4.1 Schneiderman

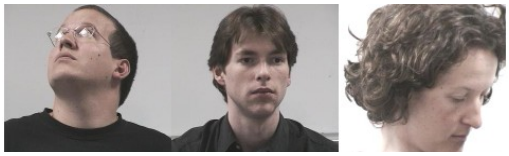
Tento dataset je klasickým zástupcem datasetu určeného na trénování univerzálního detektoru obličejů ve frontálním pohledu. Lze to odvodit z jeho složení. Na **obrázku 12** například vidíme, že se zde vyskytují různě staří lidé, různých ras, s brýlemi, ale také snímky při různých světelných podmínkách a obličeje s různými výrazy.

Obrázek 12: Ukázka z datasetu Schneiderman

Využitím tohoto datasetu tedy lze získat detektor, který bude relativně univerzální a bude relativně kvalitně fungovat za různých podmínek na různé typy obličejů s různým výrazem.

4.2 Cambridge

Tento dataset obsahuje 15 různých subjektů, které jsou každý nafocený v různých úhlech natočení. Pro každý subjekt existuje sada fotografií obsahující kombinace mezi vertikálním (od -60 do +60 stupňů náklonu) a horizontálním (od -90 do +90 stupňů) náklonem. Některé subjekty jsou nafoceny i vícekrát, například varianta s brýlemi a bez brýlí. Tato sada barevných fotografií je pořízena v interiéru za stejných světelných podmínek, kdy byly po místnosti umístěny nálepky na které se subjekty měly zaměřit a poté byly vyfoceny.



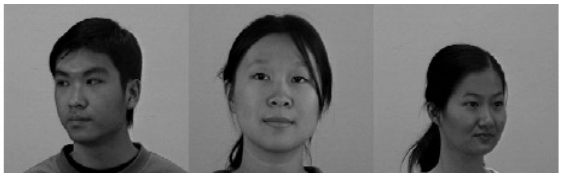
Obrázek 13: Ukázka datasetu Cambridge

Tento dataset je zajímavý svým širokým záběrem různých náklonů focených subjektů, bohužel však z důvodu malého počtu různých subjektů není příliš vhodný pro sestavení trénovací sady (viz Výběr datasetu) ale spíše pro vytvoření testovací sady detektoru.

Dostupný na: [14]

4.3 Our Database

Dataset se skládá ze dvou sad, každá obsahuje 90 nafocených subjektů, každý v horizontálním natočení od -90 do +90 stupňů v kroku po 5 stupních. Fotografie jsou pořízeny v interiéru za relativně podobných světelných podmínek. Jedná se o osoby stejné rasy, takže tím je dataset částečně chudší svým rozptylem různosti nafocených subjektů.



Obrázek 14: Ukázka datasetu Our Database

Výhodou tohoto datasetu je relativně velké množství snímků pro jednotlivá natočení, což jej činí vhodným kandidátem na dataset pro experimenty s různě natočenými obličeji.

Dostupný na: [15]

4.4 Kvalita datasetů

Nedílnou vlastností datasetů je také jejich kvalita. Kvalitu datasetů lze například popsat jako soubor vlastností, z nichž vliv některých bude dále popsán. Potřeba brát v úvahu níže uvedené vlastnosti vyplývá z chování použité metody, které lze laicky popsat jako snahu nalézt podobnosti napříč trénovací sadou.

Pro natrénování klasifikátoru potřebujeme v určité třídě natočení dostatečné množství vzorků buďto anotovaných, nebo automaticky anotovatelných (například tak, že se objekty v obraze nachází na stejném místě). První vlastností tedy je umístění hledaného objektu v trénovacích vzorcích. Na **obrázku 15** vidíme příklad různé pozice subjektu na snímcích, které oba představují subjekt nafocený za stejných podmínek:

Obrázek 15: Různá pozice subjektu v obraze

Ze vzorků je patrné, že aby trénovací algoritmus mohl mezi snímky hledat podobnosti, je nutné buďto vyříznout konkrétní oblasti zájmu, nebo sadu anotovat (dát trénovacímu algoritmu například seznam souřadnic, na kterých nalezne detekované objekty).

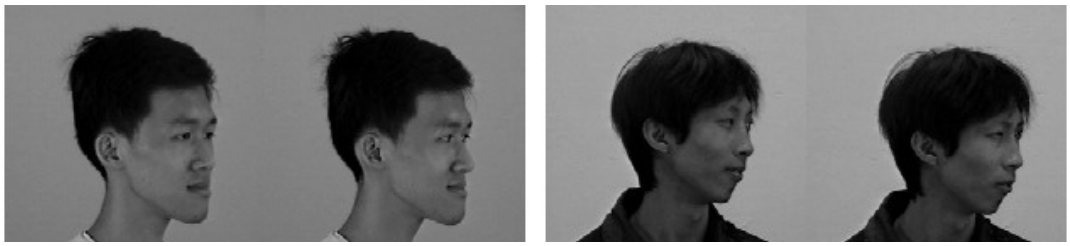
Další kvalitativní vlastností datasetů je pozadí vyplňující zbytek plochy, pokud odmyslíme detekovaný objekt. Z povahy funkce zvolené detekční metody by bylo ideální, kdyby pozadí bylo co nejrůznorodější. V takovém případě by byla relativně malá šance, že bude slabý klasifikátor zkonstruován například na hranici hledaného objektu a pozadí. Pokud by k tomu došlo, byl by objekt špatně klasifikovatelný, pokud by pozadí v provozu mělo jiný odstín než pozadí při trénování. Tento vliv je eliminován u datasetů tvořených jako sbírka fotografií z různých prostředí (viz. dataset Schneiderman). U vybraného datasetu je pozadí uniformní a proto (a také z výše uvedeného důvodu) je vhodné dataset upravit vyřezáním pouze zájmových oblastí se snahou vyhnout se ploše pozadí.

V neposlední řadě kvalitu datasetu ovlivňují samotné snímané subjekty. V mnoha případech jsou na fotografiích obsaženy různé rušivé vlivy, jako jsou brýle, naslouchátka, neobvyklé účesy, mimika, ruce a další, jak ukazuje **obrázek 16**:

Obrázek 16: Rušivé vlivy datasetů

Všechny tyto rozmanitosti na natrénovaného klasifikátoru znamenají větší univerzálnost (v dotčených oblastech nebude těmto různostem věnována taková pozornost), avšak zároveň mohou snížit kvalitu klasifikátoru (naopak důležité detaily mohou být zamaskovány). Může tedy být vhodné určité hodně netypické snímky z datasetu vyřadit.

Obdobným způsobem můžou také být nedodrženy deklarované vlastnosti jednotlivých snímků. Například úhel natočení může být kritický. Jednou z metod uměle vytvářených datasetů je situace, kdy jsou v ateliéru rozmístěny značky, na které se má subjekt dívat a přitom je pořízen snímek. Nezřídka se stává, že subjekty se na značky dívají pouze očima, hlavu mají přitom namířenu jinam, případně se na značku nedívají vůbec. Demonstrují to snímky na **obrázku 17**. Na první dvojici se jedná o natočení 50 a 65 stupňů, kdy v druhém případě subjekt pozoroval cíl spíše očima než pohybem hlavy, na druhé dvojici se jedná dokonce o rozdíl 25 stupňů (natočení 50 a 75).



Obrázek 17: Nedodržení deklarovaných parametrů

Tyto nepřesnosti udávají určitou chybu při rozdělení snímků do tříd podle natočení. Pozitivem však je, že díky této chybě lze tříd natočení vytvořit relativně málo a tak získat v každé z nich větší množství snímků.

4.5 Výběr datasetu

Při výběru datasetu hraje velkou roli typ experimentu, pro který dataset vybíráme. V případě této práce bylo důležité vyzkoušet princip úpravy detektoru tak, aby detekoval různě natočené obličeje. Aby byl co nejvíce potlačen vliv různých podmínek (světelných, typů lidí...) a byla výpovědní hodnota výsledků co největší, zdá se vhodné využít jeden konkrétní dataset (nevytvářet nový například smícháním více již existujících, čímž by se zvětšil rozptyl různých rysů výsledného

datasetu). Soustředil jsem se spíše na technologii detektoru než jeho univerzálnost za různých podmínek. Je tedy vhodné využít dataset, kde jsou subjekty foceny za podobných světelných podmínek, ideálně stejné rasy. Takové podmínky lépe splňují datasety vytvořené uměle, přímo pro tyto účely, než datasety sestavené z různých fotografií pořízených nahodile při různých situacích (viz. dataset Schneiderman). Dalším důležitým hlediskem při výběru datasetu je jeho obsáhlost. Protože je využit pouze jeden dataset, je třeba nalézt takový, který obsahuje dostatečně velký počet snímků pro jednotlivá natočení a přitom dostatečný počet tříd natočení. Tato kombinace z dostupných datasetů výrazným způsobem eliminuje výběr. I z velkého množství datasetů lze jen těžko vybrat takový, který by splňoval ideální podmínky (byl volně dostupný, obsahoval dostatečné množství snímků pro jednotlivá natočení, obsahoval velké množství různých subjektů a byl anotovaný). Nejlépe tyto vlastnosti splňuje Our dataset (i když není anotovaný), který navíc obsahuje všechny kombinace natočení pro všechny subjekty dvakrát, což nám jej dovoluje rozdělit na dvě poloviny. Toho je možné s výhodou využít tak, že polovina datasetu je využita pro trénování klasifikátoru a druhá polovina při testování jeho kvality. Z těchto důvodů byl tento dataset zvolen jako nejlepší z nalezených pro dané účely.

Celá sada obsahuje dva komplety snímků A_*S*_*X*.jpg a B_*S*_*X*.jpg. V obou případech *S* označuje číslo subjektu a *X* označuje úhel horizontálního natočení obličeje na snímku. Oba komplety obsahují 90 stejných nafocených osob. Pro každou osobu v každém kompletu existuje řada 37 snímků po pěti stupních natočení od -90 do +90 stupňů. Díky tomu, že sada obsahuje dva komplety snímků, bylo možné jeden (A) využít pro trénink klasifikátorů a druhý (B) pro jejich testování. Při testování bylo využito pouze jedné poloviny sady pro ověřování nacházení pozitivních výskytů a sady negativních snímků pro měření hodnoty false alarm. Toto omezení vyplynulo z toho, že vzhledem k malému množství dostupných vzorků a strojového času nebylo možné natrénovat univerzální klasifikátor, který by fungoval například pro jiné rasy lidí.

5 Realizace

Následující kapitola popisuje praktické postupy, kterých je využito k realizaci detektoru obličejů.

5.1 Způsob využití OpenCV

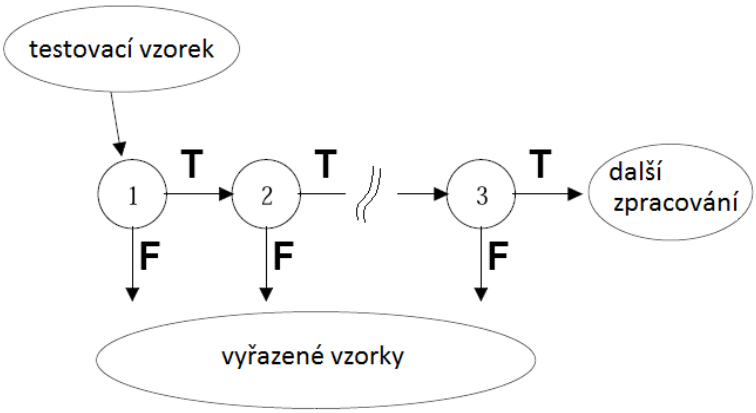
Použitá knihovna již obsahuje přímo funkcionality pro využívání natrénovaných klasifikátorů a zpracovávání výsledků. Jádrem práce byly experimenty s trénovacími daty a algoritmy. Testování probíhalo jednoduchým programem, jehož úkolem bylo načíst ze souboru uložený klasifikátor a spustit detekci nad vstupním obrázkem. Výstupem byla vždy informace, na kterých místech obrazu byl detekován pozitivní výskyt. Zjednodušená kostra programu vypadá následovně:

1. Načíst klasifikátor(y)
2. Načíst vstupní obrázek
3. Detekovat výskyty v obraze
4. Označit pozitivní výskyty

Pro porovnávání jednotlivých postupů byl jedním z hlavních hledisek celkový čas detekce. Čas jednotlivých metod byl porovnáván jako celkový čas detekční fáze nad referenční sadou testovacích obrazů.

5.2 Implementace AdaBoostu v OpenCV

Implementaci detekce pomocí AdaBoostu v OpenCV představil Paul Viola a později vylepšil Rainer Lienhart. Využívá se zde silných klasifikátorů (vzniklých pomocí AdaBoostu ze slabých příznaků Haarových příznaků), kterých je vytvořeno více a jsou sestaveny do sériové kaskády. Tato myšlenka ještě dále zvětšuje přesnost a urychluje práci detektoru. Každý ze silných příznaků je nad určitým vstupním testovacím vzorkem schopen rozhodnout, zda se jedná či nejedná o detekovaný objekt. V případě, že klasifikátor označí vzorek za neodpovídající, je vzorek vyřazen. V opačném případě, kdy je vzorek označen za vyhovující klasifikátoru, je předán k testu následujícímu klasifikátoru v kaskádě. Pouze pokud je testovaný vzorek postupně všemi klasifikátory kaskády označen za pozitivní, je celkově označen jako detekovaný objekt, jak naznačuje následující diagram na **obrázku 18**:



Obrázek 18: Diagram kaskády v OpenCV, převzato z [6]

Díky tomuto způsobu je zvýšena rychlost i přesnost. Největší zbytečná spotřeba strojového času je věnována porovnávání klasifikátorů v částech obrazu, kde se detekovaný objekt nenachází. Uvedeným způsobem je v těchto oblastech porovnávání eliminováno pouze na první klasifikátory, které vzorek vyřadí jako negativní, takže je ušetřen čas na zbytečném porovnávání zbylými klasifikátory. Zároveň je zvýšena i kvalita detektoru jako celku, protože pravděpodobnost, že negativní vzorek projde celou kaskádou je dána součinem pravděpodobností vyřazení vzorku všech klasifikátorů. Také pravděpodobnost vyřazení pozitivního vzorku je relativně malá, protože pravděpodobnost jednotlivých klasifikátorů vyřadit pozitivní vzorek je relativně nízká.

5.3 Trénování kaskády v OpenCV

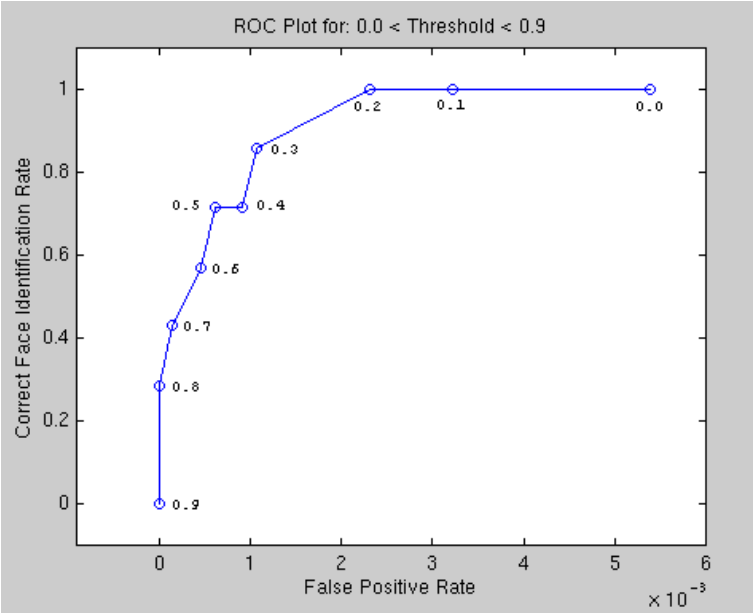
Se zvolenou knihovnou OpenCV je dodáván i nástroj **haartraining** pro trénování klasifikátorů. Pro natrénování klasifikátoru na určitý objekt je třeba sady pozitivních a sady negativních vzorků. Výběru datasetu pozitivních vzorů pro trénování se věnuje kapitola Výběr datasetu. Jako dataset negativních vzorků je možné použít dostatečně rozsáhlou databázi fotografií, které neobsahují detekované objekty.

Nástroj pro trénování klasifikátorů požaduje na vstupu dataset negativních vzorků, dataset pozitivních vzorků a dodatečná nastavení. Jako dataset negativních vzorků poslouží textový soubor, v němž jsou uvedeny cesty k jednotlivým negativním vzorkům. Dataset pozitivních vzorků je dodán v souboru .vec, což je katalog zmenšenin pozitivních vzorků. Na vytvoření katalogu je v balíku OpenCV připraven nástroj **createsamples**.

Během procesu trénování je možno sledovat postup, jak jsou tvořeny jednotlivé silné klasifikátory ze slabých a postupně se mění hodnoty false alarm a hit rate. False alarm představuje míru pravděpodobnosti případu, kdy je pozice, na které se objekt nenachází, označena jako výskyt. Hit rate naopak označuje pravděpodobnost správného označení výskytu. Jedním z kvalitativních parametrů natrénovaného klasifikátoru tedy můžou být právě tyto hodnoty. Ideální situace nastane tehdy, když je false alarm 0% (tedy nejsou nahlášeny falešné výskyty) a hit rate je 100% (tedy není opomenut žádný skutečný výskyt). V dalším textu budou tyto pojmy využívány v tomto významu.

Trénovací nástroj vytvoří složku, do které ukládá mezivýsledky, což je velká výhoda při potřebě přerušit běh trénování. Po dokončení je vytvořen XML soubor obsahující specifikaci natrénovaného klasifikátoru.

Kvalita klasifikátorů lze porovnávat například ROC (Receiver Output Characteristics) křivkou, která vyjadřuje závislost false alarmu (tedy mylně označených snímků za pozitivní) a hit rate (tedy správně označených pozitivních výskytů) na vybraném proměnném parametru. Při trénování AdaBoost klasifikátoru si můžeme za proměnný parametr zvolit například váhový význam konkrétního slabého klasifikátoru. Na **obrázku 19** je příklad ROC křivky detektoru určitého typu objektů.

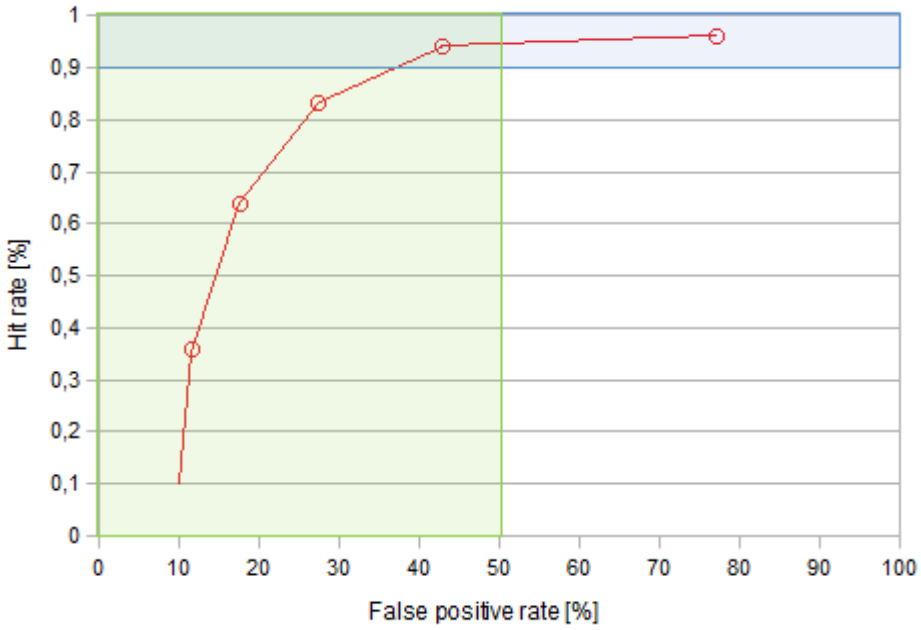


Obrázek 19: ROC křivka, převzato z [16]

Vidíme, že správná funkce klasifikátoru je kompromisem mezi velkým množstvím chybně detekovaných výskytů a velkým množstvím vynechaných výskytů. Nejlepší pozicí nastavení parametru je tedy někde kolem prudkého zlomu charakteristiky, kdy klasifikátor vynechává relativně únosné množství výskytů. Nastavení těchto parametrů pak závisí na účelu použití detektoru.

V případě, kdy je třeba zachytit všechny (ideálně) výskyty, například využití v detektivním systému, je třeba se při dané kvalitě klasifikátoru smířit s určitým množstvím špatně detekovaných míst. V opačném případě, například v situaci, kdy je potřeba obrazy rozpoznávat v reálném čase a pak je zpracovávat (hlavním kritériem je čas), je možné zvolit opačný extrém, kdy se smíříme s tím, že mnoho výskytů nebude detekováno. V takovém případě je při detekovaném objektu vyšší pravděpodobnost, že se zde objekt opravdu nachází.

Implementace kaskády AdaBoost klasifikátorů v OpenCV řeší tento problém multiplikací klasifikátorů na úkor času. Při trénování se zadává parametr maximálního false alarmu a minimálního hit rate. Tyto hodnoty jsou směrodatné při vytváření každého silného klasifikátoru kaskády. Na **obrázku 20** je příklad, kdy je trénovacímu nástroji zadáno, že výsledný klasifikátor musí splňovat podmínku, kdy alespoň 90% skutečných výskytů musí být nalezeno (modrá oblast) a smí být označeno maximálně 50% mylně (zelená oblast):



Obrázek 20: Hledání optimálního bodu pomocí ROC křivky

Trénovací algoritmus AdaBoost pak upravuje slabé klasifikátory a jejich váhy v cyklu tak dlouho, dokud nejsou splněny obě podmínky a tudíž není vybrán bod křivky ležící v ploše průniku modré i zelené plochy. Průběh algoritmu je naznačen následujícím pseudokódem:

```
cílový_false_alarm = nastavený_false_alarm_klasifikátoru ^ nastavený_počet_klasifikátorů;
while (∏ false_alarmy_dosavadních_klasifikátorů > cílový_false_alarm)
{
        while (false_alarm_tvořeného_klasifikátoru > nastavený_false_alarm_klasifikátoru ||
                hit_rate_tvořeného klasifikátoru < nastavený_hit_rate_klasifikátoru)
        {
                přidej_slabý klasifikátor();
                uprav_váhy_slabých_klasifikátorů();
        }
        přidej_vytvořený_klasifikátor();
}
```

Pseudokód 2: Tvorba kaskády klasifikátorů v OpenCV

Z takto vytvořených silných klasifikátorů je pak sestavena kaskáda, pro kterou platí, že pokud je například false alarm 50% a hit rate 99% a je vytvořena kaskáda 20 klasifikátorů splňujících tyto podmínky, získáme klasifikátor zajímavých vlastností. Protože pro pozitivní označení vzorku je třeba aby vzorek byl označen všemi prvky kaskády jako pozitivní, získáme celkový false alarm jako $0{,}5^{20} \approx 9{,}5*10^{-7}$ a celkový hit rate jako $0{,}99^{20} \approx 0{,}82$. Po uplatnění této kaskády jsme potom získali na charakteristice bod, který leží na 82% hit rate a velice blízko 0% false alarmu. Uvedená metoda má tedy v porovnání s jedním silným klasifikátorem výrazně vyšší relevanci, avšak na úkor pracovního času. Díky tomu, že se vyhodnocují jednotlivé klasifikátory kaskády postupně, je většina negativních výskytů vyřazena již v prvních členech kaskády.

5.4 Další použité nástroje

Kromě knihovny OpenCV jsem k vytvoření demo aplikací potřeboval ještě vývojové prostředí. Pro zpracování aplikací jsem si vybral Microsoft Visual Studio a jazyk C#. Protože knihovna OpenCV je implementována v jazyce C, neobsahují její překompilované knihovny rozhraní, které by bylo možné přímo z jazyka C# využít. Na internetu se mi však podařilo najít řešení v podobě wrapperu pro Visual Studio [17]. Tímto řešením je knihovna, která zprostředkovává rozhraní mezi knihovnami OpenCV a Visual Studiem.

Pro implementaci dávkových aplikací pro měření času jsem využil prostředí Dev-C++ a řadu jednorázových skriptů, které sloužily například pro:

- automatickou tvorbu výřezů
- anotaci knihovny výřezů
- práci s histogramem
- výpočet průměrných podob vzorků
- a další

6 Experimenty a vyhodnocení

Experimenty uvedené v této kapitole mohou čtenáři přinést náhled na účinnost jednotlivých návrhů uvedených v kapitole Návrh řešení.

Jednotlivé experimenty byly pro možnost porovnání prováděny nad referenční testovací sadou dat (jedna polovina použitého datasetu). Využité klasifikátory byly vždy natrénovány nad vzorky druhé poloviny použitého datasetu.

Porovnávané vlastnosti metod použitých v experimentech jsou:

1. Celkový čas detekce nad celou testovací sadou
2. Počet správně označených pozitivních výskytů
3. Počet nesprávně označených negativních výskytů

6.1 Trénování klasifikátorů

Prvním úkolem je natrénování klasifikátoru jedné třídy natočení tak, aby bylo dosaženo co nejlepších kvalitativních parametrů klasifikátoru. Je tedy třeba zjistit, jakou podobu mají mít trénovací data, aby z nich trénovací aplikace mohla vytvořit co možná nejkvalitnější klasifikátor. Po vyřezání vzorků z původních snímků jsem získal trénovací sadu obsahující například následující vzorky:

Obrázek 21: Ukázka trénovacích výřezů

Klasifikátor natrénovaný nad touto sadou (cca 600 snímků) produkuje následující výsledky prezentované **tabulkou 1** a **obrázkem 22**.

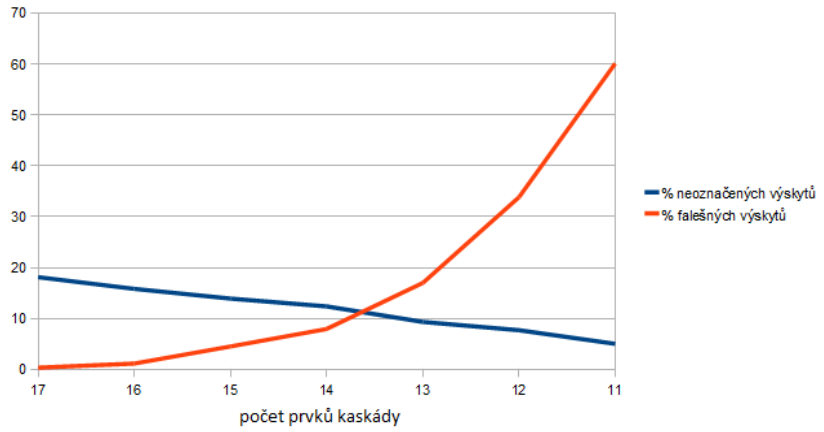
	Celkem testováno	**Chybně (ne)klasifikováno**	**Poměr**
Neobsahující obličej	1101	3	0,27%
Obsahující obličej	1044	189	18,10%

Tabulka 1: Výsledky testu frontálního klasifikátoru

Obrázek 22: Použití frontálního klasifikátoru

K natrénování klasifikátoru byly využity výřezy, testováno bylo na testovací polovině datasetu. Jak bylo popsáno v kapitole Výběr datasetu, detekční okno je posunováno po ploše obrazu, a každou takto vzniklou kombinaci polohy a zvětšení lze označit jako pozitivní či negativní výskyt. V tomto textu budou výskytem či falešným výskytem nadále myšleny již celé oklasifikované obrazy jako celky. Pokud tedy bude zmíněn správně označený výskyt, byl detektorem odhalen správný počet obličejů v obraze na správných pozicích. V opačném případě buďto v obraze byly označeny obličeje na místech kde se nevyskytují, nebo některé výskyty nebyly označeny. V takovém případě obraz jako celek hodnotím jako špatně ohodnocený. Z uvedených výsledků je patrno, že chyba, kdy je chybně detekován objekt v místě kde ve skutečnosti není, se vyskytuje relativně málo (tedy hodnota celkového false alarmu je malá) oproti chybě druhé, kdy nejsou zachyceny všechny výskyty (tedy hodnota hit rate je příliš malá). Větší pozornost tedy věnuji snaze zvýšit parametr hit rate klasifikátoru.

Z logiky kaskády vyplývá, že uvedené hodnoty lze ovlivnit jednak nastavením hodnoty false alarm jednotlivých klasifikátorů kaskády a pak také jejich celkovým počtem. Následuje analýza závislosti uvedených hodnot na počtu prvků kaskády. Je třeba si však uvědomit, že pro různá data a různě natrénované klasifikátory může být tato závislost jiná. Proto čtenář může považovat graf na **obrázku 23** pouze za orientační.



Obrázek 23: Závislost hodnot false alarm a hit rate na celkovém počtu klasfikátorů kaskády

Na vodorovné ose je vynesen postupně se snižující počet klasifikátorů v testovací kaskádě. Vzhledem k pozvolna klesající křivce představující chybu neoznačených výskytů a prudce stoupající křivce falešných výskytů lze odvodit, že spíše než o nevhodně nastavené cílové parametry se jedná o nevhodně upravená trénovací data.

Zkusil jsem se tedy zaměřit na tvorbu klasifikátoru pro profilový pohled na lidský obličej. Protože se v pohledu z profilu ve velké míře uplatňují vlivy jako je tvar hlavy, účes, poloha ucha, proporční rozdělení a další, rozhodl jsem se otestovat účinnost klasifikátoru, který bude natrénován nad vzorky v podobě **obrázku 24**. Jedná se o výřezy z pohledů v úhlech 75 až 90 stupňů.

Obrázek 24: Profilové trénovací snímky

Testovací klasifikátor byl natrénován za použití 719 pozitivních snímků ve tvaru jako na **obrázku 24** a 650 negativních snímků. V ideálním případě by klasifikátor měl správně detekovat obličeje v pohledu z profilu. Jak však ukazuje **obrázek 25**, úspěšnost nebyla příliš velká.

Obrázek 25: Použití profilového klasifikátoru

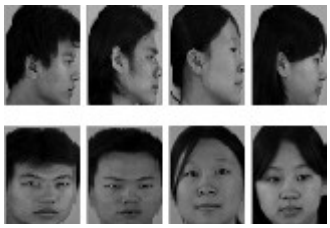
Klíčem neúspěchu dle pozdějších zjištění bylo, že katalogizér OpenCV **createsamples** nezohledňuje poměr stran trénovacích snímků. Snímky jsou do .vec katalogu uloženy tak, že dané zdrojové výřezy jsou roztaženy v obou rozměrech tak, aby vyplňovaly zadaný rozměr. Pokud je tedy zadán rozměr například 20 x 20 pixelů, jsou trénovací snímky které nemají poměr stran 1:1 zdeformovány. A protože klasifikátor natrénovaný nad zdeformovanými snímky není při detekci deformován zpět, takový klasifikátor pak detekuje pouze objekty v obraze, které jsou obdobným způsobem deformované. Z tohoto důvodu je třeba při vytváření katalogu zadávat skutečný poměr stran jednotlivých snímků (průměr).

Obdobnými experimenty byly stanoveny následující požadavky na trénovací snímky:

- dostatečné množství snímků (empiricky ověřené minimum je cca 200 pro dostatečnou funkci pro testování nad druhou polovinou datasetu)
- nezáleží na roztažení histogramu snímků
- záleží na poměru stran snímků (vzorky jsou při detekci zvětšovány pouze v poměru stran 1:1, není tedy měněn poměr stran klasifikátoru vůči detekovanému obrazu)
- snímky musí obsahovat co nejmenší množství mezi snímky konstantního pozadí, aby nemohlo být považováno za charakteristický znak hledaného objektu
- jednotlivé části objektu musí na snímcích mít podobnou polohu

Výše získané informace byly získány během experimentů s datovými sadami a trénovací a testovací aplikací. Protože, jak jsme zjistil, je třeba uvažovat poměr stran snímků, je třeba, aby profilové a frontální trénovací snímky (a tedy i klasifikátor) měly podobný (ideálně stejný) poměr stran. Tento požadavek vyplývá z faktu, že podle návrhu Strom klasifikátorů je třeba vytvořit univerzální klasifikátor, v němž jsou použity jak snímky frontálního, tak profilového pohledu. Kdyby tyto snímky nesplňovaly tuto podmínku, nastal by problém s roztahováním trénovacích dat zmíněný v předchozích odstavcích. Výslednou podobu trénovacích dat pro vyhodnocení jednotlivých návrhů předvádí **obrázek 26**. Vypočítáním průměru poměrů stran jednotlivých trénovacích dat jsem určil výsledný poměr stran jednotlivých snímků. Na tento pak byly všechny snímky upraveny, některé

byly roztaženy, jiné zase mírně zúženy.



Obrázek 26: Konečná podoba trénovacích dat

6.2 Postupná klasifikace

Po úspěšném natrénování klasifikátorů na jednotlivé třídy natočení lze přistoupit k druhému kroku, a sice odzkoušení prvního návrhu, který je triviálním postupným vyhodnocováním jednotlivými klasifikátory. Pro tento test tedy byl natrénován klasifikátor dle parametrů uvedených v **tabulce 2**.

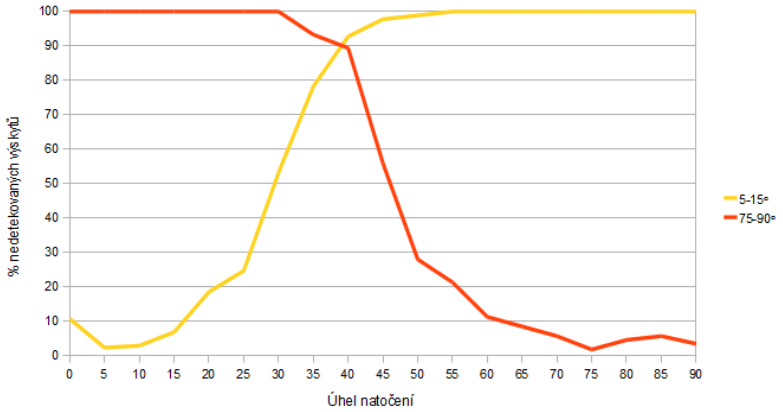
Parametr	Frontální klasifikátor	Profilový klasifikátor
Použité snímky	+5...+15, -5...-15(zrcadlené)	+75...+90, -75...-90(zrcadlené)
Pozitivních snímků	625	719
Negativních snímků	800	800
Cílový false alarm	$0{,}5^{20}$	$0{,}5^{20}$
Počet klasifikátorů kaskády	9	9

Tabulka 2: Trénovací parametry frontálního a profilového klasifikátoru

Pro testování detekce různých natočení byla vytvořena baterie testovacích sad pro různá natočení, jak naznačuje **obrázek 27**.

Obrázek 27: Testovací sady pro jednotlivá natočení

Pro každé natočení je k dispozici téměř 180 testovacích snímků. Pro jednotlivá natočení můžeme v sadách detekovat testovaným detektorem. Získá se tak charakteristika tolerance klasifikátorů (**obrázek 28**).



Obrázek 28: Uhlová toleranční charakteristika klasifikátorů

Z grafu je patrná využitelnost klasifikátoru natrénovaného na objekty v pohledu z úhlu α pro detekci téhož objektu v pohledu z úhlu β, kde α≠β. Čím je rozdíl mezi α a β větší, tím se přirozeně účinnost detektoru snižuje.

V procesu detekce byl každý vstupní snímek detekován postupně oběma klasifikátory. Na grafu na **obrázku** 28 lze pozorovat překryv působnosti obou klasifikátorů. Při vývoji detektoru podle tohoto návrhu pro konkrétní situaci je pak třeba přihlédnout k požadovaným vlastnostem výsledného produktu. Pokud je kladen požadavek na malé množství promeškaných výskytů, je třeba vytvořit větší množství klasifikátorů a tím hustěji zaplnit vodorovnou osu grafu, což na druhou stranu přináší delší detekční čas.

Průměrně detekováno frontálních výskytů	86,70%
Průměrně detekováno profilových výskytů	100,00%
Celkový detekční čas	220ms
Informace o třídě natočení výskytu	ANO

Tabulka 3: Celkový výsledek experimentu

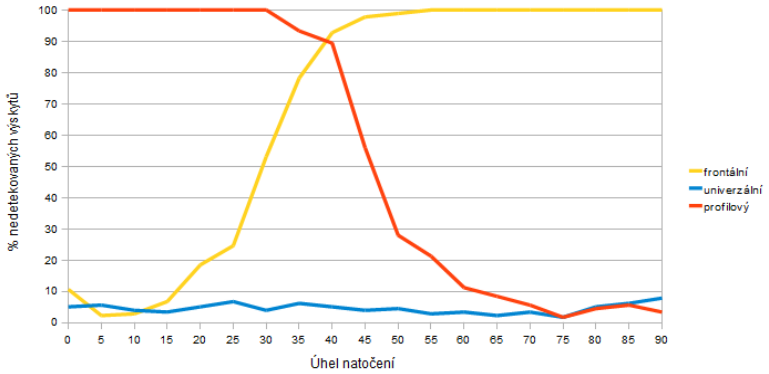
6.3 Strom klasifikátorů

Druhý návrh je vylepšením prvního návrhu, jehož cílem je zmenšit detekční dobu. Byl tedy natrénován univerzální klasifikátor s parametry uvedenými v **tabulce 4**. Jako trénovací data byla použita podmnožina dat, ze kterých byly vytvářeny frontální a profilový klasifikátor.

Parametr	Univerzální klasifikátor
Použité snímky	+5...+15, +75...+90 -5...-15, -75...-90(zrcadlené)
Pozitivních snímků	500
Negativních snímků	450
Cílový false alarm	$0,5^{20}$
Počet klasifikátorů kaskády	10

Tabulka 4: Parametry univerzálního klasifikátoru

Tento univerzální klasifikátor má za cíl odfiltrovat co nejvíce potenciálních výskytů, které výskyty ve skutečnosti nejsou. Tím se zmenší počet pozic, na které je třeba použít dva (či více, podle počtu tříd natočení) klasifikátory. Je třeba však nalézt kompromis mezi situací, kdy univerzální klasifikátor vytváří tak velké množství false alarmů, že metoda ztrácí výhodu své menší časové náročnosti a situací, kdy univerzální klasifikátor má příliš malý hit rate. Graf na **obrázku 29** porovnává klasifikátory jednotlivých natočení z předchozího návrhu s informací o univerzálním klasifikátoru.



Obrázek 29: Uhlová toleranční charakteristika klasifikátorů

Průměrně detekováno frontálních výskytů	100,00%
Průměrně detekováno profilových výskytů	100,00%
Celkový detekční čas	130ms
Informace o třídě natočení výskytu	NE

Tabulka 5: Celkový výsledek experimentu

6.4 Varianty metody Strom klasifikátorů

Při bližším pohledu na operace prováděné v předchozí variantě můžeme odvodit několik možností zpracování univerzálně oklasifikovaného výskytu. Univerzální klasifikátor vrací polohy označené jako výskyty univerzální podoby objektu. K těmto datům se můžeme chovat následovně:

1. Označit je jako konečná s tím, že se zde vyskytuje objekt v některém z natočení. Jistota tohoto faktu je však zatížena pravděpodobností vyplývající z kompromisu uvedeném v předchozím návrhu. Na **obrázku 31** v tomto případě vidíme výsledek detekce.

Obrázek 31: Detekce pouze univerzálním klasifikátorem

2. Nad označeným výřezem detekovat jednotlivými specializovanějšími klasifikátory podrobněji třídu objektu. Protože však byl univerzální klasifikátor trénován nad jinými daty (mix vzorků jednotlivých natočení), je pravděpodobné, že bude univerzální výskyt označen s jistou poziční tolerancí oproti výskytu, který by byl detekován klasifikátorem určeným pro konkrétní třídu natočení. Tuto situaci je možno řešit například následujícími způsoby:

    1. Kolem nalezeného univerzálního výskytu vytvořit toleranční okno (rozšířit výřez) a v tomto okně se pokusit specializovanějšími klasifikátory detekovat objekt v konkrétním natočení. Výsledek je na **obrázku 32**.

Obrázek 32: Detekce univerzálním klasifikátorem (červená) a následně v tolerančním okně frontálním (zelená) a profilovým (modrá) klasifikátorem

2. Detekční funkce OpenCV v obraze vrací pouze tzv. silné kandidáty. Jedná se o situaci, kdy je na podobném místě v určitém rozptylu více kandidátů na výskyt detekovaného objektu. OpenCV tyto sobě blízké slabé kandidáty shlukuje. Pokud je počet blízkých slabých kandidátů roven nebo větší než nastavená hranice, je zde ohlášen silný kandidát. Výstupem detekční funkce jsou potom pouze silní kandidáti. Počet slabých kandidátů pro shlukování však lze nastavit. Nabízí se tedy možnost místo tolerančního okna (viz předchozí varianta) specializovanější klasifikátory spouštět pouze nad všemi slabými kandidáty.

Obrázek 33: Detekce specializovanými klasifikátory na přesné pozici detekce univerzálního klasifikátoru

Na **obrázku 33** je patrný výsledek takového pokusu. Tímto experimentem bylo ověřeno, že výskyt detekovaný univerzálním klasifikátorem se nepřesně kryje s výskyty detekovanými specializovanými klasifikátory.

7 Závěr

V počátečních kapitolách práce byly popsány dosavadní přístupy k detekci objektů v obraze, byly představeny některé dosud vyzkoušené postupy detekce různě natočených obličejů a byly charakterizovány vybrané datasety. Jako vhodný kandidát z detekčních metod byl vybrán algoritmus AdaBoost. Jako rozšíření pro detekci obličejů v různých natočeních byla navržena metoda postupné klasifikace, kdy byl obraz postupně detekován klasifikátory pro jednotlivé třídy natočení. Druhou navrženou metodou byl strom klasifikátorů, který představuje seskupení klasifikátorů do různých úrovní od jednoho univerzálního k více specializovaným. Při implementaci daných rozšíření byla využita knihovna OpenCV, která již obsahuje připravené funkcionality pro využití algoritmu AdaBoost.

Po implementaci byly měřeny kvalitativní a časové charakteristiky uvedených metod. Jednotlivé návrhy se od sebe liší hodnotami hit rate a false alarm, detekčním časem a druhem výstupní informace. Závěrem lze říci, že jednotlivé metody jsou v praxi použitelné pro detekci různě natočených obličejů, ale výběr konkrétního návrhu velmi záleží na cílové aplikaci.

Pokud cílová aplikace vyžaduje co nejvyšší hit rate, může to znamenat prodloužení detekčního času či zvýšení hodnoty false alarm. V případě použití metody postupné klasifikace velice záleží na trénovacích datech a na počtu tříd natočení (tedy počtu klasifikátorů). V případě stromu klasifikátorů se počet použitých klasifikátorů oproti postupné klasifikaci zmenšuje. Pokud aplikace vyžaduje informaci o úhlu natočení objektu, je potom potřeba, aby strom klasifikátorů měl více hladin. Celkově je v konkrétní aplikaci třeba určit prioritní parametr, zda jím je čas nebo kvalita klasifikace. Od toho se pak odvíjí vybraná metoda, počet tříd natočení, uspořádání klasifikátorů, hloubka klasifikačního stromu, počet prvků kaskád jednotlivých klasifikátorů, nastavení hodnot false alarm a hit rate jednotlivých klasifikátorů a další parametry. Přímo se porovnávat s jinými pracemi na toto téma je problematické, protože každá se zaměřuje na jiný cíl (některé například celkový čas detekce neuvažují jako hodnotící kritérium). Je však třeba říci, že například u metody stromu klasifikátorů časový zisk oproti postupné klasifikaci hodně záleží na různých parametrech, například na počtu obličejů, které se na obraze skutečně nachází. Pokud totiž na obraze nebude žádný obličej, postupná klasifikace bude na celý obraz aplikovat dva klasifikátory (při dvou třídách natočení), zatímco strom klasifikátorů aplikuje na celý obraz pouze jeden univerzální klasifikátor. Tím je (za ideálních podmínek) strom klasifikátorů dvakrát rychlejší. Pokud však bude obraz kompletně zaplněn obličeji, metoda postupné klasifikace stále pouze pro celý obraz použije dva klasifikátory, zatímco metoda stromu klasifikátorů (v podobě, jaká byla použita v experimentu) na takový obraz bude muset využít tři klasifikátory, čímž se již stává o polovinu pomalejší, než metoda postupné

klasifikace. Je tedy patrné, že výhody jednotlivých metod jsou závislé nejen na charakteru cílové aplikace ale také na obvyklých podmínkách, za jakých bude detektor provozován. A možná právě to je důvodem, proč se v pracích na toto téma neuvažuje parametr detekčního času.

V uvedených experimentech natrénované klasifikátory jsou trénovány pouze nad malým počtem relativně uniformních dat, proto nebudou moc dobře fungovat pro nahodilé snímky. Pro vytvoření takového klasifikátoru by bylo třeba vytvořit univerzální datasety (jako je například Schneiderman) pro jednotlivé třídy natočení. Potom by vše pravděpodobně fungovalo s úspěchem i pro nahodilá data.

8 Zdroje informací

[1] Sezin Kaymak: Face Detection, Recognition and Reconstruction using Eigenfaces. 2003, class presentation

[2] http://cmp.felk.cvut.cz/cmp/courses/recognition/Labs/pca/index.html, 16.5.2010

[3] Tomáš Jelínek: Detekce pohybujících se objektů ve video sekvenci. Brno, 2007, diplomová práce, FIT VUT v Brně

[4] Heikkilä, M., Pietikäinen, M.: A texture-based method for modeling the background and detecting moving objects. 2005

[5] F. Suard, A. Rakotomamonjy, A. Bensrhair: Pedestrian Detection using Infrared images and Histograms of Oriented Gradients. Tokyo, 2006, Intelligent Vehicles Symposium

[6] P. Viola, M. Jones: Robust Real-time Object Detection. 2001, Kanada, SECOND INTERNATIONAL WORKSHOP ON STATISTICAL AND COMPUTATIONAL THEORIES OF VISION – MODELING, LEARNING, COMPUTING, AND SAMPLING

[7] Jan Šochman, Jiří Matas: AdaBoost, http://cmp.felk.cvut.cz/~sochmj1/adaboost_talk.pdf, 16.5.2010

[8] Chang Huang, Haizhou Ai, Yuan Li, Shihong Lao: High-Performance Rotation Invariant Multiview Face Detection, 2007, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE

[9] Shihong Lao a kol.: A Fast 360-Degree Rotation Invariant Face Detection System. Japonsko

[10] Phuong-Trinh Pham-Ngoc, Quang-Linh Huynh: ROBUST FACE DETECTION UNDER CHALLENGES OF ROTATION, POSE AND OCCLUSION. Vietnam

[11] Chang Huang, Haizhou AI, Yuan LI and Shihong Lao: Vector Boosting for Rotation Invariant Multi-View Face Detection. Čína

[12] Domovská stránka projektu OpenCV. http://opencv.willowgarage.com, 23.10.2009

[13] http://robotics.csie.ncku.edu.tw/Databases/FaceDetect_PoseEstimate.htm, 17.10.2009

[14] N. Gourier, D. Hall, J. L. Crowley: Estimating Face Orientation from Robust Detection of Salient Facial Features. Cambridge, 2004, International Workshop on Visual Observation of Deictic Gestures

[15] http://robotics.csie.ncku.edu.tw/Databases/OurDatabase/all.zip, 17.10.2009

[16] http://users.cecs.anu.edu.au/~ssanner/Software/Vision/RocLabel.GIF, 12.3.2010

[17] http://www.iib-chemnitz.de/cvwrapper/index.php, 20.4.2010