VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

AUTOMATICKÁ IDENTIFIKACE TVÁŘÍ V REÁLNÝCH PODMÍNKÁCH

AUTOMATIC FACE RECOGNITION IN REAL ENVIRONMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. PAVOL KIČINA
AUTHOR

VEDOUCÍ PRÁCE Ing. JIŘÍ PŘINOSIL, Ph.D.
SUPERVISOR

BRNO 2011

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Pavol Kičina	***ID:***	98528
Ročník:	2	***Akademický rok:***	2010/2011

NÁZEV TÉMATU:

Automatická identifikace tváří v reálných podmínkách

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte moderní metody číslicového zpracování obrazových signálů orientované na detekci a rozpoznání tváří. Na základě získaných teoretických znalostí navrhněte metodu pro identifikaci osob podle tváří s uvážením reálných podmínek při snímání obrazu. Navrženou metodu následně implementujte ve vhodném programovacím jazyku a stanovte podmínky její použitelnosti.

DOPORUČENÁ LITERATURA:

[1] Li, S. Z., Jain, A. K.: Handbook of Face Recognition, Springer, ISBN: 0-387-40595-X, 2004.
[2] Delac, K., Grgic, M.: Face recognition, I-Tech Education and Publishing, ISBN: 978-3-902613-03-5, 2007.

Termín zadání:	7.2.2011	***Termín odevzdání:***	26.5.2011

Vedoucí práce: Ing. Jiří Přinosil, Ph.D.

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

Abstrakt

Táto diplomová práca sa zaoberá identifikáciu tvárií v reálnych podmienkach. Obsahuje prehľad súčasných metód detekcie tvárií pomocou klasifikátorov. Taktiež obsahuje rôzne metódy na detekciu tváre. V druhej časti práce je popis dvoch programov určených na identifikáciu osoby. Prvý program pracuje v reálnom čase v laboratórnych podmienkach, kde pomocou web kamery získava obrázky uživatelovej tváre. Tento program je zameraný na rýchlosť rozpoznania osoby. Druhý program pracuje už na statických obrázkoch, v reálnych podmienkach. Hlavná podstata u tejto metódy je úspešnosť rozpoznania osoby, preto sa kladie dôraz na výpočetnú náročnosť. V programoch som použil naštudované metódy PCA,LDA a kernel PCA (KPCA). Prvý program pracuje len s PCA metódou, ktorá má dobré výsledky vzhľadom na úspešnosť a rýchlosť rozpoznania. V druhom programe sa porovnávajú metódy, kde najlepšie obstála metóda KPCA.

Abstract

This master‘s thesis describes the identification faces in real terms. It includes an overview of current methods of detection faces by the classifiers. It also includes various methods for detecting faces. The second part is a description of two programs designed to identify persons. The first program operates in real time under laboratory conditions, where using web camera acquires images of user's face. This program is designed to speed recognition of persons. The second program has been working on static images, in real terms. The main essence of this method is successful recognition of persons, therefore the emphasis on computational complexity. The programs I used a staged method of PCA, LDA and kernel PCA (KPCA). The first program only works with the PCA method, which has good results with respect to the success and speed of recognition. In the second program to compare methods, which passed the best method for KPCA.

Kľúčové slová

Detekcia tváre, identifikácia tváre, Haarove príznaky, AdaBoost, extrakca príznakov, klasifikátor, trénovacia tvár, testovacia tvár, databáza

Keywords

Face detection, face identification, Haar features, AdaBoost, feature extraction, classifier, training face, test face, database

KIČINA, P. *Automatická identifikace tváří v reálných podmínkách.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. XY s. Vedoucí diplomové práce Ing. Jiří Přinosil, Ph.D.

Prehlásenie

Prehlasujem, že svoj diplomový projekt na tému „Automatická identifikace tváří v reálných podmínkách“ som vypracoval samostatne pod vedením vedúceho diplomového projektu a s použitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedeného semestrálneho projektu ďalej prehlasujem, že v súvislosti s vytvorením tohto projektu som neporušil autorské práva tretích osôb, predovšetkým som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona č. 121/2000 Sb., vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovení § 152 trestného zákona č. 140/1961 Sb.

V Brne dňa ....

podpis autora

Poďakovanie

Ďakujem vedúcemu diplomovej práce Ing. Jiřímu Přinosilovi, Ph.D. za veľmi užitočnú metodickú pomoc a cenné rady pri spracovaní práce.

V Brne dňa ....

podpis autora

OBSAH

Úvod…… 3
1 Biometria 4
1.1 Fyziologické metódy 5
1.2 Behaviorálne metódy 8
2 Rozpoznávanie objektov v obraze 9
2.1 Extrakcia príznakov 9
2.1.1 Extrakcia príznakov pomocou PCA 10
2.1.2 Extrakcia príznakov pomocou LDA 14
2.1.3 Extrakcia príznakov pomocou kernel PCA 15
2.2 Klasifikátory 17
2.2.1 Klasifikátor SVM 18
2.2.2 Neurónové siete 19
2.2.3 Klasifikátor najbližšieho suseda 20
3 Detekcia tváre 23
3.1 Detekcia tváre pomocou farby pokožky 23
3.2 Detekcia tváre pomocou Viola & Jones 24
3.2.1 AdaBoost a Haarove príznaky 24
3.2.2 Integrálny obraz 26
3.2.3 Zapojenie klasifikátorov do kaskády 27
4 Databáza tvárí 28
5 Implementácia 30
5.1 Program na rozpoznanie tváre 30
5.1.1 Implementačný jazyk 31
5.1.2 Vlastný program 32
5.2 Program na porovnanie metód 36
5.2.1 Implementačný jazyk 37
5.2.2 Vlastný program 37
6 Dosiahnuté výsledky 43
Záver 44
Použitá literatúra 46

ZOZNAM SKRATIEK

CCD	Charge-Coupled Device
DNA	Deoxyribonucleic acid
HSV	hue, saturation, and lightness
KPCA	Kernel Principal Component Analzsis
LDA	Linear Discriminant Analysis
PCA	Principal Component Analysis
RGB	Red,green and blue
SVM	Support Vector Machines

ZOZNAM OBRÁZKOV

Obrázok 1-1: Rozpoznávanie podľa a.)geometrie ruky b.)odtlačku prsta c.)geometrie tváre .. 4
Obrázok 1-2: Odtlačok prsta s riadiacimi bodmi 5
Obrázok 1-3: a.) sken sietnici b.) sken dúhovky 6
Obrázok 1-4: Geometria dlane 7
Obrázok 1-5: Rozpoznávanie tváre 8
Obrázok 2-1: Dvojstupňový systém 9
Obrázok 2-2: Efektívne vyjadrenie dát pomocou PCA 10
Obrázok 2-3: Výpočet podpriestoru 11
Obrázok 2-4: Priemerný obrázok z trénovacej sady 11
Obrázok 2-5: Eigenfaces trénovacej sady 13
Obrázok 2-6: a) lineárne neseparovatelné triedy b) pridaním ďalšej dimenzie lineárne separovatelné 16
Obrázok 2-7: Proces rozpoznávania 17
Obrázok 2-8: Separácia dát pomocou SVM 18
Obrázok 2-9: 20
Obrázok 2-10: Princíp hľadania najbližších k susedov 21
Obrázok 2-11: K-D strom hľadania susedov 22
Obrázok 3-1: a) zosnímaný obrázok , b) detekcia kože 23
Obrázok 3-2: Typy Haarových príznakov 25
Obrázok 3-3: Integrálny obraz : a.) hodnota bodu A reprezentuje šedú oblasť , b.) oblasť pri využití rovnice 3 26
Obrázok 3-4: Schématické zobrazenie kaskády 27
Obrázok 4-1: Výber tvárií z databázy ORL Database 28
Obrázok 4-2: Vstupné databázy programu 29
Obrázok 5-1: Vývojový diagram rozpoznania tváre vo videu 32
Obrázok 5-2: Okno so zosnímaným obrázkom 33
Obrázok 5-3: Zdetekovaná tvár 34
Obrázok 5-4: Rozpoznaná tvár osoby 36
Obrázok 5-5: Hlavné okno programu 37
Obrázok 5-6: Pridanie tváre do novo vytvorenej triedy 38
Obrázok 5-7: Výber zobrazených trénovacích tried 39
Obrázok 5-8: Načítanie testovacej tváre 40
Obrázok 5-9: Rozpoznanie tváre 41
Obrázok 5-10: Nájdenie dvoch najbližších obrázkov 41
Obrázok 5-11: Nerozpoznanie testovacej tváre 42

ÚVOD

Detekcia tváre je rozšírená najmä vo fotoaparátoch a kamerách, no dostáva sa do popredia aj v iných sférach, ako napr. letiská alebo kasína. Výhody, ktoré takýto druh identifikácie osôb má, pozostávajú z nenáročnosti na hardvér a jednoduchosti integrácie s existujúcimi systémami.

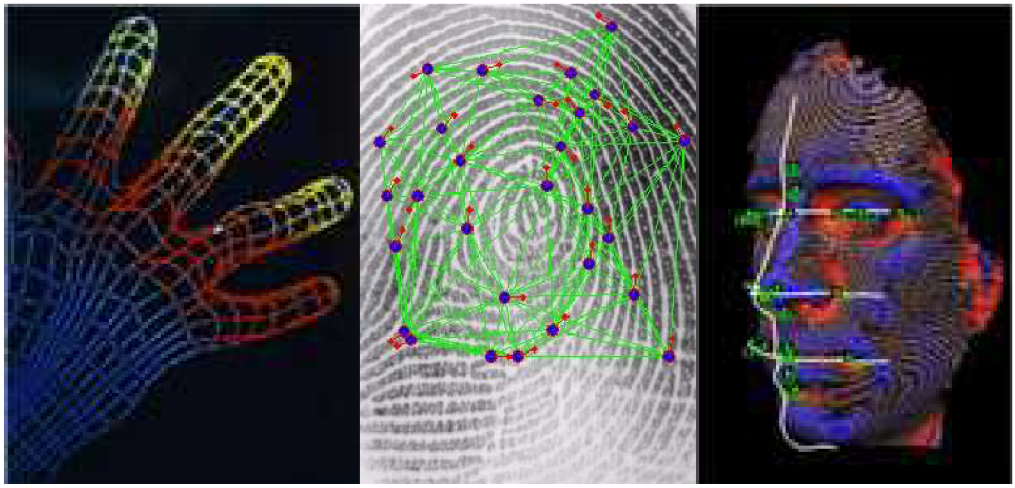
Moja práca sa zaoberá identifikáciou osoby na základe detekcie ľudskej tváre. Práca obsahuje rôzne detekcie tváre, jednotlivé metódy potrebné na rozpoznanie tváre osoby a vlastne naprogramované dva programy, ktoré slúžia pre oboznámenie sa s touto problematikou. Je členená jednotlivo do kapitol, ktoré popisujú potrebné informácie k dosiahnutiu požadovaného cieľa. Popis vlastných programov s vyhodnotením úspešnosti je na konci práce.

Prvým programom je detekcia zosnímanej tváre pomocou web kamery užívateľom a následným rozpoznaním osoby. Táto aplikácia pracuje v reálnom čase, a teda dôraz sa kladie na rýchlosť spracovania údajov a vyhodnotenia. Tu sa pracuje s dynamicky meniacimi sa obrázkami získanými počas snímania kamery. Druhý program si kladie za požiadavky úspešnosť rozpoznanej osoby. Program nepracuje v reálnom čase a nevyužíva kameru na zosnímanie tváre. Pracuje so statickými obrázkami uloženými v databáze.

V dnešnej dobe existuje viacero spôsobov na identifikáciu človeka pomocou tváre. Vytvorenie novej metódy je zdĺhavý a náročný proces, preto sa budem venovať porovnávaním rôznych metód.

1 BIOMETRIA

Pod pojmom biometria rozumieme jednoznačnú identifikáciu na základe jedinečných fyziologických znakov človeka. Biometrický systém je taký systém, kde sa rozpoznávajú vzory, získavajú biometrické dáta od jednotlivcov, extrahujú sa z nich množiny znakov ktoré sa následne porovnávajú s vytvorenou databázou príznakov. Ide teda o určitú formu elektronického kľúča, kde kľúčom je samotný človek.



a.) b.) c.)

Obrázok 0-1: Rozpoznávanie podľa

a.) geometrie ruky b.) odtlačku prsta c.) geometrie tváre

Biometria využíva jedinečné a opakovateľne identifikovateľné vlastnosti človeka (odtlačok prsta, črty tváre, DNA, tvar očného pozadia), čo znázorňuje obr.1 [1]. Jednotlivé biometrické znaky človeka sa prevedú do elektronickej podoby a je s nimi možné narábať, ako s jedinečným kódom. Tieto údaje sú prenositeľné a overiteľné a je ich možné zakomponovať do identifikačných dokladov a pod. Základné biometrické údaje sa používali už skôr (výška, farba očí) napríklad v pasoch. Rozvoj informačných technológií umožnil sledovať aj oveľa zložitejšie štruktúry a tvary.

Na spoľahlivú identifikáciu sa v biometrii využívajú čítačky. Biometrické čítačky sa používajú všade tam, kde potrebujeme zabezpečiť vysokú spoľahlivosť, transparentnosť, bezpečnosť a samozrejme jednoduchosť a komfort. Teda môžu sa využiť v rôznych dochádzkových systémoch, na kontrolu originality podpisu, zabezpečenie ochrany a pod [2]. Ich markantnou výhodou je, že nepotrebujú žiadnu réžiu spojenú s prevádzkou

identifikačných predmetov. Na identifikáciu nepotrebujeme žiadne predmety, preto nemusíme nič nosiť pri sebe a odpadá riziko straty alebo zabudnutia.

1.1 FYZIOLOGICKÉ METÓDY

Odtlačok prsta

Identifikácia osôb na základe odtlačkov prstov je jedna z najpoužívanejších a najznámejších biometrických metód. Identifikácia vo svojom funkčnom princípe pozostáva z dvoch samostatných fáz. Prvým krokom je získanie obrazca odtlačku, čo sa uskutočňuje snímaním CCD čipom (Charge-Coupled Device chip) alebo lineárnym kapacitným senzorom. Takto získaný digitalizovaný obrazec je potrebné následne vyhodnotiť, a jednoznačne identifikovať. Treba brať ohľad na mnoho faktorov, kam zaraďujeme drobné poranenia, meniacu sa pozíciu snímaného prstu na snímači a mnoho ďalších vplyvov.

Obrázok 0-2: Odtlačok prsta s riadiacimi bodmi [4].

Softvérové algoritmy sú v súčasnom štádiu vývoja pomerne úspešné, pričom ich spoľahlivosť sa v súčasnosti pohybuje na úrovni 99,93 percenta.

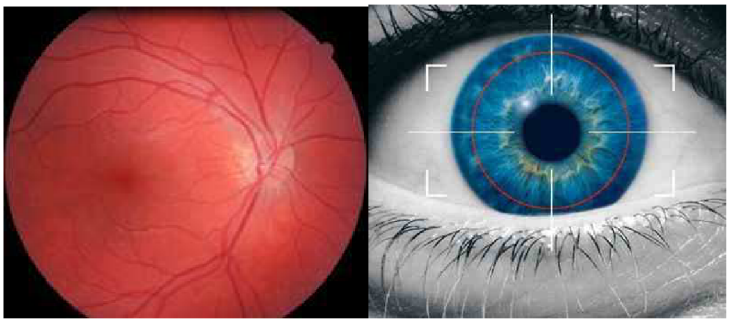
V praxi sa využívajú tri princípy: optický (založený na zmenách odrazivosti skla v miestach, kde sa dotkne línia odtlačku prsta. Snímanie je uskutočnené CCD čipom), kapacitný (meria kapacitu jednotlivých častí odtlačkov, pomocou lineárneho kapacitného senzora) a ultrazvukový (používa ultrazvuk na zistenie povrchu prsta). Využitie odtlačkov prstov je

pomerne jednoduché a nenáročné, o čom svedčí aj nasadenie tejto technológie pri identifikácii používateľa v informačných systémoch [3]. V súčasnosti je na trhu množstvo komponentov (špecializované snímače, snímače implementované na myši, či klávesnice) i komplexných riešení, ktorých nasadenie je naozaj jednoduché. Rozpoznanie je uskutočnené od 0,2 do 1 sekundy.

Oko

Sken očnej dúhovky sa vykonáva zväčša prostredníctvom čiernobielej kamery. Princíp snímania je síce pomerne jendoduchý a nekladie na identifikovanú osobu zvláštne nároky, i tak je však táto procedúra o niečo zložitejšia ako v prípade odtlačkov. S jej nasadením sa však experimentovalo najmä v bankomatoch. Čas rozpoznania býva väčšinou okolo 2 sekúnd.

Sken očnej sietnice je v porovnaní s predchádzajúcou metódou podstatne náročnejší, a to tak na technické vybavenie, ako aj na identifikovanú osobu. Miesto kamery sa totiž v tomto prípade používa laser, ktorý je namierený priamo do oka. Praktické nasadenie tejto metódy je preto veľmi obmedzené. Jediná doposiaľ potvrdená implementácia je v amerických väzeniach, kde sa využíva na identifikáciu väzňov. Táto metóda vyžaduje 1,5 až 4 sekundy na rozpoznanie.[2]



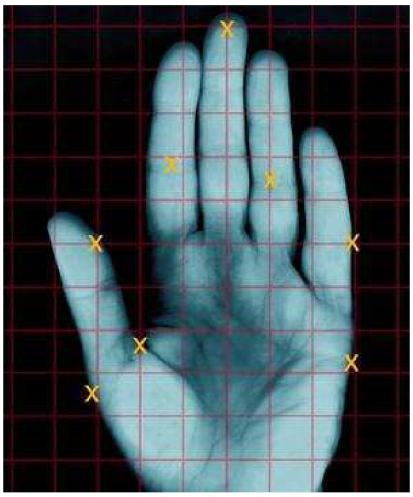
a.) **b.)**

Obrázok 0-3: a.) sken sietnice [5] b.) sken dúhovky [6]

Geometria ruky a prstov

Každý človek má odlišnú geometriu ruky a prstov od inej osoby. Preto sa vyvinula metóda, ktorá je založená na tomto fakte. Metódy, ktoré sa používajú môžme rozdeliť na mechanické a detekciu hrán v obraze. Obe tieto metódy produkujú odhady určitého

kľúčového merania ruky (dĺžka prstov a palca, ich šírky a pod.). Obraz sa zvyčajne sníma CCD kamerou za pomoci zrkadiel, ktoré zabezpečujú 3D obraz. Ako zdroj svetla je použitý IR laser. Pre získanie relevantného vzoru, ktorý sa potom používa pre porovnanie, je vykonaných 90 meraní.[1]



Obrázok 0-4: Geometria dlane [3]

Obraz tváre

Táto metóda je jedna z menej náročných biometrických metód, zakladá sa na princípe identifikácie osôb prostredníctvom rozprávania hlavných čŕt tváre. Základom systému je farebná kamera, ktorá by mala mať čo najvyššie rozlíšenie (čím viac detailov dokáže zachytiť, tým je identifikácia spoľahlivejšia). Na identifikáciu sa v tomto prípade používajú viaceré „parametre“ tváre. Základným je vzájomná poloha očí, nosa, úst a uší. V závislosti od komplikovanosti systému sa využívajú niektoré alebo aj všetky spomenuté prvky. V praxi sa toto riešenie využíva na rozpoznávanie známych tvárí v dave (letiská, verejné priestranstvá, úrady a pod). Okrem identifikácie teroristov majú toto riešenie nasadené aj niektoré letecké spoločnosti na identifikáciu VIP zákazníkov.

Obrázok 0-5: Rozpoznávanie tváre [6]

Analýza DNA

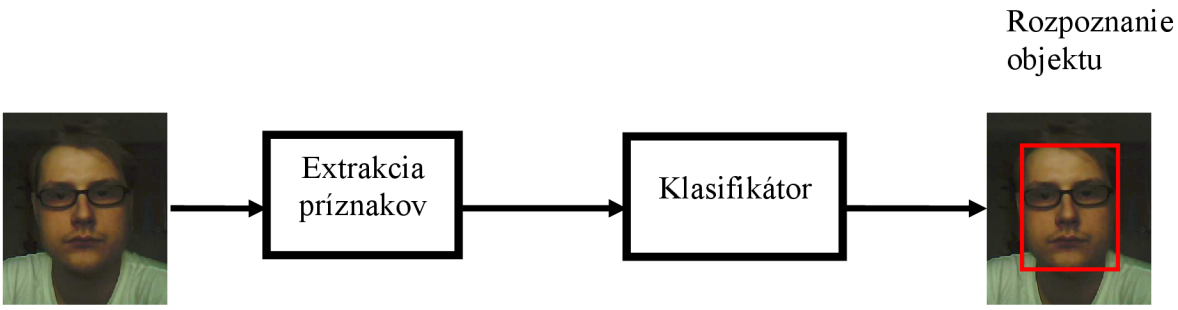
Analýza DNA sa odlišuje tým, že ju nemožno vykonávať v reálnom čase a spočíva v porovnávaní vzoriek, t. j. nespočíva vo vytvorení šablóny.[2] Táto metóda sa zatiaľ najviac využíva v kriminalistike na identifikáciu páchateľov.

1.2 BEHAVIORÁLNE METÓDY

Medzi najpoužívanejšie behaviorálne metódy patrí podpis, skúmanie hlasu. Z pohľadu miery zabezpečenia sú najslabšími alternatívami, preto sa zvyčajne používa aj dodatočné zabezpečenie vo forme statických hesiel a pod. Hlasová verifikácia využíva charakteristiku hlasu, basové a výškové tóny, vibrácie hlasu, hrdelné a nosné tóny. Zvuk je pri snímaní analyzovaný z viacerých hľadísk – frekvenčné spektrum, rytmus, intonácia. V prípade hlasovej indispozície však nie je možné osobu identifikovať.[1]

2 ROZPOZNÁVANIE OBJEKTOV V OBRAZE

V súčasnosti sa pre rozpoznávanie objektov alebo vzorov používa viacero spôsobov – systémov. Medzi základné patria jednostupňový a dvojstupňový systém. Princíp jednostupňového systému, alebo taktiež aj priamej metódy, spočíva v tom, že vstupný obrázok vstupuje hneť do klasifikátora, ktorý identifikuje objekty na základe zvoleného algoritmu. Avšak nevýhodou tohto princípu je pomalosť pri použití vstupných dát veľkých rozmerov. Preto sa zaviedol dvojstupňový systém, ktorého schému môžme vidieť na obr.2-1. Odstránili sa tým nedostatky priamej metódy.



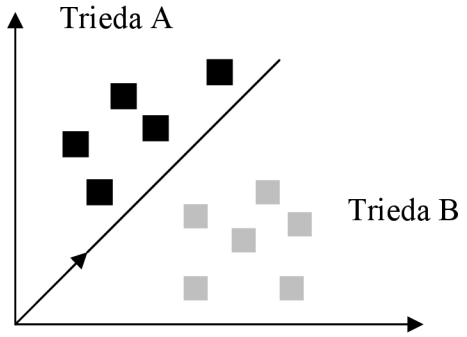
Obrázok 2-1: Dvojstupňový systém

2.1 EXTRAKCIA PRÍZNAKOV

Dvojstupňový system na rozdiel od jednostuňového používa extrakciu príznakov. Tá slúži na zníženie dimenzie vstupných dát (počet pixelov obrázku). Znížením tejto dimenzie sa dosiahne rýchlejší výpočet len tých najdôležitejších dát. Medzi extrakciu patria viaceré metódy, medzi nimi aj PCA (Principal Component Analysis) , LDA (Linear Discriminant Analysis) a taktiež kernelovské metódy KPCA (kernel PCA) a KLDA (kernel LDA). Ja sa budem zaoberať len PCA, LDA a KPCA.

2.1.1 Extrakcia príznakov pomocou PCA

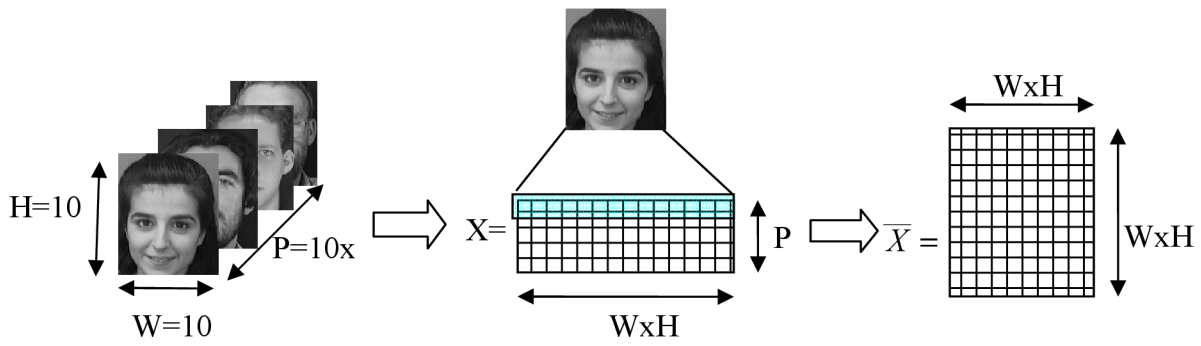
Táto metóda predstavuje tzv. Dvojstupňový systém rozpoznávania. Trénovacie a testovacie dáta premieta (transformuje) do podpriestoru vytvoreného pomocou vlastných vektorov kovariančnej matice na základe trénovacích dát. Obr.2-2 zobrazuje zjednodušený 2D prípad rozdelenia dát pomocou PCA. A teda metóda PCA hľadá najefektívnejšie vyjadrenie dát, pričom maximalizuje rozptyl dát, a to bez ohľadu na triedy dát [8]. V prípade rozpoznávania tvárí to znamená, že neprihliada na to, ktoré dáta patria ktorému subjektu.



Obrázok 2-2: Efektívne vyjadrenie dát pomocou PCA

Aby sme mohli lepšie pochopiť celý proces rozpoznávania tváre pomocou PCA, uvediem najprv hlavné pojmy a následne rozoberiem problematiku.

Majme sadu tvárí o počte P (taktiež trénovacia sada), kde všetky snímky majú rovnaké rozmery WxH. Z týchto obrázkov zostrojíme maticu o počte riadkov P a o počte stĺpcov rovném číslu $n = WxH$. Obrázky sú si viac-menej podobné, a preto existuje medzi nimi nenulová korelácia. Ako bolo spomenuté, PCA hľadá vektory, ktoré najlepšie popisujú distribúciu obrázku tvárí v celom priestore obrázkov (trénovacej sade). Tieto nájdené vektory dimenzie n definujú podpriestor, ktorý sa taktiež nazýva eigenspace. Tieto vektory sú vlastnými vektormi kovariančnej matice, ktoré korešpondujú k pôvodným obrázkom a pretože sú podobné obrázkom tvárí, nazývajú sa eigenfaces. Na obr 2-3 môžme vidieť výpočet podpriestoru.



Obrázok 2-3: Výpočet podpriestoru

Teraz sa môžme zaoberať samotným procesom rozpoznávania. Ak máme sadu tvárí (obrázkov) $x_1, x_2, \ldots x_P$, kde každý obrázok je vektor. Vypočítame priemerný obrázok tejto sady podľa vzťahu (2.1) a takto vypočítaný obrázok môžme vidieť na obr 2-4.

$$M = \frac{1}{P} \cdot \sum_{i=1}^{P} x_i \tag{2.1}$$

Obrázok 2-4: Priemerný obrázok z trénovacej sady

Následne sa od každého obrázku odpočíta priemerný obrázok podľa vzťahu:

$$\overline{x}_i = x_i - M \tag{2.2}$$

Matica X obsahuje usporiadané vektory, kde každý riadok matice popisuje jeden z vektorov x_i. Táto Matica je vstupom do analýzy hlavných komponentov, ktorá hľadá množinu vektorov, ktoré najlepšie popisujú distribúciu vstupných dát. Hľadané vektory sú vlastné vektory kovariančnej matice C', ktorú získame nasledovne:

$$C' = \overline{X}.\overline{X^T} \tag{2.3}$$

Predpokladom pre zostrojenie kovariančnej matice je, že $P >> n$. To znamená, že počet obrázkov (tvárí) v sade je omnoho menší ako počet bodov v jednom obrázku a teda aj v jednom vektore. V opačnom prípade sa musí zmenšiť rozmer obrázkov v sade.

V ďalšom kroku sa vypočítajú (2.4) vlastné vektory V' (eigenvectors) usporiadané v riadkoch matice a vlastné číslá λ (eigenvalues) kovariančnej matice C'. Tento krok nieje jednoduchý, avšak v programe MATLAB na to slúži funkcia eig(), ktorá nájde z dát eigenvalues a eigenvectors a uloží do premenej.

$$C'.V' = V'.\lambda' \tag{2.4}$$

Matica V' obsahujúca v riadkoch vlastné vektory má rovnaký rozmer ako kovariančná matica C', teda PxP. K vlastným vektorom v matice V kovariančnej matice sa dopracujeme následovně :

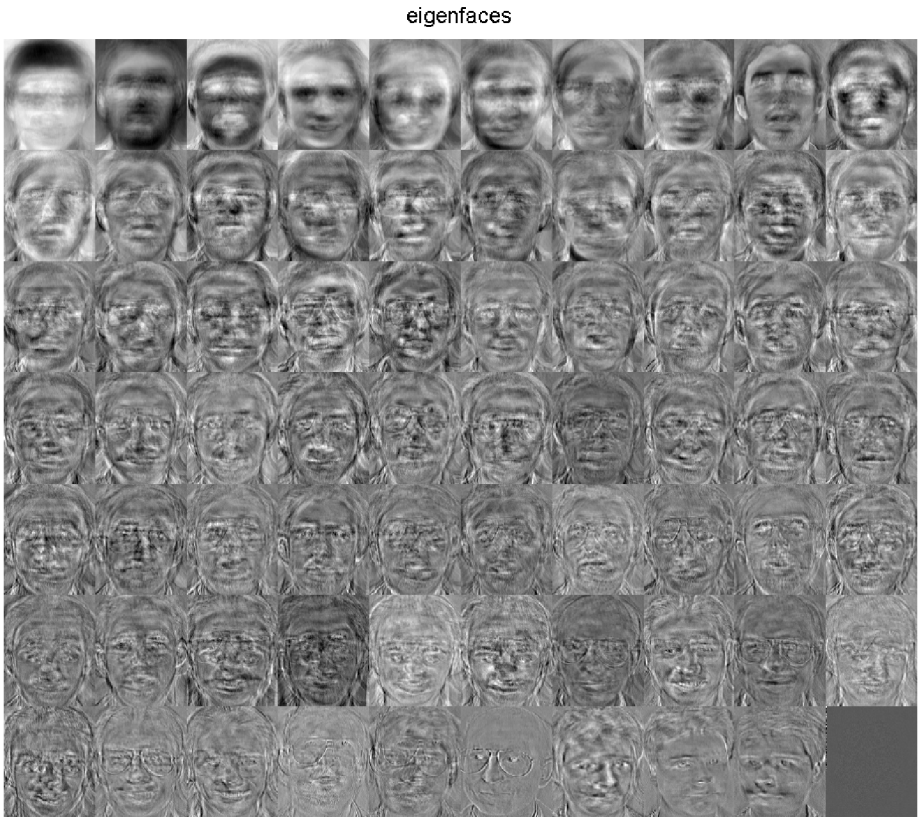
$$V = X.V' \tag{2.5}$$

Dostaneme maticu V, ktorá obsahuje v P riadkoch vlastné vektory o dĺžke n, ktoré tvoria bázu priestoru eigenspace. Vypočítané hodnoty vektorov sa zotriedia od najväčšieho po najmenší a každý vektor v_i z matice V prevedieme výpočtom podľa (2.6), aby sme normalizovali hodnotu na jednotkovú veľkosť.

$$v_i = \frac{v_i}{\|v_i\|} \tag{2.6}$$

Po tomto kroku už máme určenú bázu priestoru eigenspace, ktorá je tvorená maticou *V*. Následne sa transformujú trénovacie obrázky umiestnené v matici *X* do eigenspace. Predpokladajme, že jednotlivé transformované eigenfaces (obr. 2-5) vkladáme po riadkoch do matice Z. Transformácia je nasledovná:

$$z_i = x_j . V^T \tag{2.7}$$



Obrázok 2-5: Eigenfaces trénovacej sady

V tomto okamžiku sme sa dopracovali do bodu kde môžme začať s rozpoznávaním testovacieho obrázku neznámej tváre *u*. Tento obrázok prevedieme do eigenspace a to dvomi krokmi:

- Vycentrujeme obrázok : $\bar{u} = u - M$
- Vypočítame eigenface: $\tilde{u} = \bar{u} - V^T$

Teraz rozhodneme ktorému obrázku z trénovacej sady najviac odpovedajú vypočítané hodnoty. Určíme vzdialenosť trénovacích eigenfaces (obr 2-5) z_i
k eigenface $\tilde{u}$. Z nájdenej minimálnej vzdialenosti nakoniec určíme odpovedajúci tvar trénovacej sady. K určeniu vzdialenosti sa používajú rôzne metriky. Ja som v mojej práci použil Euklidovu metriku pre výpočet minimálnej vzdialenosti, ktorá má tvar :

$$d_E(x_i \tilde{\tilde{u}}) = \sqrt{\sum_{j=0}^{P-1} (\tilde{\tilde{u}}[j] - x_i[j])^2} \tag{2.8}$$

V rámci rozpoznávania je podstatný prah úspešnosti rozpoznávania, ktorý je chápaný ako medzná hodnota vzdialenosti obrázku po rozpoznaní od obrázku z trénovacej sady. Ak je hodnota väčšia ako prah, testovací obrázok nie je správne rozpoznaný. Na záver tejto metódy by som ju zhrnul do troch základných krokov, v ktorých pracuje :

1. najprv sa vypočíta podpriestor, do ktorého sa budú premietať všetky obrázky

2. trénovací obrázok sa prevedie do tohto podpriestoru a vypočítajú sa hodnoty

3. testovací obrázok sa prevedie do toho istého podpriestoru a porovná vzdialenosť vypočítaných hodnôt od hodnôt testovacieho obrázku

2.1.2 Extrakcia príznakov pomocou LDA

Tento princíp je podobný ako PCA, avšak je rozdielny v transformácii dát. LDA minimalizuje vnútrotriedny a zároveň maximalizuje medzitriedny rozptyl dát [8]. V problematike rozpoznávania tvárí to znamená, že transformácia sa snaží rozptýliť čo najviac dát rôznych subjektov a zároveň minimalizuje vzdialenosť dát patriacich tomu istému subjektu.

Celý proces rozpoznávania je založený na výpočte eigenvalues a eigenfaces ako u PCA, avšak s mizernou modifikáciou. Matica rozptylu medzi triedami je definovaná ako :

$$S_B = \sum_{i=1}^{C} N_i(\mu_i - \mu) \tag{2.9}$$

A matica rozptylu vnútri tried:

$$Sw = \sum_{i=1}^{C} \sum_{x_k \in P_i} (x_k - \mu_i).(x_k - \mu_i)^T \tag{2.10}$$

Kde μ_i je stredný obraz triedy X_i a μ je stredný obraz celej trénovacej sady. N_i je počet vzorkov (tvárí) v triede P_i. Ak je matica S_B regulérna (švorcová matica kde determinant je rôzny od nuly a stĺpce a riadky sú lineárne nezávislé) je optimálna projekcia W_{opt} volená ak matica ortonormálnych vektorov, ktorá maximalizuje pomer determinantov matíc S_B a S_w.

$$W_{opt} = \arg\max \frac{W^T S_B W}{W^T S_w W} = [w_1, w_2, \ldots\ldots w_m] \tag{2.11}$$

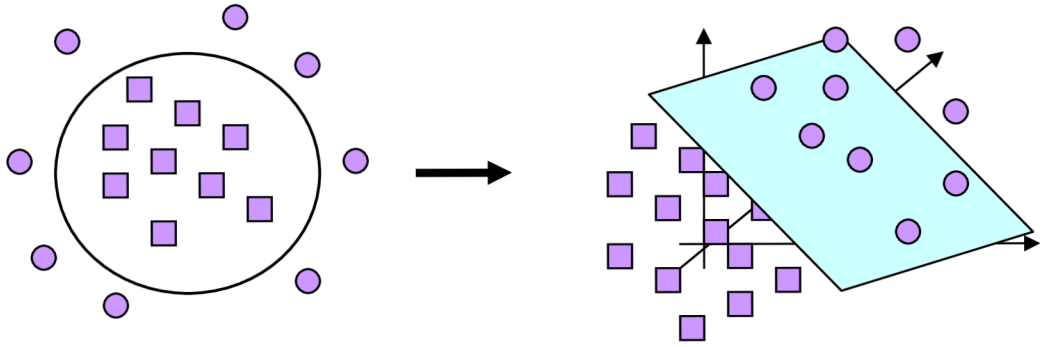
kde w_i je množina vlastných vektorov (eigenvectors) S_B a S_w. Matica W obsahuje vypočítané vlastné vektory z trénovacej sady pomocou PCA metódy. Z matíc $W^T S_B W$ a $W^T S_w W$ sa vypočítajú zobecnené vlastné vektory, ktoré sa zoradia od najväčšej po najmenšiu hodnotu.

Projekcia dát (trénovacích a testovacích) sa prevedie do tejto množiny vektorov a po klasifikácii sa vypočíta najmenšia vzdialenosť ako pri metóde PCA.

2.1.3 Extrakcia príznakov pomocou kernel PCA

Hlavnou myšlienkou kernelských metód je transformácia dát do vysokorozmerového priestoru. Využíva sa vtedy, ak triedy sa nedajú lineárne separovať. Vhodnou transformáciou môžme aj lineárne separovateľný problém rozdeliť lineárnou hyperrovinou. Na obr. 2-3 je znázornený problém tejto metódy, kde a) nedá sa použiť lineárny klasifikátor. Pridaním ďalšej dimenzie sa jednotlivé triedy dajú lineárne separovať, viď obr 2-3 b).

Postup KPCA je podobný ako u PCA. Transformáciu do vysokorozmerového priestoru označíme Φ. Po prevedení dát (sady tvárí) do vyššej dimenzie nám vznikne novo vytvorená sada vektorov $\Phi(x_1)$…. $\Phi(x_i)$. Táto dimenzia zachováva vzťah vzdialeností medzi vzorkami v pôvodnom a novom priestore.



Obrázok 2-6: a) lineárne neseparovateľné triedy b) pridaním ďalšej dimenzie lineárne separovateľné

Následne sa vytvorí kovariančná matica C :

$$C = \frac{1}{l}\sum_{j=1}^{l}\Phi(x_j).\Phi(x_j)^T \tag{2.9}$$

Predpoklad, že počet trénovacích obrázkov je menší ako počet bodov v jednom obrázku musí byť zachovaný aj u tejto metódy.

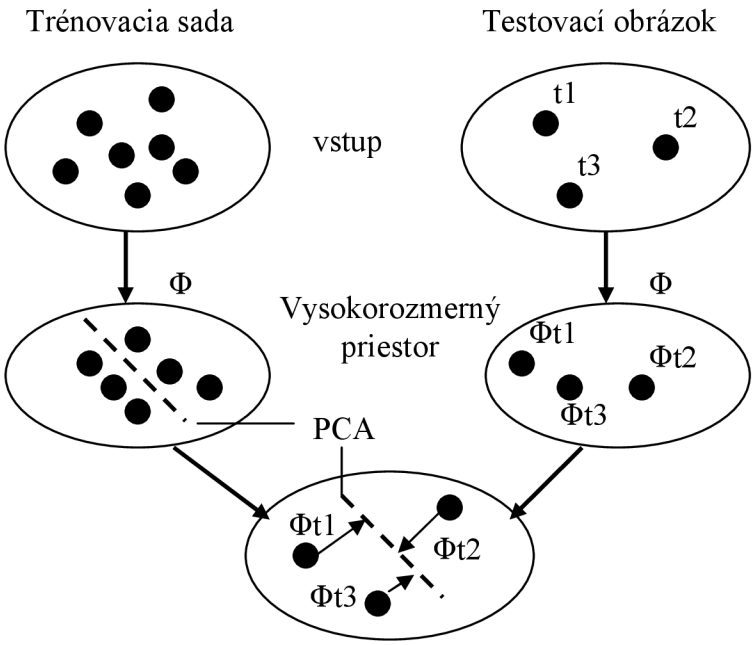
V nasledujúcom kroku sa vypočítajú eigenvalues $\lambda > 0$ a eigenfaces V podľa vzťahu (2.4), no s miernou modifikáciou:

$$(\Phi(x_k).C.V) = \lambda(\Phi(x_k).V) \tag{2.10}$$

Dôležitým faktorom u tejto metódy je výpočet kernel matice K. Použije sa vzorec :

$$K_{ij} = \Phi(x_j)^T.\Phi(x_i) \tag{2.11}$$

Lepšie predstavenie procesu rozpoznávania nám zobrazuje obr 2-7.



Obrázok 2-7: Proces rozpoznávania

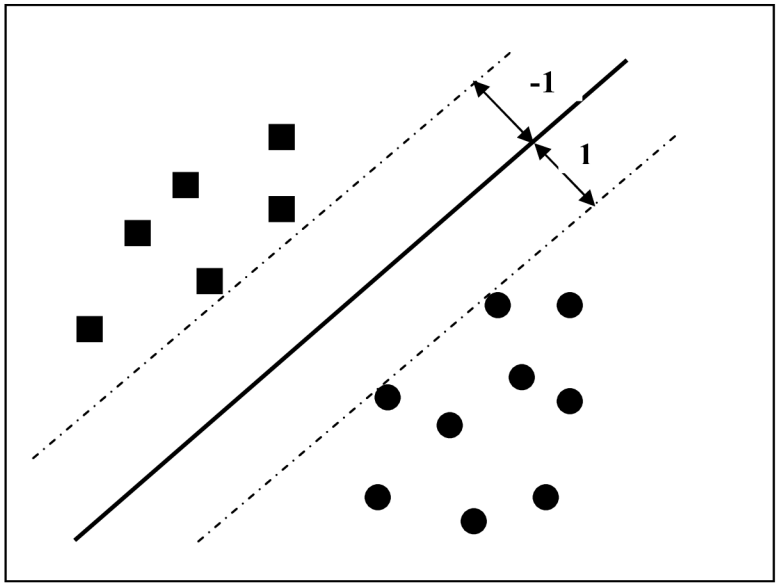
2.2 KLASIFIKÁTORY

Pod pojmom klasifikátory môžme rozumieť funkciu alebo modul, ktorý dokáže zhodnotiť s akou pravdepodobnosťou sa na danom mieste nachádza skúmaný objekt [7]. Klasifikátory môžme rozdeliť na :

- binárne - vracajú dve hodnoty, „áno“ alebo „nie“ v závislosti na výsledku klasifikátora
- viachodnotové – vracia hodnotu v určitom rozmedzí, ktorá reprezentuje pravdepodobnosť s akou sa detekovaný objekt nachádza v obraze

2.2.1 Klasifikátor SVM

Táto metóda je postavená ako lineárny klasifikátor. Neskôr bola modifikovaná pomocou kernelových metód, takže umožňuje aj klasifikáciu lineárne neseparovateľných problémov. SVM (Support Vector Machines) rozdeľuje p-rozmerné dáta pomocou p-1 rozmernej separačnej hyperroviny [8]. Na obr 2-8 je znázornený model, ktorý predstavuje najjednoduchší problém, kedy je možné lineárne separovať vzorky patriace 2 rôznym triedam.



Obrázok 2-8: Separácia dát pomocou SVM

SVM rozdeľuje p-rozmerné dáta pomocou p-1 rozmernej hyperroviny a to tak, že maximalizuje vzdialenosť jednotlivých tried dát. Vzdialenosť rozdeľujúcej hyperroviny (na Obr. 2-8 znázornené celou čiarou) od najbližších výskytov dát jednotlivých tried má byť čo najväčšia. Za týmto účelom sú najbližšími výskytmi dát vedené tzv. Support Vektory (na Obr. 2-8 znázornené prerušovanými čiarami). SVM hľadá takú separačnú hyperrovinu, ktorej vzdialenosť od oboch Support Vektorov je maximálna, a tým maximalizuje rozptyl [8].

2.2.2 Neurónové siete

Do rady klasifikátorov taktiež spadajú neurónové siete. Je to vlastne paralerný systém, kde sa modelujú vzťahy medzi vstupmi a výstupmi, alebo sa porovnáva pomocou vzoru. Počet vstupov a výstupov musí byť minimálne 1. Táto metóda sa využíva najmä vtedy, ak je potrebné riešiť zložité matematické úlohy. Takáto sieť pozostáva z jednotlivých častí, ktoré sa nazývajú neuróny a uzly. Neurón prijíma N vstupov a vracia M výstupov [7]. Na obrázku 2-9a môžme vidieť 3 typy neurónov:

- vstupné – neuróny, ktorých vstupmi sú signály z prostredia
- skryté – neuróny, ktoré sú vstupmi aj výstupmi spojené s inými neurónmi
- výstupné – neuróny, ktorých výstup vedie do prostredia

Podľa [7] sa neurón spracovávací informáciu riadi nasledujúcim pravidlom:

$$o_i^{k+1} = f\left(\sum_{j=1}^{N} w_{ij}^k \times o_j^k - \Theta_i^{k+1}\right), \qquad (2.12)$$

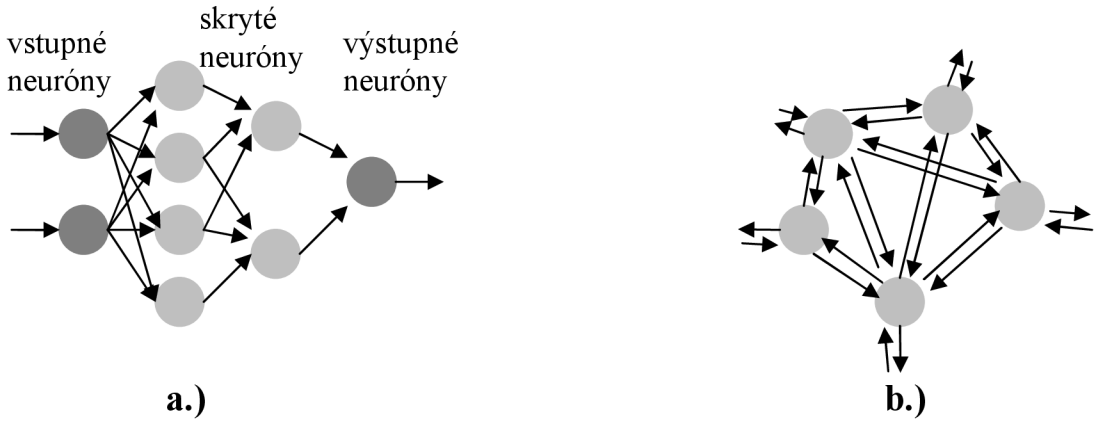
kde :

$0 < i \leq M$	výstup
$0 < j \leq M$	vstup
o_i^{k+1}	vstupná hodnota i-tého neurónu k + 1 vrstvy
k	index vrstvy
Θ_i^{k+1}	prah exitácie i-tého neurónu k + 1 vrstvy
w_{ij}^k	váha spojenia medzi j-tým neurónom k-tej vrstvy a i-tým neurónom k + 1-tej vrstvy
f()	ľubovoľná monotónna funkcia

Neurónovú sieť môžme rozdeliť do dvoch vrstiev. V závislosti na toku dát sa delí na :

- dopredné – signál sa šíri od vstupných neurónov cez skryté neuróny na výstup. Príklad obr. 2-9a.

- rekurentné – signál sa môže pohybovať aj smerom od výstupov ku skrytým častiam alebo dokonca až výstupom. Príklad obr. 2-9b.



Obrázok 2-9:

a.) dopredná neurónová sieť. Obsahuje 4 vrstvy b.) rekurentná neurónová sieť obsahujúca neuróny, ktoré sú vstupné a výstupné zároveň

Hlavnou vlastnosťou neurónových sietí je schopnosť učiť sa. Učenie je dôležitá fáza, ktorá sa nachádza pri každej tvorbe siete. V tejto fáze sa upravujú jednotlivé váhy spojené neurónmi tak, aby dosiahli požadované výsledky, alebo výsledky ktoré majú najbližšie k tejto hodnote [7].

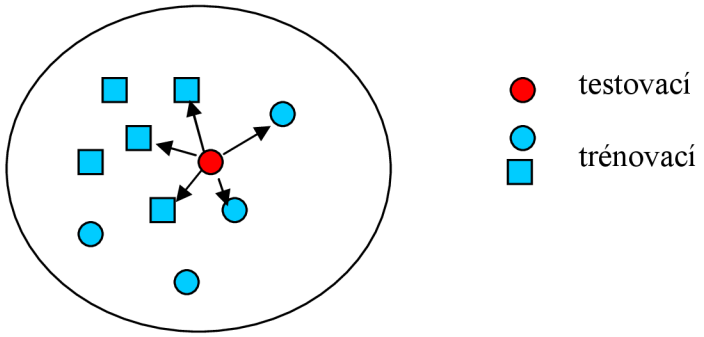
2.2.3 Klasifikátor najbližšieho suseda

Táto metóda sa taktiež označuje aj ako 1NN klasifikácia (z angl. slov Nearest Neighbor). Na rozdiel od predchádzajúcej metódy sa neučí a len rozhoduje. Ako vstupné dáta sa použije trénovacia sada a porovnáva sa s testovacím obrázkom. Vyberie sa najlepšia miera podobnosti trénovacích a testovacích dát. Na rozhodovanie sa používajú rôzne funkcie :

- Euklidovská vzdialenosť $$dis(x_i\ x_j) = \sqrt{\sum_{k=1}^{K}\left(x_{ik} - x_{jk}\right)^2} \qquad (2.13)$$

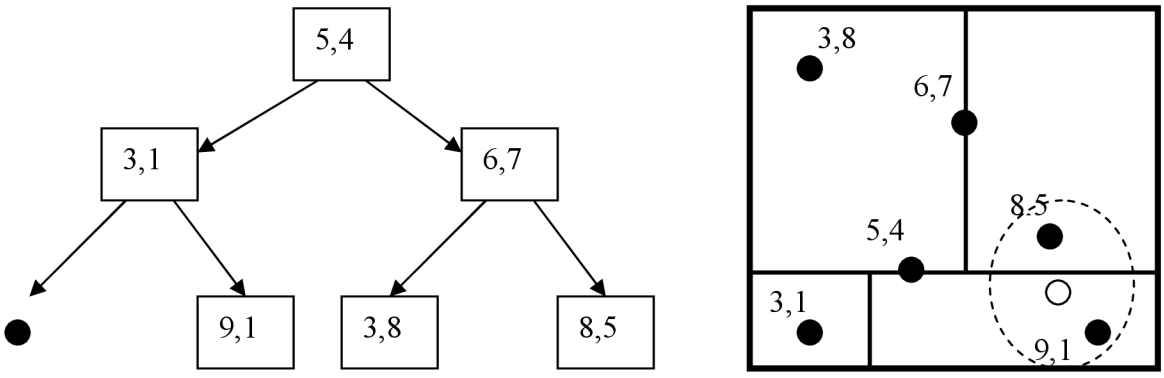
- Manhattanská vzdialenosť $$dis(x_i\ x_j) = \left|x_{jk} - x_{ik}\right| \qquad (2.14)$$

Môže nastať prípad, že sa vyskytne viacero hodnôt s rovnakou minimálnou vzdialenosťou. Postupuje sa tak, že sa vyberie jeden. Tento princíp porovnáva testovací obrázok s každou trénovacou inštanciou, čo má za následok pomalosť systému [14]. Zovšeobecnením dotazu na najbližšieho suseda môžme dostať hľadanie *k* najbližších susedov. Ide o rovnaký princíp ako u hľadania najbližšieho suseda s rozdielom, že sa hľadá ľubovoľný počet najbližších objektov. Obrázok 2-11 predstavuje algoritmus K-D tromu hľadania najbližšieho suseda a obr. 2-10 znázorňuje priestorové určenie jednotlivých hodnôt hľadaním *k* najbližších susedov.



Obrázok 2-10: Princíp hľadania najbližších k susedov[13]

K-D stromy rozdeľujú priestor podľa bodov (hodnôt). Vyžaduje malú pamäťovú náročnosť. Požiadavka na prehľadanie K-D stromu v R^d má časovú náročnosť $O(d.n^{(1-1/d)}+k)$, kde *k* je počet nájdených bodov.



Obrázok 2-11: K-D strom hľadania susedov [14]

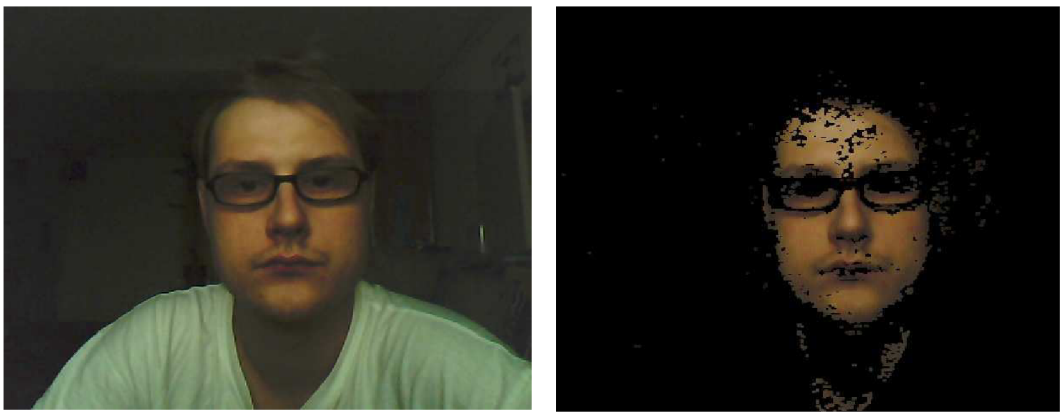
3 DETEKCIA TVÁRE

Pre ľudí je úplne prirodzené vedieť rozpoznávať rôzne objekty, rozlíšiť tvár od iných objektov a tak isto identifikovať osobu podľa tváre. Vo svete techniky tomu tak nie je. Proces rozpoznávania objektov resp. identifikácie osoby podľa tváre je komplikovaná záležitosť. Táto oblasť techniky sa začala dynamicky rozvíjať iba nedávno, vďaka vzrastajúcemu výkonu počítačov a rozvoju digitálnych videokamier a fotoaparátov.

Preto sa vyvinulo niekoľko metód, ktoré dokážu rozpoznať, či sa jedná o ľudskú tvár. Môžme si ich rozdeliť na dva základné smery. Prvý je založený na obrazovom základe. Využíva techniky ako neurónové siete, lineárne podpriestorové metódy s učením pozadia alebo štatistické prístupy. Druhý smer sa zameriava na vlastnosti obrazu. Sem patria informácie o okrajoch, o farbe pokožky, o pohybe, merania symetrie a meniteľné šablóny.

3.1 DETEKCIA TVÁRE POMOCOU FARBY POKOŽKY

Každá farba ľudskej kože je výnimočná aj napriek množstvu jednotlivých variácií. Na pokožku môže dopadať svetlo pod rôznymi uhlami, snímacie zariadenie môže byť nastavené rôznymi spôsobmi alebo samotná pigmentácia pokožky človeka môže byť rozdielna. Preto sa väčšinou spájajú rôzne metódy, pre vznik invariantného systému detekcie. Napr. na zdetekovanie tváre môžme použiť detekciu farby kože ako predspracovanie obrázku na špecifikovanie potencionálnych objektov. Detekciu kože môžme vidieť na obr. 3-1.



a.) **b.)**

Obrázok 3-1: a) zosnímaný obrázok , b) detekcia kože

Pravdepodobný model farby kože *p(farba|tvár)* je odvodený zo vzoriek farby kože. Toto sa môže previesť v HSV alebo RGB priestore farieb. Pixel znázorňuje farbu kože, ak pravdepodobnosť *p(H|tvár)* je väčšia ako prahová hodnota 0,3 a hodnoty S a V sú medzi určitými hornými a spodnými hranicami.
Takto môžme získať mapu pokožky v obraze[8]. Táto metóda je výpočetne jednoduchá. Avšak táto metóda má medzery v tom, že v obraze sa môžu nachádzať aj predmety farby ľudskej kože. Môžme to odstrániť doplnkovou metódou (detekcia a analýza hrán), avšak výpočetná náročnosť nám stúpne.

3.2 DETEKCIA TVÁRE POMOCOU VIOLA & JONES

Ako prví prišli s touto myšlienkou P.Viola a M. Jones v roku 2001, ktorú prezentujú v dokumente [9]. Ich myšlienka pozostávala z 3 základných vecí, vďaka ktorým je detekcia efektívna v rýchlosti a vykazuje nízku chybovosť.
Sú to:

- AdaBoost využívajúci Haarove príznaky
- Integrálny obraz
- Zapojenie klasifikátorov do kaskády

3.2.1 AdaBoost a Haarove príznaky

AdaBoost

AdaBoost je skratka pre slovo adative boosting. Tento algoritmus výrazne znižuje chybu učiaceho algoritmu, ktorý dáva výsledky lepšie ako náhodné tipovanie. AdaBoost je adaptabilný z dôvodu využívania viacerých slabých klasifikátorov a pri učení sa využíva množina, predtým zle zaradených objektov, na natrénovanie nasledujúcich klasifikátorov. Výsledným výtvorom je jeden robustný klasifikátor[7].

Algoritmus ukladá váhy jednotlivých klasifikátorov na základe zle klasifikovaných objektov. Ak nastane prípad, že sa práve zle sklasifikuje objekt, je váha klasifikátora

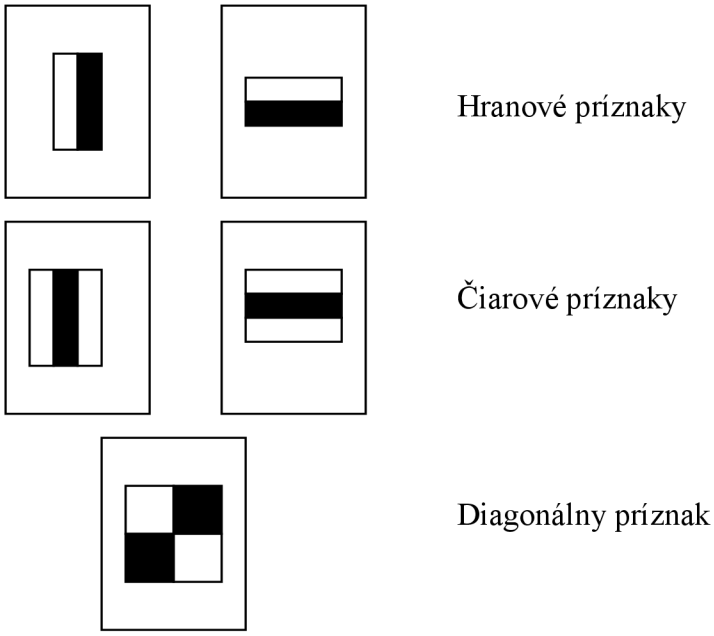
posilnená, čím sa dosiahne, že v nasledujúcom trénovaní algoritmus dokáže opraviť svoje rozhodnutie. Samotný algoritmus je však citlivý na všetky objekty. A teda sa snaží zaradiť každý objekt do správnej množiny. Avšak táto vlastnosť nie je vhodná pre trénovanie klasifikátorov na detekciu tvárí.

Haarove príznaky

Tieto príznaky sú jednoduché obdĺžnikové oblasti, kde hodnota sa vypočíta z intenzity obrazu. Na obr. 3-2 môžme vidieť rôzne typy Haarových príznakov. Za pomoci rotácií sa dajú vygenerovať ostatné typy príznakov. Výpočet jedného príznaku sa dá určiť zo vzorca (1), kde zo sumy bielych oblastí sa odpočíta suma čiernych oblastí.

$$f(x) = \sum_{w \in W} x(w) - \sum_{b \in B} x(b) \quad (3.1)$$

Pixel o intenzite x, ktorý sa nachádza v oblasti skúmaného Haarovho prínaku, patrí do skupiny w v prípade, že sa nachádza v bielej oblasti. Ak sa nachádza v čiernej oblasti Haarovho príznaku, tak patrí do skupiny b.



Obrázok 3-2: Typy Haarových príznakov

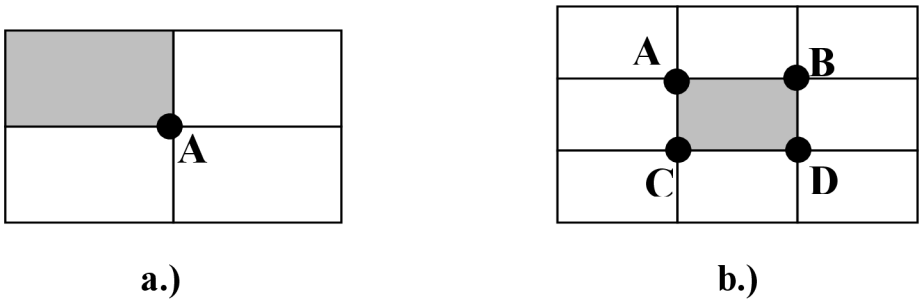
3.2.2 Integrálny obraz

Integrálny obraz dokáže rýchlo a efektívne vypočítať hodnotu jednotlivých príznakov z obdĺžnikových oblastí. Veľkosť integrovaného obrazu je o jeden stĺpec širší a o jeden riadok vyšší. Hodnoty integrovaného obrazu *II* (3.2) sú súčtom hodnôt intenzity pixelov obrazu *I* naľavo a navrch do pozície pixelov v obraze, čo nám znázorňuje obr. 3-3a. Pixel umiestnený v pravom dolnom rohu je súčtom intenzít v celom obraze, zatiaľ čo prvý riadok a stĺpec majú hodnotu 0 [7].

$$II(x,y) = \sum_{i,j}^{x,y} I(i,j) \,. \tag{3.2}$$

Ak poznáme pozíciu rohových bodov (A,B,C,D), môžeme pomocou vzorca (3.3) vypočítať intenzitu obdĺžnikovej časti, obr 3-3b.

$$I_{obd} = A - B - C + D \,. \tag{3.3}$$

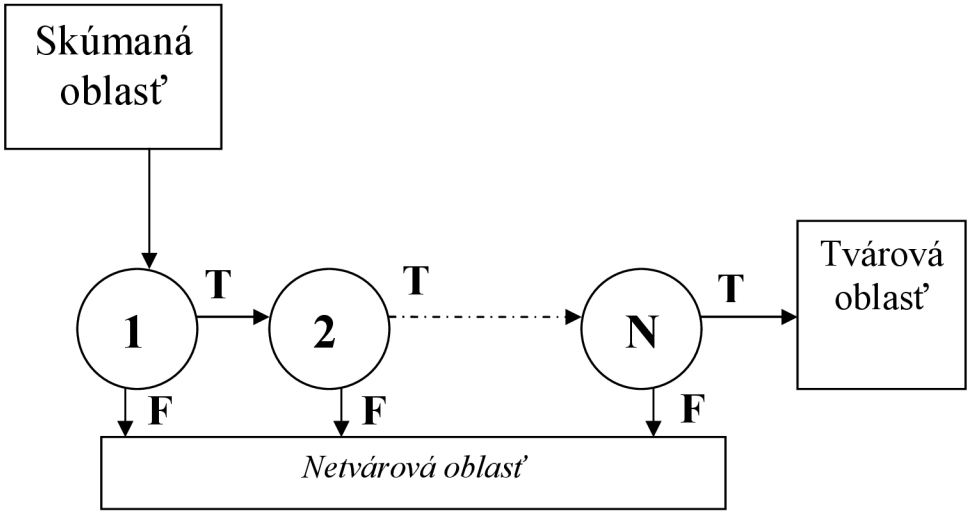


Obrázok 3-3: Integrálny obraz : a.) hodnota bodu A reprezentuje šedú oblasť , b.) oblasť pri využití rovnice 3

3.2.3 Zapojenie klasifikátorov do kaskády

Vo väčšine skúmaných oblastiach obrazu sa nenachádza tvár. Preto sa začalo využívať kaskádové zapojenie, v snahe o zníženie chybovosti a zvýšenia efektivity.

Popis kaskádového zapojenia vychádza z dokumentu [9]. Stupne v kaskáde sú natrénované pomocou AdaBoostu, kde každý stupeň rozhoduje, či sa nachádza v danom výreze tvár. Teda obraz sa nespracováva ako celok ale po častiach. Tieto časti sa vyberajú pomocou posuvného pod-okna. Toto okno mení svoju pozíciu x,y po celom skúmanom obraze a taktiež mení svoju veľkosť v závislosti na predpokladanej veľkosti detekovaného objektu. Vstupná oblasť vybraná pod-oknom je skúmaná klasifikátorom a následne rozhodne či sa v skúmanom objekte nachádza tvár. Ak sa nenachádza, predá oblasť ďalšiemu klasifikátoru. Ten preskúma znova oblasť a znova rozhoduje. Takto sa to opakuje až kým sa objekt nedostane k poslednému klasifikátoru. Princíp je znázornený na obr. 3-4.



Obrázok 3-4: Schematické zobrazenie kaskády

Výhodou tohto zapojenia je odstránenie oblastí už na začiatku, kde sa nenachádza tvár. Tam môžu byť klasifikátory jednoduchšie a rýchlejšie, ako klasifikátory na konci kaskády, kde sa môžu nachádzať s podstatne väčšou zložitosťou. Práve táto myšlienka je výhodou na ušetrenie času detekcie.

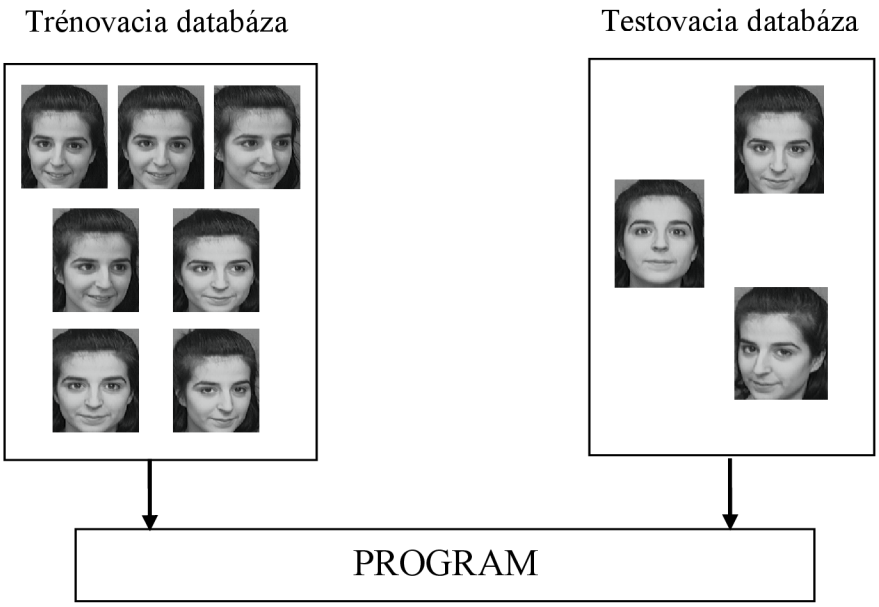
4 DATABÁZA TVÁRIÍ

Použil som databázu tvárií ORL Database. Táto sada tvárí vznikla na univerzite v Cambridge a slúži na rozpoznávanie tvárí. V databáze sa nachádza 40 typovo rôznych ľudí. Každý človek je zachytený na 10 snímkach v rôznych nakloneniach hlavy, s alebo bez okuliarov, s alebo bez úsmevu, so zavretými očami alebo otvorenými a pri rôznych svetlostiach. Takže táto databáza obsahuje 400 snímok tvárí.

Každá snímka je v prevedení stupni šedej (grayscale) s rozmerom 92 x 112 a formát je typu PGM. Táto databáza je voľne dostupná na internete [11]. Obrázok 4-1 znázorňuje časť tvárí z databázy.

Obrázok 4-1: Výber tvárí z databázy ORL Database [11]

V mojom programe som použil 7 rôznych tvárí - tried. Na trénovanie som použil 7 z každej triedy a na testovanie vždy 3 z rovnakej triedy, no iný obrázok. Ako vstupné dáta pre program slúžia dve databázy. Prvá je trénovacia a druhá testovacia kde sú umiestnené práve zvyšné tváre z trénovacej sady. Pre lepšiu ilustráciu nám zobrazuje obrázok 4-2 testovaciu databázu.



Obrázok 4-2: Vstupné databázy programu

5 IMPLEMENTÁCIA

V tejto kapitole sa budem zaoberať implementáciou vlastných programov. Oba moje programy vychádzajú z analýzy popísanej v predchádzajúcich kapitolách. Každý z programov bol navrhnutý v inom programovacom jazyku, na základe požadovaných kritérií.

Prvý program pracuje ako jednoduché rozpoznávanie tvárí, uložených v databáze, pomocou jednej najzákladnejšej metódy PCA. Tento program pracuje v reálnom čase zachytávajúc tvár osoby pomocou web kamery. Dôraz sa kladie najmä na rýchlosť rozpoznávania. Úspešnosť v tomto systéme nehrá veľkú úlohu, keďže rozpoznávanie prebieha zo zosnímaného videa, kde rôznym natočením hlavy môžme zvýšiť percento úspešnosti rozpoznania. Pracuje teda s obrázkami tváre, získanými v reálnom čase užívateľom. V druhom programe som sa už nezameral na detekciu tváre, ale snažil som sa o porovnanie jednotlivých metód z hľadiska úspešnosti identifikácie osôb. Tu sa nekladie dôraz na rýchlosť vyhodnocovania, ale na úspešnosť. Pracuje sa už so statickými obrazmi tváre.

5.1 PROGRAM NA ROZPOZNANIE TVÁRE

Táto aplikácia sa zaoberá identifikáciou osoby na základe detekcie ľudskej tváre. Postup môžme rozdeliť do dvoch krokov. Prvým krokom je detekcia tváre z uloženého snímku pomocou web kamery. Ďalší krok obsahuje už samotnú identifikáciu osoby. Použil som metódu PCA. Princíp tejto metódy je popísaný v kapitole 2.1.1. Následne 1NN klasifikáciu, čo predstavuje nájdenie najbližšieho suseda pomocou Euklidovskej funkcie, princíp je v kapitole 2.2.3.

Táto metóda vykazovala priemerné výsledky. Pri vypočítaní vzdialenosti medzi testovacím a trénovacím obrázkom nie vždy určila správne hodnoty. Dochádzalo občas k nerozpoznaniu osoby podľa prahovej hodnoty alebo k rozpoznaniu inej osoby ako zdetekovanej z kamery. Tieto testy som uskutočňoval v “laboratórnych” podmienakch kde sa nepočítalo s rôznym osvetlením scény, naklonením tváre a pod.

5.1.1 Implementačný jazyk

Základom je výber implementačného jazyka, v ktorom bude daný program vytvorený. Vzhľadom na požadovanú rýchlosť systému, keďže rozpoznávanie sa uskutočňuje v reálnom čase, je lepšie použiť kompilované jazyky na rozdiel od skriptovacích jazykov, ktoré sú náročnejšie na dobu vyhodnocovania. Preto som sa rozhodol použiť jazyk *C/C++*.

V jazyku *C/C++* môžeme využiť knižnicu OpenCV. Celý názov je *Open Source Computer Vision Library.* Obsahuje okolo 500 funkcií a preto sa značne používa v mnohých spoločnostiach ako aj výskumných centrách a taktiež uľahčuje prácu v systéme, ktorý je zameraný na spracovanie videa [10]. Knižnica pozostáva zo štyroch modulov:

- CV – obsahuje všetky hlavné obrazové funkcie pre spracovanie obrazu a počítačového videnia

- CXCORE – Obsahuje základé dátové štruktúry, podporu pre operácie lineárnej algebry, funkcie pre kreslenie a podporu pre XML

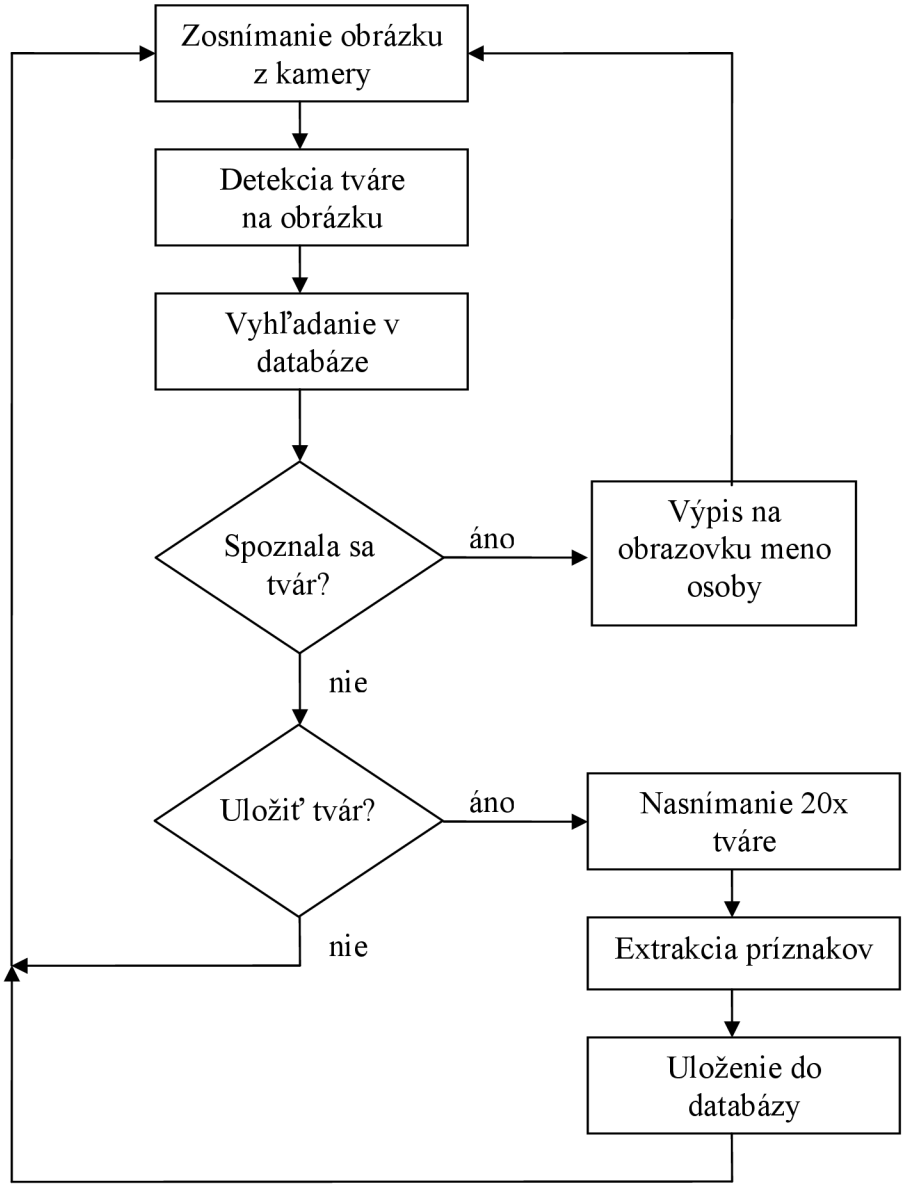
- HIGHGUI – GUI funkcie ako napr. zobrazovanie okna, načítanie a zobrazenie videa, práca s myšou a pod.

- CVAUX – obsahuje doplňujúce funkcie OpenCV

OpenCV implementuje široký výber nástrojov na interpretáciu obrazu. Napriek niektorým primitívam ako sú binarizácia, filtrovanie, pyramídy, je OpenCV predovšetkým knižnica vysokého stupňa implementujúca algoritmy pre kalibračné techniky (Camera Calibration), detekciu črty (Feature) a sledovanie (Optical Flow), analýzu tvaru (Geometry, Contour Processing), analýzu pohybu (Motion Pemplates, Estimators), 3D rekonštrukciu (View Morphing), segmentáciu a rozpoznávanie objektu (Histogram, Embedded Hidden Markov Models, Eigen Objects) [10].

5.1.2 Vlastný program

Pre lepšiu ilustráciu programu nám slúži obr. 5-1. Ako môžme vidieť, program pozostáva z jednotlivých blokov, ktoré budú podrobnejšie vysvetlené v ďalších kapitolách.



Obrázok 5-1: Vývojový diagram rozpoznania tváre vo videu

Zosnímanie obrázku z kamery

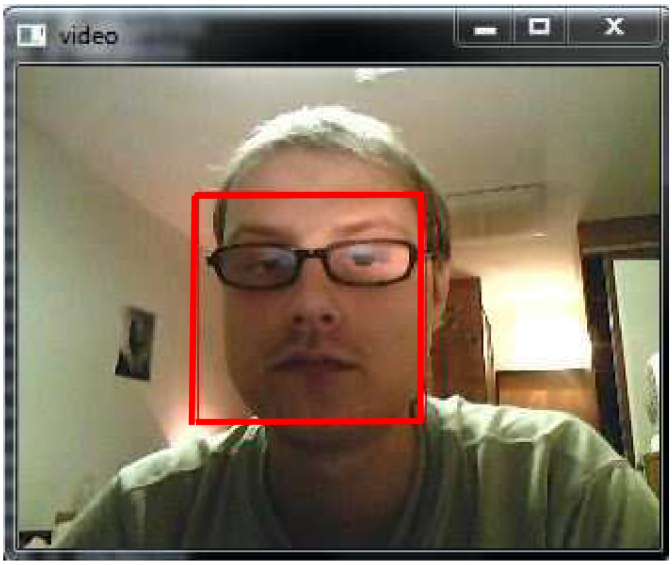
Na zosnímanie videa som použil web kameru s rozlíšením 1,3 Mpixels. V programe som využil knižnicu OpenCV a jej podporovanú časť HIGHGUI na zosnímanie a následné zobrazenie videa v okne, čo môžme vidieť na obr. 5-2.

Obrázok 5-2: Okno so zosnímaným obrázkom

Detekcia tváre na obrázku

Aby sme mohli rozpoznať osobu na obrázku, najprv musíme zdetekovať tvár. Na detekciu som použil metódu Viola-Jones, ktorá je popísaná v kapitole 3.2. V programe som využil funkciu knižnice OpenCV Haar Cascade Face Detector, ktorá umožňuje zdetekovať frontálnu tvár v obrázku. Táto knižnica obsahuje v sebe radu ďalších klasifikátorov na detekciu, ako napr. detekciu očí, detekciu nosa, atď. Aby sme mohli aplikovať túto funkciu, musíme si najprv zosnímaný obrázok predspracovať. A teda obrázok zmenšíme a prevedieme ho do odtieňa šedej. Je to z dôvodu, že takýto obrázok obsahuje len jeden kanál farby, kdežto farebný obsahuje 3 kanály farby (R,G,B). Takto prevedený obraz má menšiu výpočetnú náročnosť. Následne po úprave aplikujeme funkciu *cvHaarDetectObjects* z knižnice OpenCV. Tá nám nájde všetky tváre v snímke, ale ďalej sa pracuje len s najväčšou zdetekovanou tvárou. Odstránia sa tým nežiadúce tváre v pozadí a pracuje sa len s frontálnou tvárou pred kamerou.

Táto detekcia tváre nieje veľmi náročná časovo na detekciu tváre. Systém potreboval cca 60 ms na zdetekovanie tváre, ak bola pod správnym uhlom natočenia ku kamere. Ak bola tvár odvrátená od kamery alebo snímka neobsahovala žiadnu tvár, systému trvalo cca 450 ms na detekciu, čo je vlastne pol sekundy. Obr. 5-3 zobrazuje zdetekovanú tvár.



Obrázok 5-3: Zdetekovaná tvár

Rozpoznanie tváre

Ak je už zdetekovaná tvár, môže sa začať s rozpoznávaním. Na to slúži databáza nájdených a uložených tvárí. Databáza pozostáva z nasledujúcich súborov:

- *train.txt* – Súbor, do ktorého sa ukladá číslo osoby, meno osoby a cesta uloženého zosnímaného obrázku.
- *facedata.xml* – Súbor, kde sa zapisujú hodnoty po porovnaní testovacieho a trénovacieho obrázku.
- *data* – Zložka, do ktorej sa ukladajú obrázky zdetekovaných tvárí na testovanie

Ak databáza neobsahuje zdetekovanú tvár, musí sa do nej najprv uložiť. Ukladanie do databázy prebieha vo viacerých krokoch. Prvým krokom je nasnímanie 20x tváre a zmeniť nasnímaný obrázok na jednotnú veľkosť. Musí sa dodržať rovnaká veľkosť trénovacieho a testovacieho obrázku. Takto zmenená snímka sa uloží do adresára *data.* Tieto obrázky môžme nazvať aj trénovacími. Následne sa uloží do súboru *train.txt* identifikačné číslo tváre, meno ktorej zadá užívateľ a cestu uloženého snímku. Ďalším krokom je extrakcia príznakov z uložených obrázkov. Tomuto kroku budem venovať väčšiu pozornosť.

Extrakcia príznakov

Na extrakciu príznakov som sa rozhodol použiť analýzu hlavných komponentov (PCA). Je to jedna z prvých metód, ktoré sa používali.

Postup metódy PCA je popísaný v kap. 2.1.1. Premietnutie do podpriestoru a výpočet vlastných vektorov a vlastných hodnôt by bol výpočetne náročný , preto sa využila knižnica OpenCV, v ktorej je implementovaná funkcia. Tieto vypočítané hodnoty sa uložia spolu s menom osoby a identifikačným číslom tváre do súboru *facedata.xml.* Takže celý proces ukladania tváre pre jasnejšiu ilustráciu môžme zhrnúť do nasledovných krokov:

1. Zosnímanie a zdetekovanie tváre
2. uloženie tváre do adresára a informácie do súboru *train.txt*
3. výpočet podpriestoru
4. premietnutie trénovacieho obrázku do podpriestoru
5. uloženie informácií do súboru *facedata.xml*

Rozpoznanie tváre

Po uložení tváre do databázy nastane proces samotného rozpoznávania. Zosníma sa testovací obrázok z web kamery a znovu sa zmení veľkosť na preddefinovaný tvar, aby bol zhodný s testovacím. Testovací obrázok sa premietne do podpriestoru a vypočítajú sa vlastné hodnoty a vlastné vektory. Na tento proces sa znovu využíva knižnica OpenCV. Po získaní hodnôt sa načítajú uložené dáta zo súboru *facedata.xml* a hľadajú sa najbližšie spoločné hodnoty. Prah rozpoznania je od 0 do 1. Určuje nám, či sa jedná o podobnú tvár alebo o inú

tvár, napr. ak by hodnota rozpoznania vykazovala 0,9 s najväčsou pravdepodobnosťou sa jedná o podobnú tvár, ale ak by hodnota bola v rozmedzí 0 ÷ 0,6 trénovacia tvár by bola rozdielna od testovacej. Ak teda systém vyhodnotí testovaciu a trénovaciu tvár za podobné, nasleduje výpis mena zdetekovanej osoby,čo môžme vidieť na obr.5-4.

Obrázok 5-4: Rozpoznaná tvár osoby

Tieto vzájomné vzdialenosti, ktoré reprezentujú podobnosť obrázkov sú určené na základe metriky. V mojom programe som pracoval s metrikou nazývanou Euklidovská, no existujú viaceré metriky, ktoré vykazujú rôzne vlastnosti.

5.2 PROGRAM NA POROVNANIE METÓD

Tento program sa zaoberá porovnaním jednotlivých metód popísaných v kapitole 2.1. Je hlavne zameraný na kvalitu identifikácie tváre z hľadiska vyhodnocovania údajov medzi testovacím a trénovacím obrázkom. Tentokrát sa pracuje s reálnymi podmienkami, a teda počíta sa s rôznym osvietením scény, naklonením tváre a atď.

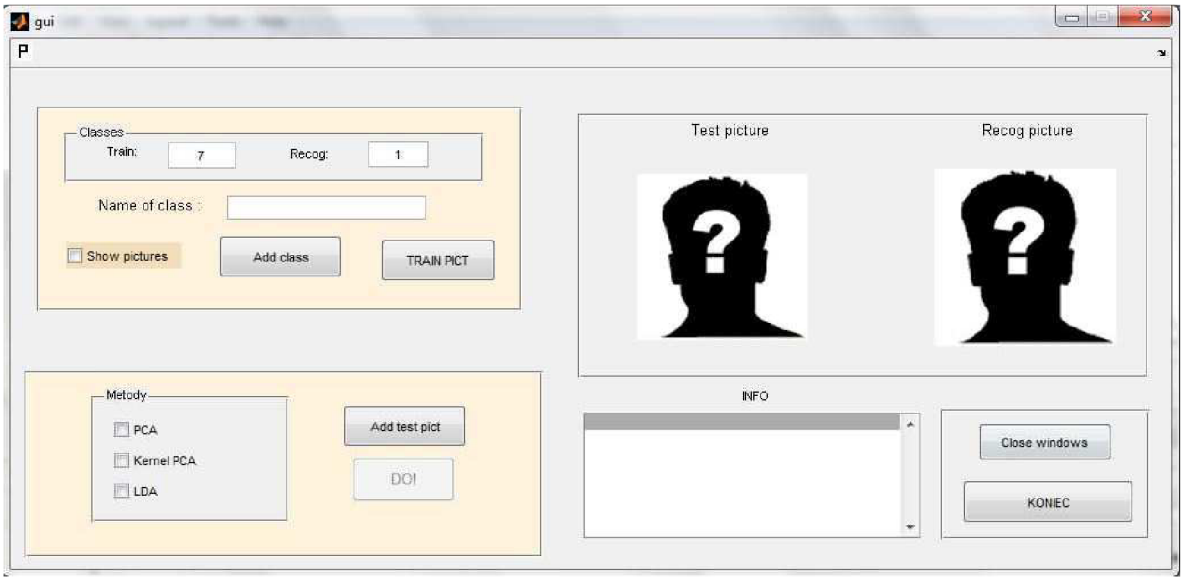
5.2.1 Implementačný jazyk

Na rozdiel od predchádzajúceho programu nepracujeme už v reálnom čase, pracuje sa hlavne s výpočetnými funkciami a teda vhodnejšou voľbou je tentokrát program MATLAB, ktorý rieši výpočetné operácie lepšie.

Názov MATLAB vznikol skrátením slov MATrix LABoratory (voľne preložené „laboratórium s maticami“), čo zodpovedá skutočnosti, že kľúčovou dátovou štruktúrou pri výpočtoch v MATLABe sú matice. Je to programové prostredie (programovací jazyk) pre vedeckotechnické numerické výpočty, modelovanie, návrhy algoritmov, počítačových simulácií, analýzu a prezentáciu dát, merania a spracovania signálov, návrhy riadiacich a komunikačných systémov.

5.2.2 Vlastný program

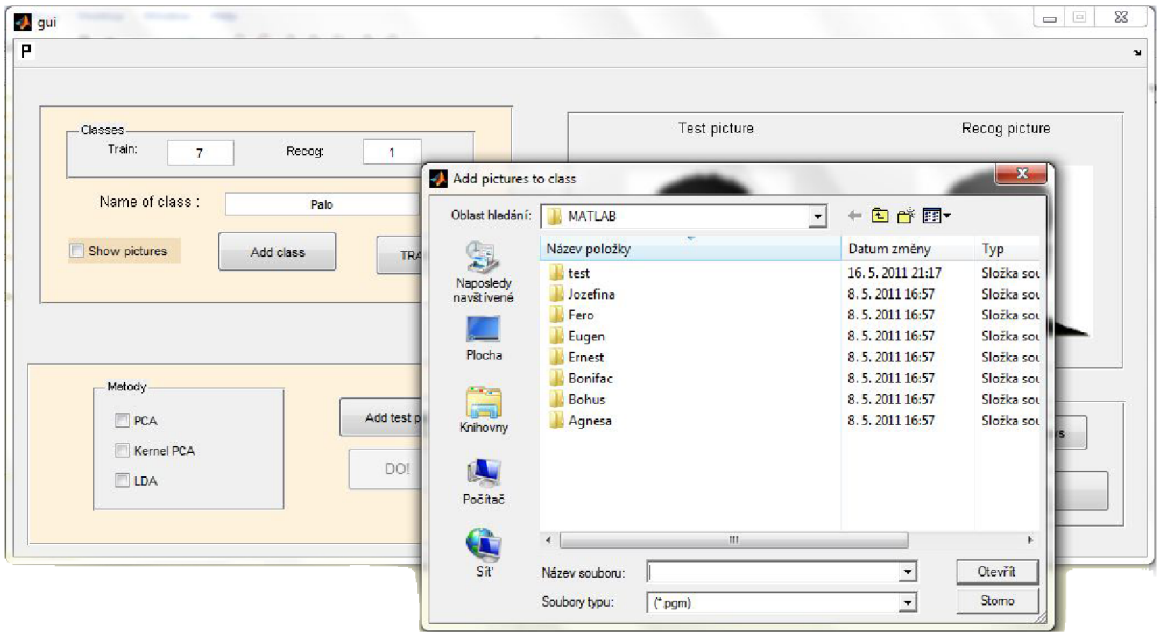
Celkový program pozostáva z jednej GUI aplikácie situovanej na jednom hlavnom okne. V programe sú jednotlivé bloky rozdelené podľa významu alebo funkcie. Tieto funkcie a bloky systému budú rozpísané v nasledujúcich kapitolách. Na obrázku 5-5 je hlavné okno programu.



Obrázok 5-5: Hlavné okno programu

Blok natrénovania obrázkov

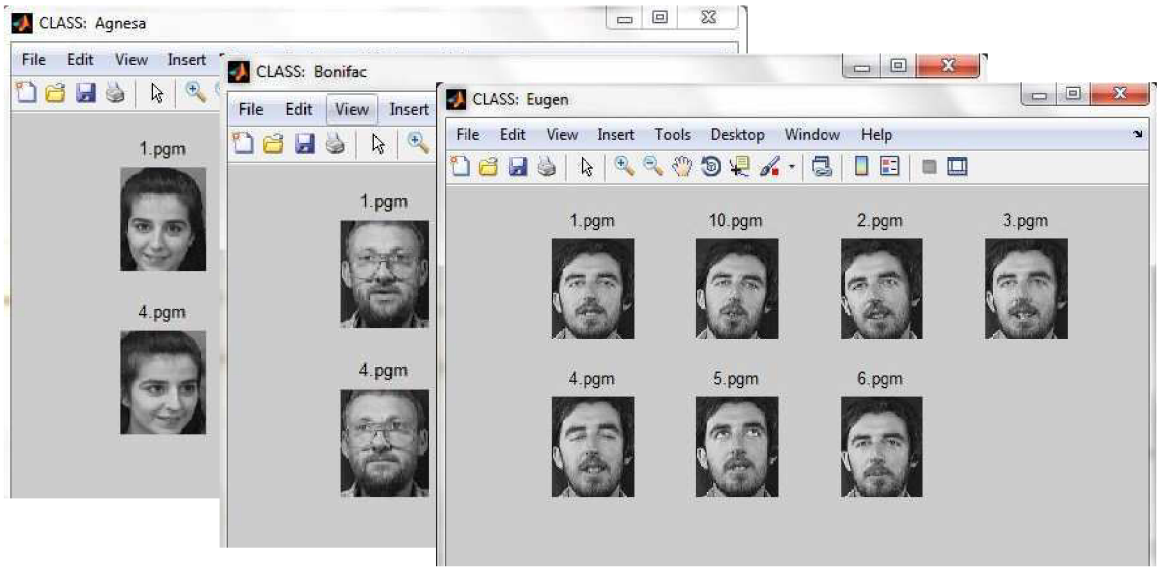
Tento blok sa nachádza v ľavom hornom rohu aplikácie a slúži ako vstupná jednotka programu. Umožňuje nám vkladať tváre do novo vytvorenej triedy v trénovacej databáze a následne ich natrénovať. Program neumožňuje akúkoľvek prácu (rozpoznávanie) preskočením tejto funkcie - natrénovanie dát. Na pridanie obrázku tváre slúži tlačidlo *Add class.* Ak by sme chceli vytvoriť novú sadu tvárí (triedu), napíšeme do poľa *Name of class* príslušný názov identický pridávanej osobe a po stisknutí tlačidla *Add class* sa otvorí dialógové okno, kde si môžme vyhľadať a pridať obrázky. Obrázky sa ukladajú vždy implicitne na miesto určené pre trénovaciu databázu obrázkov pre program. Pridávanie tváre je znázornené na obr. 5-6.



Obrázok 5-6: Pridanie tváre do novo vytvorenej triedy

Po úspešnom vložení novej sady tvárí musíme však zmeniť údaj v informačnom okne *Classes,* ktorá nás informuje o počte trénovacích a testovacích tried. A teda položku *Train* inkrementujeme o hodnotu pridaných tried. Ak by sme tak neučinili, program by sa riadil len hodnotou *Train* a nami novo pridané sady tvárí by ignoroval. V tomto kroku je systém pripravený na natrénovanie tvárí z trénovacej databázy.

Samotné trénovanie testovacích obrázkov pozostáva z niekoľkých krokov. Prvým je načítanie obrázkov do trénovacej matice. Tu máme na výber zobrazenia si trénovacích tvárí v jednotlivých triedach. Ak zvolíme políčko *Show pictures*, pri načítaní sa nám zobrazia jednotlivé triedy s tvárami na samostatných oknách.



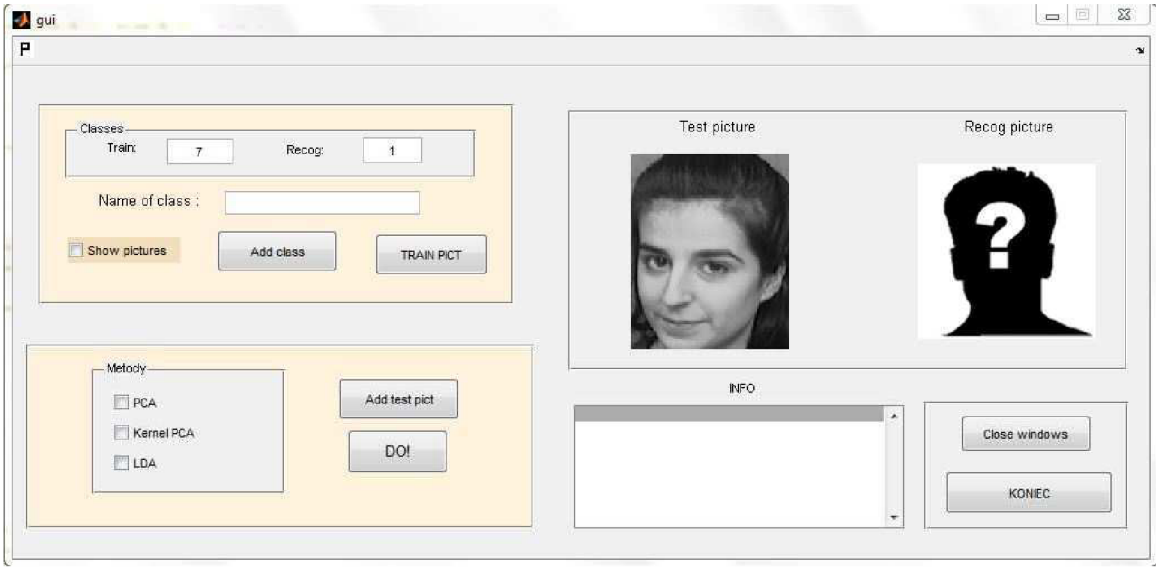
Obrázok 5-7: Výber zobrazených trénovacích tried

Po načítaní obrázkov nastáva najdôležitejšia fáza, a to samotné natrénovanie tvárí. Postup trénovania obrázkov je popísaný v kapitole 2.1. Všetky tváre z databázy sa natrénujú všetkými tromi metódami a vypočítané hodnoty sa uložia do súborov. Výpočetný proces je hlavne pri metóde LDA zdĺhavý a náročný, preto ukladanie výsledkov do súborov odľahčí systém pri opätovnom spustení a trénovaní. Úspešné natrénovanie sa oznámi v informačnom okne programu *Info* vetou *TRAIN pictures successfull.* Ak by sme po opätovne spustenom programe zadali trénovať obrázky, dáta by sa už len načítali z uložených súborov a v informačnom okne by sa zobrazila hláška *LOAD pictures successfull.* Vypočítané eigenvalues a eigenvectors spolu s vypočítaným priemerným obrázkom sa po PCA natrénovaním uložia do súboru *StoredSvm.mat,* čo vlastne predstavuje matlabovský súbor s hodnotami. Vypočítané hodnoty pomocou LDA metódy sa uložia do súboru *StoredScat.mat a* kernel PCA (KPCA) do súboru *StoredKpca.mat*.

Ak sa do databázy pridá nová sada tvárí, systém zdetekuje zmenu v databáze a po stlačení tlačidla *Train pict* proces trénovania s uložením nových hodnôt prebehne znovu.

Blok rozpoznávania testovacej tváre

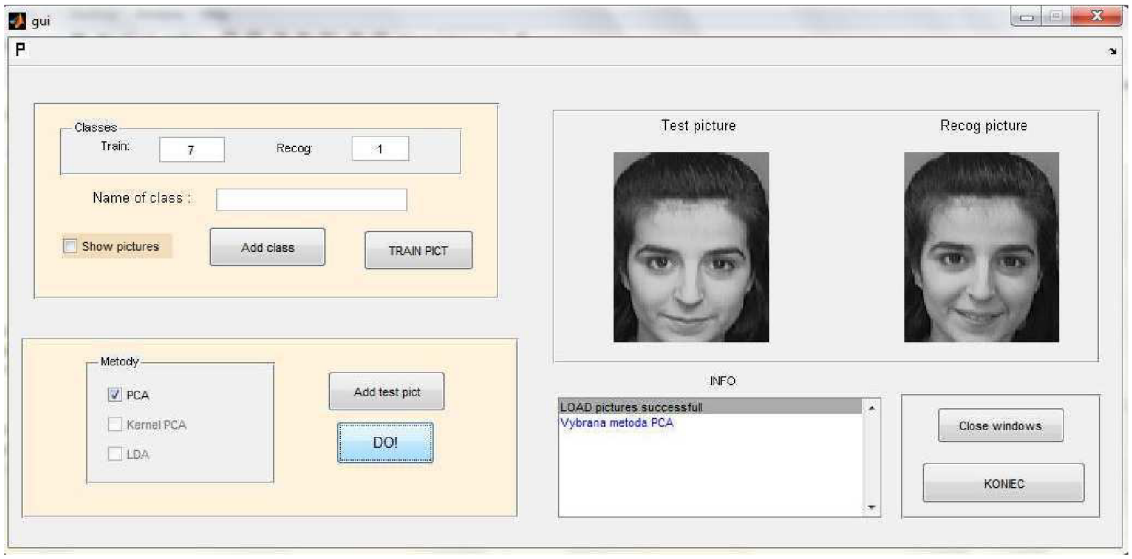
V tomto bloku sa odohráva načítanie testovacej tváre a porovnanie s testovacou sadou tvárí v databáze. Nachádza sa v ľavom dolnom rohu programu. Ako prvým krokom vo fáze rozpoznávania sa načíta testovací obrázok. Pomocou tlačidla *Add test pict* sa otvorí dialógové okno, kde si zvolíme tvár z testovacej databázy. Táto databáza obsahuje z každej triedy 3 rôzne tváre. Viď kapitola 4. Po načítaní sa nám zobrazí zvolená tvár v okne *Test picture,* čo nám znázorňuje obr. 5-8.



Obrázok 5-8: Načítanie testovacej tváre

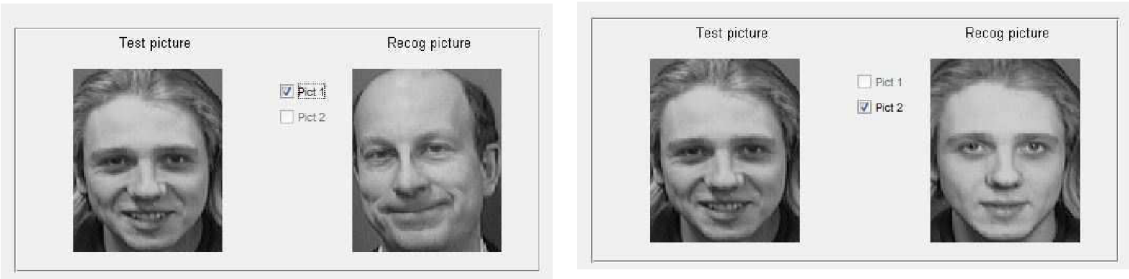
Zvolením patričnej metódy z ponuky vyberieme jednu zaškrtnutím políčka. Program umožňuje porovnávať práve len jednou zvolenou metódou. Aby sme sa dostali k samotnému rozpoznávaniu, musí sa absolvovať prvá fáza natrénovaním trénovacej sady, aby nám program umožnil použitie tlačidla *DO!*.

Pokiaľ by neprebehla prvá fáza, tlačidlo by bolo neaktívne. Proces trénovania testovacej tváre je totožný s trénovaním trénovacej sady. S rozdielom, že vypočítané hodnoty sa neukladajú do súboru, ale porovnávajú s natrénovanými hodnotami. Následne sa vypočíta najpravdepodobnejšie totožný obrázok k testovaciemu obrázku. Ako výstup sa zobrazí v okne *Recog picture* tvár vypočítaná zvolenou metódou, čo môžme vidieť na obrázku 5-9.



Obrázok 5-9: Rozpoznanie tváre

Ako bolo spomínané v kapitole 2.2.3. pri výpočte najbližšieho suseda (určenie obrázku), môže nastať problém s viac nájdenými totožnými minimálnymi hodnotami. Ak program zistí, že sa také hodnoty nachádzajú, indikuje stav hláškou *Found more pictures!* a poskytne na výber oba obrázky. Na obrázku 5-10 môžme vidieť dvojicu najbližších obrázkov k testovaciemu obrázku.



Obrázok 5-10: Nájdenie dvoch najbližších obrázkov

Rozhodovanie sa či testovací obrázok spadá do jednej z trénovacích tried sa uskutočňuje pomocou vhodne zvolenej prahovej hodnoty. Ak je vypočítaná hodnota väčšia ako prahová, systém vyhodnotí obrázok, že nepatrí do žiadnej triedy. Neklasifikovanie testovacieho obrázku znázorňuje obrázok 5-11.

Obrázok 5-11: Nerozpoznanie testovacej tváre

Blok zobrazenia informácií

Tento blok sa nachádza na celej pravej strane programu. Slúži na oboznámenie užívateľa s prebiehajúcimi výpočtami a výsledkami. Testovací a rozpoznaný obrázok sa zobrazuje v okne *Test picture* a *Train picture*. Informačné okno *INFO* poskytuje výstražné alebo informačné hlášky. Tlačidlo *Close windows* slúži na zavretie všetkých okien s trénovacími tvárami, otvorenými počas načítavania po aktivovaní *Show pictures*. Program sa ukončuje tlačidlom *KONIEC*. Obrázok 5-11 zachytáva celý tento blok.

6 DOSIAHNUTÉ VÝSLEDKY

Porovnávané metódy vykazujú každá zvlášť rôzne výsledky úspešnosti. Testy sa uskutočňovali na databáze tvárí ORL Database. Celkové množstvo tvárí v trénovacej databáze bolo 49 a v testovacej databáze 21. Pri klasifikovaní testovacieho obrázku majú výrazný vplyv na úspešnosť dôležité parametre:

- *CuttOff* - je počet zachovaných bázových vektorov transformačnej matice. Dôvodom je zníženie výpočtovej náročnosti metód a odstránenie bázových vektorov, ktoré neobsahujú informácie využiteľné pri rozpoznávaní tvárí

- *Dropped from front* – počet orezaných bázových vektorov transformácie zo začiatku transformačnej matice (vlastné vektory zodpovedajúce najväčším vlastným číslam). Tieto vektory sa nebudú využívať pri projekcii obrázkov do PCA resp. LDA podpriestoru. Dôvodom je, že prvé vektory transformačnej matice sú nositeľmi informácie o svetelných podmienkach [8] a nie sú teda užitočné pri rozpoznávaní tvárí.

V mojom programe pri výpočtoch som sa rozhodol použiť pri faktore *CuttOff* pri PCA a KPCA metóde 100% a pri LDA ~20% z dôvodu výpočetnej náročnosti pri množstve trénovacích obrázkov. Parameter *Dropped from front* som zachoval nemenný, a teda použil som všetky bázové vektory.

Po uskutočnení testov som sa dopracoval k nasledujúcim výsledkom, ktoré sú uvedené v tabuľke 5-1.

Metóda	Úspešnosť [%]
PCA	72,5
KPCA	66,25
LDA	82

Tabuľka 6-1: Úspešnosť jednotlivých metód

ZÁVER

Táto diplomová práca mala za úlohu automaticky identifikovať tvár v reálnych podmienkach. Práca je rozdelená na dva programy, ktoré slúžia na rozpoznanie tváre, ale každá pracuje s rôznymi podmienkami a požiadavkami. Prvý program pracuje s obrázkami tváre zosnímanej z web kamery v reálnom čase a druhý program vyhodnocuje úspešnosť identifikácie rôznych tvárí získaných z databázy tvárí.

V prvom programe na to, aby systém mohol zdetekovať tvár osoby snímajúcu z kamery, musel som naštudovať a pochopiť princíp detekcie tváre. Existuje viacej spôsobov ako detekovať. V kapitole 3. som popísal detekciu na základe farby pokožky a Viola&Jones. Ja som sa rohodol pre detekciu tváre pomocou Viola&Jones. Táto metóda vykazovala lepšie a presnejšie výsledky. Pri výbere detekcie kože systém dobre vyhodnocoval farbu kože, avšak bol náchylný na cudzie predmety, ktoré mali farbu identickú ľudskej pokožky. Pri použití Viola&Jones sa vyselektoval tento jav. Detekcia tváre touto metódou trvala niečo okolo 60 ms takže je pomerne rýchla.

Na samotné rozpoznanie tváre som použil extrakciu príznakov pomocou PCA. Princíp tejto metódy je popísaný v kapitole 2.1.1. Táto metóda vykazovala priemerné výsledky. Pri vypočítaní vzdialenosti medzi testovacím a trénovacím obrázkom nie vždy určila správne hodnoty. Dochádzalo občas k nerozpoznaniu osoby podľa prahovej hodnoty alebo k rozpoznaniu inej osoby ako zdetekovanej z kamery. Tieto testy som uskutočňoval v "laboratórnych" podmienakch, kde sa nepočítalo s rôznym osvetlením scény, naklonením tváre a pod.

Druhý program pracuje v reálnych podmienkach, kde osvetlenie tváre, rôzne predmety na tvári (okuliare, brada, fúzy) a naklonenie hlavy určovali úspešnosť detekcie osoby. Neprebieha tu už proces detekcie tváre. Tento systém vyžaduje databázu tvárí, ktoré metódami popísanými v kapitole 2.1. natrénuje a databázu testovacích tvárí, ktoré nie sú obsiahnuté v trénovacej databáze. Proces rozpoznávania neprebieha na obrázkoch zosnímaných z web kamery, a teda nepracuje sa v reálnom čase, preto systém je zaťažený hlavne na výpočetnú náročnosť. Jednotlivé metódy pri procese rozpoznávania mali značné problémy s obrázkami tvárí, ktoré boli pootočené do strán. Takéto obrázky program nedokázal zaradiť do správnej triedy alebo vôbec nevyhodnotil ako tvár. Tento problém vyselektovala metóda KPCA, ktorej sa darilo určovať tváre o niečo lepšie ako druhým metódam, čo znázorňuje aj tabuľka 6-1. Celkovo sa javila najlepšie metóda KPCA, pretože používa kernelovské metódy, ktoré sú viac sofistikovanejšie ako PCA a LDA. Pracuje s

nelineárnymi prvkami, ktoré prevádza do mnohorozmerného priestoru, kde táto schopnosť absentuje u PCA metódy, a preto umožní efektívne pracovať v mnohorozmernom priestore zaujímavých vlastností

Vytvorením týchto dvoch programov som zistil, aký systém má v čom nedostatky a prednosti. Prvý program slúžil na rýchlu detekciu tváre pomocou web kamery. Takýto systém sa môže použiť pri vstupných kontrolách. Vykazuje menšiu výpočetnú náročnosť a teda aj lepšiu aplikovanosť na hardware. Druhý program sa zaoberal mierou úspešnosti, a teda je vhodnejší do prostredia, kde sa pracuje s dobre známymi obrázkami v databáze a kde sa nevyžaduje rýchlosť, ale precíznosť rozpoznania.

POUŽITÁ LITERATÚRA

[1] www.biometria.sk [online]. 2010 [cit. 2010-11-08]. Čo je biometria?. Dostupné z WWW: <http://www.biometria.sk/co-je-biometria.html>.

[2] Biometria : www.euractiv.sk. In Biometria : Čítanie biometrických údajov [online]. 2010-01-02. , 2010-01-02 [cit. 2010-11-08]. Dostupné z WWW: <http://www.euractiv.sk/informacna-polocnost/zoznam_liniek/biometria>.

[3] DEBORAH, Gough Schools to fingerprint students for security. In Schools to fingerprint students for security [online]., 2010-11-02 [cit.2010-11-08].Dostupné z WWW: <http://www.theage.com.au/news/national/schools-to-fingerprint-dents-for-security/2007/09/01/1188067438 565.html>.

[4] Posts Tagged 'passport' : Biometric passport promise revived. In Posts Tagged 'passport' [online]., 2010-03-09 [cit. 2010-11-08]. Dostupné z WWW:<http://kevinvisser.com/wp-content/uploads/2010/03/fingerprint_biometric_dsu_lg.jpg>.

[5] Ray small optometrics : Eye Examinations. In Ray small optometrics [online]. [s.l.] : [s.n.], 2010 [cit. 2010-11-08]. Dostupné z WWW: <http://www.raysmallopt.co.uk/images/retinal-scan.gif>.

[6] The New World Order. Biometrics : Face Scan [online]. 2010, 1, [cit. 2010-11-08].Dostupný z WWW: < http://thecomingnewworldorder.blogspot.com/2007/05/new-biometric-smart-cards.html >.

[7] ŠTRBA, M.: Detekce obličeje. Bakalářská práce, Vysoké učení technické v Brne, Fakulta informačních technologiíÚstav počítačové grafiky a multimédií, 2008, ved. Michal Hradiš.

[8] MELIŠEK, Bc. Martin; MAZANEC, Bc. Ján; ORAVEC, Doc. Dr. Ing.Miloš. SVM, PCA a LDA v rozpoznávaní ľudských tvárí [online]. 2007. [cit. 2010-11-17].

[9] VIOLA, P.; JONES, M.: Rapid Object Detection using a Boosted Cascade of Simple Features. cvpr, ro_cn__k 1, 2001: str. 511, ISSN 1063-6919

[10] OpenCV2005Q4tutorial.ppt,http://sourceforge.net/project/showfiles.php?groupid=22870&package id=16949&release id=17485

[11] The Database of Faces [online]. 2002 [cit. 2011-05-18]. The Database of Faces. Dostupné z WWW: <ttp://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.htm

[12] SCHÖLKOPF, Bernhard ; SMOLA, Alexander; M ULLER, Klaus-Robert. Kernel Principal Component Analysis. In Kernel Principal Component Analysis [online]. [cit. 2011-05-20]. Dostupné z WWW: <http://www.eecs.berkeley.edu/~wainwrig/stat241b/scholkopf_kernel.pdf>.

[13] ČEPEK:, Miroslav . Vytěžování dat [online]. [s.l.], 2010. 28 s. Cvičení. České vysoké učení technické v Praze. Dostupné z WWW: <http://cw.felk.cvut.cz/lib/exe/fetch.php/courses/y336vd/cviceni/files/vd-2010-cv4-knn.pdf>.

[14] KOSKOVÁ, Gabriela . Klasifikácia a regresia. In Klasifikácia a regresia [online]. 13. marca 2007 [cit. 2011-05-21]. Dostupné z WWW: <http://www2.fiit.stuba.sk/~polcicova/ZZ/prednasky/04_klasifikacia.pdf>.

ZOZNAM PRÍLOH

PRÍLOHA 1 : Zdrojový kód pre výpočet PCA

PRÍLOHA 2 : Zdrojový kód pre výpočet LDA

PRÍLOHA 3 : Zdrojový kód pre výpočet KPCA

PRÍLOHA 4 : Obsah priloženého CD

PRÍLOHA 1 : Zdrojový kód pre výpočet PCA

```
// vypočítanie eigenvalues a eigenvectors pre trénovaciu sadu
// pre vypočítanie testovacieho obrázka je postup podobný ako pre výpočet
// trénovacích obrázkov, len sa ukladajú hodnoty do premennej recog

X = double(reshape(train.I,[Nx*Ny M]))./256;  // načítanie tvárií do matice
me = mean(X,2);                               // zpriemerovanie tvárií
A = X - repmat(me,[1 nEigFace]);              // odpočítanie priem. obr.
clear X                                       // uvolnenie pamete
[U,E,V] = svd(A,0);                     // výpočet eigenvalues a eigenvectors
eigVals = diag(E);                         // zoradenie eigenvalues vzostupne
eigVecs = U;
clear U V                                     // uvolnenie pamete
save(svdFile,'eigVecs','eigVals','me','A')    // uloženie hodôt do súboru

P = eigVecs(:,1:nEigFace);        // výstupné eigenfaces

// projekcia každej tváre trénovacej sady na eigenfaces a uloženie váhy

train.wt = P'*A;

// nájdenie najmenšej vzdialenosti pomocou Euklidovej vety

for l=1:nEigFace   // pre každý testovací obrázok
        recog.euDis(l,j) = sqrt(sum((recog.wt(:,l) - train.wt(:,l)).^2));
end
```

PRÍLOHA 2 : Zdrojový kód pre výpočet LDA

```
// vypočítanie medzitriednu vzdialenosť, Sb
// vypočítanie vnútrotriednu vzdialenosť,  Sw


for i = 1:numOfClass
        row = trainClass.mean{i} - me;
        prod = row * row';
        Sb = Sb + prod;
end

for i = 1:numOfClass
      for j = (trainClass.classStartIndex(i)):(trainClass.classEndIndes(i))
            row = X(:,j) - trainClass.mean{i};
            prod = row * row';
            Sw = Sw + prod;
      end
end
    clear prod row

    save(scatFile,'Sb','Sw')   // uloženie hodôt do súboru
end

Sbb = P1.'*Sb*P1;             // použitie PCA na projekcu do podpriestoru
Sww = P1.'*Sw*P1;
clear Sb Sw                   // uvolnenie pamete

[V,D] = eig(Sbb,Sww);         // Nájdenie eigenvalues a eigenvectors

Ds = diag(D);           // extrakcia eigenvalues
[tmp,ndx] = sort(abs(Ds));  // zoradenie od največšiu po najmenšiu hodnotu
eigVals = Ds(ndx);
eigVecs = P1*V(:,ndx);
clear D Ds V            // uvolnenie pamete

P = eigVecs(:,1:Mp);                          // výstupné fisherfaces
P = P./repmat(sum(P.^2).^0.5,Nx*Ny,1);       // normalizovanie


// projekcia každej tváre trénovacej sady na eigenfaces a uloženie váhy
train.wt = P.'*A;
```

PRÍLOHA 2 : Zdrojový kód pre výpočet KPCA

```
// vytvorenie kernel matice trénovacej sady
    for i=1:size(kTrain,1)
        for j=i:size(kTrain,2)
            kTrain(i,j)=gKernel(trainData(:,i),trainData(:,j));
            kTrain(j,i)=kTrain(i,j);
        end
    end

// vytvorenie kernel matice testovacieho obrázku

    for i=1:size(kTest,1)
        for j=1:size(kTest,2)
            kTest(j,i)=gKernel(testData(:,i),trainData(:,j));
            %kTest(j,i)=gKernel(testData(:,j),trainData(:,i));
        end
    end

// vycentrovanie kernel matic

oneM=ones(M,M)/M;
kTrain=kTrain-oneM*kTrain-kTrain*oneM+oneM*kTrain*oneM;
kTest=kTest-oneM*kTrain-kTest*oneM+oneM*kTrain*oneM;


// vypočítanie kernel eigenvalues
[kVectors, kValues]=eig(kTrain);
kValues=diag(kValues);

 [kValues, IX]=sort(kValues,'descend'); // Zoradenie eigenvalues zostupne
kVectors=kVectors(:,IX);
IX=kValues>=0;
kValues=kValues(IX);
kVectors=kVectors(:,IX);

// normalizácia kernel vectors
for i=1:size(kVectors,2)
    kVectors(:,i)=kVectors(:,i)/sqrt(kValues(i));
end


// Projekcia trénovacích a testovacích dát do PCA podpriestoru

// pre každý trénovací obrázok, pre testovací je rovnaký s premennou
// csTestData()

for i=1:size(kTrain,2)
    // vypočítanie projekcie pre každý obrázok
    for j=1:size(kVectors,2)
       temp=0;
        for k=1:size(kTrain,2)
            temp=temp+kVectors(k,j)*kTrain(k,i);
        end
        csTrainData(j,i)=temp;
    end
end
```

PRÍLOHA 4 : Obsah priloženého CD

Súbory potrebné k správnemu fungovaniu programu v prostredí MATLAB:

- gui.m
- gui.asf
- gui.fig
- priečinok **test** s obrázkami testovacích tvárí
- priečinok **train** s obrázkami trénovacích tvárí
- ORL databáza

Aby program pracoval správne, je nutné v programe prepísať cestu k trénovacím obrázkom.

Súbory potrebné k správnemu fungovaniu prvého programu v prostredí *C/C++:*

- main.cpp
- projekt.vcxproj
- projekt.vcxproj.filters
- projekt.vcxproj.user
- train.txt
- facedata.xml
- priečinok **data** so zaznamenanými tvárami

Diplomová práca:

- DP_Kicina.pdf