VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

ODHALENIE MORPHOVANÝCH ODTLAČKOV PRSTOV

MORPHED FINGERPRINT DETECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE **Bc. DENIS DOVIČIC**
AUTHOR

VEDOUCÍ PRÁCE **Ing. ONDŘEJ KANICH, Ph.D.**
SUPERVISOR

BRNO 2022

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2021/2022

Zadání diplomové práce

Student: **Dovičic Denis, Bc.**
Program: Informační technologie a umělá inteligence
Specializace: Strojové učení
Název: **Odhalení morphovaných otisků prstů**
Morphed Fingerprint Detection
Kategorie: Zpracování obrazu
Zadání:

1. Prostudujte literaturu týkající se rozpoznávání podle otisků prstů a daktyloskopie. Seznamte se s metodami morphingu využívaných v biometrii, zejména s metodou řezu.
2. Navrhněte algoritmus, který odhalí morphované otisky prstů (nabývající více identit) vytvořené pomocí metody řezu. Detekujte řeznou linií a jednotlivé oblasti původních otisků prstů.
3. Implementujte navržený algoritmus z předchozího bodu.
4. Otestujte algoritmus z předchozího bodu na databázi normálních a morphovaných otisků prstů (s využitím metody řezu). Analyzujte přesnost odhalení i detekce řezné linie.
5. Dosažené výsledky shrňte a diskutujte. Uveďte možná rozšíření či vylepšení jak v oblasti odhalení tak i tvorby morphovaných otisků.

Literatura:

- Drahanský, M.: *Hand-Based Biometrics: Methods and technology*, IET 2018, pages 430, ISBN 978-1-78561-224-4.
- Maltoni, D., Maio, D., Jain, A.K. and Prabhakar, S.: *Handbook of Fingerprint Recognition*. Springer, 2009, p. 512. ISBN 978-1-8488-2254-2.
- Dovičic, D.: *Morphing otisků prstů*, Bakalářská práce, FIT VUT v Brně, Brno, 2020.

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/
Vedoucí práce: **Kanich Ondřej, Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2021
Datum odevzdání: 18. května 2022
Datum schválení: 3. listopadu 2021

Abstrakt

Táto diplomová práca sa zaoberá detekciou morphovaných odtlačkov prstov v náväznosti na moju bakalársku prácu a projektovú prax. Doposiaľ neexistuje žiadna verejná práca venujúca sa detekcií morphovaných odtlačkov prstov. Morphované a obyčajné odtlačky sú považované za stochastické textúry a sú popísané binarizovanými štatistickými vlastnosťami odhadnutými ako nezávislé komponenty scény. K dosiahnutiu cieľa bol zvolený klasifikačný algoritmus support vector machine, pracujúci s histogrammi modelujúcimi pravdepodobnosť výskytu BSIF deskriptoru v textúre. Testovanie prebehlo na morphovaných odtlačkoch prstoch, ktoré vznikli z nikoľkých dátových sád odtlačkov: Bergdata, Sagem MSO, SecuGen a syntetické. Ako najoptimálnejší detektor sa ukázal SVM s polynomickou kernelovou funkciou s presnosťou 97,12 % na dátovej sade zo sensoru Bergdata, 98,49 % na Sagem MSO, 94,97 % na SecuGen a 100 % na syntetických odtlačkoch. Na adaptívnej metóde morphingu z mojej projektovej praxe je presnosť detekcie 98,85 % na dátovej sade zo sensoru Bergdata, 98,49 % na Sagem MSO, 94,97 % na SecuGen a 100 % na syntetických odtlačkoch. Vylepšená metóda morphingu z mojej projektovej praxe nepreukázala vplyv na znemožnenie detekcie morphingu, aj napriek tomu, že generuje silnejšie dvojité identity.

Abstract

This diploma thesis deals with the detection of morphed fingerprints, following my bachelor's thesis and project practice. There is no public work dealing with the detection of morphed fingerprints so far. Morphed and normal fingerprints are considered stochastic textures and described by binarized statistical properties, which are estimated as independent components of the scene. To achieve the goal, the classification algorithm support vector machine was chosen. SVM learns features from the histograms modeling the probability of occurrence of the BSIF descriptor in the texture. Testing was performed on morphed fingerprints, which originated from several fingerprint datasets: Bergdata, Sagem MSO, SecuGen and synthetic. SVM with polynomial kernel function proved to be the most optimal detector with an accuracy of 98 % on the Bergdata sensor dataset, 98.5 % on the Sagem MSO, 94.97 % on the SecuGen and 100 % on the synthetic fingerprints. Detection accuracy on the adaptive morphing method from my project practice is 98.85 % on the Bergdata sensor dataset, 98.49 % on the Sagem MSO, 94.97 % on the SecuGen and 100 % on the synthetic fingerprints. The improved merphing method from my project practice has not shown an effect on preventing morphing detection, although it generates stronger double identities.

Kľúčové slová

morphing odtlačkov prstov, bsif rysy, nezávislosť, deskriptor textúry, detekcia morphingu

Keywords

morphing of fingerprints, bsif features, independance, texture descriptor, detection of morphing

Citácia

DOVIČIC, Denis. *Odhalenie morphovaných odtlačkov prstov*. Brno, 2022. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Ondřej Kanich, Ph.D.

Odhalenie morphovaných odtlačkov prstov

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána Ing. Ondřeje Kanicha, Ph.D. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Denis Dovičic
17. mája 2022

Poďakovanie

Chcel by som tu poďakovať svojmu vedúcemu Ing. Ondřejovi Kanichovi, Ph.D. za odbornú pomoc, usmerňujúce rady a čas, ktorý mi venoval pri vypracovávaní práce. Tiež by som chcel poďakovať svojej rodine za podporu.

Obsah

Kapitola 1

Úvod

Používanie identifikácie odtlačkov prstov bolo istú dobu oveľa spoľahlivejšie ako iné druhy populárnych metód osobnej identifikácie založených na podpise, tvári a reči [1]. Aj keď sa overovanie odtlačkov prstov obvykle spája s trestnou činnosťou a policajnou prácou, v súčasnosti sa stalo populárnym využitie tejto formy verifikácie aj v bežných aplikáciách, ako sú kontrola prístupu do zariadenia, finančná bezpečnosť a iné. [2]

Široké rozvinutie biometrického rozpoznávania vyvolalo niekoľko obáv týkajúcich sa ich bezpečnosti. Jednou z hrozieb sú útoky prostredníctvom morfovacích techník. Ide o skombinovanie vlastností viacerých odtlačkov, či iných identifikačných prostriedkov, do jedného výsledného umelého obrázku. Tieto novo vzniknuté obrázky sú potom pozitívne priradené k viacerým subjektom, oproti ktorým prebieha overenie totožnosti (napríklad hľadaní zločinci môžu použiť autentický pas na vstup do krajiny s falošnou totožnosťou). [3]

Táto práca skúma, v náväznosti na moju bakalársku prácu [4] a projektovú prax [5], automatizovanú detekciu takto vytvorených morphovaných odtlačkov, bez toho, aby bol potrebný ľudský faktor. Doposiaľ existuje len jedna morphovacia technika odtlačkov prstov, použitím reznej línie, zo štúdie [6] a mojej bakalárskej práce [4], a jej modifikácia zabezpečujúca efektívnejšiu evaluáciu reznej línie z mojej projektovej praxe [5]. Na druhú stranu však neexistuje žiadna verejná práca, ktorá by sa venovala detekcií morphovaných odtlačkov prstov, takže táto diplomová práca bude úplne prvá, skúmajúca možnosti detekcie morphingu. Keďže morphing je možné vykonávať nad obrázkami reálnych odtlačkov, tradičné algoritmy pre metódy detekcie falzifikátov či syntetických odtlačkov, ako absencia pórov, pravidelnosť gradientu či papilárnych línií a zmeny v šedotónových hodnotách, v tomto prípade nebude možné použiť.

Kapitola 2 sa zaoberá všeobecným popisom odtlačku prsta, popisom odtlačku z biologického hľadiska, jeho využitiu v biometrických systémoch, jeho vlastnosťami používaných k identifikácií a tzv. morphingom odtlačkov prstov. V kapitole 3 sa podrobne zoznámime so spracovávaním textúr a popisom textúry deskriptormi, ktoré budú využité v kapitole 5. Kapitola 4 obsahuje úvod do strojového učenia a základné koncepty. Nasledujúca kapitola 5 obsahuje návrh spracovania odtlačkov, morphing odtlačkov a návrh detekcie morphovaných odtlačkov prstov s využitím deskriptorov textúr k efektívnejšej klasifikácií morphovaných a obyčajných odtlačkov prstov. Práca ďalej pokračuje implementačnými detailami navrhnutého riešenia v kapitole 6 a vyhodnotením výsledkov získaných pomocou implementovaných skriptov v kapitole 7. Posledná kapitola 8 obsahuje záver.

Kapitola 2

Odtlačok prsta

Odtlačok prsta je vzor, ktorý je zanechaný na objekte po dotyku ľudským končekom prsta. Za zanechanie vzoru na povrchu po dotyku môže vlhkosť a mastnota na prste. Okrem toho je odtlačky možné aj odobrať atramentom, alebo inými látkami nanesením na povrch končeku prsta a následne odtlačením na hladký povrch, napríklad papier. [4] [7] [8]

Ľudské odtlačky prstov sú detailné, takmer jedinečné, len málo sa menia a sú trvanlivé po celý život jednotlivca, vďaka čomu sú vhodné ako dlhodobé identifikátory ľudskej identity. Môže ich zaznamenávať polícia alebo iné orgány na identifikáciu jednotlivcov, ktorí zatajujú svoju totožnosť, alebo na identifikáciu osôb, ktoré sú nespôsobilé alebo mŕtve, a teda sa nemôžu identifikovať. [7] [9]

V tejto kapitole bude popísaná biometria v rámci spracovávania odtlačkov prstov a faktory, ktoré biometrickú charakteristiku ovplyvňujú (kapitola 2.1), podstata daktyloskopie a jej tri zákony (kapitola 2.2) a všeobecný popis odtlačku prsta, ktorý zahŕňa papilárne línie (kapitola 2.2.1), rozdelenie odtlačkov do určitých tried na základe globálnej alebo lokálnej úrovne pozorovania (kapitola 2.2.2)

2.1 Úvod do biometrie

Biometria v rámci tejto práce znamená automatizované rozpoznávanie jedincov na základe ich špecifických charakteristík. Je ich možné rozdeliť do dvoch skupín: anatomické charakteristiky a behaviorálne charakteristiky [2] [10]. Medzi fyzické charakteristiky patria črty jedincov, ktoré sú tvorené anatómiou tela, napr. odtlačok prsta, dlane, geometria tváre, sieť žíl na rukách, DNA a podobne. Medzi behaviorálne charakteristiky patrí napríklad chôdza, podpis či hlas. [4] [10]

Hlavnou výhodou biometrických systémov je, že charakteristiku využitú na identifikáciu jedinca nie je možné stratiť alebo zabudnúť (napríklad v porovnaní s občianskym preukazom). Tento fakt je zároveň aj nevýhodou, pretože charakteristiku nie je možné zmeniť. Ďalšou nevýhodou je, že niektoré charakteristiky môžu povedať veľa o zdravotnom stave jedinca a v tom prípade ide o zásah do súkromia. [2]

Množstvo biometrických systémov je používaných v rôznych aplikáciách a pre danú aplikáciu má skúmaná biometrická charakteristika určité výhody a nevýhody. Výber biometrickej charakteristiky teda závisí na zvážení niekoľkých možností, aby bol splnený výkon rozpoznávania. Všeobecne existuje 8 faktorov, ktoré ovplyvňujú výber [4] [10]:

1. **Univerzálnosť** - každý jedinec pristupujúci k aplikácií by mal vlastniť charakteristiku.

2. **Jedinečnosť** - charakteristika by mala byť dostatočne odlišná medzi jednotlivcami, aby nedochádzalo k nepravdivej identifikácii jedinca.

3. **Trvalosť** - charakteristika jedinca by nemala byť závislá na čase, to znamená, že v priebehu času by sa nemala významne meniť. V opačnom prípade by mohlo dôjsť buď k nepravej zhode alebo k nemožnosti identifikovať jedinca.

4. **Merateľnosť** - malo by byť možné získať a digitalizovať biometrickú charakteristiku pomocou vhodných zariadení, ktoré nespôsobujú jednotlivcovi neprimerané nepríjemnosti. Získané nespracované údaje by okrem toho mali byť spracovateľné. Tento faktor významne ovplyvňuje frekvenciu porúch a presnosť rozpoznávania.

5. **Výkon** - okrem presnosti rozpoznávania by výpočtové zdroje potrebné na dosiahnutie tejto presnosti a priepustnosti (počet transakcií, ktoré je možné spracovať za jednotku času) biometrického systému mali spĺňať aj obmedzenia stanovené aplikáciou.

6. **Prijateľnosť** - jedinci pristupujúci k aplikácií by mali byť ochotný predať charakteristiku systému.

7. **Bezpečnosť** - miera, ako ľahko je možné charakteristiku podvrhnúť, alebo sa vyhnúť zhode.

8. **Cena** - mal by byť dodržaný pomer cena/kvalita.

2.2 Daktyloskopia

Podstatu daktyloskopie tvoria vedecké poznatky o fyziologických vlastnostiach kože človeka. Je charakterizovaná ako náuka o obrazcoch papilárnych línií vytvorených na vnútornej strane článkov prstov, na dlaniach, na prstoch na nohách a na chodidlách. [4] [11]

Skúmanie papilárnych línií je podriadené trom fyziologickým zásadám, takzvaným daktyloskopickým zákonom, ktoré spočívajú v individuálnosti papilárnych línií, ich relatívnej nemennosti a relatívnej neodstrániteľnosti [4] [11]:

- **Neopakovateľnosť** - na zemi s vysokou pravdepodobnosťou neexistujú žiadny dvaja jedinci, ktorý majú úplne zhodné obrazce papilárnych línií. Sir Francis Galton zistil, že pravdepodobnosť dvoch zhodných odtlačkov je 1 ku 64 miliard [2].

- **Nemennosť** - po celý život jedinca sa obrazce, tvorené papilárnymi líniami, nemenia.

- **Neodstrániteľnosť** - papilárne línie sú neodstrániteľné za predpokladu, že nie je poškodená nižšia vrstva kože (podrobnejšie prebrané v kapitole 2.2.1).

2.2.1 Papilárne línie

Odtlačok prstu je jeden z biometrických charakteristík človeka. Jedná sa o unikátnu priestorovú kresbu vyvýšením povrchovej štruktúry pokožky, tzv. papilárnych línií. Nachádzajú sa na vnútornej strane článkov prstov, na dlaniach, na prstoch nôh a chodidiel. Na ich vrchole sa nachádzajú nervové zakončenia hmatových nervov a vyústenia potných žliaz. Výška vyvýšených reliéfov je zhruba 0,1-0,4 mm a šírka cca 0,2-0,7 mm [9] [11]. Na iných častiach povrchu ľudského tela papilárne línie nie sú vytvorené. [4]

Obrázok 2.1. Ukážka štruktúry pokožky papilárnych línií. [12]

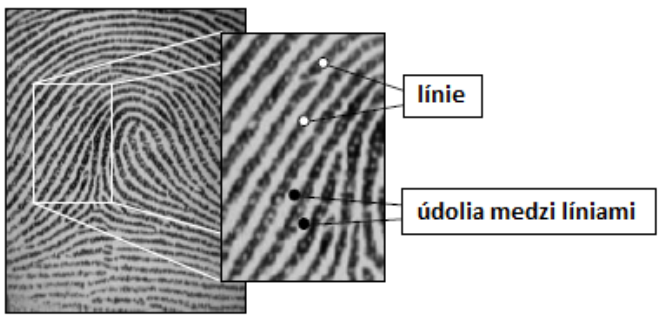
Významnou skutočnosťou je, že papilárne línie nemajú žiadne iné živočíšne tvory okrem primátov a nález odtlačkov a stôp preto skoro jednoznačne svädčí o ľudskom pôvode.[4] [9] [10]

Papilárne línie vznikajú v nižšej vrstve kože z drobných výbežkov, papilár. V tejto nižšej vrstve, derme, sú takisto aj preddefinované ich zakrivenia. Výsledné obrazce, ktoré sa nazývajú dermatoglyfy, sú teda len projekciou z nižšej vrstvy kože (ako je ukázané na obrázku 2.1) a v tom prípade majú schopnosť regenerácie pri ľahkom poškodení. Ak v prípade hlbšieho zranenia dôjde k poškodeniu vrstvy dermis, papilárne línie sa na tomto mieste už nikdy neobnovia. [4] [10] [12]

Ďalšou významnou vlastnosťou dermatoglyfov je to, že ich na prvý pohľad hladké zakrivenia obsahujú významné detaily pre daktyloskopiu. Sú to rôzne prekríženia a prerušenia papilárnych línií, čo vytvára charakteristické znaky odtlačku prsta, takzvané markanty (bude bližšie popísané v kapitole 2.2.3). [4] [10] [12]

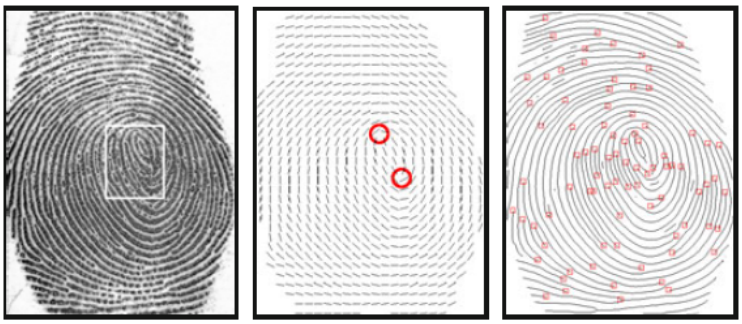
2.2.2 Klasifikácia odtlačkov prstov

Na obrázku odtlačkov prstov sú obvykle línie tmavé, zatiaľ čo údolia medzi líniami svetlé (obrázok 2.2). Línie a údolia majú väčšinou paralelný priebeh, avšak niekedy sa rozdvojujú a niekedy sa ukončujú. Pri analýze na globálnej úrovni má vzor odtlačkov prstov jednu alebo viac oblastí, v ktorých čiary hrebeňa vykazujú charakteristické tvary (vyznačujúce sa vysokým zakrivením, častým ukončením atď.). Tieto regióny (nazývané singularity) možno klasifikovať do troch typológií: slučka, delta a závit (bude bližšie prebrané v nasledujúcich častiach) [8] [9] [11]. Singulárne oblasti zapadajúce do kategórií slučiek, delta a závit sa typicky označujú ako $\cap$, Δ a O. Niekedy nie sú výslovne uvádzané vlastnosti typu závit, pretože typ závit sa dá opísať ako „singularita“ s dvoma čelnými slučkami [9] [11]. Ďalej v práci sa bude uvádzať už len typ jadro (zahŕňa závit a slučku) a typ delta. [4]



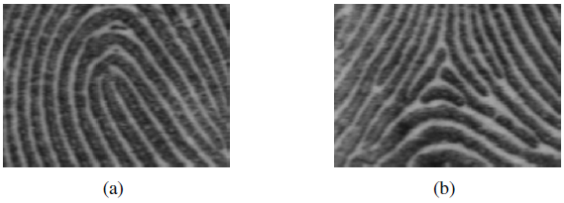
Obrázok 2.2. Línie a údolia medzi líniami. [13]

Detaily odtlačkov prstov je možné rozdeliť do dvoch rôznych skupín podľa spôsobu rozpoznávania. Prvou skupinou je rozdelenie podľa dermatoglyfov, obrazcov, ktoré odtlačky vytvárajú. V tejto skupine sú odtlačku reprezentované mapou lokálnych orientácií ako je ukázané na obrázku 2.3 (b). Mapa zvyčajne obsahuje miesta kde sa lokálne orientácie náhle menia a vytvárajú tak špeciálne úseky zvané singulárne body [8] [9] [11]. Druhou skupinou sú prerušenia a prekríženia papilárnych línií, takzvané markanty (obrázok 2.3 (c)), ktoré budú bližšie popísané v kapitole 2.2.3. [4] [8]



Obrázok 2.3. Úrovne detailov odtlačku. [10]

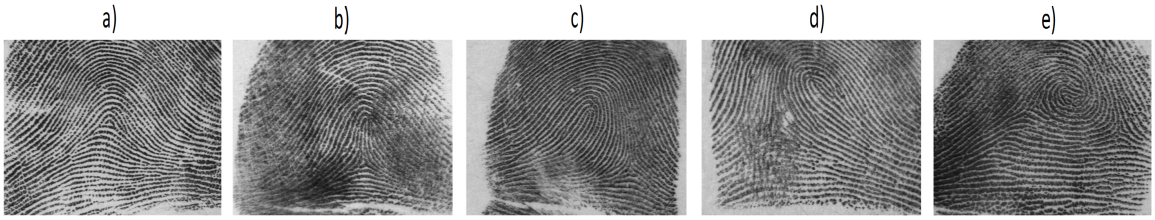
Teraz si popíšeme vyššie uvedené singulárne body - jadro a deltu. Oba z bodov sú vizuálne charakteristické. Jadro, tiež nazývané aj slučka, tvoria papilárne línie ktoré vstupujú z jedného smeru a vrátia sa tým istým (obrázok 2.4 (a)). Táto singularita sa môže využiť ako orientačný bod na zarovnanie odtlačku. Deltu tvoria tri papilárne línie, ktoré sa zbiehajú na jedno miesto a vytvárajú tak vzor trojuholníka (obrázok 2.4 (b)). [4] [8] [9]



Obrázok 2.4. Singulárne body: (a) slučka, (b) delta. [10]

Súbor singulárnych bodov, je možné vnímať ako abstraktnú mapu orientácií, pretože od ich množstva a umiestnenia je možné predpovedať zvyšok odtlačku. Väčšina klasifikačných algoritmov vychádza z Galton-Henryovej klasifikačnej schémy [14] a teda existuje päť najbežnejších tried, do ktorých je možné rozdeliť odtlačky podľa singulárnych bodov (obrázok 2.5) [4]:

1. **Oblúk** - neobsahuje ani deltu ani slučku.
2. **Vypuklý oblúk** - jadro je priamo nad deltou.
3. **Ľavá slučka** - jadro je orientované na ľavú stranu a delta sa nachádza napravo.
4. **Pravá slučka** - jadro je orientované na pravú stranu a delta sa nachádza naľavo.
5. **Závit** - obsahuje dve delty a dve slučky.



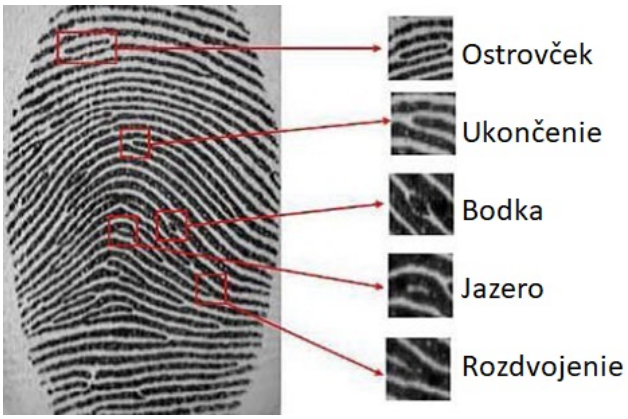
Obrázok 2.5. Príklad bežných 5 tried odtlačkov prstov: a) oblúk, b) stanový oblúk, c) ľavá slučka, d) pravá slučka e) závit. [15]

Rozdelenie tried nie je dostatočné, pretože by dochádzalo k falošným zhodám v identifikácií jedinca. Charakteristika, ktorá dokáže dostatočne detailne rozlíšiť odtlačky a vedie k správnej identifikácií, sa nazýva markanty odtlačku prstu (kapitola 2.2.3). [4]

2.2.3 Markanty

Markanty odtlačku prsta sú drobné detaily tvoriace špecifické znaky papilárnych línií, pomocou ktorých je možné jednoznačne identifikovať jedinca. Práve na základe detekcie markantov je založená väčšina rozpoznávacích metód alebo algoritmov odtlačkov prstov. Ide o spôsoby narušenia kontinuality papilárnych línií, kedy na nich dochádza napríklad k náhlemu ukončeniu, k môžnému rozdeleniu na dve atď. Sir Francis Galton (1822 - 1911) bol prvým, kto kategorizoval markanty a pozoroval, že počas života jednotlivca zostávajú nezmenené. Markanty sa na jeho počesť niekedy nazývajú „Galtonove detaily“. [4] [8]

Pre identifikáciu markantov sú zanedbateľné geometrické a rozmerové hodnoty línií. Podstatnými sú len polohy na odtlačku, kde sa línia vynára, končí, rozdeľuje alebo sa spája s inou. Okrem miesta, kde sa charakteristika nachádza, má ešte dve ďalšie vlastnosti: smer a typ [8] [10]. Smer markantu je určený lokálnou orientáciou papilárnej línie, na ktorej sa nachádza. Typ markantu je definovaný miestami, kde sa línie krížia alebo končia, čo výtvárá niekoľko druhov čŕt. Patria sem: zakončenia, ostrovček, bodka, rozdvojenie, jazero, háčik, most a prekríženie. Niektoré typy sú zobrazené na obrázku 2.6. [4] [8] [16]



Obrázok 2.6. Ukážka niekoľkých markantov odtlačkov prsta. [16]

Na rozoznanie odtlačku prsta sú využívané len 2 typy markantov, postačujúce na jednoznačnú identifikáciu a to rozdvojenie a zakončenie. Pri snahe analyzovať všetky typy markantov by sa zbytočne výrazne predĺžilo celkové spracovanie odtlačku, pretože popisovanými dvoma typmi je možné všetky ostatné nahradiť. [4] [2] [10]

2.2.4 Charakteristiky markantov

Pre presnejšiu verifikáciu pomocou odtlačkov prstov sú okrem typu markantu zaznamenávané aj ďalšie charakteristiky a to poloha markantu a smer markantu. Poloha markantu predstavuje súradnicu $[x, y]$ obrázku odtlačku na ktorej bol markant detekovaný. Smer markantu je odvodený zo smeru papilárnej línie. Pre výpočet smeru markantu je potrebné zvážiť dve možné varianty, pretože rozlišujeme dva rôzne typy markantov (zakončenie a rozdvojenie). Pre zakončenie línií je smer markantu definovaný, ako smer postupujúci zo zakončenia línie. V prípade rozdvojenie je smer markantu definovaný ako priemerná hodnota smeru dvoch línií, ktoré postupujú smerom z rozdvojenia ďalej. [4] [17]

2.3 Morphing

Experimenty na Západnej virgínskej univerzite (WVU) ukazujú, že morphing odtlačkov prstov má niekoľko výhod [4] [18]:

- Možno ho použiť na generovanie virtuálnej identity z dvoch rôznych prstov.
- Možno ho použiť na zakrytie informácií prítomných na snímke odtlačkov prstov jednotlivca, pred ich uložením do centrálnej databázy.
- Môže sa použiť na vytvorenie potlačiteľnej šablóny odtlačkov prstov, t. j. šablóna sa môže resetovať, ak je zmiešaný odtlačok prstov ohrozený.

Morphovať odtlačky je taktiež možné niekoľkými spôsobmi. Prvou možnosťou je kombinovať na úrovni detailov, kedy sa proces začína extrakciou lokálnych orientácií, frekvencií a markantov z pôvodných odtlačkov prstov a následne sa vygeneruje syntetický obrázok odtlačku. Druhou možnosťou je skombinovať odtlačky na úrovni obrázku a to priamym spájaním častí pôvodných odtlačkov. V oboch prípadoch však morphing odtlačkov prstov očakáva nasledujúce požiadavky [4] [6]:

- Vlastnosti odtlačkov (distribúcia orientácií, priestorovo-frekvenčná doména, markanty), by mali byť skombinované tak, aby algoritmus rozpoznávania odtlačkov prstov priradil výsledný odtlačok prsta k obidvom subjektom.
- Vizuálne by mal byť realistický (tj. bez zjavných artefaktov), a tak aby ľudské oko nerozpoznalo že ide o umelo vytvorený falzifikát.

K morphingu sa pristupuje analytickým vyhodnotením najoptimálnejšej reznej línie, t. j. línie rozdeľujúcej dva rôzne odtlačky tak, že v mieste rezu majú papilárne línie odtlačkov, čo najpodobnejšie orientácie, frekvencie a na oboch stranách je dostatočný počet markantov. V prvom rade sú teda jednotlivé odtlačky zanalyzované a potrebné prvky k morfovaniu extrahované, t. j. distribúcia orientácií, priestorovo-frekvenčná doména a markanty. Následne sú odtlačky zarovnané na základe ich lokálnych orientácií s minimálnym prekrytím určeným prahovou hodnotou min_{vr} (v percentách). Pre zarovnané odtlačky sa odhadne optimálna línia rezu, rozdeľujúca zarovnané odtlačky na pozitívne a negatívne strany, a v jej okolí (okolie je dané maximálnou vzdialenosťou od reznej línie) sú spriemerované orientácie a frekvencie. Orientácie a frekvencie mimo okolia reznej línie (na oboch stranách línie je vždy na výber z dvoch, keďže rezná línia je vedená cez dva zarovnané odtlačky) sú zvolené

na základe početnosti markantov jednoduchým pravidlom, väčší počet vyhráva. Teda pre dané dva odtlačky je teda nutné vykonať nasledujúce kroky [6]:

1. Zistiť podobnosť orientačných polí odtlačkov a čo najlepšie ich zarovnať.
2. Odhadnúť optimálnu reznú líniu odvodenú z podobnosti odtlačkov.
3. Vygenerovať šablónu odvodenú z optimálnej reznej línie odtlačkov.
4. Vygenerovať morphovaný odtlačok z predpripravenej šablóny.

Tento postup je znázornený na obrázku 2.7.

Obrázok 2.7. Ukážka morphingu dvoch odtlačkov prstov. [4]

Kapitola 3

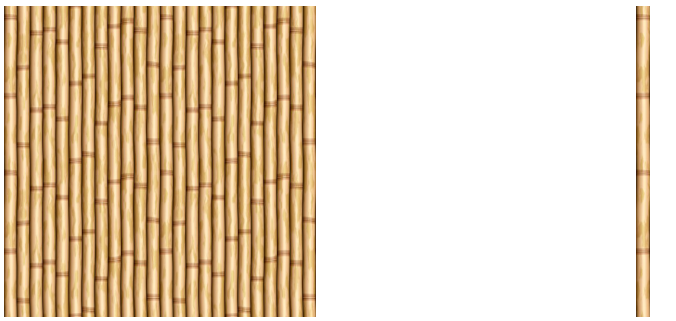
Analýza textúr

Textúra určitého objektu je rozpoznateľná hmatom alebo zrakom. Hmat sa vzťahuje na štruktúru povrchu materiálu, z ktorého je objekt tvorený (napr. hladký, drsný, kovový) a vizuálna textúra sa vzťahuje na pozorovaný tvar (napr. okrúhly, hranatý) či obsah daného objektu (napr. farba, vzor). V počítačovom videní je nutné sa vysporiadať s rôznymi štrukturálnymi charakteristikami obrazu alebo videa. Textúra je jednou z hlavných charakteristík tohto druhu, ktoré sa používajú na identifikáciu objektov alebo oblastí záujmu. [19]

V tejto kapitole budú popísane vizuálne textúry z pohľadu počítačového videnia a ich rozdiel oproti textúram v grafike. Pozrieme sa na ich základné rozdelenie v kapitole 3.1 a na metódy štatistickej, štrukturálnej analýzy textúr a analýzy na základe transformácie bežne používaných v praxi, v kapitolách 3.2, 3.3 a 3.4.

3.1 Textúry v počítačovom videní

V počítačovom videní môže byť textúra definovaná ako funkcia priestorovej zmeny intenzity jasu pixelov. Textúra predstavuje práve tieto zmeny, ktoré určujú charakteristiky ako hladkosť, drsnosť a pravidelnosť povrchu. Obrázky textúr v počítačovom videní sa vzťahujú na obrázky, v ktorých sa špecifický vzor rozloženia textúry postupne opakuje v celom obraze. Nižšie na obrázku 3.1 je znázornená textúra s opakujúcim sa špecifickým vzorom. [19] [20]



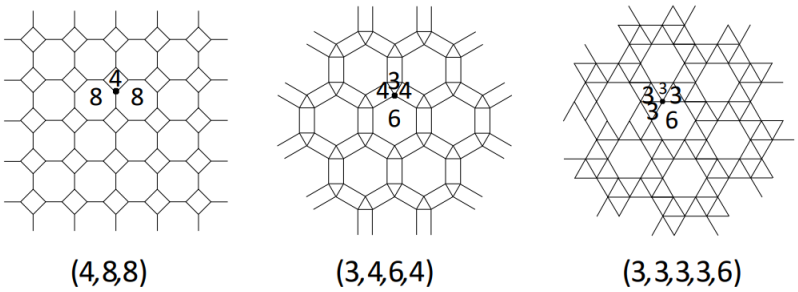
(a) (b)

Obrázok 3.1. Ukážka obrázku textúry: (a) Obrázok textúry. (b) Opakujúci sa vzor.

Textúry v počítačovej grafike predstavujú potlač 3D objektov, kde primitívu textúry texel predstavuje jeden pixel, ktorého farba je daná osvetlením, tieňovaním a hodnotou

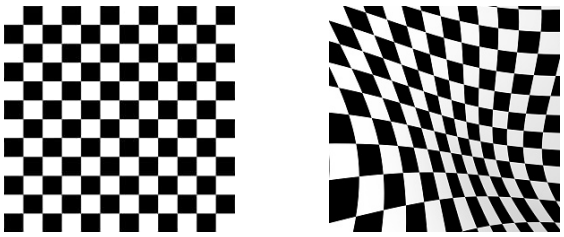
textúry namapovanej na pozíciu, ktorú pixel predstavuje. V počítačovom videní primitívou textúry je texton. [19] [21]

- **Pravidelné textúry:** pravidelné textúry s dobre definovanou štruktúrou, ktorých textony sú ľahko rozlíšiteľné. Maju teda dobre definované primitívy (mikrotextúra) a zároveň hierarchiu priestorových usporiadaní primitív (makrotextúra). Môžu byť reprezentované napríklad sieťou polygónov (viď obrázok 3.2).



Obrázok 3.2. Textony reprezentované ako sieť polygónov, kde každý texton definuje počet susediacich polygónov.

- **Čiastočne pravidelné textúry:** textúry obsahujúce nízku úroveň nepravidelnosti, ale zároveň stále definujúce istú štruktúru (viď obrázok 3.3).



(a) (b)

Obrázok 3.3. (a) Pravidelná štruktúra. (b) Čiastočne pravidelná textúra.

- **Stochastické textúry:** úplne náhodné textúry, v ktorých je náročné rozpoznať štruktúru a aj samotné textony. Reprezentované sú nepriamo prostredníctvom nedeterministických vlastností, ktoré riadia distribúcie a vzťahy medzi úrovňami šedej v obraze (nutná transformácia obrazu do úrovní šedej). Patria sem najmä textúry z reálneho života ako napríklad, kôra stromov, tráva, oblaky atď. (viď obrázok 3.4)

(a) (b)

Obrázok 3.4. Na obrázkoch (a) aj (b) je možné vidieť stochastickú textúru.

3.1.1 Analýza textúr

Pre strojové porozumenie textúram je nutné jednotlivé textony textúry popísať vektormi vlastností. Popis textúry umožňuje priradenie neznámej vzorky textúr z obrázka do akejkoľvek preddefinovanej triedy textúr. Toto je dôvod zavedenia analýzy textúr k extrakcií ich vlastností a naučeniu sa tak samotnej štruktúry textúry a popisu jej textonov. Samotná analýza čelí dvom zásadným problémom, ktoré pôsobia na jej výstup negatívnym vplyvom [19]:

- **Rotácia textúry.**
- **Šum v textúre.**

Tieto problémy v analýze textúr môžu mať rôzne deštruktívne vplyvy. Ak teda na textúry aplikujeme analytické metódy a následne textúru rotujeme alebo pridáme šum, analýza nebude invariantná, tzn. výstup analýzy bude odlišný od pôvodnej bez rotácie či šumu. V praxi môže byť úroveň výkonu procesu analýz ešte výrazne nižšia. Vždy chceme, aby bol proces analýzy a kategorizácie obrázkov robustný a stabilný a zároveň neutralizoval účinok týchto negatívnych vplyvov. [19]

Taktiež môžu existovať rôzne situácie, kedy sa obrázky navzájom líšia, v mierke (zväčšenie/zmešenie), uhle pohľadu, jase alebo intenzite svetla. Formálne to spôsobuje problémy v klasifikácii textúr. Na zníženie vplyvov týchto problémov boli zavedené rôzne metódy založené na učení sa rotačne či šumovo invariantných textonov, alebo na priamom popise textúry funkciou. Rozdelenie metód a ich príklady je možné vidieť v nižsie uvedenej tabuľke 3.1. [19] [22]

Typ	Metódy
Štatistické	vlastnosti histogramov lokálne binárne deskriptory založené na registrácií obrazu matice spoluvýskytu zákony o energii textúr
Štrukturálne	merania primitív vlastnosti okrajov reprezentácia kostry morfologické operácie transformácie vlastností invariantne voči zväčšeniu
Založené na modeli	autoregresívne modely fraktálne modely modely náhodných polí textúrne modely
Založené na transformácii	spektrálne transformácie gáborove transformácie vlnkové transformácie krivkové transformácia

Tabuľka 3.1: Tabuľa rozdelenia typov metód analýz/klasifikácií textúr a ich príklady.

V počítačovom videní neexistuje všeobecná definícia textúry, väčšina doposiaľ vyvinutých techník pristupuje k textúre hlavne zo štrukturálneho alebo štatistického hľadiska.

V štatistických prístupoch je textúra opísaná priestorovým rozložením pixelov v obraze, zatiaľ čo v štrukturálnych prístupoch je textúra definovaná ako relačné usporiadanie textúrových primitív, no oba prístupy často predpokladajú obraz v úrovniach šedej. [19] [23]

V tejto práci sú bližšie obsiahnuté vybrané metódy štatistickej a štrukturálnej analýzy, ktoré v každom prípade očakávajú na vstupe obraz transformovaný do úrovní šedej, v kapitolách 3.2 a 3.4.

3.2 Analýza štatistickými vlastnosťami

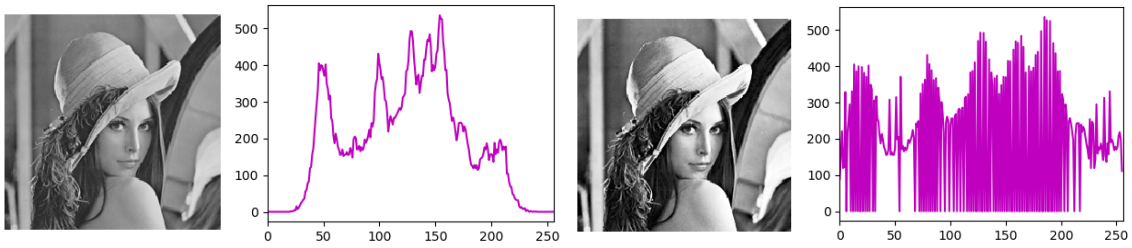
Metódy štatistického spracovania textúr tvoria mnohé z metód prezentovaných v oblasti počítačového videnia. Tieto metódy na analýzu textúry obrázkov vykonávajú sériu štatistických výpočtov rozloženia intenzity svetla pixelov. Všeobecnosti do tejto skupiny patria metódy používané na odvodenie vektora vlastností. Medzi tieto metódy patria štatistické charakteristiky [19]:

- **Prvého rádu:** špecifikácia jedného pixelu je spočítaná funkciou nezávislou na akomkoľvek inom pixeli v obraze.
- **Druhého rádu:** špecifikácia jedného pixelu je spočítaná funkciou závislou na inom pixeli v obraze.
- **Vyššieho rádu:** špecifikácia jedného pixelu je spočítaná funkciou závislou na viacerých pixelov v obraze.

V nasledujúcich kapitolách budú popísané vybrané štatistické metódy analýzy obrazu.

3.2.1 Histogramové vlastnoti

Jedná sa o štatistickú metódu prvého rádu, kde extrahovaný vektor vlastností je odvodený z hodnôt pixelov nezávisle medzi sebou. Jeden štatistický ukazateľ teda odpovedá hodnote úrovne šedej jedného pixelu v obraze, tzn. medzi pixelmi neexistuje priestorová závislosť. Histogram šedotónového obrazu je dvojrozmerný graf znázorňujúci zastúpenie jednotlivých hodnôt šedej a teda umožňuje grafické znázornenie množstva svetla a tmavosti. V prípade histogramu farebného obrazu je možné vidieť zastúpenie farieb, poprípade je možné na histogram použiť rôzne operácie ako napríklad jeho ekvalizácia (roztiahnutie hodnôt pixelov do plného rozsahu – zvýšenie kontrastu obrazu). Ukážku histogramu a operácie ekvalizácie je možné vidieť na obrázku 3.5. [24]



(a) Pôvodný histogram.

(b) Ekvalizovaný histogram.

Obrázok 3.5. Ukážka ekvalizácie histogramu (zvýšenie kontrastu). V oboch grafoch predstavuje os x úroveň šedej v rozsahu $\langle 0, 256)$ a os y početnosť pixelov danej úrovne šedej.

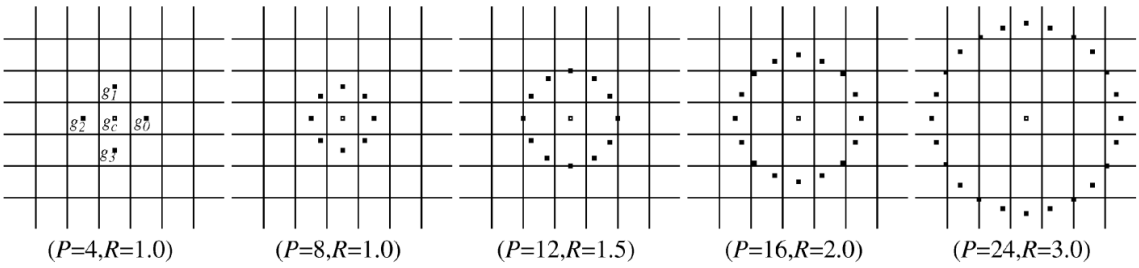
3.2.2 Lokálny binárny vzor

Lokálny binárny vzor (ďalej len LBP, odvodené z angl. local binary pattern) je jedným z deskriptorov textúry, ktorý definuje lokálnu priestorovú štruktúru a lokálny kontrast obrazu. Jedná sa o štatistickú metódu minimálne druhého rádu (bude vysvetlené nižšie). LBP je motóda s jednoduchou implementáciou a extrakciou správnych vlastností s veľmi presnou klasifikáciou. Poprednou vlastnosťou tejto metódy je jej invariantnosť v monotónnych transformáciách stupňou šedej (ak je pri transformácií zachované poradie pixelov) a v zmenách rotácie textúry. [19] [25]

Pre potrebu tejto metódy je nutné najprv previesť textúru $\mathbf{T}$ do stupňov šedej. Ńásledne je možné pre každý pixel textúry $\mathbf{g}_c$ skúmať jeho lokálne okolie úrovní šedej do určitej vzdialenosti definovanou rádiusom R, kde $R > 0$. Lokálne okolie pixelu $\mathbf{g}_c$ je možné previesť do spojitého rozloženia úrovní šedej P okolitých pixelov, kde $P > 1$, nasledovne [19] [25]:

$$\mathbf{T} = t(\mathbf{g}_c, \mathbf{g}_0, \mathbf{g}_1, \mathbf{g}_2, ..., \mathbf{g}_{P-1}), \tag{1}$$

kde $\mathbf{g}_c$ je úroveň šedej stredového pixelu a $\mathbf{g}_p$ sú úrovne šedej okolitých pixelov rádiusu R. Na obrázku 3.6 je možné vidieť znázornenie výberu pixelov pre určité počty P analyzovaných susedných úrovní šedej a veľkosťou spracovávaného okolia R. Potrebné šedotónové hodnoty okolia, ktorých umietnenia neodpovedajú priamo súradniciam pixelov sú odhadnuté interpoláciou. [19] [25]



Obrázok 3.6. Kruhové susedstvá stredových pixelov g_c, pre rôzne počty šedotónových hodnôt P a rôzne veľkosti rádiusu R. [19]

Ďalším krokom k dosiahnutia invariancie voči zmenám úrovne šedej je odčítanie od každej susednej šedotónovej hodnoty, hodnotu úrovne šedej stredového pixelu $\mathbf{g}_c$. Potom rozloženie bude mať tvar [19] [25]:

$$\mathbf{T} = t(\mathbf{g}_0 - \mathbf{g}_c, \mathbf{g}_1 - \mathbf{g}_c, \mathbf{g}_2 - \mathbf{g}_c, ..., \mathbf{g}_{P-1} - \mathbf{g}_c), \tag{2}$$

ktoré je následne potrebné previesť do binárnej podoby zavedením prahovej funkcie f s prahovou hodnotou nula. [19] [25]:

$$\mathbf{T} = t(f(\mathbf{g}_0 - \mathbf{g}_c), f(\mathbf{g}_1 - \mathbf{g}_c), f(\mathbf{g}_2 - \mathbf{g}_c), ..., f(\mathbf{g}_{P-1} - \mathbf{g}_c)), \tag{3}$$

kde

$$f(x) = \begin{cases} 1, & x \geq 0, \\ 0, & inak. \end{cases} \tag{4}$$

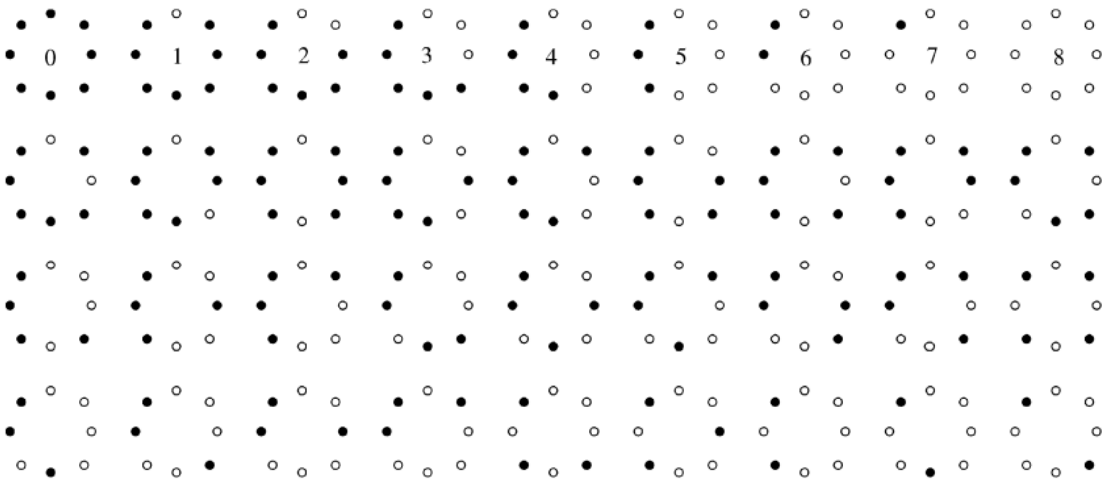
Takto zavedená klasifikácia okolitých stupňov šedej vytvára 2^P rôznych binárnych vzorov tvorených P pixelmi. V prípade otočenia textúry sa budú okolité šedotónové hodnoty, odpovedajúce pixelu so šedotónovou hodnotou $\mathbf{g}_c$, pohybovať po obvode kruhu definovanom rádiusom R a premené $\mathbf{g}_p$ z rovnice 2 budú prirodzene nadobúdať iné hodnoty. K vyriešeniu tohto rotačne variantného problému sa zavádza pridanie binomického faktora 2^p ku každému znaku s indexom p v generovanom vzore, čim je zaručené získavanie jedinečných vzorov na základe rozloženia binarizovaných hodnôt produkovaných funkciou f [19] [25]:

$$\mathbf{F}_{unique} = \sum_{p=0}^{P-1} f(\mathbf{g}_p - \mathbf{g}_c)2^p \tag{5}$$

a

$$LBP = \min\left\{Shift(\mathbf{F}_{unique}(x), i) | i = 0, ..., P-1\right\}, \tag{6}$$

kde funkcia $Shift$ zabezpečuje bitové posunutie z prava hodnoty $\mathbf{F}_{unique}$ toľko krát, koľko šedotóvých hodnôt sa na obvode rádiusu R nachádza. Z tejto množiny sa následne vyberie najmenšia hodnota, čím je zabezpečená invariantnosť voči rotácií textúry (viď obrázok 3.7). [19] [25]



Obrázok 3.7. 36 jedinečných $\mathbf{F}_{unique}$ vzorov generovanych nastavením $P = 8$. V prvom riadku čísla vo vzoroch predstavujú ich jedinečné a rotačne invariantné LBP hodnoty. [25]

3.2.3 Lokálna fázová kvantizácia

Operátor lokálnej fázovej kvantizácie (ďalej len LPQ, odvodené z angl. local phase quantization) bol pôvodne navrhutý na popis textúr a ukázalo sa, že je robustnejší voči rozmazaniu a výkonnejší ako operátor LBP v klasifikácii textúr [26][27]. LPQ sa vo veľkej miere používa na analýzu obrazov alebo textúr signálov v aplikácii, ako je analýza výrazu tváre či rozpoznávanie tváre [26] alebo bol využitý aj pri detekovaní živosti odtlačku prstu [28].

Táto metóda začína krátkou furierovou transformáciou (ďalej len STFT, odvodené z angl. short-term fourier transform) lokálneho okolia (podobne ako v predchádzajúcej kapitole 3.2.2). Pre každý pixel x obrazu $\mathbf{G}$ je STFT odhadnuté na štvorcovej lokálnej oblasti

o rozmeroch $M \times M$ pixelov definujucej susedov $\mathbf{M}_x$ analyzovaného pixelu x umietneného v strede oblsati [26] [27] [28]:

$$\mathbf{G}(\mathbf{u}, x) = \sum_{y \in \mathbf{M}_x} g(x - y) e^{j2\pi \mathbf{u}^T y}, \tag{7}$$

kde $\mathbf{u}$ je frekvencia popísaná vektorom štyroch koeficientov zodpovedajúcim 2D frekvenciám $\mathbf{u}_1 = [a, 0]^T$, $\mathbf{u}_2 = [0, a]^T$, $\mathbf{u}_3 = [a, a]^T$, $\mathbf{u}_4 = [a, -a]^T$ (len vrámci LPQ), kde a je skalárna frekvencia vyjadrená ako $a = \frac{1}{m}$ [28]. Pre každé x obrazu G je výstup v tvare [26] [27] [28]:

$$\mathbf{G}_x = [\mathbf{G}(\mathbf{u}_1, x), \mathbf{G}(\mathbf{u}_2, x), \mathbf{G}(\mathbf{u}_3, x), \mathbf{G}(\mathbf{u}_4, x)] \tag{8}$$

a

$$\mathbf{H}_x = [Re(\mathbf{G}_x), Im(\mathbf{G}_x)]^T, \tag{9}$$

kde $Re(.)$ vráti reálnu časť a $Im(.)$ imaginárnu časť komplexného čísla produkovaného zo STFT. Výsledné LPQ sa potom odhadne len ako súčet binarizovaných hodnôt q_i komplexnej zložky h_1 vektoru H_x vynásobenej binomickým faktorom:

$$LPQ(x) = \sum_{i=1}^{M^2-1} \mathbf{q}_i(x) 2^{i-1}, \tag{10}$$

kde

$$q_i(x) = \begin{cases} 1, & \mathbf{h}_i(x) \geq 0, \\ 0, & inak. \end{cases} \tag{11}$$

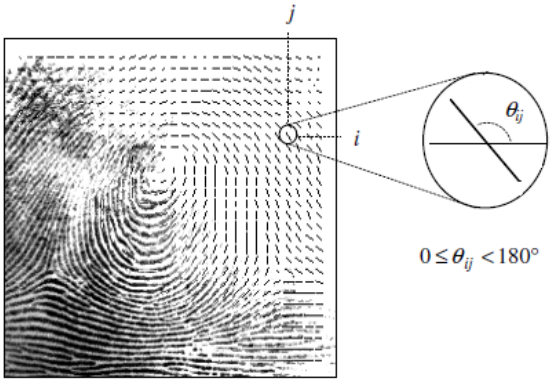
kde v prípade zvolenia $M = 3$ $\mathbf{q}_i(x)$ predstavuje 8 binárnych koeficientov reprezentovateľných celočíselnou hodnotou v rozsahu $\langle 0, 256)$.

Pri kvantovaní je informácia zachovaná v plnom rozsahu, ak sú vzorky, ktoré sa majú kvantovať, štatisticky nezávislé. V prípade obrazu majú susedné pixely vysokú koreláciu, čo má za následok závislosť medzi fourierovými koeficientmi $\mathbf{h}_i$ kvantovanými v LPQ. Avšak tento problém je riešiteľný pridaním dekorelácie k LPQ. [26] [27]

3.2.4 Distribúcia orientácií

Orientácie okrajových hrán sa najprv využívali na segmentáciu obrazu, vyhľadávanie a sledovanie objektov, pričom aplikácia princípu orientácií hrán v obraze je možná aj na klasifikáciu textúr [23] [29] [30]. Klasifikácia textúry je založená na hybridnom štatisticko-štrukturálnom prístupe, kde popis textúry distribúciou orientácií je odhadnutý na úrovni pixelov v obraze, t. j. prvok distribúcie orientácií odpovedá jednému pixelu. [23]

Najrozšírenejším spôsobom výpočtu orientačného poľa je algoritmus najmenších štvorcov (metóda založená na gradiente). Gradient $\boldsymbol{\partial}(x_i, y_j)$ v bode $[x_i, y_j]$ obrázku $\mathbf{I}$, je dvojdimenzionálny vektor $[\boldsymbol{\partial}_x(x_i, y_j), \boldsymbol{\partial}_y(x_i, y_j)]$, kde $\boldsymbol{\partial}_x$ a $\boldsymbol{\partial}_y$ komponenty sú deriváty pre $\mathbf{I}$ v $[x_i, y_j]$ s ohľadom na smer v osi x a y. Odhad hodnôt orientácie línie je založený na vzťahu gradientu medzi susednými pixelami a preto sa jedná o metódu vyššieho rádu. Keďže gradienty obrazu na úrovni pixelov popisujú monotónnosť úrovní šedej, v určitom lokálnom okolí, orientácie hrán $\boldsymbol{\theta}_{xy}$ sú kolmé na stredný fázový uhol zmien týchto hodnôt pixelov. Uhol $\boldsymbol{\theta}_{xy}$ nadobúda iba hodnoty $0° - 180°$ pretože hrany sami o sebe nemajú smer a vo výsledku by lokálna orientácia s uhlom napríklad $80°$ bola identická lokálnej orientácií s uhlom $260°$ (viď obrázok 3.8).[4] [23] [8]



Obrázok 3.8. Obrázok orientačného poľa zodpovedajúceho odtlačku prsta vypočítaného pomocou okna s veľkosťou $\omega \times \omega$. Každý prvok označuje lokálnu orientáciu papilárnych línií odtlačku prsta. [8]

Algoritmus najmenších štvorcov závisí na významných parametroch: 1) Horizontálny a vertikálny operátor gradientu $\boldsymbol{\partial}_x$ a $\boldsymbol{\partial}_y$; 2) veľkosť $\omega \times \omega$ priemerovacích blokov. Pre výpočet lokálnych orientácií je teda v prvom rade potrebné odhadnúť gradienty, v smeroch osí x a y, $\boldsymbol{\partial}_x(x,y)$ a $\boldsymbol{\partial}_y(x,y)$ pre každý pixel obrazu $[x,y]$. Následne sa pre výpočet lokálnych orientácií v blokoch $\omega \times \omega$, centrovaných na pixeloch $[x,y]$ použijú nasledujúce rovnice [8] [23] [31]:

$$\boldsymbol{v}_x(x,y) = \sum_{u=x-\frac{\omega}{2}}^{x+\frac{\omega}{2}} \sum_{u=y-\frac{\omega}{2}}^{y+\frac{\omega}{2}} 2\boldsymbol{\partial}_x(u,v)\boldsymbol{\partial}_y(u,v), \tag{12}$$

$$\boldsymbol{v}_y(x,y) = \sum_{u=x-\frac{\omega}{2}}^{x+\frac{\omega}{2}} \sum_{u=y-\frac{\omega}{2}}^{y+\frac{\omega}{2}} (\boldsymbol{\partial}_x^2(u,v) - \boldsymbol{\partial}_y^2(u,v)), \tag{13}$$

kde $\boldsymbol{v}_x(x,y)$ je fázová zmena v smere osi x a $\boldsymbol{v}_y(x,y)$ je fázová zmena v smere osi y, potom stredný fázový uhol sa bude rovnať [8][31] [32] :

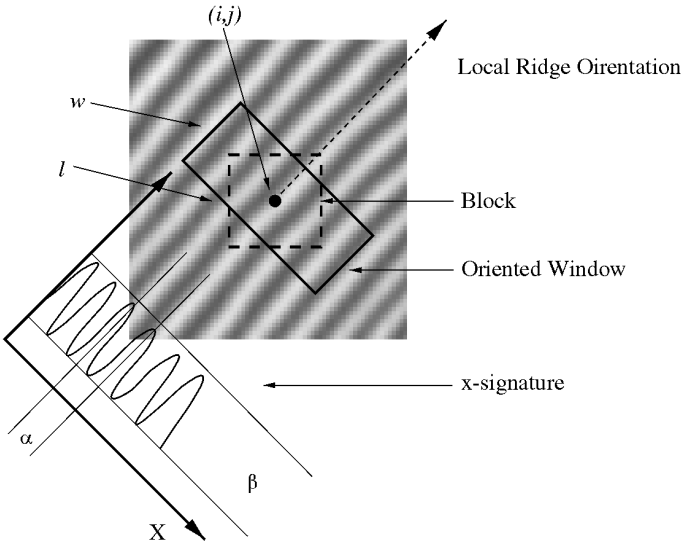
$$\boldsymbol{\phi}(x,y) = \frac{1}{2} tan^{-1}\left(\frac{v_x(x,y)}{v_y(x,y)}\right) \tag{14}$$

a výslednú orientáciu pixelu hrany $\theta(x,y)$ spočítame ako kolmicu na tento fázový uhol zmien $\phi(x,y)$ [8] [31] [32]:

$$\boldsymbol{\theta}(x,y) = 90° + \boldsymbol{\phi}(x,y). \tag{15}$$

3.2.5 Distribúcia priestorovo-frekvenčnej domény

Ďalšou metódou vyššieho rádu analýzy/klasifikácie textúr je štatistická reprezentácia textúry v priestorovo-frekvenčnej doméne, ktorej forma reprezentácie je invariantná voči zmenám v stupňoch šedej a rotácií obrazu. Jedná sa o priestorovo-frekvenčnú doménu z dôvodu počítania frekvencie pre každý pixel $[x,y]$ obrazu $\mathbf{G}$ priestorovým filtrom zo susedstva o veľkosti bloku $2\omega \times \omega$. K extrakcií frekvenčných vzorov je následne možné použiť priestorový filter definujúci prekrývajúcu oblasť obrazu $\mathbf{G}$ ako sínusoidu, z ktorej je extrahovaný frekvenčný vzor, nazývaný x-signatúra, spracovávaného pixelu $[x,y]$ obrazu $\mathbf{G}$ (viď obrázok 3.9). [8] [33] [34]



Obrázok 3.9. Priestorový filter a x-signatúra. [8]

Pre dosiahnutie podobného výsledku je potrebné získať smer hrán vlnenia, pre každý pixel $[x, y]$ obrazu $\mathbf{G}$. K tomu je možné využiť metódu klasifikácie textúr z kapitoly 3.2.4 zabezpečujúcu popis textúry distribúciou orientácií $\boldsymbol{\theta}(x, y)$. X-signatúru $\mathbf{x}[k]; k = 0, ..., 2\omega$ extrahovanú priestorovým filtrom je potom možné vyjadriť ako [8] [35]:

$$\mathbf{x}[k] = \frac{1}{w} \sum_{d=0}^{w-1} \mathbf{G}(u, v), \tag{16}$$

$$u = x + \left(d - \frac{w}{2}\right) \cos\left(\boldsymbol{\theta}(x, y)\right) + \left(k - \frac{1}{2}\right) \sin\left(\boldsymbol{\theta}(x, y)\right), \tag{17}$$

$$v = y + \left(d - \frac{w}{2}\right) \sin\left(\boldsymbol{\theta}(x, y)\right) + \left(\frac{1}{2} - k\right) \cos\left(\boldsymbol{\theta}(x, y)\right). \tag{18}$$

3.2.6 Vlastnosti prírodných scén

Predpokladom využitia prírodných scén na štatistický popis obrazu distribúciou určitých vlastností je, že ľudské oči sú prostredníctvom evolúcie prispôsobované na spracovanie zrakového vstupu. O prírodných scénach teda existuje domnienka, že má podobnú štatistickú štruktúru, na ktorú sú samotné oči prispôsobené. Ľudské oči sa vyvinuli v prostredí výrazne odlišnom od toho dnešného. Pravdepodobne aj obrázky mrakodrapov, áut a iných moderných entít nijako významne neovplyvnili ľudský zrakový vnem. Toto tvrdenie dokazuje aj minimálny vplyv výberu prírodných scén alebo mestských častí na výsledky analýzy. Väčšina obrazových množín zhromaždených na účely analýzy prírodných scén poskytuje veľmi podobné výsledky v štatistickom modelovaní avšak sú zvyčajne úplne odlišné od umelých a náhodne generovaných. [36]

Z týchto dôvodov sa zavádzajú filtre zostavené tak, aby mali na prírodných scénach čo najvyššiu odozvu. Filtre je nutné zostaviť tak, aby pre určité váhy alebo funkcie W (kernel)

bola jeho odozva s na obraze G maximálna [36]:

$$s = \sum_{i,j} \mathbf{W}(i,j)\mathbf{G}(i,j), \tag{19}$$

kde $W(i,j)G(i,j)$ predstavuje operáciu konvolúcie kernelu W vycentrovaného na súradnici $[i,j]$ obrazu G.

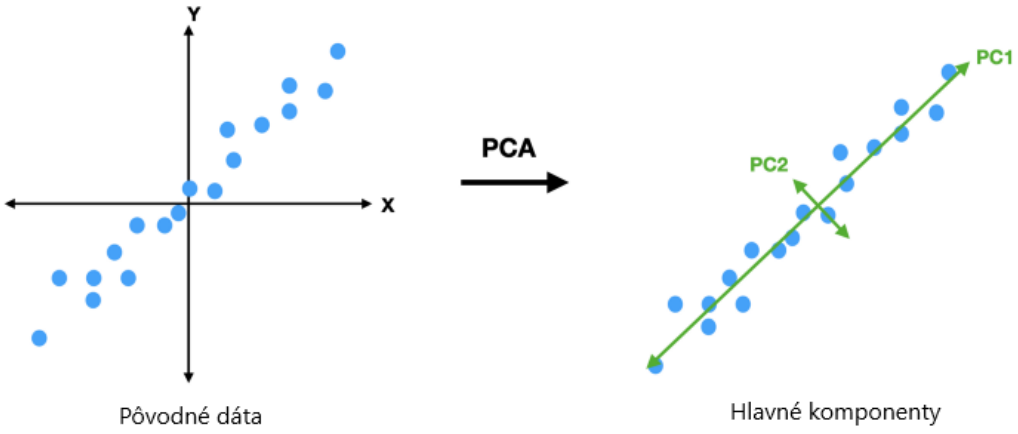
Existuje veľa rôznych funkcií, ktoré možno použiť. Typické možnosti zahŕňajú Fourierove funkcie (mriežky), vlnky (kapitola 3.2.5), Gáborove funkcie (kapitola 3.4.1), funkcie diskrétnej kosínusovej transformácie a mnohé ďalšie. Všetky tieto funkcie majú spoločné to, že sa snažia reprezentovať všetky možné obrazy, nielen prírodné scény (na ktorých boli zostavené), optimálny spôsobom. [36]

3.2.7 Analýza hlavných komponentov

Analýza hlavných komponentov (ďalej len PCA, odvodené z angl. principal component analysis) už neprebieha priamo nad textúrov, ale očakáva určitý typ predspracovania, ako napríklad prevod šedotónovej textúry do histogramu (viď kapitola 3.2.1). Metóda znižuje rozmernosť a redundanciu dát v smere ktorý maximalizuje ich rozptyl, t. j. definuje novú os vo vstupnom priestore, pozdĺž ktorej je rozptyl maximalizovaný (viď obrázok 3.10). Pri použití PCA je dôležitý prevod vstupných dát $\mathbf{x}$ do formy nulového priemeru, t. j. odčítanie priemeru μ od všetkých dát $\mathbf{X}$ [36] [37]:

$$\mathbf{Z} = \mathbf{X} - \mu. \tag{20}$$

To zaisťuje, že stred každej novo odhadnutej osi (stred je v bode nula), zodpovedá priemeru. Tento proces sa nazýva vybielenie dát.



Obrázok 3.10. Ukážka aplikácie PCA na dáta.

Nasledujúci postup PCA je matematicky ekvivalentný rozkladu kovariačnej matice pôvodných dát na vlastné vektory a vlastné hodnoty. Z týchto komponentov je následne možné určiť hlavné zložky (dimenzie) dát. K odhadu kovariačnej matice na vybielených dátach je možné použiť nasledujúci vzťah, v ktorom priemernou hodnotou μ bude stred súradnicovej

sústavy (μ obsahuje len nuly), takže je možné ju vynechať [36] [37]:

$$\mathbf{C} = \frac{1}{N} \sum_{i=0}^{N} \mathbf{z}_i \mathbf{z}_i^T, \tag{21}$$

kde N je počet vektorov v $\mathbf{Z}$. Kovariančná matica $\mathbf{C}$ je štvorcová matica s rozmermi $d \times d$, kde d znamená „dimenzia“ (počet hodnôt jedného vektoru zo $\mathbf{Z}$), a zobrazuje koreláciu medzi jednotlivými dátami. Vlastný vektor je vektor, ktorého smer zostáva nezmenený pri aplikácií lineárnej transformácie a je možné ho získať z rozkladu kovariačnej matice $\mathbf{C}$ spolu s vlastnými hodnotami nasledovne [36] [37]:

$$\mathbf{C} = \mathbf{V}\mathbf{L}\mathbf{V}^T, \tag{22}$$

kde V predstavuje maticu vlastných vektorov (každý stĺpec je jeden vlastný vektor) a $\mathbf{L}$ diagonálnu maticu vlastných hodnôt. Dôležitou informáciou je, že vlastné vektory reprezentujú smer variácie v priestore a vlastné hodnoty jej veľkost.

Teraz sú k dispozícií všetky informácie potrebné k redukcií redundantných dimenzií. Zoradením vlastných vektorov vo $\mathbf{V}$ zostupne podľa vlastných hodnôt z $\mathbf{L}$ a zvolením m prvých vlastných vektorov s najvyššími vlastnými hodnotami je definovaný nový priestor s vyššou variáciou dát než bola v pôvodnom. Transformácia celej množiny $\mathbf{Z}$ do priestoru definovaného práve týmito vlastnými vektormi prebieha nasledovne [36] [37]:

$$\mathbf{z}_{i_{new}} = \left[\mathbf{z}_i \mathbf{V}_0^T, ..., \mathbf{z}_i \mathbf{V}_m^T\right], \tag{23}$$

Počet zvolených vlastných vektorov m je vždy závislý od daného problému a nie je možné ho všeobecne určiť.

3.2.8 Analýza nezávyslých komponentov

Analýza nezávislých komponentov (ďalej len ICA, odvodené z angl. independent component analysis) je ďalšia metóda štatistickej analýzy textúr zabezpečujúca extrakciu skrytých nezávislých komponentov. Napríklad pre dva extrahované nezávislé komponenty $\mathbf{s}_1$ a $\mathbf{s}_2$, t. j. komponent $\mathbf{s}_1$ nesmie mať vplyv na komponent $\mathbf{s}_2$ a opačne, musí existovať faktorizačný prepis spoločnej distribúcie pravdepodobnosti $p(\mathbf{s}_1, \mathbf{s}_2)$ na súčin ich marginálnych distribúcií pravdepodobností [36] [38]:

$$p(\mathbf{s}_1, \mathbf{s}_2) = p(\mathbf{s}_1)\, p(\mathbf{s}_2). \tag{24}$$

Generatívny model v ICA je definovaný lineárnou transformáciou skrytých nezávislých komponentov $\mathbf{s}_i$. Tzn. z pohľadu tejto analýzy obraz vzniká konvolúciou vlastností $\mathbf{A}_i$ a skrytých nezávyslých komponentov $\mathbf{s}_i$, takže generovanie obrazu $\mathbf{G}$ so šedotónovými honotami môžme zapísať ako [36] [38]:

$$\mathbf{G}(x, y) = \sum_{i=0}^{m} \mathbf{A}_i(x, y) s_i, \tag{25}$$

pre každé x a y obrazu $\mathbf{G}$. V rámci ICA je tiež potrebné podotknúť, že v prípade generovania viacerých rôznych obrazov $\mathbf{G}_t$ bude pre každý obraz rovnaký kernel vlastností $\mathbf{A}_t$ (ďalej len $\mathbf{A}$) ale rôzne skryté nezávyslé komponenty $\mathbf{s}_t$ ($\mathbf{s}_t$ môžu byť považované za nositeľov zdrojových informácií obrazov). V ICA sú dva kľúčové predpoklady skrytých nezávislých komponentov [36] [38]:

1. Sú štatisticky nezávislé - východiský bod, kedy sa snaží rozčleniť informácie k odvodeniu nezávislých faktorov (ICA očakáva viacero modelov z ktorých boli dáta generované).

2. Ich distribúcie funkcií pravdepodobnosti sú negaussovské - myšlienka spočívajúca v centrálnom limitnom teoréme, ktorý hovorí, že súčet dvoch nezávislých náhodných premenných má distribúciu, ktorá je bližšie ku gaussovskej než ktorákoľvek z pôvodných premenných.

Tieto predpoklady sú dostatočné na to, aby umožnili odhad modelu, t. j. pri dostatočne veľkej vzorke obrazov $\mathbf{I}_t$, $t = 0, ..., T-1$, je môžné získať nejakú rozumnú aproximáciu kernelu $\mathbf{A}$ bez toho, aby boli vopred známe hodnoty latentných komponentov $\mathbf{s}_{t_i}$. Potom ako bolo odvodený kernel $\mathbf{A}$ je možné odvodiť skryté nezávyslé komponenty $\mathbf{s}_i$ šedotónového obrazu $\mathbf{G}$ nasledovne [36] [38]:

$$\mathbf{s}_i = \sum_{x,y} \mathbf{W}_i(x, y)\mathbf{G}(x, y), \tag{26}$$

kde $\mathbf{W}$ je inverzný kernel ku kernelu $\mathbf{A}$.

$\mathbf{W}$ môže byť tiež považovaný za váhy maximalizujúce negaussovskosť skrytých nezávyslých komponentov s_i. K odhadu podobého kernelu je možné použiť aproximačnú analýzu *kurtosis*, v preklade špičatosť. Meria vzdialenosť distribúcie dát od gaussovského rozloženia a vo výstupe môže mať tri možnosti [39]:

1. Nula - distribúcia je gaussovská.

2. Kladná hodnota - distribúcia je super-gaussovská (špičatejšia než gaussovská).

3. Záporná hodnota - distribúcia je sub-gaussovská (plochejšia než gaussovská).

3.3 Analýza štrukturálnymi vlastnosťami

V tomto type metód sa textúra popisuje niekoľkými základnými primitívami a ich priestorovým rozložením. Týmto primitívom môže byť jednoduchá jednotka ako pixel alebo niečo mierne zložitejšie typu oblasť alebo čiara (rovná, oblá, atď). Ich pravidlá priestorového usporiadania sú dané buď geometrickými vzťahmi alebo skúmaním ich štatistických vlastností. Inými slovami, štrukturálne metódy považujú textúru za kombináciu niekoľkých počiatočných primitív (vzorov). Tieto metódy sú optimálne pre textúry s pravidelnou štruktúrou, ale pre textúry s nepravidelnou textúrou sú nepoužiteľné. [19] [40]

V nasledujúcich kapitolách budú popísané vybrané metódy analýzy textúr štruktúrálnymi vlastnosťami.

3.3.1 Skeletonizácia

Skeletonizácia je technika globálnej priestorovej domény určená na reprezentáciu tvaru. Kostra má atraktívne vlastnosti, vďaka ktorým sú je možné rozpoznávanie štruktúrnych vzorov. Existujú dva typy metód skeletonizácie: a) založené na pixeloch b) a bez pixelov. Pri metóde založenej na pixeloch sa v procese skeletonizácie používajú všetky pixely vo vnútri tvaru. Metódy založené na pixeloch často využívajú techniky stenčenia alebo transformácie vzdialenosti. V metóde, ktorá nie je založená na pixeloch, sa na skeletonizáciu používajú

iba obrysové rysy spracovávaného tvaru. Kostra tvaru je potom analyticky odvodená od jeho obrysu. [22] [40]

Digitálne kostry, generované stenčovacími algoritmami, sa často používajú na reprezentáciu objektov v binárnom digitálnom obraze na analýzu a klasifikáciu tvaru. Kostra je však definovaná ako súbor tenkých čiar, oblúkov a kriviek (zvyčajne s hrúbkou jeden pixel), ktoré sú navzájom spojené tak, že musia byť zachované geometrické a topologické vlastnosti jej pôvodného objektu. [22] [41] [42]

Metóda predpokladá binarizovaný obraz $\mathbf{G}$ na vstupe, čo môže byť uskutočniteľné jednoduchým kvantizátorom a určením prahu t. Hodnota pixelu $[x, y]$ potom bude vyzerať:

$$\mathbf{G}_b(x,y) = \begin{cases} 1, & \mathbf{G}(x,y) \geq t, \\ 0, & inak, \end{cases} \tag{27}$$

kde $\mathbf{G}_b$ je binarizovaný obraz $\mathbf{G}$. Pomocou masky veľkosti 3×3 je z obrazu extrahovaná kostra s predpokladom, že stredový pixel masky bude vždy práve jeden. Takáto maska môže pokryť pixely v obraze s 2^8 kombináciami. Nie je však zohľadnené primitívum kostry znázornené na obrázku 3.11, pretože sa s vo väčšine situácií s vysokou pravdepodobnosťou

0	0	0
0	1	0
0	0	0

Obrázok 3.11. Maska textúry nedefinujúca žiadne primitívum ksotry, pravdepodobne sa jedná len o šum. [22]

jedná len o šum. Celkovo teda bude 255 možných kombinácií pixelov na maske 3×3. Z týchto kombinácí sú následne vyhodnocované primitívy textúry - čiary, oblúky, krivky alebo body kostry. [22]

3.4 Analýza na základe transformácie

Pri týchto metódach sa obraz pomocou transformačnej funkcie prevedie do nového priestoru. Nový priestor zvyčajne umožňuje jednoduchšiu analýzu textúry, pretože transformovaná textúra je lepšie rozlíšiteľná a často sú odkryté v pôvodnom priestore neviditeľné vlastnosti. [19]

V nasledujúcich kapitolách budú popísané vybrané metódy analýzy textúr na základe transformácie.

3.4.1 Gáborova transformácia

Gáborova transformácia je podobná vlnkovej transformácii [43], v ktorej funkcie majú Gaussovu prirodzenú bázu. Gáborova vlnka je optimálna transformácia na minimalizáciu dvojrozmernej neistoty vo frekvenčnej doméne, čo umožňuje jej využitie pri odkrývaní hrán a čiar [44]. Gáborov filter čiastočne zapadá aj do štatistickej analýzy vyššieho rádu, pretože štatistické vlastnosti tejto transformácie môžu byť použité tiež na určenie štruktúry a vizuálneho obsahu obrazu. Vlastnosti tejto transformácie sa používajú v niekoľkých aplikáciách analýzy obrazu, vrátane kategorizácie a segmentácie textúr, rozpoznávania obrazu, rozpoznávania abecedy atď. [19]

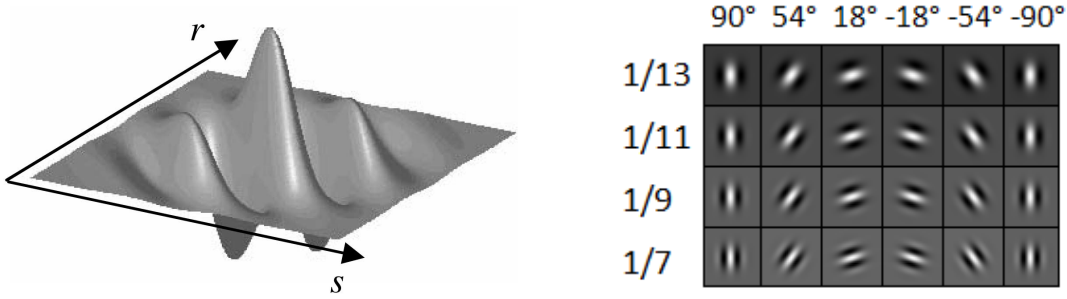
Keďže Gáborov filter má frekvenčne aj orientačne selektívne vlastnosti, je možné ho použiť ako pásmovú priepustnosť šedotónových pixelov $[x, y]$ obrazu $\mathbf{G}$. Rovnomerný Gáborov filter je všeobecne definovaný nasledovne [4] [35] [45]:

$$h(x, y : \theta, f) = exp\left\{-\frac{1}{2}\left[\frac{x_\theta^2}{\delta_x^2} + \frac{y_\theta^2}{\delta_y^2}\right]\right\} \cos(2\pi f \cdot x_\theta) \tag{28}$$

$$x_\theta = x\cos(\theta) + y\sin(\theta) \tag{29}$$

$$y_\theta = -x\sin(\theta) + y\cos(\theta) \tag{30}$$

kde θ je orientácia Gáborovho filtra, f je frekvencia, δ_x a δ_y sú priestorové konštanty Gaussovej obálky pozdĺž osí x a y. Ako orientácie Gáborového filtra je možné využiť distribúciu orientácií obrazu G z kapitoly 3.2.4 a ako frekvenciu filtra je možné použiť distribúciu priestorovo-frekvenčnej domény obrazu G z kapitoly 3.2.5. Kedže dvojrozmerné Gáborove filtre sú definované v priestorovo-frekvenčnej doméne, použitie distribúcie priestorovo-frekvenčnej domény je pre filter optimálne [45]. Výber hodnôt δ_x a δ_y zahŕňa kompromis. Čím väčšie sú hodnoty, tým robustnejšie sú filtre, ale zároveň aj vzrastá šanca, že filtre vytvoria falošné príznaky obrazu. So zvolením nižších hodnôt síce klesá šanca na vytvorenie falošných príznakov, ale taktiež nebude filter tak efektívny v získavaní vlastností a pri rozlíšiteľnosti textúry. Grafickú reprezentáciu popisovanej rovnice 28 je možné vidieť na obrázku 3.12. [4] [44] [45]



(a) (b)

Obrázok 3.12. Grafická ukážka banky Gáborovho filtra. (a) 3D reprezentácia Gáborovho filtru. [46] (b) Ukážka bánk Gáborovho filtru pre rôzne frekvencie (zvislá os) a orientácie (vodorovná os). [4]

Kapitola 4

Strojové učenie

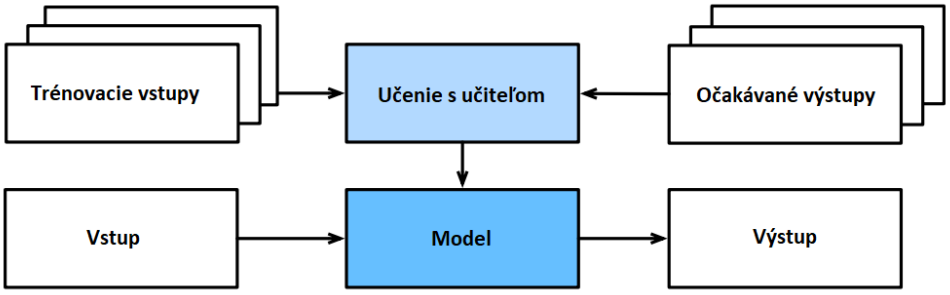
Strojové učenie sa zvyčajne delí na učenie s učiteľom, kedy sa algoritmu učenia predávajú okrem vstupných dát aj očakávané výstupy, a na učenie bez učiteľa, kedy sú algoritmu predané len vstupné dáta bez akýchkoľvek očakávaných výstupov. Učenie bez učiteľa sa často používa v úlohách ako je zhlukovanie, segmentácia obrazu alebo pri analýze hlavných komponentov. [24]

Algoritmus učenia s učiteľom sa zvyčajne snaží odhadnúť určité premenné tak, aby mali čo najlepšiu odozvu na vstupné dáta. Tieto premenné je možné nazvať parametrami definujúce model a jeho najlepšia odozva predstavuje čo najpodobnejšie výstupy očakávaným triedam t_i. Parametre predstavujú dvojicu $(\mathbf{w}, b)$ a celý model môže byť vyjadrený ako:

$$f(\mathbf{x}_i; \mathbf{w}, b) = \mathbf{w}\mathbf{x}_i + b, \tag{31}$$

kde $\mathbf{x}_i$ predstavuje vektor vstupného dáta. Chyba medzi odpoveďou modelu $f(\mathbf{x}_i; \mathbf{w}, b)$ a očakávanou triedou t_i sa nazýva loss L.

Ako je znázornené na obrázku 4.1, algoritmus učenia očakáva trénovacie dvojice v tvare $\mathbf{T} = \{(x_i, t_i)|i = 1, ..., N\}$, kde N predstavuje veľkosť dát. Výstup môžu byť diskrétne hodnoty $\mathbf{C} = \{c_k|k = 1, ..., M\}$ popisujúce triedy, do ktorých vstupné dáta zapadajú (M predstavuje počet týchto tried), alebo spojité hodnoty y, ktorými je možné predpovedať určitú skutočnosť, napríklad predpoveď ceny nehnuteľnosti na základe kriminality v okolí. Prvý prípad sa nazýva klasifikácia, keďže sa snažíme predpovedať náležitosť dát do určitej triedy a druhý regresia. [24]



Obrázok 4.1. Znázornenie procesu učenia s učiteľom.

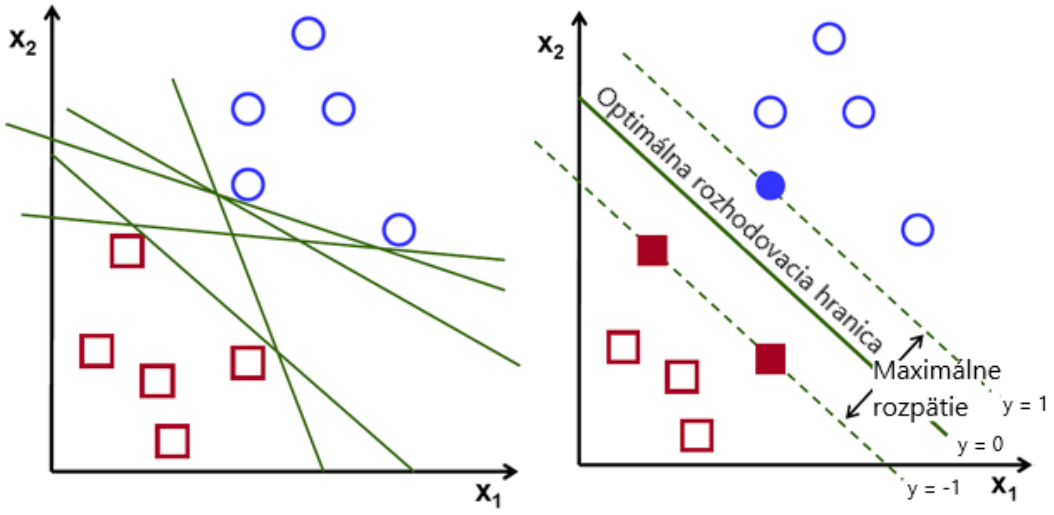
Po fáze učenia, nazývanej aj trénovanie, keď všetky dáta z trénovacej množiny boli spracované (zvyčajne viac krát), vznikol natrénovaný model schopný predpovedať výstupy na

doteraz nevidených dátach. Často býva celý postup trénovania modelu rozšírený ešte o testovaciu fázu, ktorá zavádza metriku presnosti predpovedí modelu. V tejto fáze sú potrebné nevidené dáta a tiež ich očakávané výstupy, ale na ich základe neprebieha aktualizácia parametrov modelu $(\mathbf{w}, b)$, ale je počítaná len presnosť samotných výstupov $f(\mathbf{x}_i)$ oproti očakávaným triedam $\mathbf{t}_i$. [24]

4.1 Support Vector Machine

Support vector machine (ďalej len SVM) je algoritmus učenia s nižším výpočetným výkonom a s významnou presnosťou. Je ho možné použiť ako pre regresné, tak pre klasifikačné úlohy. V tejto kapitole bude popísané využitie SVM v klasifikačných úlohách. [24] [47]

Cieľom algoritmu je nájsť hyperrovinu v N-rozmernom priestore, tak aby čo najlepšie klasifikovala vektory predstavujúce dátové body (kde N predstavuje počet hodnôt jedného vektora). Ako je možné vidieť na obrázku 4.2a pre oddelenie dvoch tried existuje nekonečne veľa možných hyperrovín. SVM zabezpečuje zvolenie takej hyperroviny oddeľujúcej dve triedy, ktorá maximalizuje vzdialenosť medzi nimi, t. j. nachádza sa najďalej od oboch tried zároveň (viď obrázok 4.2b). Optimálna rozhodovacia hyperrrovina odhadnutá maximalizáciou vzdialeností od tried taktiež zabezpečuje väčšiu istotu pri klasifikácií. [24] [47]



(a) (b)

Obrázok 4.2. Dátové body predstavujú vektory dvoch hodnôt (dvojdimenzionálny priestor), hyperrovinou je teda obyčajná línia ktorá má tvar rovnice 31. (a) Možné klasifikačné línie. (b) Klasifikačná línia zvolená pomocou SVM, vyplnené dáta predstavujú support vektory.

Podporné vektory, alebo support vektory, sú dátové body, ktoré sú najbližšie k hyperrovine a majú priamy vplyv na jej polohu a rotáciu. Ich pomocou je maximalizovaný klasifikátor a v prípade odstránenia týchto vektorov sa zmení poloha a smer hyperroviny. Support vektory tvoria hlavný princíp algoritmu SVM a sú to tie vektory, pre ktoré platí [24] [47]:

$$f(\mathbf{x}_i; \mathbf{w}, b) = \mathbf{w}\mathbf{x}_i + b, \tag{32}$$

$$|f(\mathbf{x}_i; \mathbf{w}, b)| \geq 1. \tag{33}$$

V nerovnici 33 je potrebná absolútna hodnota z dôvodu potrebných support vektorov z oboch strán klasifikačnej hyperroviny, kde pre jednu stranu má funkcia 32 záporné hodnoty a pre druhú kladné (viď obrázok 4.2). K odhadnutiu optimálnej hyperroviny je potrebné nájsť najmenší normovaný váhový vektor $\mathbf{w}$, ktorý spĺňa funkciu 32 [24] [47]:

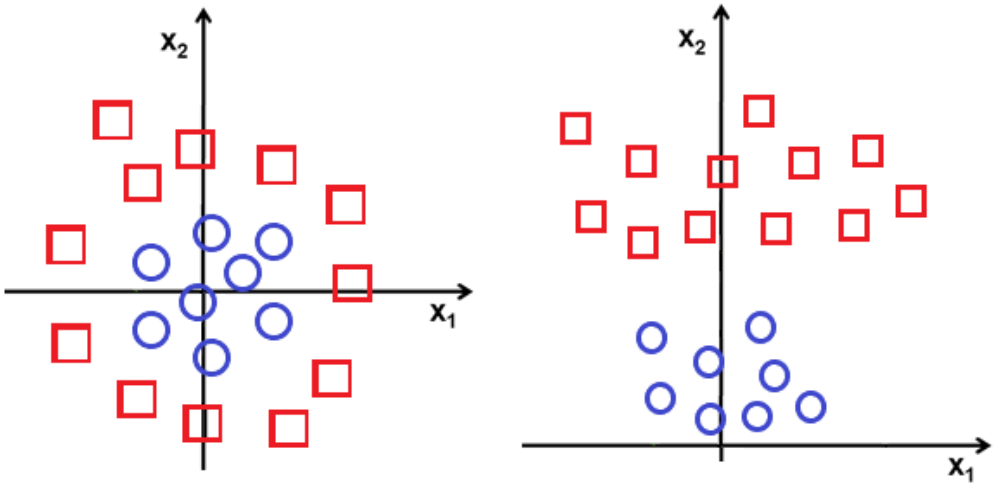
$$\arg\min_{\mathbf{w},b} ||\mathbf{w}||^2. \tag{34}$$

Ako je vidieť na obrázku 4.3a, nie je vždy možné dátové body lineárne klasifikovať. V tomto prípade je nutné dáta transformovať do takého priestoru, v ktorom lineárne klasifikovatelné budú. K tomuto sa v rámci SVM používajú tzv. kernelové funkcie. Jednoduchý príkladom je nasledujúca kernel funkcia [24] [47]:

$$z(\mathbf{x}) = x_1^2 + x_2^2, \tag{35}$$

kde doplnková súradnica z predstavuje vzdialenosť dátového bodu od počiatku súradnicovej sústavy. Názornú ukážku transformácie dvojdimenzionálnych dát je možné vidieť na obrázku 4.3. Na takto transformovaných dátach je už možná lineárna klasifikácia pomocou SVM ale funkcia modelu sa zmení z rovnice 32 na [24] [47]:

$$f(\mathbf{x}_i; \mathbf{w}, b) = \mathbf{w}z(\mathbf{x}_i) + b = \mathbf{w}z(x_{i_1} + x_{i_2}) + b. \tag{36}$$



(a) (b)

Obrázok 4.3. Ukážka transformácie dát rovnicou 35. (a) Pôvodné dáta, (b) transformované dáta do lineárne klasifikovatelnej formy.

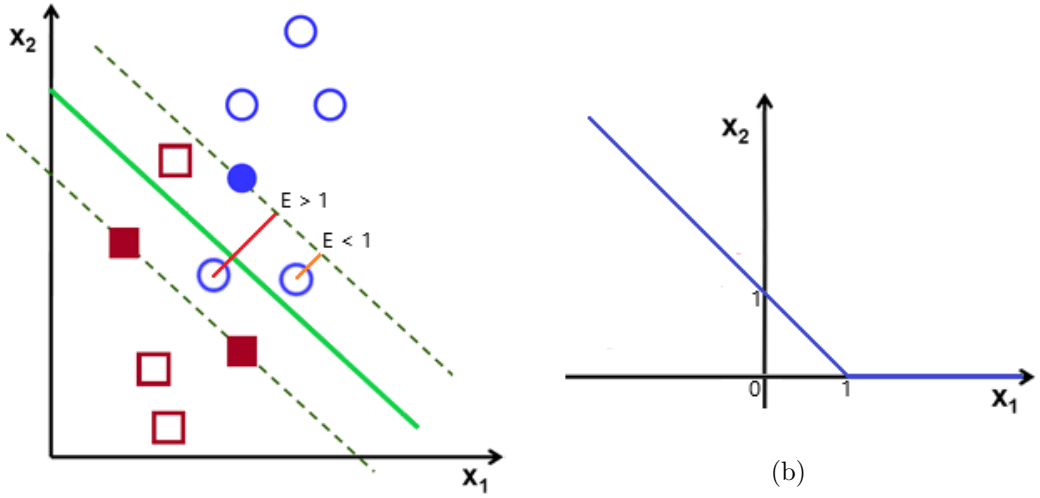
4.1.1 Neseparovatelné trénovacie dáta

Doteraz boli brané do úvahy len dáta, ktoré sú separovatelné, t. j. dáta pre ktoré existuje rozhodovacia hranica korektne klasifikujúca všetky trénovacie dáta, ktoré majú priradené triedy $\mathbf{T} = \{\mathbf{t}_i | t \in -1, 1, i = 1, ..., N\}$. V niektorých prípadoch môže dôjsť k tomu, že dáta nie sú separovatelné ani po transformácií do iného priestoru. SVM je možné aplikovať aj

na takto prekrývajúce sa rozloženia jednotlivých tried. V algoritme ale dôjde k niekoľkým modifikáciám, kedy je nerovnica 33 nahradená za hinge loss penalizáciu [24] [47]:

$$E_{HL}(f(\mathbf{x}_i; \mathbf{w}, b), t_i) = max(0, 1 - t_i f(\mathbf{x}_i; \mathbf{w}, b)). \tag{37}$$

Hinge loss je 0 vždy, ak je nerovnica 33 splnená a väčšia ako nula v prípade, že sa nerovnica nesplní. Veľkosť hinge loss chyby rastie s vyššou mierou nesplnenia nerovnice 33. Na obrázku 4.4a je možné vidieť znázornený odhad chyby E_{HL}, kde nesprávne klasifikované dáta majú chybu vyššiu ako jedna a dáta nachádzajúce sa na správnej strane klasifikácie ale príliš blízko klasifikačnej hyperroviny s chybou v rozsahu $(0, 1\rangle$ (support vektory sú prvými vektormi v najkratšej vzdialenosti od klasifikačnej hyperroviny s chybou nula). Obrázok 4.4b znázorňuje priebeh chyby E_{HL}. [24] [47]



(a) (b)

Obrázok 4.4. Ukážka hinge loss chyby. (a) Chyba na dátach, kde E predstavuje hinge loss E_{HL}. (b) Priebeh hinge loss funkcie.

Pre odhad optimálnych parametrov $(\mathbf{w}, b)$, je nutné minimalizovať chybu odhadnutú ako súčet chýb všetkých trénovacích dát s reguralizátorom $\lambda||w||^2$ [24] [47]:

$$E_{SV}(\mathbf{w}, b) = \sum_i E_{HL}(f(\mathbf{x}_i; \mathbf{w}, b), t_i) + \lambda||w||^2 \tag{38}$$

Kapitola 5

Návrh aplikácie

Hlavným cieľom aplikácie je detekcia morphovaných odtlačkov prstov. Na dosiahnutie tohto je v prvom rade nutné odtlačky predspracovať a extrahovať z nich potrebné informácie k morphingu. Po vygenerovaní morphovaných odtlačkov budú tieto odtlačky transformované z obrazového priestoru do priestoru vlastností, čím bude umožnená podrobnejšia analýza.

Pred samotnou implementáciou algoritmu detekcie morphovaných odtlačkov prstov, je nutné najprv vygenerovať dataset morphovaných odtlačkov, aby bolo možné sa učiť ich vlastnosti. K morphingu odtlačkov je potrebné ich predspracovanie, čo zahŕňa extrakciu niektorých primitívnych vlastností, ako je distribúcia orientácií, priestorovo-frekvenčná doména a markanty. Predspracovanie bolo implementované v mojej bakalárskej práci [4] a zahrňuje tieto kroky:

1. Normalizácia - bude popísané v kapitole 5.1.1.
2. Extrakcia distribúcie orientácií - štatistická analýza textúry z kapitoly 3.2.4.
3. Extrakcia frekvenčnej domény - štatistická analýza textúry z kapitoly 3.2.5.
4. Skeletonizácia - štrukturálna analýza textúry z kapitoly 3.3.1.
5. Extrakcia markantov - bude popísané v kapitole 5.1.5.

Po vykonaní vyššie uvedených krokov, máme všetky informácie o odtlačkoch potrebné k morphingu. Samotný morphing odtlačkov predstavuje kombináciu dvoch alebo viacerých odtlačkov prstov do jedného výsledného odtlačku, tak že výsledný sa bude zhodovať s každým z pôvodných. Ďalšími podmienkami takéhoto odtlačku, ako bolo podrobnejšie popisované v kapitole 2.3, je realistický vzor odtlačku pre ľudské oko a zároveň musí mať dostatok aspektov z oboch pôvodných odtlačkov, aby bol voči nim overiteľný. Avšak aby bolo možné samotný morphing odtlačkov uskutočniť je potrebné extrahované informácie vhodne roztriediť a vybrať také, z ktorých bude vygenerovaný výsledný morfovaný odtlačok spĺňať všetky potrebné podmienky. Zvolenými metódami morphingu je metóda rezom implementovaná v rámci mojej bakalárskej práce [4] a jej modifikácia implementovaná v rámci projektovej praxe [5]. Obe metódy však zahŕňajú rovnaké hlavné kroky:

1. Zarovnanie odtlačkov prstov na základe ich distribúcie orientácii.
2. Odhad optimálnej línie rezu.
3. Vytvorenie šablóny morphovaného odtlačku a vygenerovanie morphovaného odtlačku.

Posledným krokom je vygenerovanie datasetu morphovaných odtlačkov. K tomuto sa využije databáza odtlačkov prstov transformovaná na databázu ich primitivných vlastností, ktorá bude využitá k odhadu optimálnej morphovacej línie náhodne vyberaných dvojíc odtlačkov. Následne po odhade optimálnych morphovacích liniií pre dvojice odtlačkov, je vykonaný samotný morphing, alebo zmiešanie, obrázkov odtlačkov prstov. Teraz je možné začať učenie vlastností morphovaných odtlačok prvok tak, aby bola možná detekcia a rozpoznanie morphovaných odtlačkov od obyčajných odtlačkov.

Neexistuje doposiaľ žiadna verejná práca venujúca sa detekcií morphovaných odtlačkov, takže v tejto diplomovej práci je prvý krát prebraný návrh a implementácia metódy zabezpečujúce tento cieľ. V prvom rade je potrebné si uvedomiť že morphing je možné vykonávať aj nad obrázkami reálnych odtlačkov prstov, takže rôzne metódy určené k detekcií falošných odtlačok typu testovania rozloženia šumu, pravidelnosti gradianetov či zmien šedej v obrázku odtlačku, sú v tomto prípade nepoužiteľné. Preto na detekciu morphovaných odtlačkov navrhujem použiť abstraktnejšiu analýzu odtlačkov a to použitím binarizovaných deskriptorov textúr [48], ktoré už boli použité pri hodnotení živosti odtlačkov prstov (preukázaná funkčnosť na obrázkoch reálnych odtlačkov) [49] a s obdivuhodným výkonom použité na detekciu morphovaných tvárií, kde bolo len 3,46 % morphovaných tvári označených za normálne a žiadna normálna tvár nebola označená ako morphovaná [50]. Takže učenia detekcie bude prebiehať v nasledujúcich krokoch:

1. Generovanie datasetu morphovaných odtlačkov prstov.
2. Predspracovanie datasetu a odstránenie všetkých morphovaných odtlačkov prstov, ktoré nenadobúdajú dve identity - učenie vlastností prebehne len na valídnych morphovaných odtlačkoch.
3. Odhad množiny optimálnych filtrov na obrazoch prírodných scén - spôsob odhadu bude popísaný v kapitole 5.3.1.
4. Generovanie diskriptorov textúry (obrázku) morphovaných odtlačkov ako binarizovaný vektor odvodený pre každý pixel pomocou množiny optimálnych filtrov odvodených z prírodných scén - bude popísané v kapitole 5.3.2.
5. Transformácia deskriptorov textúr do histogramov a učenie sa vlastností linéarnym SVM - bude poopísané v kapitole 5.3.3

V kapitole 5.1 budú bližšie popísané kroky predspracovania odtlačkov z mojej bakalárskej práce [4]. V kapitole 5.2 bude popísaný spôsob morphingu odtlačkov z mojej bakalárskej práce [4] a jeho modifikácia z mojej projektovej praxe [5]. V poslednej kapitole 5.3 bude popísaný celý navrhovaný spôsob detekcie morphovaných odtlačkov prstov, v rámci tejto diplomovej práce, postupujúci podľa vyššie uvedených bodov.

5.1 Predspracovanie odtlačku

V tejto kapitole bude bližšie popísaný spôsob riešenia z mojej bakalárskej práce [4]. Ako už bolo uvedené vyššie, predspracovanie odtlačku a extrakcia primitívnych vlastností zahŕňa určité kroky, ktoré budú popísane bližšie v nasledujúcich bodoch.

5.1.1 Normalizácia odtlačku

V prípade, že odtlačok prstu nie je šedotónový obrázok, je nutné ho do tohto formátu previesť (všetky algoritmy nad odtlačkom očakávajú šedotónový vstup). Okrem toho normalizácia zahŕňa nasledujúce operácie znázornené na obrázku 5.1:

1. **Rozmazanie** - jednoducho dosiahnuteľné aplikáciou Gaussovho filtra na odtlačok prsta **G**. Rozmazanie zabezpečuje odstránenie prudkých zmien odtieňov šedej čo zlepšuje viditeľnosť papilárnych línií a tým aj odhad jeho vlastností ako sú distribúcia orientácií a frekvenčná doména. Gaussov filter na pozícii okna $\mathbf{G}(x, y)$ a o veľkosti okna $i \times j$ s odchýlkou σ bude mať tvar:

$$\mathbf{G}(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x-\frac{i}{2})^2+(y-\frac{j}{2})^2}{2\sigma^2}}. \tag{39}$$

2. **Ekvalizácia histogramu** - ďalšia operácia zabezpečujúca vyhladenie kolísavých hodnôt pozdĺž papilárnych línií. Jej účelom je roztiahnutie histogramových hodnôt do plného rozsahu ako je popísané v kapitole 3.2.1. Ekvalizovaná hodnota pixelu $[x, y]$ obrázku **F** potom bude mať tvar:

$$\mathbf{F}_{ekv}(x, y) = \begin{cases} m_0 + \sqrt{\frac{v_0(\mathbf{F}(x,y)-m)^2)}{v}} & \mathbf{F}(x, y) > m, \\ m_0 - \sqrt{\frac{v_0(\mathbf{F}(x,y)-m)^2)}{v}} & inak, \end{cases} \tag{40}$$

   kde m a v je stredná hodnota a smerodajná odchýlka obrazu **F** a m_0 a v_0 je požadovaná stredná hodnota a smerodajná odchýlka.

3. **Segmentácia odtlačku** - poslednou úpravou je oddelenie odtlačku od pozadia, kedy je pre každý pixel ekvalizovaného obrázku $\mathbf{F}_{ekv}$ odhadnutý rozptyl $\mathbf{V}^2_{x,y}$ v okolí $\omega \times \omega$. Mapa segmentácie $\mathbf{F}_{seg}$ potom bude mať tvar:

$$\mathbf{F}_{seg}(x, y) = \begin{cases} 1 & \mathbf{V}^2_{x,y} \geq v_0^2, \\ 0 & inak, \end{cases} \tag{41}$$

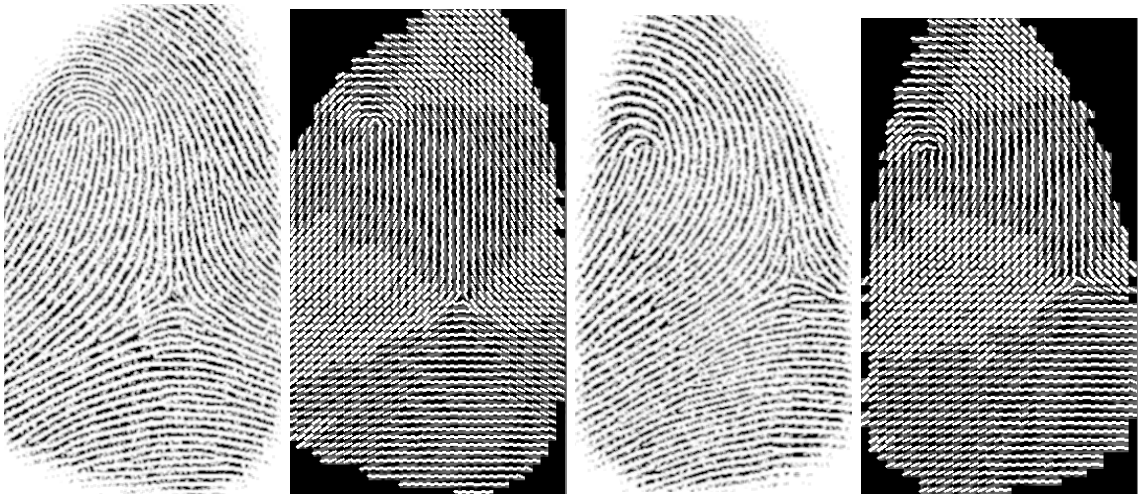
   kde hodnota jedna predstavuje popredie (odtlačok) a nula pozadie.

5.1.2 Odhad distribúcie orientácií

Odhad orientácií papilárnych línií je vykonaný štatistickou analýzou odtlačku z pohľadu stochastickej textúry. Na odtlačok je aplikovaný algoritmus z kapitoly 3.2.4, čím je získaná distribúcia orientácií O_F papilárnych línií odtlačku. Avšak kvôli prítomnosti rušivých aspektov na odtlačku (napr. šum, poškodené línie, markanty...) odhadnutá orientácia nemusí byť vždy správna. Keďže pre orientácie papilárnych linií nie je typické, že prudko menia smer (okrem miest obsahujúcich singulárne body), je možné odhadnuté orientácie vyhladiť pomocou Gaussového filtra z rovnice 39. Distribúciu orientácií papilárnych línií je možné vidieť na obrázku 5.2.

(a) (b) (c) (d)

Obrázok 5.1. Ukážka prípravy obrázku (a) k spracovaniu. (a) Pôvodný odtlačok, (b) rozmazaný odtlačok, (c) normalizovaný odtlačok, (d) segmentácia odtlačku. [4]



Obrázok 5.2. Ukážka odhadnutia lokálnych orientácií z odtlačkov prstov.

5.1.3 Odhad priestorovo-frekvenčnej domény

Nasledujúcou primitívnou vlastnosťou odtlačku je priestorovo-frekvenčná doména alebo hustota papilárnych línií. Táto vlastnosť je tiež odhadnuteľná zo štatistickej analýzy textúr použitím metódy z kapitoly 3.2.5, ktorá sa snaží modelovať malé okolie ako sínusoidu. Tento model je ideálny pre samotný odtlačok, pretože prierez papilárnych línií tvorí sínusoidu taktiež. Avšak metóda z kapitoly pre každý pixel obrazu odhadne deskriptor textúry v podobe x-signatúry - diskrétnej sínusoidy. Tento deskriptor následne musí byť transformovaný do frekvencie, ktorá je odhadnutá ako $\frac{1}{t(x,y)}$, kde $t(x, y)$ je priemerný počet bodov medzi sused-

nými vrcholmi x-signatúry pre pixel $[x, y]$ obrazu $\mathbf{F}_{ekv}$. Získame tak priestorovo-frekvenčnú doménu $\mathbf{T}_F$, ktorá je znázornená na obrázku 5.3.

Obrázok 5.3. Ukážka odhadnutia priestorovo-frekvenčnej domény z odtlačkov prstov. [4]

5.1.4 Skeletonizácia

K extrakcií kostry odtlačku je potrebné vyhladiť papilárne línie tak, aby neobsahovali žiadny šum ani prerušenia. K dosiahnutiu takýchto kvalitných papilárnych línií je možné transformovať odtlačok do nového priestoru, v ktorom pixely obrazu presne definujú všetky papilárne línie. Metóda zabezpečujúca podobnú transformáciu je Gáborová transformácia z kapitoly 3.4.1 a je ju možné použiť na tieto účely s mienym prispsôbením parametrov. V rovniciach:

$$h(x, y : \theta, f) = exp\left\{-\frac{1}{2}\left[\frac{x_\theta^2}{\delta_x^2} + \frac{y_\theta^2}{\delta_y^2}\right]\right\}\cos(2\pi f \cdot x_\theta) \tag{42}$$

$$x_\theta = x\cos(\theta) + y\sin(\theta) \tag{43}$$

$$y_\theta = -x\sin(\theta) + y\cos(\theta) \tag{44}$$

sú parametre f a θ definujúce frekvenciu a smer banky Gáborovho filtra (viď obrázok 3.12b). Nastavenie týchto parametrov bude prebiehať prostredníctvom distribúcie orientácií $\mathbf{O}_F$ a priestorovo-frekvenčnej domény $\mathbf{T}_F$. Napríklad pre transformáciu pixelu $\mathbf{F}_{ekv}(x, y)$ (v prípade, že sa jedná o popredie - $\mathbf{F}_{seg}(x, y) = 1$) bude frekvencia $f = \mathbf{T}_F(x, y)$ a smer banky $\theta = \mathbf{O}_F(x, y)$.

Takto transformovaný odtlačok $\mathbf{F}_q$ je následne nutné normalizovať do rozsahu $\langle 0, 1\rangle$ a binarizovať pomocou prahovej hodnoty b. Výsledný odtlačok $\mathbf{F}_b$ je potom vo formáte vhodnom pre skeletonizáciu a je ho možné vidieť na obrázku 5.4. Skeletonizácia prebieha pomocou metódy z kapitoly 3.3.1, ktorá zabezpečuje extrakciu štrukturálnej vlastnosti z textúry – kostry (viď obrázok 5.4).

Obrázok 5.4. Ukážka aplikácie Gáborovho filtru, binarizácie a skeletonizácie na odtlačok prstu. [4]

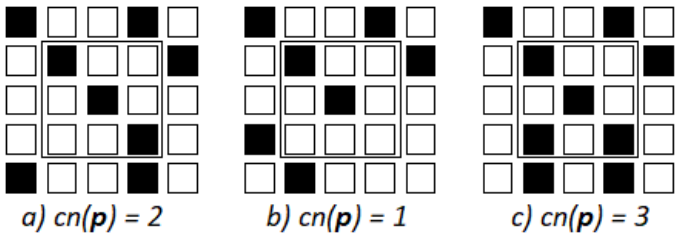
5.1.5 Extrakcia markantov

Po získaní binárnej kostry, jednoduché skenovanie obrázku umožňuje detekciu pixelov, ktoré zodpovedajú markantom. Pixely zodpovedajúce markantom sú charakterizované špeciálnou hodnotou (tzv. *crossing number*), ktorá je rôzna od 2. *Crossing number* $cn(p)$, pixelu p, je definované v binárnom obrázku ako polovica súčtu rozdielov medzi dvojicami pixelov v susedstve ôsmych pixelov $\mathbf{p}$. *Crossing number* je definované nasledovne: [8]

$$cn(p) = \frac{1}{2}\sum_{i=1}^{8} \left| val(\mathbf{p}_{i \bmod 8} - val(\mathbf{p}_{i-1}) \right| \tag{45}$$

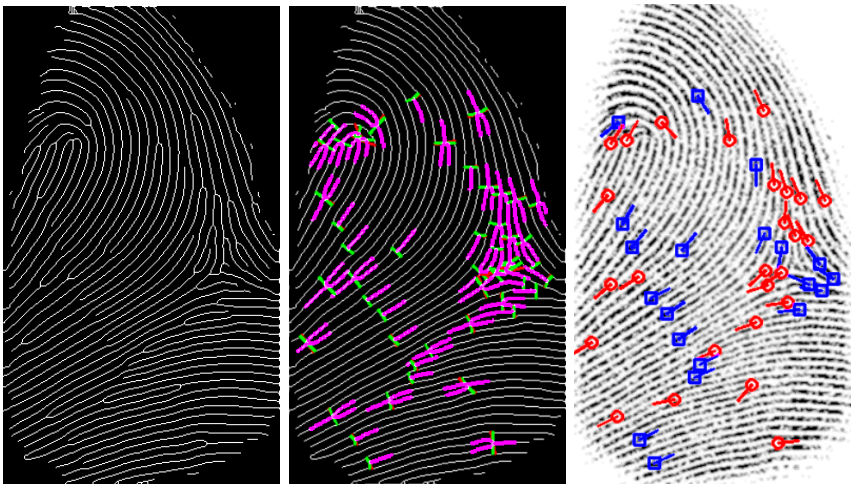
kde pixely $p_1, p_2, ..., p_7$ patria do zoradenej postupnosti pixelov, ktoré definujú susedstvo pixelu p a $val(p) \in 0, 1$ je hodnota pixelu. Z rovnice vyplýva že pixel p s hodnotou val(p) rovnou jedna [8]:

- Ak $cn(p) = 2$ - je pixel kostry papilárnej línie, neoznačujúci markant (obrázok 5.5 a)).
- Ak $cn(p) = 1$ - zodpovedá markantu typu ukončenie (obrázok 5.5 b)).
- Ak $cn(p) = 3$ - zodpovedá komplexnejšiemu markantu, rozdvojeniu (obrázok 5.5 c)).



Obrázok 5.5. Typy markantov detekovateľných pomocou *crossing number*. [8]

Detekciu markantov je možné vidieť na obrázku 5.6.



Obrázok 5.6. Ukážka získavania markantov z odtlačku. [4]

5.2 Morphing odtlačkov

V tejto práci bude podrobne prebraný postup spracovania dvoch rôznych odtlačkov prstov a postup vygenerovania morfovaného odtlačku na úrovni obrázku pomocou metódy rezu popisovanej v kapitole 2.3. V prvom rade sa jednotlivé odtlačky zanalizujú a extrahujú sa potrebné prvky k morfovaniu, tj. distribúcia orientácií, priestorovo-frekvenčná doména a markanty odtlačkov. Pre optimálne zarovnané odtlačky na základe ich distribúcií orientácií sa odhadne optimálna rezná línia a zmiešajú sa ich primitívne vlastnosti, ktoré sú následne využité k vygenerovaniu nového morfovaného odtlačku s identitami oboch pôvodných. Pre dané dva odtlačky $\mathbf{F}^1$ a $\mathbf{F}^2$ je teda nutné vykonať nasledujúce kroky:

1. Zistiť podobnosť orientačných polí odtlačkov (kapitola 5.2.1).
2. Zistiť optimálnu reznú líniu odvodenú z podobnosti odtlačkov (kapitola **??**) - v tomto bode budú popísané dve rôzne evaluačné metódy, jedna z mojej bakalárskej práce [4] a druhá z projektovej praxe [5].
3. Vygenerovať morfovanú šablónu odvodenú z optimálnej reznej línie odtlačkov a vygenerovať morfovaný odtlačok z predpripravenej šablóny (kapitola 5.2.4).

5.2.1 Zarovnanie odtlačkov prstov

Táto technika zarovná dva odtlačky určením najvyššej podobnosti orientácií línií v ich priesečníkoch, pre akékoľvek primerané posunutie a rotáciu. Využíva k tomu metódu získavania podobností distribúcie orientácií odtlačkov prstov $\mathbf{O}_1$ a $\mathbf{O}_2$ (odhadnuté postupom v kapitole 5.1.2), ktorá je definovaná nasledovne:

$$s(\mathbf{O}_1, \mathbf{O}_2) = \frac{\sum_{(i,j)\in(\mathbf{F}^1_{seg}\vee\mathbf{F}^2_{seg})}(\mathbf{F}^1_{seg_{i,j}} + \mathbf{F}^2_{seg_{i,j}}) \cdot \psi(\mathbf{O}^1_{i,j}, \mathbf{O}^2_{i,j})}{\sum_{(i,j)\in(\mathbf{F}^1_{seg}\vee\mathbf{F}^2_{seg})}(\mathbf{F}^1_{seg_{i,j}} + \mathbf{F}^2_{seg_{i,j}})}, \tag{46}$$

kde $\mathbf{F}^1_{seg}$ a $\mathbf{F}^2_{seg}$ je segmentácia odtlačkov prstov z kapitoly 5.1.1, $\mathbf{F}^1_{seg} \vee \mathbf{F}^2_{seg}$ predstavuje takú množinu bodov odtlačku, v ktorej aspoň jedna zo segmentácií nadobúda hodnotu

jedna (popredie). $\psi(\mathbf{O}^1_{i,j}, \mathbf{O}^2_{i,j})$ je podobnosť medzi dvoma hodnotami orientácií θ_1, θ_2:

$$\psi(\mathbf{O}^1_{i,j}, \mathbf{O}^2_{i,j}) = 1 - \frac{2|\theta_1 - \theta_2|}{\pi} \tag{47}$$

Aby sa našlo najlepšie zarovnanie, sú určené všetky možné posunutia $\mathbf{S}$ (v krokoch δ_{st} pixelov) a rotácie (v krokoch δ_{rot}) orientačného poľa $\mathbf{O}^2$ s ohľadom na $\mathbf{O}^1$. Následne sú vyradené zarovnania s veľmi malým prekrytím, podľa predom určeného prahu prekrytia:

$$\frac{|\mathbf{S}_{\mathbf{O}^1} \cap \mathbf{S}_{\mathbf{O}^2}|}{max(|\mathbf{S}_{\mathbf{O}^1} \cap \mathbf{S}_{\mathbf{O}^2}|)} \geq min_{SR} \tag{48}$$

Obrázok 5.7. Ukážka zarovnaných odtlačkov prstov. [4]

Pre zarovnania, ktoré prekročili prah prekrytia odtlačkov sa otočia predom odhadnuté lokálne orientácie odtlačku, tak aby boli odpovedajúce pre aktuálnu rotáciu odtlačku a následne sa odhadne ohodnotenie podobnosti odtlačkov v jednotlivých zarovnaniach, podľa rovnice 46 (po otočení orientačného poľa, je nutné jednotlivé orientácie natočiť späť o uhol rotácie poľa, aby odpovedali lokálnym orientáciám rotovaného odtlačku). Ako ideálne zarovnanie odtlačkov sa zvolí to, ku ktorému je priradené najvyššie ohodnotenie podobnosti lokálnych orientácií. Príklad optimálneho zarovnania je vidieť na obrázku 5.7, z ktorého prekrytá oblasť bude vybraná k ďalším krokom morfovania.

5.2.2 Pôvodný odhad reznej línie

Optimálna línia rezu je založená na dvoch ohodnoteniach:

1. Podobnosť papilárnych línií v kritickej oblasti (oblasť línie rezu).
2. Zachovaný dostatočný počet markantov z oboch pôvodných odtlačkov.

Ako bolo písané vyššie v tejto kapitole budú navrhnuté dve evaluačné metriky optimálnej reznej línie. Prvá bude pôvodná z mojej bakalárskej práce [4] a štúdie [6] a druhá z mojej projektovej praxe [5].

Rezná línia l prechádza cez stred ťažiska zarovnaných odtlačkov prstov, kde obidva obsahujú pixely popredia, t. j. oblasť je daná ako $\mathbf{F}^A_{seg} = \mathbf{F}^1_{seg} \wedge \mathbf{F}^{2A}_{seg}$, kde $\mathbf{F}^{2A}_{seg}$ je rotovaná a posunutá segmentácia na základe zarovnania z predošlej kapitoly. V okolí reznej línie sa nachádza tzv. priesečníková oblasť so šírkou d_{max}, ktorá sa spolu s reznou líniou l v krokoch δ rotuje po priestore zarovnaných odtlačkov, čo generuje niekoľko rezných línií. K jednotlivým líniám sa odhadne ohodnotenie priesečníkových oblastí a ako optimálna línia rezu sa zvolí línia s najvyšším ohodnotením. Príklad odhadnutej línie rezu je vidieť na obrázku 5.8.

Rovnicu priamky, ktorá predtavuje líniu rezu prechádzajúcu cez ťažisko $\boldsymbol{\rho} = (\rho^x, \rho^y)$ s uhlom β je možné získať nasledujúcim spôsobom:

$$\begin{aligned} &a_l \cdot x + b_l \cdot y + c_l = 0 \\ &a_l = sin(\beta), b_l = cos(\beta), \\ &c_l = -\rho_x \cdot sin(\beta) - \rho_y \cdot cos(\beta) \end{aligned} \tag{49}$$

Ohodnotenie línie rezu je možné získať jednoduchým váhovaním jej prierezu primitívnych vlastností odtlačkov F^1 a F^2:

$$S_c = \omega_o \cdot S_o + \omega_t \cdot S_t + \omega_m \cdot S_m \tag{50}$$

kde:

- S_o a S_v merajú podobnosť distribúcií orientácií a priestorovo-frekvenčnej domény popri línií rezu l (s cieľom vygenerovania vzoru odtlačku realistického pre ľudské oko), ktoré vychádzajú z nasledujúcich rovníc:

$$S_o = \frac{\sum_{(i,j)\in C}(\mathbf{F}^1_{seg_{(i,j)}} + \mathbf{F}^2_{seg_{(i,j)}}) \cdot \psi(\mathbf{O}^1_{i,j}, \mathbf{O}^{2A}_{i,j}))}{\sum_{(i,j)\in C}(\mathbf{F}^1_{seg_{(i,j)}} + \mathbf{F}^2_{seg_{(i,j)}})} \tag{51}$$

$$S_t = \frac{\sum_{(i,j)\in C}\left(1 - \frac{|\mathbf{T}^1_{F_{(i,j)}} + \mathbf{T}^2_{F_(i,j)}|}{(max_F - min_F)}\right)}{|\mathbf{C}|} \tag{52}$$

  kde $\mathbf{C}$ obsahuje súradnice, pre ktoré platí $d_l \leq d_{max}$ (hranica priesečníkovej oblasti od línie rezu, d_l je vzdialenosť od línie rezu a d_{max} predstavuje maximálnu vzdialenosť) a miesto kde obidve orientačné polia predstavujú nenulové prvky:

$$\mathbf{C} = \{(i,j) | (i,j) \in (\mathbf{S}_{O^1} \cap \mathbf{S}_{\hat{O}^{2A}}) \wedge dist_l(i,j) \leq d_{max}\} \tag{53}$$

$$dist_l(x,y) = \frac{|a_l \cdot x + b_l \cdot y + c_l|}{\sqrt{a_l^2 + b_l^2}} \tag{54}$$

- S_m je hodnotenie odvodené z dvoch šablón markantov (bude popísané v nasledujúcich odsekoch) s cieľom vygenerovať odtlačok prsta, ktorý sa zhoduje s oboma pôvodnými odtlačkami:

$$S_m = max(\zeta_m(\mathbf{K}^1, \mathbf{K}^2), \zeta_m(\mathbf{K}^2, \mathbf{K}^1)) \tag{55}$$

$$\zeta_m(\mathbf{A}, \mathbf{B}) = \frac{Z(|\mathbf{A}|^P_l, \mu_m, \tau_m) + Z(|\mathbf{B}|^N_l, \mu_m, \tau_m)}{2} \tag{56}$$

  kde $|\mathbf{K}|^P_l$ a $|\mathbf{K}|^N_l$ označujú kardinalitu markantov $\mathbf{K}$, ktoré spadajú na pozitívnu alebo negatívnu stranu línie l, pre každé zvlášť:

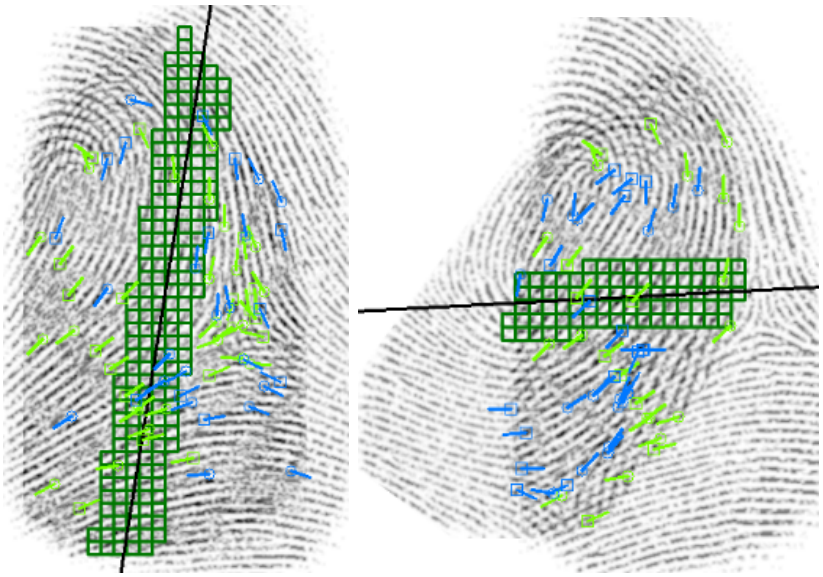
$$|\mathbf{K}|^P_l = |\{m \in \mathbf{K} | \Phi_l(m_x, m_y) \geq 0)\}| \tag{57}$$

$$|\mathbf{K}|_l^N = |\{m \in \mathbf{K} | \Phi_l(m_x, m_y) < 0)\}| \tag{58}$$

kde Z je sigmoidná funkcia, riadená dvoma parametrami (μ_m a τ_m), ktorá obmedzuje príspevok kardinálneho operátora ($|\cdot|$) a zabezpečuje aby výsledná hodnota bola v rozsahu $[0, 1]$. Sigmoidná funkcia je definovaná nasledovne:

$$Z_{(v,\mu,\tau)} = \frac{1}{1 + e^{-\tau(v-\mu)}} \tag{59}$$

- $\omega_o, \omega_v, \omega_m \in [0, 1], \omega_O + \omega_v + \omega_m = 1$ sú tri váhové faktory.



Obrázok 5.8. Ukážka odhadnutia optimálnej línie rezu. Modré markanty patria zarovnávanému odtlačku a zelené odtlačku, voči ktorému sa zarovnáva. Markanty sú zobrazené len v morfovanej oblasti. [4]

5.2.3 Modifikovaný odhad reznej línie

Prvá modifikácia sa nachádza v hodnotení početností markantov v rovnici 56, kvôli chybnému hodnoteniu spôsobenému priemerovaním. K chybe dochádza hlavne pri príliš vysokých rozdieloch početností markantov na jednotlivých stranách línie, teda ak je na jednej strane málo markantov a rovnica 59 to ohodnotí hodnotou 0.1, na druhej strane príliš veľa s ohdonotením 0.9, výsledný priemer nadobudne hodnotenie 0.5, čo často býva jedno z maximálnych hodnotení početnosti rozloženia markantov. Z tohto dôvodu bolo nutné zaviesť evaluačnú metriku, ktorá je funkciou oboch hodnotení sigmoidnej funkcie a zároveň zabezpečí pokles výsledného skóre v prípade príliš nízkeho jedného hodnotenia. Ako vhodná metrika bola navrhnutá metrika inšpirovaná F_1 skóre, bežne používané v strojovom učení, dané rovnicou:

$$F_1 = 2\frac{precission \cdot recall}{precission + recall}, \tag{60}$$

kde precision a recall sú ďalšie metriky, ktoré v tomto prípade nie sú podstatné a sú nahradené za ohodnotenie početnosti $Z(|\mathbf{A}|_l^P, \mu_m, \tau_m)$ a $Z(|\mathbf{B}|_l^N, \mu_m, \tau_m)$ a teda výsledné hodno-

tenie početnosti markantov bude vyzerať:

$$\zeta_m(\mathbf{A},\mathbf{B}) = 2\frac{Z(|\mathbf{A}|_l^P,\mu_m,\tau_m)\cdot Z(|\mathbf{B}|_l^N,\mu_m,\tau_m)}{Z(|\mathbf{A}|_l^P,\mu_m,\tau_m)+Z(|\mathbf{B}|_l^N,\mu_m,\tau_m)} \tag{61}$$

Druhá modifikácia odhadu reznej línie sa nachádza v jej fixne nastavenej polohe na ťažisko zarovnanej oblasti $\mathbf{F}^A$. Často však dochádza k tomu, že sa práve v ťažisku nachádza singulárny bod odtlačku, ktorý pri morphingu spôsobuje zbytočné artefakty. Toto je dôvodom zavedenia nového parametra do algoritmu $dist_\rho$ určujúci maximálnu vzdialenosť od ťažiska. Do tejto vzdialenosti sú prehľadávané alternatívne rezné línie nadobúdajúce lepšie hodnotenia. Hľadanie optimálnej reznej línie s pohybujúcim sa stredom je možné len s metrikou z rovnice 61 eliminujúcou chyby spôsobené priemerovaním v rovnici 56.

5.2.4 Vytvorenie a generovanie z šablóny morfovaného odtlačku

Po získaní optimálnej reznej línie zarovnaných odtlačkov prstov je posledným krokom vytvorenie šablóny morfovaného odtlačku. Keďže línia rezu rozdeľuje oba zarovnané odtlačky na dve časti, pozitívnu a negatívnu, je potrebné rozhodnúť, aká časť z ktorého odtlačku bude využitá pre vygenerovanie šablóny. Toto rozhodnutie prebieha na základe markantov. Jednoducho povedané, vyberú sa tie časti odtlačkov, ktoré spoločne obsahujú viac markantov:

$$(p,n) = \begin{cases} (1,2) & \zeta_m(\mathbf{K}^1,\mathbf{K}^2) \geq \zeta_m(\mathbf{K}^2,\mathbf{K}^1) \\ (2,1) & \text{inak} \end{cases} \tag{62}$$

Po výbere ideálnych častí zarovnaných odtlačkov určených k morfovaniu je potrebné zlúčiť informácie, ktoré obsahujú. Zlúčenie však zahŕňa vyváženie morfovania v kritickej oblasti pomocou váhového faktora pre šablónu markantov $\tilde{\mathbf{K}}$:

$$\tilde{\mathbf{K}} = \{m \in \mathbf{K}^p, \Phi_{l_{max}}(m_x,m_y) \geq 0\} \cup \{m \in \mathbf{K}^n, \Phi_{l_{max}}(m_x,m_y) \geq 0\} \tag{63}$$

Po aplikovaní popísanej techniky zlúčenia informácií obsiahnutých v odtlačkoch je pripravená šablóna pre vygenerovanie morfovaného odtlačku. Keďže cieľom práce je vytvorenie morfovaného odtlačku s čo najrealistickejším vzhľadom, k morfovaniu sú využívané pôvodné obrázky odtlačkov. Získaním potrebných morfovaných šablón s informáciami z pôvodných odtlačkov, je následne jednoducho rozhodnuté ktorá časť z ktorého odtlačku bude zvolená k morfovaniu. V mieste kde je spočítaná ideálna línia rezu dochádza k prepojeniu pôvodných odtlačkov vyvážením pixelových hodnôt pôvodných obrázkov. K vyváženiu pixelových hodnôt je využitý nasledujúci vzťah:

$$\mathbf{D}^I(x,y) = w_{x,y}^{l_{max}} \cdot \hat{\mathbf{F}}^p(x,y) + (1 - w_{x,y}^{l_{max}}) \cdot \hat{\mathbf{F}}^n(x,y) \tag{64}$$

kde $w_{x,y}^{l_{max}} \in [0,1]$ je váhový faktor na vyrovnanie morfovania v blízkosti línie rezu l_{max}:

$$w_{x,y}^{l_{max}} = \begin{cases} 1 - \max\left(0, \frac{d_{max} - dist_{l_{max}}(x,y)}{2 \cdot d_{max}}\right) & \text{ak } a_{l_{max}} \cdot x + b_{l_{max}} \cdot y + c_{l_{max}} \geq 0 \\ \max\left(0, \frac{d_{max} - dist_{l_{max}}(x,y)}{2 \cdot d_{max}}\right) & \text{inak} \end{cases} \tag{65}$$

Celý proces morfovania je možné vidieť na obrázku 5.9, kde posledný krok je popisovaný postup generovania morfovaného odtlačku z odhadnutej reznej línie.

Obrázok 5.9. Ukážka celého procesu morfovania dvoch odtlačkov. [4]

5.3 Detekcia morphovaných odtlačkov

Na popis textúry boli zvolené binárne štatistické obrazové vlastnosti (ďalej BSIF, odvodené z angl. binarized statistical image features), inšpirované metódami LBP a LPQ z kapitoly 3.2.2 a 3.2.3. Tieto metódy opisujú okolie každého pixelu binárnou hodnotou, ktorý sa získa najprv konvolúciou obrazu pomocou lineárnych filtrov a potom binarizáciou odoziev filtrov. Avšak na rozdiel od pristupov LBP a LPQ, nie sú použité manuálne preddefinované filtre, ale natrénované filtre pomocou štatistických vlastností prírodných scén. Tento prístup poskytuje možnosť odvodenia filtrov, ktoré je užitočné na popis obrázkov s nezvyčajnými vlastnosťami. [48] [49]

5.3.1 Učenie vlastností prírodných filtrov

V prvom rade je potrebné odvodiť filtre z kapitoly 3.2.6 tak aby mali čo najlepšiu odozvu na N prírodných scénach. K tomu je možné vyžiť mierne modifikovanú rovnicu 19, v ktorej nebude maximalizovaná odozva s jedného filtru, ale vektor $\mathbf{s}$ N rôzných filtrov. Rovnicu je teda potrebné prepísať do tvaru:

$$\mathbf{s}_n = \sum_{i,j} \mathbf{W}_n(i,j)\mathbf{G}(i,j) \tag{66}$$

Vzhľadom na n lineárnych filtrov $\mathbf{W}_n$ ich môžeme naskladať do matice $\mathbf{W}$ veľkosti $n \times l \times l$ a vypočítať všetky odozvy naraz, t.j. $\mathbf{s} = \mathbf{W}\mathbf{x}$ a bitový reťazec $\mathbf{b}$ získame binarizáciou každého prvku z $\mathbf{s}$, ako je uvedené vyššie. Takže vzhľadom na detektory lineárnych znakov $\mathbf{W}_n$ je výpočet bitového reťazca $\mathbf{b}$ jednoduchý. Tiež je jasné, že bitové reťazce pre všetky obrazové oblasti veľkosti $l \times l$, obklopujúce každý pixel obrazu, možno pohodlne vypočítať pomocou n konvolúcií.

Aby bola získaná sada filtrov $\mathbf{W}_n$, sú odhadnuté filtre maximalizáciou štatistickej nezávislosti prvkov $\mathbf{s}$. Aby však bolo možné použiť štandardné algoritmy analýzy nezávislých komponentov (viď kapitola 3.2.8) na odhadovanie nezávislých komponentov, je potrebné rozložiť filtračnú maticu $\mathbf{W}$ na dve časti pomocou:

$$\mathbf{s} = \mathbf{W}\mathbf{x} = \mathbf{P}\mathbf{Q}\mathbf{x} = \mathbf{P}\mathbf{z} \tag{67}$$

kde $\mathbf{z} = \mathbf{Q}\mathbf{x}$ a $\mathbf{P}$ je $n \times n$ štvorcová matica, ktorá je odhadnutá pomocou ICA, a matica $\mathbf{Q}$ vykonáva súčasné bielenie a zmenšenie rozmerov trénovacích vzoriek $\mathbf{x}$ (viď kapitola 3.2.7).

Predspracovanie používa analýzu hlavných komponentov (PCA z kapitoly 3.2.7) nasledovne. Trénovacia množina malých častí $l \times l$ náhodne odobraných z prirodných scén sa najprv prevedú do formátu s nulovým priemerom (t. j. odpočíta sa priemerná intenzita pixelu každého obrázku) a potom sa ich dimenzia zníži zachovaním iba n prvých hlavných komponentov, ktoré sa ďalej delia ich štandardnou odchýlkou, aby sa získali vybielené vzorky údajov $\mathbf{z}$. Vzhľadom na rozklad kovariačnej matice $\mathbf{C}$ na vlastné vektory:

$$\mathbf{C} = \mathbf{V}\mathbf{L}\mathbf{V}^T \tag{68}$$

je matica $\mathbf{Q}$ definovaná:

$$\mathbf{Q} = \left(\mathbf{L}^{-1/2}\mathbf{V}^T\right)_{0:n}, \tag{69}$$

kde hlavná diagonála $\mathbf{L}$ obsahuje vlastné hodnoty $\mathbf{C}$ v zostupnom poradí a $()_{0:n}$ označuje prvých n riadkov matice.

Potom, ak vezmeme do úvahy vzorky z vybielených údajov s nulovým priemerom, je možné použiť štandardné algoritmy analýzy nezávislých komponentov na odhadnutie ortogonálnej matice $\mathbf{P}$, pomocou ktorej sa získajú nezávislé komponenty z trénovacích dát. Inými slovami, keďže $\mathbf{z} = \mathbf{P}^{-1}\mathbf{s}$, nezávislé zložky umožňujú reprezentovať dátové vzorky $\mathbf{z}$ ako lineárnu superpozíciu základných vektorov definovaných stĺpcami $\mathbf{P}^{-1}$. Nakoniec, ak sú dané $\mathbf{P}$ a $\mathbf{Q}$, získa sa matica filtra $\mathbf{W} = \mathbf{P}\mathbf{Q}$, ktorú možno priamo použiť na výpočet vlastností BSIF.

5.3.2 BSIF

Táto metóda transformuje priestor obrazu do priestoru vlastností, v ktorom je každý pixel popísaný binárnym vektorom. Vektorová hodnota pixelu je považovaná ako lokálny deskriptor intenzity obrazu v malom okolí pixelu. Za predpokladu, že časť obrázku $\mathbf{X}$ veľkosti $l \times l$ a lineárne filtre $\mathbf{W}_n$ rovnakej veľkosti, odpoveď filtrov s_n je získaná ako:

$$\mathbf{s}_n = \sum_{i,j} \mathbf{W}_n(i,j)\mathbf{X}(i,j) = \mathbf{W}_n^T\mathbf{X}, \tag{70}$$

Hodnota každého prvku (t. j. bitu) v binárnom deskriptore pixelu sa vypočítava binarizáciou odozvy $\mathbf{s}$ lineárnych filtrov s prahom nastaveným na nulu:

$$\mathbf{b}_n = \begin{cases} 1 & \mathbf{s}_n > 0, \\ 0 & inak. \end{cases} \tag{71}$$

Každý bit je spojený s iným filtrom a požadovaná dĺžka bitového reťazca určuje počet použitých filtrov. Sada filtrov je odhadnutá z trénovacej sady prírodných scén maximalizáciou štatistickej nezávislosti odpovedí filtra, ktorá je popísaná v predošlej kapitole 5.3.1.

5.3.3 SVM klasifikácia

K učeniu vlastností morphovaných odtlačkov prstov je potrebné najprv vygenerovať dataset morphovaných odtlačkov. V prvom kroku je potrebné rozšíriť dataset obrázkov reálnych odtlačkov prstov o ich primitívne vlastnosti - tento krok má len optimalizačné účely, aby nebolo nutné predspracovanie odtlačkov pred morphingom každej dvojice. Následne bude prebiehať morphing obrázkov odtlačkov $\mathbf{F}_1$ a $\mathbf{F}_2$, kedy bude morphovaný každý odtlačok

s každým v rámci databáze daného senzoru, pričom musí platiť, že $\mathbf{F}_1 \neq \mathbf{F}_2$ (morphing rovnakých odtlačkov nemá zmysel).

Po vygenerovaní databáze morphovaných odtlačkov prstov $\mathbf{M}$ je potrebné databázu premiešať s databázov normálnych odtlačkov $\mathbf{N}$ a označiť ich, triedami identifikujúce daný odtlačok či sa jedná o morphovaný alebo nie. Kedže ako algoritmus učenia bude použité lineárne *SVM*, podľa kapitoly 4.1 bude platiť, že triedy $t \in \{-1, 1\}$, kde $t = -1$ predstavuje normálny odtlačok a $t = 1$ morphovaný. Takže dátou sadu odtlačkov $\mathbf{D}$ budú tvoriť dvojice $(\mathbf{F}, t)$, kde $\mathbf{F}$ predstavuje histogram BSIF deskriptorov a bude vyzerať nasledovne:

$$\mathbf{D} = \{(\mathbf{F}_i, \mathbf{t}_i) \,|\mathbf{F}_i \in \mathbf{M} \cup \mathbf{N},\ t_i = -1 \Rightarrow \mathbf{F}_i \in \mathbf{N} \vee \mathbf{t}_i = 1 \Rightarrow \mathbf{F}_i \in \mathbf{M},\ 0 \leq i < T\}, \tag{72}$$

kde T predstavuje počet prvkov datasetu.

Po tom čo bol definovaný dataset, pre efektívne učenie vlastností je potrebné ešte určité predspracovanie. Predspracovanie zahŕňa extrakciu vlastností prostredníctvom n optimálnych filtrov $\mathbf{W}$ odhadnutých na prírodných scénach z kapitoly 5.3.1. Podľa rovnice 70 z kapitoly 5.3.2 priložením množiny filtrov $\mathbf{W}_n$ na každý pixel obrazu, je obraz transformovaný do priestoru vlastností, kde každý pixel namiesto šedotónovej hodnoty predstavuje vektor $\mathbf{s}$ obsahujúci n prvkov. Následne je tento vektor $\mathbf{s}$ binarizovaný rovnicou 71:

$$\mathbf{F}_i^b = b(\mathbf{F}_i), \tag{73}$$

kde $b(\mathbf{F})$ je transformácia obrazového priestoru do priestoru vlastností v ktorom sú pixely popísané binárnymi vektormi $\mathbf{b}$ a $\mathbf{F}_i^b$ je obraz transformovaný do tohto priestoru vlastností BSIF.

Posledným krokom predspracovania je transformácia obrazu popísaného BSIF vlastnosťami $\mathbf{F}_i^b$ do histogramu z kapitoly 3.2.1. Napríklad pre binárne vektory dĺžky 10 (počet filtrov z kapitoly 5.3.1) bude mať histogram dimenziu 1×1024, kedže na každej pozícií vektoru sa môžu nachádzať práve dve hodnoty - 0 alebo 1. Predspracovaný dataset $\mathbf{D}_{hist}$ pre SVM bude mať tvar:

$$\mathbf{D}_{hist} = \{(h\,(b\,(\mathbf{X}_i))\,, \mathbf{t}_i) \,|\mathbf{X}_i \in \mathbf{M} \cup \mathbf{N},\ t_i = -1 \Rightarrow \mathbf{X}_i \in \mathbf{N} \vee \mathbf{t}_i = 1 \Rightarrow \mathbf{X}_i \in \mathbf{M},\ 0 \leq i < T\}, \tag{74}$$

kde $h(\mathbf{X})$ predstavuje funkciu transformujúcu vstupné dvojrozmerné dáta $\mathbf{X}$ do histogramu $\mathbf{x}$.

Teraz je možné využiť lineárne SVM z kapitoly 4.1 k učeniu ideálnej rozhodovacej hranice medzi dátami morphovaných odtlačkov. Rozhodovacia hranica však nebude učená striktne ale pomocou hinge loss chybovej funkcie z kapitoly 4.1.1, pretože s veľkou pravdepodobnosťou dôjde k nemožnosti klasifikovať skupiny odtlačkov striktne pre každý z datasetu. Hinge loss z kapitoly 4.1.1 má tvar:

$$E_{SV}(\mathbf{w}, b) = \sum_i E_{HL}(f(\mathbf{x}_i; \mathbf{w}, b), t_i) + \lambda ||w||^2, \tag{75}$$

kde $f(\mathbf{x}_i; \mathbf{w}, b)$ predstavuje lineárnu funkciu s váhami $\mathbf{w}, b$ (váhy k učeniu - $\mathbf{w}$ je sklon priamky a b posunutie) na histograme $\mathbf{x}_i$. Optimum tejto chybovej funkcie spočíva v minimalizácií jej chyby. A teda optimálna rozhodovacia hranica má také parametre w a b, pre ktoré je E_{SV} minimálna.

Kapitola 6

Implementácia

K implementácií navrhnutej aplikácie boli zvolené jazyky C++ a python, a knižnica OpenCV (Open Source Computer Vision Library) verzia 4.1.2[1], ktorá je určená pre projekty týkajúce sa počítačového videnia a strojového učenia. Ďalej bola použitá pythonovská výpočetná knižnica NumPy 1.22.2[2] poskytujúca viacrozmerný objekt poľa, rôzne odvodené objekty (ako sú maskované polia a matice) a rad rutín pre rýchle operácie s poľami, vrátane matematických, logických, manipulácií s tvarmi, triedenia, výberu, I/O operácií, diskrétnej Fourierovej transformácie, základnej lineárnej algebry, základných štatistických operácií, náhodných simulácií a mnohých ďalších. Poslednou významnou použitou knižnicou je knižnica scikit-learn verzia 1.0.2[3] poskytujúca nástroje pre prediktívnu analýzu. Rovnako ako OpenCV pre python, tak je aj scikit-learn postavená na výpočetnej knižnici NumPy, čo zabezpečuje vzájomnú kompatibilitu. Grafy zo štatistických analýz a predpovedí sú generované pomocou knižnice Matplotlib verzia 3.5.1[4] umožňujúcu jednoduchú vizualizáciu aj pre náročné dáta.

Implementácia aplikácie umožňujúcej morphing odtlačkov prstov je prevzaná z mojej bakalárskej práce [4] a projektovej praxe [5]. Jedná sa o GUI aplikáciu implementovanú v jazyku C++, kvôli optimálnosti, a frameworku Qt[5], umožňujúcom vytvárať softvér nezávislý od platformy. K uloženiu a práci s obrázkami odtlačkov prstov je použitá dátová štruktúra *cv::Mat*, ktorú poskytuje knižnica OpenCV. V podstate sa jedná o maticu hodnôt určitého dátového typu. Táto dátová štruktúra je ďalej použitá nielen k uloženiu medzistavov a výsledkov zpracovania obrázku odtlačku, ale tiež k uloženiu získaných vlastností odtlačku, morphovacích informácií a šablón novo generovaného odtlačku. K zisteniu odpovedajúcemu skóre odtlačkov, bolo do aplikácie integrované Minutia Cylinder-Code SDK [51].

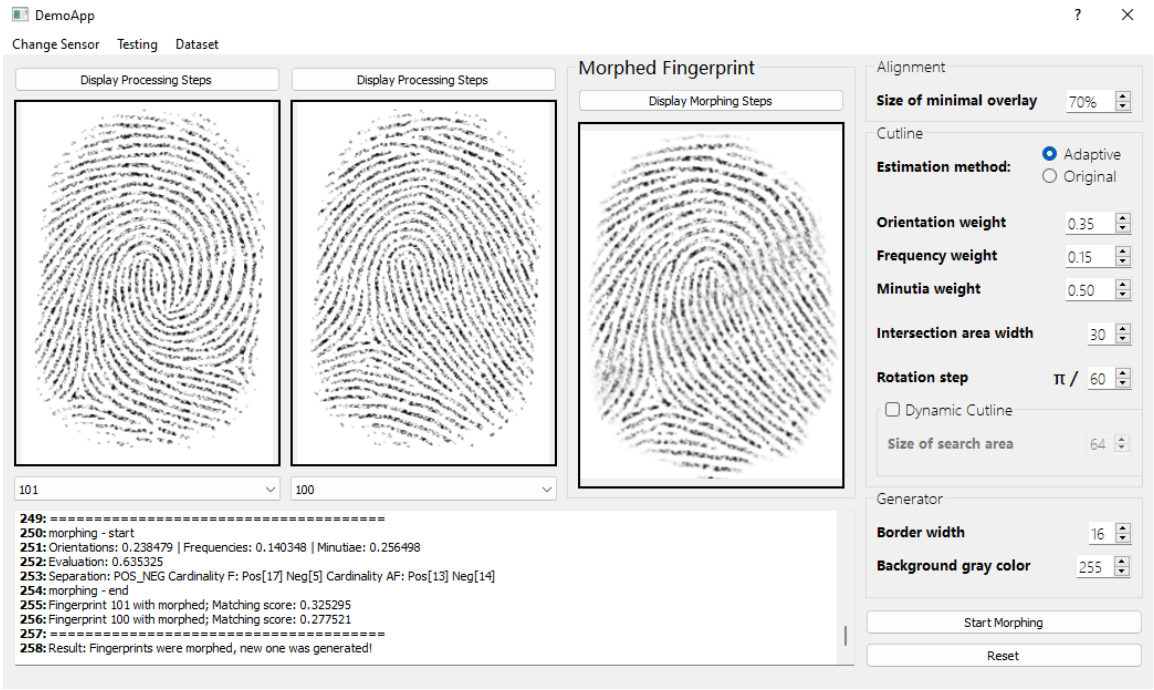
V tejto práci nebudú popísané implementačné detaily celej aplikácie prevzatej z mojej bakalárskej práce [4], ale bude priblížená špecifikácia jednotlivých častí vzhľadom na návrh aplikácie tejto diplomovej práce v kapitole 6.1. Ďalej bude kapitola 6.1 zahŕňať popis rožsírenia implementácie o generátor datasetu morphovaných odtlačkov prstov. V nasledujúcej kapitole 6.2 sa nachádza popis implementácie knižnice, obsahujúcej nástroje potrebné k detekcií morphovaných odtlačkov prstov, v jazyku python. V kapitole 6.3 sú špecifikované implementačné detaily odhadnutia filtrov zo sady prírodných obrázkov popísaných v kapitole 5.3.1. V kapitole 6.4 je špecifikácia implementácie generovania BSIF z kapitoly 5.3.2

[1] https://opencv.org/about/
[2] https://numpy.org/doc/stable/
[3] https://scikit-learn.org/stable/index.html
[4] https://matplotlib.org/
[5] https://www.qt.io/

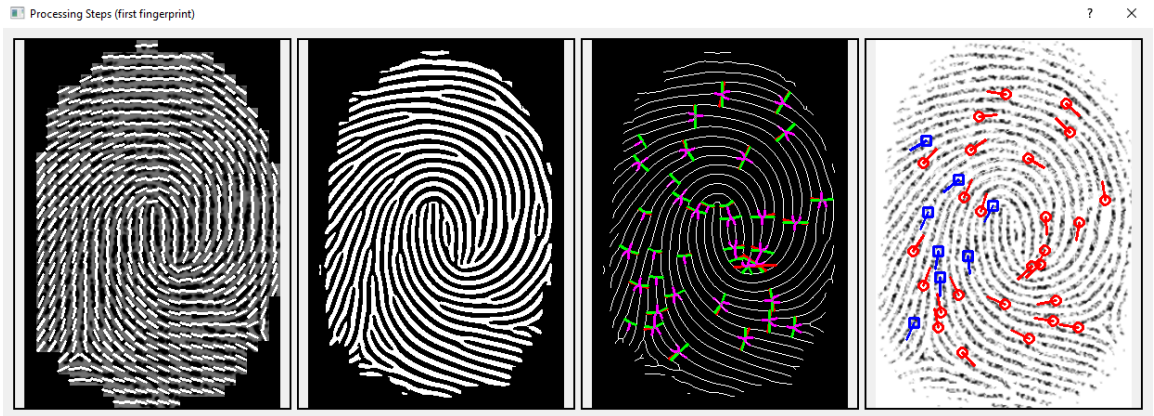
za pomoci odhadnutých filtrov z predošlej kapitoly. V poslednej kapitole 6.5 je popísaná implementácia trénovania SVM modelov z kapitoly 5.3.3 na extrahovaných BSIF z predošlej kapitoly.

6.1 GUI aplikácia morphingu

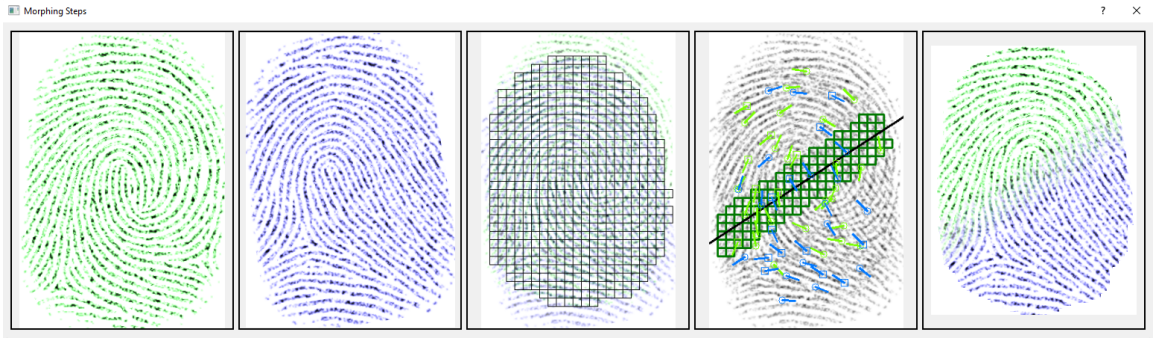
Ako už bolo spomínané jedná sa o GUI aplikáciu implementovanú v C++ schopnú vizualizovať jednotlivé kroky predspracovania odtlačkov prstov a následne ich morphing. Aplikácia umožňuje aj nastavenie parametrov pre jednotlivé kroky spracovania odtlačkov. Výzor aplikácie je možné vidieť na obrázku 6.1 a zobrazenie krokov spracovania na obrázkoch 6.2 a 6.3.



Obrázok 6.1. GUI aplikácia umožňujúca morphing odtlačkov prstov.



Obrázok 6.2. Vizualizácia krokov extrakcie vlastností odtlačku.



Obrázok 6.3. Vizualizácia krokov morphingu odtlačkov.

Celý proces morphingu bol rozdelený do troch statických knižníc, kde každá knižnica zabezpečuje špecifické nástroje spracovania:

1. **Processing** (extrakcia informácií z odtlačku prsta a vylepšenie kvality) - podporuje základné spracovanie odtlačku zahrňujúce aj vylepšenie kvality. Implementované nástroje knižnice podporujú rozmazanie, normalizáciu a segmentáciu odtlačku z kapitoly 5.1.1, odhad distribúcie orientácií z kapitoly 5.1.2, odhad priestorovo-frekvenčnej domény z kapitoly 5.1.3, vylepšenie kvality odtlačku a odhad kostry odtlačku z kapitoly 5.1.4 a odhad markantov z kapitoly 5.1.5. Funkčnosť jednotlivých nástrojov a celý proces extrakcie vlastností z odtlačku zapuzdruje trieda `FingerprintProcessor`.

2. **Morphing** (morphing odtlačkov prstov) - knižnica umožňujúca morphing odtlačkov prstov spracovaných pomocou knižnice z bodu 1. Implementované nástroje knižnice podporujú zarovnanie odtlačkov prstov z kapitoly 5.2.1, odhad línie rezu pôvodnou a vylepšenov metódov z mojej projektovej praxe [5] (viď kapitola **??**) a generátor morphovaného odtlačku z odhadnutej šablóny z kapitoly 5.2.4. Implementácia generátora morphovaného odtlačku prstu obsahuje aj rozšírenie o vyhladzovanie hranatých okrajov z mojej bakalárskej práce [4]. Funkčnosť jednotlivých nástrojov a celý proces morphingu zapuzdruje ako aj v predošlej knižnici jedna trieda, `MorphingProcessor`.

3. **Matching** (overovanie morfovaného odtlačku voči každému z pôvodných) - umožňuje overiť 2 odtlačky využitím vyššie spomenutého Minutia Cylinder-Code SDK [51]. Implementácia knižnice obsahuje nástroj vykonávajúci prevod šablóny odtlačkov prstov do formátu potrebného pre MCC SDK implementované v jazyku .Net DLL. Umožňuje vyvíjať aplikácie na overovanie odtlačkov prstov pomocou Minutia Cylinder Code algoritmov [52]. MCC SDK rozpoznáva odtlačky s ohodnotením $\langle 0, 1\rangle$, pre ktoré podľa pokynov FRONTEX[6] s cieľom simulovať realistický útok na nejaký sytém, sú prahové hodnoty rozpoznávania odtlačkov prstov nastavené na 3 miery nesprávneho overenia (FAR) [6]:

FAR (%)	1	0,1	0,01
MCC SDK	0,1083	0,1205	0,1329

Tabuľka 6.1: Ohodnotenie FAR metriky pre MCC SDK.

[6] https://frontex.europa.eu/

Ohodnotenia v tabuľke 6.1 znamenajú, že pre FAR 1 % je potrebné aby porovnanie dvoch šablón markantov, pomocou MCC SDK, nadobudlo ohodnotenie aspoň 0, 1083. Obdobne to platí pre ostatné FAR hodnoty, teda pre 0, 1 % FAR musí ohodnotiť MCC SDK otlačky skóre 0, 1205 a 0, 01 % FAR potrebuje nadobudnúť aspoň 0, 1329. Hodnotenie skóre priradenia dvoch odtlačkov prstov zapuzdruje trieda `Matcher`.

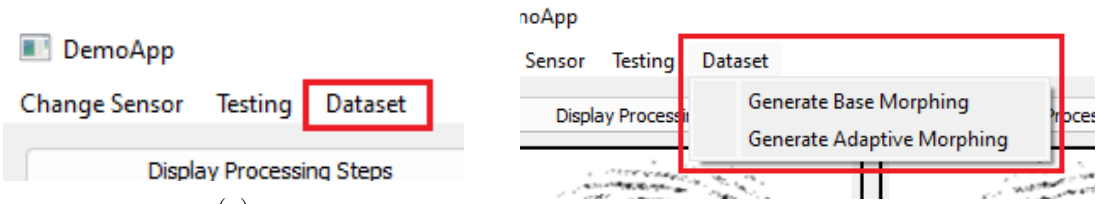
Primárnymi komponentami aplikácie sú widgety, prvky určené k vytváraniu užívateľského rozhrania. Widgety môžu zobrazovať údaje a informácie o stave, prijímať vstup od užívateľov a poskytovať kontajnery pre ďalšie widgety. Komunikácia medzi týmito užívateľskými prvkami prebieha na základe signálov a slotov. V princípe ide o zasielanie notifikácií (signálov) o zmene stavu určitého prvku, na ktorý reaguje iný prvok, očakávajúci odoslaný typ správy (správu prijíma pomocou slotu). [53]

Hlavným komponentom GUI aplikácie je rodičovská trieda `DemoApp` starajúca sa o inicializáciu GUI prostredia, o prepojenie jednotlivých widgetov aplikácie pomocou signálov a slotov, a uchováva si informáciu o aktuálne zvolenom senzore (Bergdata, SecuGen, Sagem MSO, Syntetický). V aplikácií sa tiež nachádzajú 3 okná. Dve z nich slúžia na zobrazenie odtlačkov určených na morfovanie a tretie na zobrazenie výsledku morfovania. Akciu morphingu zaobstaráva trieda `ActionManager` obsahujúca jednotlivé nástroje `FingerprintProcessor`, `MorphingProcessor` a `Matcher` z popisovaných knižníc **Processing**, **Morphing** a **Matching**. Textové pole v aplikácií je určené pre logovanie priebehu spracovávania. Loger je využitý aj pre výpis výsledkov morfovania, t. j. skóre, ktorým odpovedá morfovaný odtlačok voči každému z pôvodných.

Ďalšou funkcionalitou aplikácie je menu „Testing“ určené k hromadnému testovaniu odtlačkov. Výsledné obrázky ukladá do zložky a ohodnotenie odtlačkov do textového súboru. Menu panel bol rozšírený pre potreby tejto práce a bola pridaná nová možnosť generovania dátovej sady morphovaných odtlačkov prstov, popisovanej v kapitole 5.3.3. Bližšia špecifikácia rozšírenia panelu je popísaná v nasledujúcej kapitole 6.1.1.

6.1.1 Generovanie dátovej sady

Ako je vidieť na obrázku 6.4a, GUI aplikácií bol rozšírený panel s menu o nové políčko generovania datasetu „Dataset“. Po kliknutí na menu „Dataset“ sa zobrazia dve možnosti (widgety) generovania (viď obrázok 6.4b). „Generate Base Morphing“ generuje dátovú sadu morphovaných odtlačkov prstov pôvodnou metódou morphingu popísanou v kapitole 5.2.2 a „Generate Adaptive Morphing“ generuje dátovú sadu morphovaných odtlačkov prstov métodou morphingu z mojej projektovej praxe [5] (viď kapitola 5.2.3). V rodičovskej triede `DemoApp` sú tieto dva widgety prepojené, prostredníctvom signálu `triggered()` (signál odpovedajúci kliknutiu na daný widget), s triedou zabezpečujúcou vykonávanie akcií v aplikácií `ActionManager`.



(a) (b)

Obrázok 6.4. Ukážka rozšírenia GUI aplikácie.

`ActionManager` prostredníctvom triedy zapuzdrujúcej funkčnosť generovania dátovej sady morphovaných odtlačkov prstov `Generator`, spustí príslušný proces. V prípade zvolenia možnosti „Generate Base Morphing“, `ActionManager` spúšťa odpovedajúcu akciu `Generator::generateBaseMorphing()` a v prípade zvolenia možnosti „Generate Adaptive Morphing“, spúšťa akciu `Generator::generateAdaptiveMorphing()`. Generátoru `Generator` je taktiež sprostredkovaná informácia o aktuálne zvolenom senzore `currentSensor`. Na základe zvoleného senzora je načítaná príslušná sada odtlačkov prstov určených k morphingu.

Oba procesy generovania však vykonávajú rovnaký postup a podmienky pridania nového vzorku dátovej sady. Najprv sa zkonštruujú potrebné nástroje k morphingu a to sú `FingerprintProcessor` z knižnice **Processing**, `MorphingProcessor` z knižnice **Morphing** a `Matcher` z knižnice **Matching**. Následne pre každý odtlačok zo sady sú zvolené tri iné odtlačky, s ktorými je morphovaný. K morphingu sa využívajú nástroje `FingerprintProcessor` zabezpečujúcim extrakciu vlastností z odtlačkov a `MorphingProcessor` zabezpečujúcim morphing odtlačkov na základe extrahovaných vlastností. Pre každý odtlačok prstu zo sady teda existujú tri morphované odtlačky. Potom čo bol dokončený aktuálny morphing odtlačku sa vyhodnocuje skóre priradenia morphovaného odtlačku prstu k obom pôvodným, z ktorých vznikol. Skóre oboch priradení musí presiahnuť prahovú hodnotu t_{score} aby bol morphovaný odtlačok zaradený do generovanej dátovej sady. Hodnotenie skóre priradenia je sprostredkované nástrojom `Matcher`, ktorý porovnána šablóny morphovaného odtlačku s pôvodnými, z ktorých vznikol.

6.2 Knižnica MorphingDetection

Implementovaná knižnica sa skladá z niekoľkých modulov zabezpečujúcich konkrétne činnosti. V knižnici nebol implementovaný modul schopný vykonávať samotnú detekciu morphingu, ale jej nástroje sú prispôsobené k tomu, aby bolo možné samotnú detekciu uskutočniť. V kapitole 6.2.1 je popis implementácie databázového modulu určeného k ukladaniu a načítavaniu potrebných prostriedkov, v kapitole 6.2.2 sa nachádza špecifikácia implementácie modulu určeného k extrakcií štatistických a deskriptívnych vlastností z obrazu. V kapitole 6.2.3 je popísaná implementácia klasifikátora a trénera klasifikátora a v poslednej kapitole 6.2.4 sa nachádza popis implementácie pomocných nástrojov potrebných k detekcií morphovaných odtlačkov prstov.

6.2.1 Databáza

Modul obsahuje implementáciu dvoch základných komponentov určených k načítaniu databáze a samotnú databázu. Databázu tvorí rodičovská abstraktná trieda `DatabaseBase` definujúca základné (tiež abstraktné) operácie nad databázou ako je nastavenie nových dát `set_data()`, získavanie veľkosti `shape()`, pridanie jedného dáta `append()` a vyprázdnenie databáze `clean()`. Z abstraktnej triedy dedí trieda `Database` definujúca konkrétne operácie nad databázou. Typ dát databáze bol z optimalizačných dôvodov nastavený na numpyovské viacrozmerné pole `ndarray`, ktoré má tvar (n, w, h), kde n je počet dát, w je ich šírka a h výška. V prípade dát rôznych veľkostí je nutné nastaviť dátový typ databáze na `object` a pole dát potom bude mať tvar $(n,)$. Implementácia základných operácií bola rozšírená o niekoľko metód, najpodstatnejšou z nich je magická metóda pythonu `__geti-`

`tem__` umožňujúca priamo nad objektom `Database` iterovať, bez priameho prístupu k jeho dátam. Funkčnosť triedy `Database` ďalej rozširujú tri triedy:

1. `ImageDB2Dx1` - databáza určená k uloženiu množiny jednokanálových 2D obrázkov. Funkcionalitu rozširuje hlavne v metóde `to_patches(...)`, ktorá umožňuje transformovať množinu obrázkov do tzv. patchov (malé časti obrázkov) definovaných prostredníctvom `PatchInfo` objektu - bude popísané v kapitole 6.2.4). Z `PatchInfo` si preberá veľkosť patchu, t. j. $w \times h$, kde w predstavuje šírku a h výšku patchu, a veľkosť kroku s – krok posunutia novo extrahovaného patchu z obrázku od predošlého patchu. Extrakcia patchov z obrázkov prebieha z optimalizačných dôvodov prostredníctvom pythonovskej knižnice patchify. Ďalším podstatným rozšírením funkčnosti je metóda `flatten(...)` transformujúca obrázky/patche z 2D polí do 1D polí, tak že jednotlivé riadky su konkatenované za sebou. Sploštenie dát prebieha numpyovskou funkciou `flatten()`.

2. `FilterDB` - databáza určená k uloženiu filtrov vytvorených v kapitole 6.3. Nijak funkčnosť databáze nerozširuje, jediným významom existencie tejto triedy, je zachovanie konzistencie implementácie.

3. `HistogramDB` - databáza určená k uloženiu histogramových dát. Funkcionalitu rozširuje v metóde `normalize(...)` zabezpečujúcej normalizáciu uložených histogramov do rozsahu $\langle 0,\ 1)$ nasledujúcim spôsobom:

$$\mathbf{hist}_j = \frac{\mathbf{hist}_j}{\sum_i^N \mathbf{hist}_i}, \tag{76}$$

   kde $\mathbf{hist}_j$ je aktuálna hodnota histogramu z N prvkov.

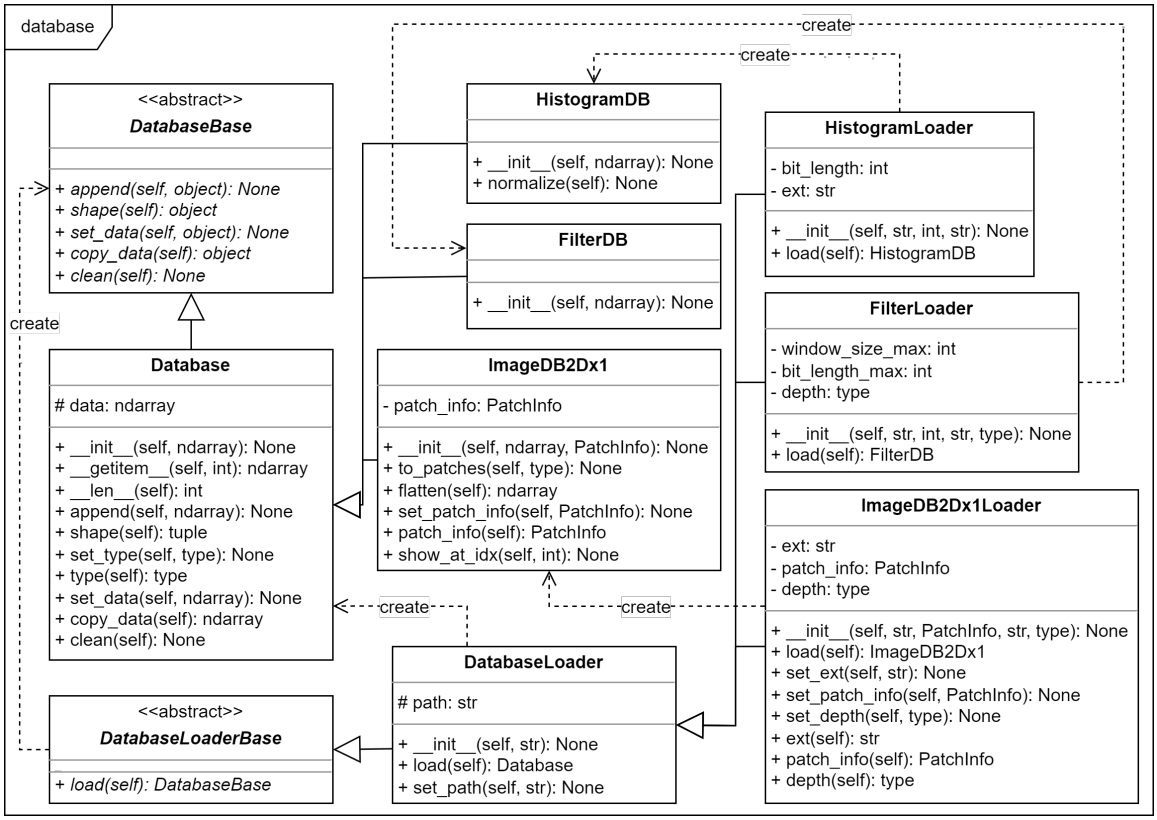
Za načítanie databáze je zodpovedná rodičovská abstraktná trieda `DatabaseLoaderBase` definujúca základné abstraktné operácie. Z nej dedí už konkrétna trieda `DatabaseLoader` definujúca načítanie dát a nastavenie parametrov databázi `Database`. Funkčnosť `DatabaseLoader` rozširujú tiež tri triedy odpovedajúce typom databáz z predošlej časti:

1. `ImageLoader2Dx1` - načítava obrázky na základe definovanej cesty k obrázkom `path`, špecifikácie patchu `patch_info`, koncovky súboru `ext` a dátového typu `depth`. Trieda z načítaných obrázkov vytvorí databázový objekt typu `ImageDB2Dx1` a nastaví jej špecifikáciu patchov `patch_info`. Pri načítavaní obrázkov prebieha konverzia do dátového typu `depth`, avšak obrázky nie sú rozdeľované do patchov, databázi je len nastavená ich špecifikácia. Dátový typ `depth` bol pomenovaný ako hĺbka z dôvodu zachovania konvencie knižnice opencv, v ktorej hĺbka obrázku je považovaná práve za dátový typ.

2. `FilterLoader` - načítava filtre vytvorené v kapitole 6.3 na základe špecifikovanej maximálnej veľkosti okna filtru `window_size_max` a maximálneho počtu okien filtru `bit_length_max`. Počet okien filtru je považovaný za bitovú dĺžku z dôvodu extrakcie vlastností BSIF z kapitoly 5.3.2, ktoré výstup okien, pri operácií konvolúcie, kvantizujú na hodnoty 0 a 1 (bity). Trieda z načítaných filtrov vytvorí databázový objekt typu `FilterDB`.

3. `HistogramLoader` - načítava histogramy na základe definovanej cesty `path`, bitovej dĺžky filtru `bit_length`, z ktorého histogram vznikol, a koncovky súboru `ext`. Z načítaných histogramov sa následne vytvorí databázový objekt typu `HistogramDB`.

Diagram popísanej implementácie databázového modulu je možné vidieť na obrázku 6.5.



Obrázok 6.5. Diagram implementácie modulu `database`.

6.2.2 Štatistické vlastnosti obrazu

Modul obsahuje implementáciu komponentov týkajúcich sa extrakcie vlastností obrazu. Je rozdelený na dve časti, štatistiky obrazu a extraktory deskriptorov obrazu. Štatistiky obrazu obsahujú implementáciu algoritmov slúžiacich k extrakcií obrazových štatistík ako PCA, ICA a kanonické predspracovanie potrebné k hľadaniu nezávislých komponentov z kapitoly 5.3.1. Z optimalizačných dôvodov triedy `PCA` a `ICA` v skutočnosti neobsahujú ručnú implementáciu týchto analýz, ale dedia ju z knižnice sklearn a jej modulu decomposition. Konkrétne sa jedná o sklearn triedy `PCA` a `FastICA`. Dôvodom podedenia implementácie z knižnice sklearn triedam štatistickému modulu bolo zachovanie konvencie v implementácií knižnice **MorphingDetection**. Kanonické predspracovanie je implementované prostredníctvom statickej triedy `Preprocessing`, ktorej činnosť vykonáva metóda `cannonical(...)` nasledujúcim spôsobom:

1. Vycentrovanie dát okolo počiatku súradnicovej sústavy:

$$\bar{\mathbf{I}}(x, y) = \mathbf{I}(x, y) - \frac{1}{M} \sum_{i,j} \mathbf{I}(i, j), \tag{77}$$

   kde $\mathbf{I}$ je patch z množiny patchov $\mathbf{X}$, $[x, y]$ je aktuálna súradnica pixelu a M je počet pixelov patchu $\mathbf{I}$.

2. Extrakcia hlavných komponentov $\mathbf{y}$ pomocou triedy `PCA` z množiny patchov $\mathbf{X}$.

3. Redukcia dimenzionality:

$$\overline{\mathbf{y}} = \mathbf{y}_{0:n}, \tag{78}$$

kde n je počet požadovaných prvých komponentov. Tento postup predpokladá zoradenie hlavných komponentov $\mathbf{y}$ zostupne podľa ich variácie.

4. Vybielenie dát:

$$\mathbf{s}_i = \frac{\overline{\mathbf{y}}_i}{\sqrt{var(\overline{\mathbf{y}}_i)}}, \tag{79}$$

kde $\mathbf{s}_i$ je vybielený hlavný komponent.

Druhá časť modulu zahŕňajúca extraktory deskriptorov obrazu obsahuje implementáciu extraktoru BSIF z kapitoly 5.3.2. Rodičovská abstraktná trieda `ImageFeatureExtractorBase` definuje základné abstraktné operácie a to nastavenie filtrov metódou `set_filters(...)` a operáciu konvolúcie metódou `convolve(...)`. Z rodičovskej triedy dedí trieda `ImageFeatureExtractor` implementujúca abstraktnú metódu `set_filters(...)`. Z nej dedí trieda implementujúca BSIF extraktor z kapitoly 5.3.2 - implementuje abstraktnú metódu `convolve(...)` nasledujúcim spôsobom:

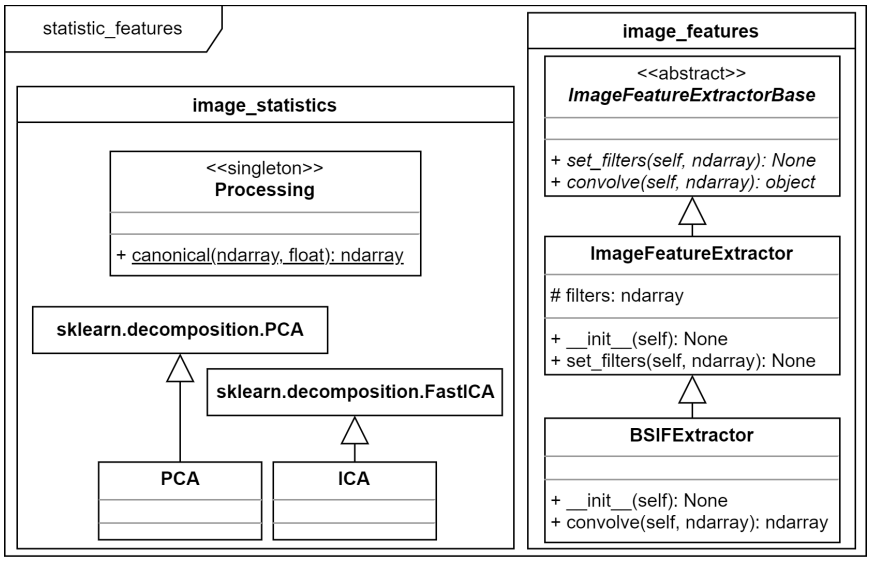
1. Pre každé okno filtra sa spúšť operácia konvolúcie pomocou opencv metódy `filter2D(...)`.

2. Výstup každého okna y, pre všetky pixely $[x, y]$ obrazu $\mathbf{I}$ je kvantizovaný podmienkou:

$$\mathbf{b}_i = \begin{cases} 1, & y_i \geq 0, \\ 0, & inak, \end{cases} \tag{80}$$

takže jeden pixel $[x, y]$ obrazu $\mathbf{I}$ má, po aplikovaní filtru, deskriptor $\mathbf{b}(x, y)$ nastavený na hodnotu napr. $[0, 1, 0, 0, 1]$. Dĺžka bitového vektora je závislá na počte okien, ktoré použitý filter obsahuje.

Diagram popísanej implementácie je možné vidieť na obrázku 6.6.



Obrázok 6.6. Diagram implementácie modulu `statistic_features`.

6.2.3 Klasifikátor

Modul obsahuje implementáciu klasifikátora SVM z kapitoly 5.3.3 a trénera klasifikátora. Diagram implementácie popísanej v nasledujúcich odsekoch je možné vidieť na obrázku 6.7. Klasifikátoru opäť kôli optimálnosti nie je implementovaný ručne, ale funkcionalita je podedená z knižnice sklearn z modulu svm.SVC triedou `SVM`. Trieda `SVM` okrem toho rozširuje funkcionalitu klasifikátora o možnosť uloženia modelu metódou `save(...)` a načítanie modelu statickou metódou `load(...)`.

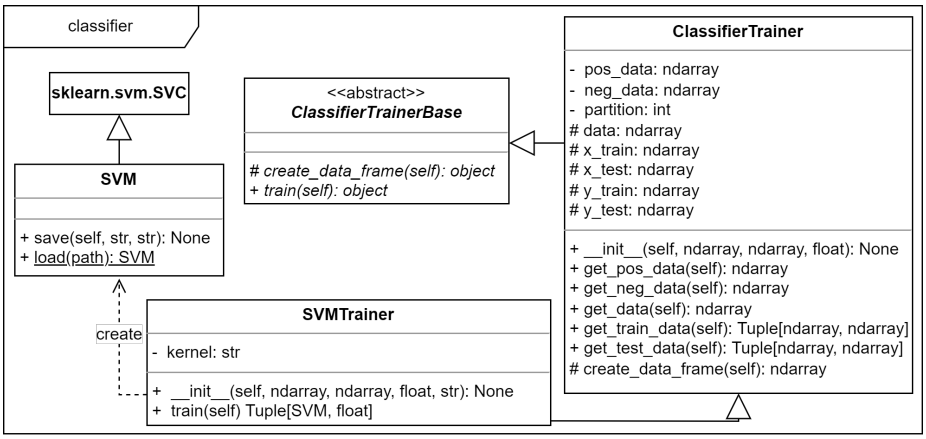
Trénera klasifikátora tvorí rodičovská abstraktná trieda `ClassifierTrainerBase` definujúca základné abstraktné operácie a to sú vytvorenie dátového rámca pre tréning metódou `create_data_frame(...)` a samotné trénovanie metódou `train()`. Z tejto abstraktnej triedy dedí trieda `ClassifierTrainer` implementujúca zdedenú abstraktnú metódu `create_data_frame(...)` nasledujúcim spôsobom:

1. `ClassifierTrainer` prostredníctvom konštruktora dostáva pozitívne dáta `pos_data`, negatívne dáta `neg_data` a percentuálnu hodnotu `percentage` udávajúcu časť dát určených k trénovaniu (zostatková časť je použitá na testovanie).

2. K pozitívnym dátam `pos_data` sú priradené jednotky a k negatívnym dátam `neg_data` sú priradené nuly. Priradené hodnoty predstavujú triedy do ktorých dáta zapadajú, čo v kontexte tejto práce znamená, že pozitívne dáta označené hodnotou jedna sú morphované odtlačky, v opačnom prípade ide o obyčajný odtlačok.

3. Dáta sú náhodne pomiešané a rozdelené pomocou `percentage` na trénovacie (videné) a testovacie (nevidené).

`SVMTrainer` rozširuje funkčnosť `ClassifierTrainer` o parameter konštruktora `kernel` a implementáciu abstraktnej metódy `train(...)`. V metóde `train(...)` vytvára objekt typu `SVM` a parametrom mu zadáva kernel funkciu, ktorú SVM využíva na transformáciu dát, k nájdeniu optimálnej rozhodovacej hranice (viď kapitola 4.1). Následne na trénovacích dátach spúšťa tréning modelu `SVM` metódou `fit(...)` a po dotrénovaní predikuje odpovede na testovacích dátach metódou `predict(...)`. Z výstupu natrénovaného modelu na testovacích dátach sa odhadne presnosť predikcie pomocou sklearn metrík metódami `accuracy_score(...)` a `classification_report(...)`, kde `accuracy_score(...)` vracia presnosť v percentách a `classification_report(...)` f1-skóre metriku pre obe predpove-



Obrázok 6.7. Diagram implementácie modulu `classifier`.

dané triedy, počet support vektorov oboch tried a priemery. Metóda `train(...)` následne vráti natrénovaný model spolu s jeho percentuálnou presnosťou a zaloguje správu z `classification_report(...)`.

6.2.4 Pomocné nástroje

Modul obsahuje implementáciu pomocných nástrojov k spracovávaniu dát, modelovaniu dát, definície typov, loger a iné. V tomto module sa nachádza aj implementácia už vyššie využívaného `PatchInfo`. Jeho komponentami sú integerová hodnota `step` a veľkosť patchu, ktorú definuje `PatchSize`. `PatchSize` dedí funkčnosť `tuple`, pod podmienkou, že nesmie mať iný počet prvkov než dva a na prvok indexu nula nastavuje `property(...)` nazvanú `width` a na prvok indexu jedna `height`. K prvkom `PatchSize` je tedá možné pristupovať okrem indexovaním aj kľúčovými slovami `PatchSize.width` a `PatchSize.height`.

Ďalšou významnou metódou je `segment(...)` vykonávajúca segmentáciu z kapitoly 5.1.1, avšak implementácia musela byť upravená tak, aby segmentácia prebehla čo najefektívnejšie (python ako interpretovaný jazyk je veľmi pomalý). Segmentácia prebieha prostredníctvom troch operácií konvolúcie použitím opencv `filter2D(...)` (dve pre zistenie distribúcie štandardných odchýlok a jedna pre segmentáciu). Najprv sa definuje priemerovací 2D kernel na základe parametru `kernel_size`, s ktorým sa vykoná konvolúcia na obraze $\mathbf{I}$, z čoho je získaný $\mathbf{I}_{mean}$. Následne sa spočíta druhá mocnina rozdielu obrázkov pre každý pixel $[x, y]$ nasledovne:

$$\mathbf{I}_{diff}(x, y) = (\mathbf{I}(x, y) - \mathbf{I}_{mean}(x, y))^2. \tag{81}$$

Znova sa na $\mathbf{I}_{diff}$ vykoná operácia konvolúcie priemerovacím filtrom, čím je získaná variácia pre každý pixel obrazu $\mathbf{I}_{var}$. Štandardná odchýlka sa potom získava nasledovne:

$$\mathbf{I}_{std}(x, y) = \sqrt{\mathbf{I}_{var}(x, y)}. \tag{82}$$

Následne je spočítaná globálna štandardná odchýlka celého obrazu t_{std}, ktorá je po vynásobení škálovacím parametrom *treshold_scale* (označený ako t_{scale}) použitá ako prahová hodnota:

$$\mathbf{B}(x, y) = \begin{cases} 1, & \mathbf{I}_{std}(x, y) > t_{std} * t_{scale}, \\ 0, & inak, \end{cases} \tag{83}$$

V poslednom kroku prebieha korekcia segmentácie konvolúciou jednotkovým kernelom na odhadnutej maske $\mathbf{B}(x, y)$, z čoho sú získané lokálne súčty $\mathbf{B}_{sums}$, na ktorých prebieha opätovným vyprahovaním korekcia segmentácie:

$$\overline{\mathbf{B}}(x, y) = \begin{cases} 1, & \mathbf{B}_{sums}(x, y) = w * h, \\ 0, & inak, \end{cases} \tag{84}$$

kde w a h je šírka a výška kernelu. Rovnica 84 hovorí, že ako popredie sú považované len tie pixely $[x, y]$, v ktorých okolí $w \times h$ sa nachádzajú len samé jednotky. Striktnosť segmentácie je možné ovládať nastavením parametru `treshold_scale`.

Predposlednou významnou metódou je metóda `histogram(...)`, vykonávajúca modelovanie dát histogramom. Obraz popísaný deskriptormi BSIF prevádza z bitových vektorov na dekadické hodnoty pomocou numpyovskej funkcie `packbits(...)`. Následne vymaskuje oblasť záujmu obrazu na základe parametru `mask`. Z vymaskovanej oblasti sa vytvorí histogram s počtom košov $2^{b_{length}}$, kde b_{length} predstavuje bitovú dĺžku deskriptora BSIF. Vytvorenie histogramu prebieha prostredníctvom numpyovskej funkcie `histogram(...)`.

V rámci vyhodnotenia BSIF deskriptorov odtlačkov prstov bola doimplementovaná metóda `utils.get_keypoints_threshold(...)` k nájdeniu ich kľúčových typov deskriptorov. Iteratívne sa odhaduje priemerný počet deskriptorov presahujúci pravdepodobnosť výskytu t_p, ktorá sa každou iteráciu zvyšuje o hodnotu `step`. V každej iterácií sa odhadne absolútny rozdiel priemerných pravdepodobností výskytov deskriptorov na histogram obyčajného odtlačku h_i^{morph} a histogram morphovaného odtlačku h_i^{orig} nasledovne:

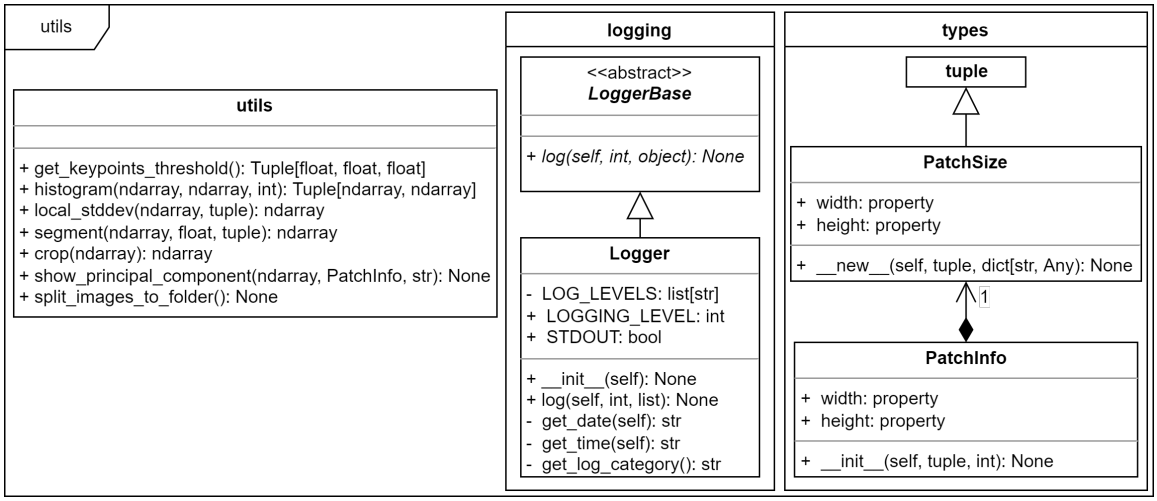
$$\mathbf{d}_i = \left| \frac{\sum_j k(\mathbf{h}_{i,j}^{morph}, t_p)}{2^{b_l}} - \frac{\sum_j k(\mathbf{h}_{i,j}^{orig}, t_p)}{2^{b_l}} \right|, \tag{85}$$

kde $k(x, t)$ predstavuje nasledujúcu funkciu:

$$k(x,t) = \begin{cases} 1, & x \geq t, \\ 0, & inak. \end{cases} \tag{86}$$

Ako finálna prahová hodnota minimálnej pravdepodobnosti výskytu deskriptoru t_{max} je nakoniec zvolená taká t_p, ktorá maximalizuje absolútny rozdiel priemerov $\mathbf{d}_i$.

Modul obsahuje implementáciu aj ďalších nástrojov určených na vizualizáciu dát, grafov, obrazov, ďalej orezávania či logger, ale vzhľadom na tú diplomovú prácu ich popis nie je opodstatnený. Diagram popísanej implementácie je možné vidieť na obrázku 6.8.



Obrázok 6.8. Diagram implementácie modulu `utils`.

6.3 Generovanie prírodných filtrov

O generovanie filtrov (kapitola 5.3.1) sa stará skript `generate_filters.py` použitím knižnice popísanej v predošlej kapitole 6.2. Generovanie prebieha iteratívne na základe definovaného rozsahu veľkostí okien filtrov `window_range`. Pomocou zadanej cesty k obrázkom `resources`, ich koncovky `ext` a veľkosti patchu `patch_info` nastaveného ako `PatchInfo(window_range[i], 1)`, je prostredníctvom `ImageLoader2Dx1` z kapitoly 6.2.1 načítaný databázový `image_db` triedy `ImageDB2Dx1` obsahujúci obrázky zo zložky `resources`. Vzhľadom na to, že obrázky na definovanej ceste `resources` môžu mať rôzne veľkosti, ich dátový typ je nastavený na `object`. Dôvodom je databáza typu `ImageDB2Dx1` obsahujúca

dáta vo formáte numpyovského `ndarray` (nevie uložiť rôzne veľkostí 2D polí - obrázkov). Následne sú obrázky transformované do patchov použitím `image_db.to_patches(float32)`, a ich typ je prekonvertovaný na 32-bitový `float` (každý patch má rovnakú veľkosť). Ďalším krokom je sploštenie patchov databáze použitím `image_db.flatten()` a je z nej vytiahnutých `n_patches` náhodných patchov. Tieto patche sú kanonicky predspracované pužitím `Preprocessing.cannonical(...)` (viď kapitolu 6.2.2), čím im je znížená dimenzionalita na základe špecifikovaného parametru `dim_reduction`. V poslednom kroku sa pre každú veľkosť okna filtra interuje na rozsahu, definovanom parametrom `bit_length_range`. Tento rozsah v každej poditerácií, definuje počet okien, ktoré budú pre aktuálnu veľkosť okna `window_range[i]` generované z `n_patches` patchov, ako nezávislé komponenty jednotlivo popisujúce rozloženia pravdepodobnosti najvzdialenejšie od gaussovského rozloženia (centrálny limitný teorém - viď kapitola 3.2.8). Túto analýzu odhadnutia nezávislých komponentov zabezpečuje trieda `ICA` popísaná v kapitole 6.2.2. Následne sa odhadnutý filter s veľkosťou okna `window_range[i]` a bitovou dĺžkou `bit_length_range[j]` uloží do zložky `output_folder` pod názvom `ixi_j.npy` (formát ukladania v .npy súbore má optimalizačné účely - rýchlejšie načítanie/uloženie). Popis je možné vidieť v pseudokóde 1.

Pseudokód 1 generate_filters.py

```
for i in window_range do
    Načítaj obrázky
    Transformuj do patchov o veľkosti i × i
    Splošti a vyber n_patches náhodných
    for j in bit_length_range do
        Nájdi j nezávislých komponentov
        Ulož filter do output_folder pod názvom ixi_j.npy
    end for
end for
```

Pre jednoduchšiu prácu so skriptom pola doimplementovaná možnosť použitia parametrov pri spustení. Všetky popisované parametre je možností vidieť v tabuľke 7.4. Príklad spustenia skriptu je nasledujúci:

```
python generate_filters.py resources --out-folder filters --window-range 3
17 --bit-length-range 1 12 --enable-stdout True
```

Parameter	Argument	Typ
resources	/cesta/k/zlozke	string
ext	--ext [ext]	string
window_range	--window-range [min] [max]	integer integer
bit_length_range	--bit-length-range [min] [max]	integer integer
n_patches	--n-patches [n_patches]	integer
dim_reduction	--dim-reduction [dim_reduction]	float
out_folder	--out-folder [folder]	string
logger.LOGGING_LEVEL	--log-level [level]	integer
logger.STDOUT	--enable-stdout [enable]	boolean

Tabuľka 6.2: Popis argumentov skriptu nastavujúce potrebné parametre algoritmu.

6.4 Generovanie BSIF

O generovanie vlastností BSIF (5.3.2) z databáze obrázkov sa stará skript `generate_bsif.py` použitím knižnice popísanej v predošlej kapitole 6.2. V prvom rade sa na základe maximálnej veľkosti okna `max_window_size` a bitovej dĺžky filtra `max_bit_length` načítajú filtre zo zložky `filters_folder`, pomocou `FilterLoader` z kaptiloy 6.2.1, ako databázový objekt `filter_db` typu `FilterDB`. Následne sa načíta databáza obrázkov `image_db`, z ktorých sa budú získavať BSIF. Databáza je typu `ImageDB2Dx1` a je vytvorená pomocou `ImageLoader2Dx1` na zákalde špecifikovanej zložky `in_folder`, v ktorej sa obrázky nachádzajú, a koncovky `ext`. Dátovy typ obrázkov je znova nastavený na `object` kvôli podpore rôznych veľkostí v databáze `image_db`.

Ďalším krokom je iterovanie cez databázu `image_db` a získanie masiek `mask` segmentáciou jednotlivých obrázkov pomocou `utils.segment(...)` (viď kapitola 6.2.4). Následne je pre každý obrázok `image_db[i]` spolu s jeho maskov `mask` extrahovaný BSIF deskriptor každého filtru `filter_db[j]`. Deskriptory obrázku `image_db[i]` sú ziskavané pomocou extraktora `extractor` typu `BSIFExtractor` (viď kapitola 6.2.2), ktorému sú postupne nastavované jednotlivé filtre `filter_db[j]` pomocou metódy `extractor.set_filters(...)`. Získanné desktriptory obrázkov metódou `extractor.convolve(...)`, spolu s maskou `mask`, sú následne prevádzané do histogramu pomocou `utils.histogram(...)` viď kapitoly 6.2.4. Histogramy a ďalšie medzivýsledky sa ukladajú do zložky špecifikovanej parametrom `out_folder` po názvom `i_counts_bl.npy` (**bl** predstavuje bitovú dĺžku filtra, s ktorým boli deskriptory extrahované). Popis je možné vidieť v pseudokóde 2.

Pseudokód 2 generate_bsif.py

```
Vytvor extraktor
Načítaj filtre filter_db
Načítaj obrázky image_db
for i, image in enum(image_db) do
    Získaj masku z image
    for filter in filter_db do
        Zisti bitovú dĺžku filtra b
        Nastav filter extraktoru
        Získaj vlastnosti BSIF
        Z vymaskovanych vlastnosti vytvor histogram
        Ulož filter do output_folder pod názvom i_counts_b.npy
    end for
end for
```

Pre jednoduchšiu prácu so skriptom pola doimplementovaná možnosť použitia parametrov pri spustení. Všetky popisované parametre je možností vidieť v tabuľke 7.7. Príklad spustenia skriptu je nasledujúci:

```
python generate_bsif.py fingerprints filters --out-folder bsif_features
--enable-stdout True
```

Parameter	Argument	Typ
in_folder	/cesta/k/zlozke	string
filters_folder	/cesta/k/zlozke	string
ext	--ext [ext]	string
max_window_size	--max-window-size [max]	integer
max_bit_length	--max-bit-length [max]	integer
out_folder	--out-folder [folder]	string
logger.LOGGING_LEVEL	--log-level [level]	integer
logger.STDOUT	--enable-stdout [enable]	boolean

Tabuľka 6.3: Popis argumentov skriptu nastavujúce potrebné parametre algoritmu.

6.5 Trénovanie klasifikátorov

O trénovanie klasifikátorov (viď kapitola 5.3.3) z histogramov sa stará skript `train_models.py` použitím knižnice popísanej v predošlej kapitole 6.2. Generovanie prebieha iteratívne na základe definovaného rozsahu veľkostí okien `window_range` a rozsahu bitových dĺžok `bit_length_range`, z ktorých histogramy vznikly (histogramy generované skriptom `generate_bsif.py`). Pomocou zadanej cesty k histogram pozitívnej triedy `pos_hists_folder` a negatívnej triedy `neg_hists_folder` (v kontexte tejto práce morphované a obyčajné odtlačky prstov), aktuálneho rozsahu `window_range[i]`, bitovej dĺžky `bit_length_range[j]` a koncovky `ext`, sú generované cesty ku konkrétnym histogramovým dátam, z ktorých sa prostredníctvom `HistogramLoader` načítavajú databáze `histogram_pos_db` a `histogram_neg_db` typu `HistogramDB` (viď kapitola 6.2.1). Dáta v databázach sú následne normalizované použitím `histogram_pos_db.normalize()` a `histogram_neg_db.normalize()`.

V ďalšom kroku sú databáze dát predané trénerovi klasifikátoru `SVMTrainer` z kapitoly 6.5, spolu s parametrom `split` (percentuálny podiel trénovacích a testovacích dát) a `kernel` (transformačná funkcia použitá na dáta). Natrénovanému modelu sa skontroluje jeho presnosť a v prípade najvyššej presnosti sa uloží ako najlepší nájdený model **best_model** do zložky špecifikovanej parametrom `out_folder` pod názvom filtra, z ktorého histogramy vznikly. Popis je možné vidieť v pseudokóde 3.

Pseudokód 3 train_models.py

```
for i in window_range do
    Vytvor cestu pos_hists_path k histogram vzniknutých z okien filtrov veľkosti i × i
    Vytvor cestu neg_hists_path k histogram vzniknutých z okien filtrov veľkosti i × i
    for j in bit_length_range do
        Načítaj pozitívne histogramy z cesty pos_hists_path a bitovej dĺžky j
        Načítaj negatívne histogramy z cesty neg_hists_path a bitovej dĺžky j
        Natrénuj SVM model na dátach
        Zisti presnosť
        Ak sa jedná o najelpší model ulož do out_folder pod názvom ixi_j.sav
    end for
end for
```

Pre jednoduchšiu prácu so skriptom pola doimplementovaná možnosť použitia parametrov pri spustení. Všetky popisované parametre je možností vidieť v tabuľke 6.4. Príklad spustenia skriptu je nasledujúci:

```
python train_models.py histograms/morphed histograms/orig --enable-stdout
True
```

Parameter	**Argument**	**Typ**
pos_hists_folder	/cesta/k/zlozke	string
neg_hists_folder	/cesta/k/zlozke	string
ext	--ext [ext]	string
window_range	--window-range [min] [max]	integer integer
bit_length_range	--bit-length-range [min] [max]	integer integer
svm_kernel	--svm-kernel [kernel]	string
split	--split [split]	float
out_folder	--out-folder [folder]	string
logger.LOGGING_LEVEL	--log-level [level]	integer
logger.STDOUT	--enable-stdout [enable]	boolean

Tabuľka 6.4: Popis argumentov skriptu nastavujúce potrebné parametre algoritmu.

Kapitola 7

Vyhodnotenie

Táto kapitola sa zaoberá testovaním a vyhodnotením implementovanej aplikácie. Vyhodnotenie prebieha na morphovaných odlačkoch prstov, vygenerovaných z databáze reálnych a syntetických odtlačkov, zloženej celkom z 1498 odtlačkov. Z nich je 949 reálnych a 549 odtlačkov bolo vygenerovaných synteticky pomocou nástroja Anguli[1]. Reálne odtlačky sa ďalej delia na: a) 330 odtlačkov zprostredkovaných senzorom Sagem MSO, b) 330 senzorom SecuGen, c) 289 senzorom Bergdata.

V prvom rade bolo nutné vygenerovať morphované odtlačky prstov, parametre a podmienky generovania sú popísané v kapitole 7.1. Popis vyhodnotenia generovania filtrov ako nezávyslých komponentov obrázkov prírody, sa nachádza v kapitole 7.2. Vyhodnotenie extrakcie BSIF deskriptorov sa nachádza v kapitole 7.3 a vyhodnotenie učenia vlastností deskriptorov SVM v poslednej kapitole 7.4.

7.1 Generovanie dátovej sady

Dátová sada morphovaných odtlačkov prstov je vygenerovaná prostredníctvom doimplementovaného modulu z kapitoly 6.1.1. Morphing prebieha vždy len nad jednou konkrétnou sadou odtlačkov prstov, t. j. odtlačky z rôznych senzorov, poprípade syntetické odtlačky, nie sú morphované dokopy. Dôvodom je výrazný vznik nových vlastnastí v morphova-

(a) (b) (c)

Obrázok 7.1. Morphing odtlačkov získaných z rôznych senzorov. (a) Odtlačok zo senzoru Bergdata, (b) odtlačok zo senzoru SecuGen, (c) morphovaný odtlačok. [4]

[1] https://dsl.cds.iisc.ac.in/projects/Anguli/userguide.html

(a) (b) (c)

Obrázok 7.2. Morphing odtlačkov prstov. (a) Prvý odtlačok, (b) druhý odtlačok, (c) morphovaný odtlačok. [4]

ných odtlačkoch, vďaka ktorým sú jednoducho rozlíšiteľné od obyčajných odtlačkov prstov (viď obrázok 7.1 a 7.2). Dôraz pri detekcií morphingu je kladený na také morphované odtlačky, ktoré sú čo najpodobnejšie obyčajným odtlačkom prstom. Generovanie dátovej sady teda prebieha oddelene pre každý senzor a syntetické odtlačky, a morphovaný odtlačok musí nadobudnúť najprísnejšie verifikačné skóre FAR 0,01 %, podľa pokynov FRONTEX z kapitoly 6.1, oproti obom pôvodným odtlačkom, z ktorých vznikol. Nastavenie parametrov potrebných k vygenerovaniu dátovej sady je možné vidieť v tabuľke 7.1.

Parameter	**Hodnota**
currentSensor	bergdata, sagemmso, secugen, synthetic
t_{score}	0,1329

Tabuľka 7.1: Parametre potrebné k vygenerovaniu dátovej sady algoritmom z kapitoly 6.1.1.

Bolo vygenerovaných 8 dátových sád morphovaných odtlačkov prstov odpovedajúcich jednotlivým kategóriám senzorov (Bergdata, Sagem MSO, SecuGen a syntetické) a metódam, ktorými boli morphované odtlačky generované (pôvodná a adaptívna z mojej projektovej praxe [5]). Následne boli z morphovaných odstránené odtlačky, ktorých skóre verifikácie nepresiahlo t_{score}. Každej sade obyčajných odtlačkov prstov bolo náhodne zvolených toľko morphovaných odtlačkov, koľko sa nachádzalo v databáze obyčajných. Napríklad pre Bergdata bolo vygenerovaných 867 morphovaných odtlačkov pôvodnou metódou, 197 bolo odstránených kvôli nenadobudnutiu dvojitej identity a zo zvyšných 670 bolo náhodne zvolených 289. Dátová sada Bergdata generovaná pôvodnou metódou teda obsahuje 578 odtlačkov prstov (50 % obyčajných a 50 % morphovaných). Prehľad celej vygenerovanej databáze pre pôvodnú a novú metódu morphingu je možné vidieť v tabuľkách 7.2 a 7.3.

	Obyčajné	Morphované			Dátová sada
		Vygenerované	Dvojitá identita	Náhodne zvolené	
Bergdata	289	867	670	289	578
Sagem MSO	330	990	791	330	660
SecuGen	330	990	784	330	660
Syntetické	549	1647	1302	549	1098

Tabuľka 7.2: Dátové sady morphovaných odtlačkov prstov generované pôvodnou metódou morphingu.

	Obyčajné	Morphované			Dátová sada
		Vygenerované	Dvojitá identita	Náhodne zvolené	
Bergdata	289	867	780	289	578
Sagem MSO	330	990	867	330	660
SecuGen	330	990	884	330	660
Syntetické	549	1647	1453	549	1098

Tabuľka 7.3: Dátové sady morphovaných odtlačkov prstov generované adaptívnou metódou morphingu z projektovej praxe [5].

7.2 Generovanie prírodných filtrov

Ako bolo popísané v kapitole 6.3, generovanie filtrov ako nezávislých komponentov obrázkov prírody zahrňuje niekoľko krokov. V prom rade je načítaných 13 šedotónových obrázkov prírody (niektoré je možné vidieť na obrázku 7.3). Tieto obrázky sú načítavané z dôvodu

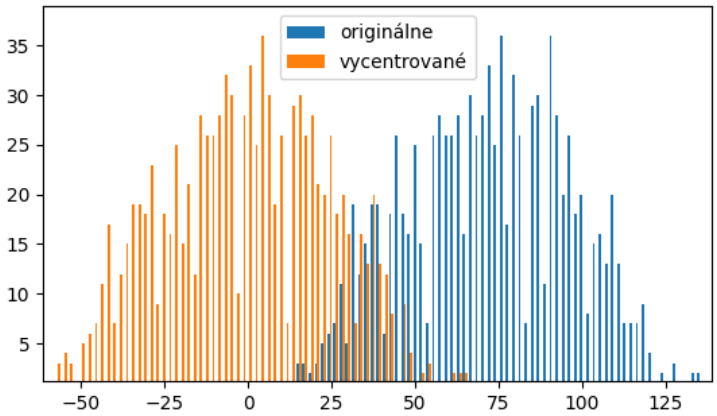
(a) (b)

Obrázok 7.3. Ukážka dvoch obrázkov prírody.

snahy modelovania takého rozloženia pravdepodobnosti dát, ktoré je čo najpodobnejšie tomu, čomu sa počas evolúcie prispôsobovalo ľudské oko (pre bližší popis teórie štatistiky obrázkov prírody viď kapitola 3.2.6). Následne sú tieto obrázky rozdelené do `n_patches` patchov o veľkosti `window_range`×`window_range`, pre každú veľkosť z definovaného rozsahu, pre krok posunutia `step` fixne nastaveného na hodnotu jedna. Krok posunutia `step` bol fixne nastavený na hodnotu jedna z dôvodu získania všetkých možných patchov z obrázkov prírody, aby nasledujúca analýza mohla zahrnúť čo najviac informácií a odhadnúť tak také komponenty, ktoré budú na sebe čo najmenej závislé. Najprv nad patchami pre-

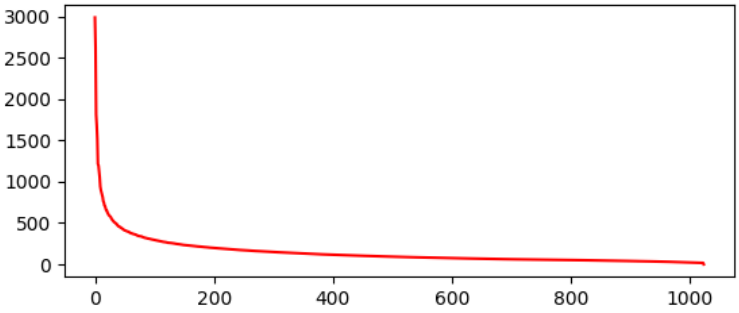
bieha kanonické predspracovanie zložené z niekoľkých krokov (bližší popis implementácie v kapitole 6.3):

1. Vycentrovanie dát - prebieha pomocou odstránenia priemeru z každého patchu. Priemer nenesie žiadnu zaujímavú informáciu pre ďalšie analýzy, takže je pre každý patch vynulovaný. Priemer patchu znamená priemerná šedotónová hodnota pixelu daného patchu. Histogram patchu pred a po odstránení priemeru je možné vidieť na obrázku 7.4.



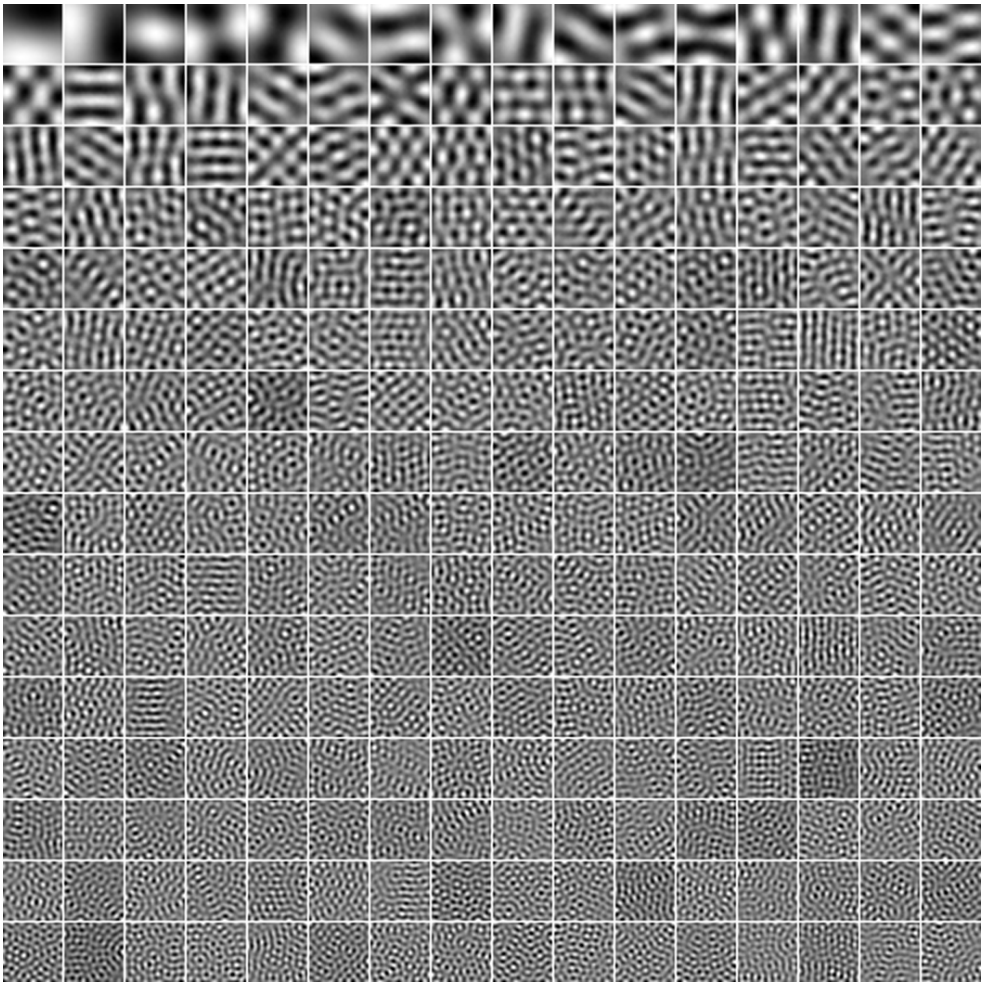
Obrázok 7.4. Histogram patchu pred a po vycentrovaní dát okolo nuly. Centrovanie predstavuje odstránenie priemernej šedotónovej hodnoty z patchu. Horizontálna os predstavuje šedotónovú hodnotu a vertikálna počet výskytov danej hodnoty v patchi.

2. Dekorelácia komponentov - prostredníctvom PCA (analýza hlavných komponentov) analýzy sú spočítané dekolerované komponenty zo všetkých patchov. Napríklad pre 2000 patchov veľkosti 32×32 prebieha transformácia do jednorozmerného poľa o 1024 prvkoch, z čoho je spočítaná kovariačná matica 1024×1024. Každý riadok matice definuje dekorelovaný komponent, pričom sú zoradené zostupne podľa ich smerodajnej odchýlky. Priebeh klesania smerodajných odchýliek je možné vidieť v grafe na obrázku 7.5.



Obrázok 7.5. Priebeh smerodajnej odchýlky komponentov odhadnutej kovariačnej matice. Horizontálna os predstavuje komponenty kovariačnej matice a vertikálna ich smerodajnú odchýlku.

3. Redukcia dimenzionality a vybielenie komponentov - ako je vidieť v predošlom grafe z obrázku 7.5, smerodajné odchýlky sa po 200. komponente menia veľmi málo a nenesú tak moc novej informácie. Z tohto dôvodu bola redukcia dimenzionality nastavená na 75 % (prvých 25 % je ponechaných). Teda napríklad pre patche veľkosti 32×32 je z 1024 dekorelovaných komponentov odhadnutej kovariačnej matice ponechaných len prvých 256, kde dochádza k naväčším zmenám smerodajnej odchýlky. Po redukcií dimenzionality je každému komponentu normalizovaná smerodajná odchýlka na jednotkovú, čim sú získané vybielené komponenty (dekorelované s jednotkovou odchýlkou). Na obrázku 7.6 je možné vidieť vizualizáciu vybielených komponentov. Je vhodné podotknúť, že vybielené komponenty sú výrazne podobné hierarchickému usporiadaniu Haarových rysov[2], ktoré majú veľmi efektívne využitie v mnohých oblastiach detekcie v počítačovom videní.

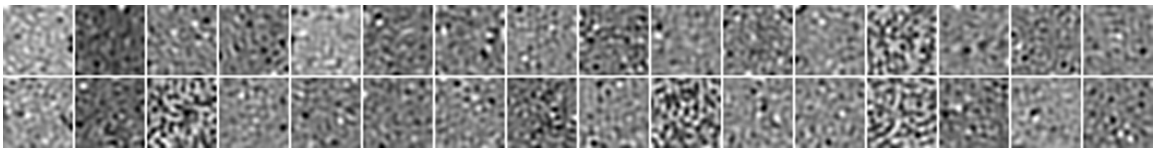


Obrázok 7.6. Prvých 256 vybielených komponentov odhadnutý prostredníctvom PCA z 50000 náhodne zvolených patchov veľkosti 32×32. V ľavom hornom rohu sa nachádza komponent s najvyššou štandardnou odchýlkou podľa PCA, smerom v pravo a po riadkoch sa postupne znižuje.

[2] https://medium.com/analytics-vidhya/what-is-haar-features-used-in-face-detection-a7e531c8332b

Posledným krokom k vygenerovaniu filtrov je odhad nezávislých kompomentov dekorelovaných dát prostredníctvom ICA (analýza nezávislých komponentov). Počet odhadnutých nezávislých komponentov je daný parametrom `bit_length_range`, t. j. pre každú veľkosť okien z rozsahu `window_range` je odhadnutý filter s rôznymi bitovými dĺžkami definovanými rozsahom `bit_length_range`. Odhad nezávislých komponentov prebieha aproximačne metódou zvanou *kurtosis*, v ktorej modelovanie distribúcie dát zabezpečuje funkcia `log-cosh` (viď kapitola 3.2.8). Funkcia bola zvolená na základe doporučení v [36].

Použitím podobnej analýzy hľadania komponentov s najvzdialenejšou distribúciou od gaussovskej (bližší popis v kapitole 5.3.1), boli vygenerované nezávislé komponenty predstavujúce okná filtru. Na obrázku 7.7 je možné vidieť okná (nezávislé komponenty) filtru veľkosti 32×32 a bitovej dĺžky 32. Parametre použité ku generovaniu je možné vidieť v tabuľke 7.4.



Obrázok 7.7. 32 nezávislých komponent o veľkosti 32×32 odhadnutých z 256 dekorelovaných komponentov na obrázku 7.6.

Parameter	**Hodnota**
window_range	3 - 32
bit_length_range	1 - 32
n_patches	50000
step	1
dim_reduction	0,75

Tabuľka 7.4: Parametre potrebné k vygenerovaniu filtrov, ako nezávyslých komponentov z obrázkov prírody, algoritmom z kapitoly 6.1.1.

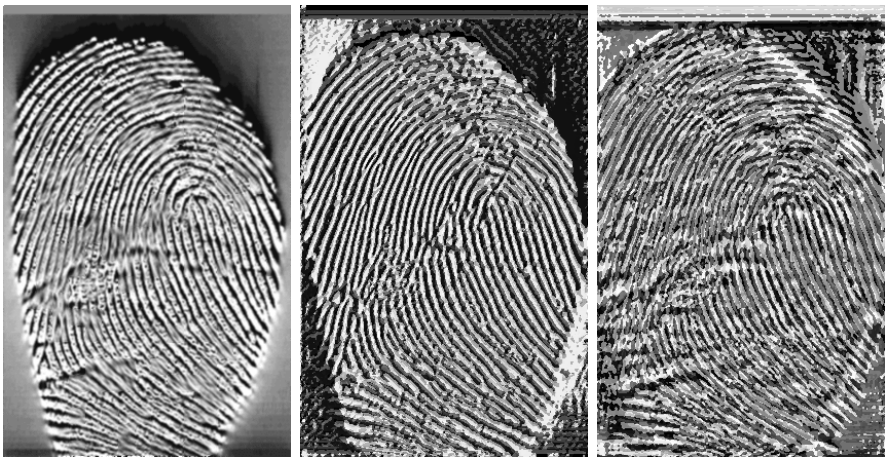
V rámci odhadu filtrov s oknami rozsahu `window_range` a počtom okien `bit_length_range` nastavených na hodnoty z tabuľky 7.4 malo byť vygenerovaných 928 rôznych filtrov. Vzhľadom na to, že ICA je aproximačná analýza, nie vždy odhad dokonvergoval k riešeniu. Podobne nesprávne odhadnuté filtre boli z vygenerovanej databáze odstránené a výsledný počet filtrov je teda 796. Prehľad generovania je možné vidieť v tabuľke 7.5

Vygenerované filtre	**Nedokonvergované filtre**	**Databáza**
928	132	796

Tabuľka 7.5: Prehľad generovania filtrov metódou ICA z dekorelovaných komponentov na obrázku 7.6.

7.3 Extrakcia BSIF deskriptorov

Ďalším krokom k uskutočneniu detekcie morphovaných odtlačkov je extrakcia BSIF deskriptorov prostredníctvom filtrov odhadnutých z obrázkov prírody (implementácia z kapitoly 6.3). V prvom rade sa načíta databáza odtlačkov prstov, postupne každá z vygenerovaných databáz morphovaných odtlačkov z kapitoly 7.1 a databáze obyčajných odtlačkov. Pre každú z načítaných databáz sa vygenerovala každým načítaným filtrom nová databáza BSIF deskriptorov rozdelených podľa filtrov rôznych veľkostí okien a rôznych bytových dĺžok. V tomto kroku neboli využité všetky filtre z rozsahu okien 3×3 - 32×32 a bitových dĺžok 1 - 32. Z priebežného testovania vyplynulo, že príliš veľké okná majú skôr devastačné účinky na extrakciu optimálnych deskriptorov z obrázkov a nenesú žiadne nezávislé vlastnosti, ktorými by prebehla úspečná detekcia morphovaných odtlačkov (viď obrázok 7.8).



(a) (b) 8×8, 10. (c) 32×32, 32.

Obrázok 7.8. Ukážka aplikácie BSIF extraktoru s rôznymi veľkosťami okien a bitovými dĺžkami filtrov. Z obrázkov je zrejmé, že pri extrakcií BSIF deskriptorov z (a), filtrom s oknom veľkým 32×32 v (c), dôjde k značnej deformácií odtlačku oproti filtru s manším oknom 8×8 v (b).

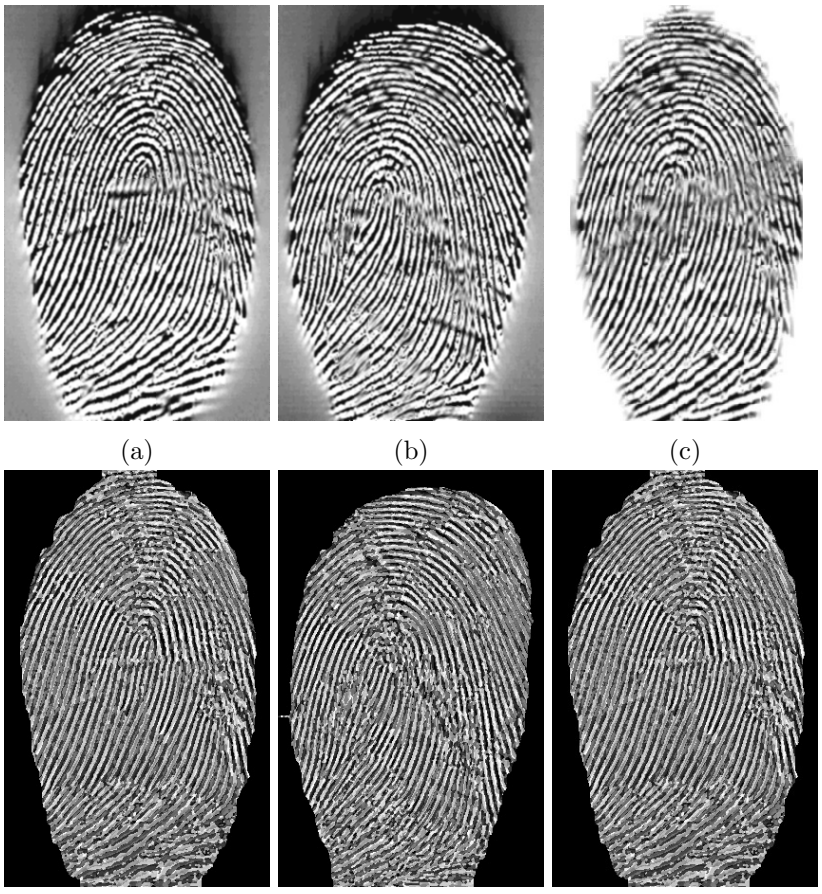
Pokus o natrénovanie klasifikátora podobne veľkými oknami končil veľmi nízkymi presnosťami rozhodovania, približne 30 %. Z tohto dôvodu bol proces generovania databáz BSIF deskriptorov obmedzený len do veľkosti okna 17×17 s maximálnou bitovou dĺžkou 16.

Z testovania tiež vyplynulo, že BSIF deskriptory s menšími oknami a bitovou dĺžkou jedna, majú celkom slušnú odozvu v binarizácií šedotónových odtlačkov prstov (viď obrázok 7.12). Dôvodom efektívnejších menších okien v binarizácií pravdepodobne bude aplkácia filtra na menšie obrázky do veľkosti 400×400 pixelov, pri použití väčších obrázkov pravdepodobne budú mať lepšiu odozvu väčšie filtre

Ostatné veľkosti filtrov popisujú obrázky bitovými vektormi, ktoré majú dĺžku v závislosti na počte okien filtra, ktoré boli odhadnuté ako kernely extrahujúce nezávislé informácie z obrázkov. Deskriptor jedného pixelu teda popisuje, či určité okolie dané veľkosťou okna, maximalizuje nezávislé vlastnosti. Príklady aplikácie filtru s veľkosťou okna 8×8 a bitovou dĺžkou 6 na obyčajný odtlačok a morphovaný odtlačok je možné vidieť na obrázku 7.10.

(a) (b) (c) (d)

Obrázok 7.9. Ukážka binarizácie odtlačkov prstov BSIF extraktorom s veľkosťou okna 8×8 a bitovou dĺžkou 1 ((a) a (b), (c) a (d)).

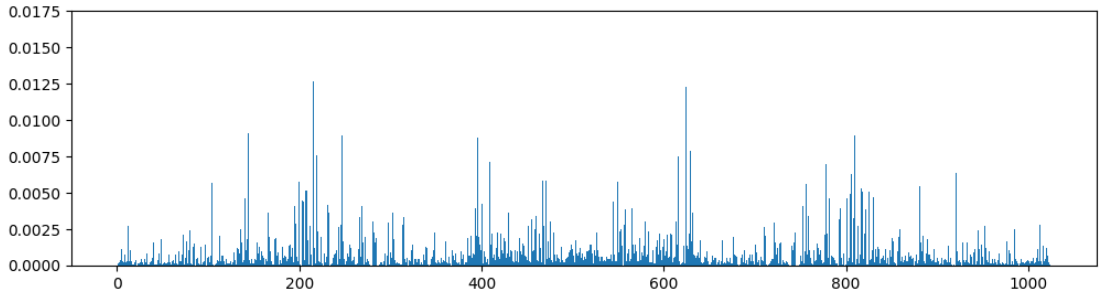


(a) (b) (c)

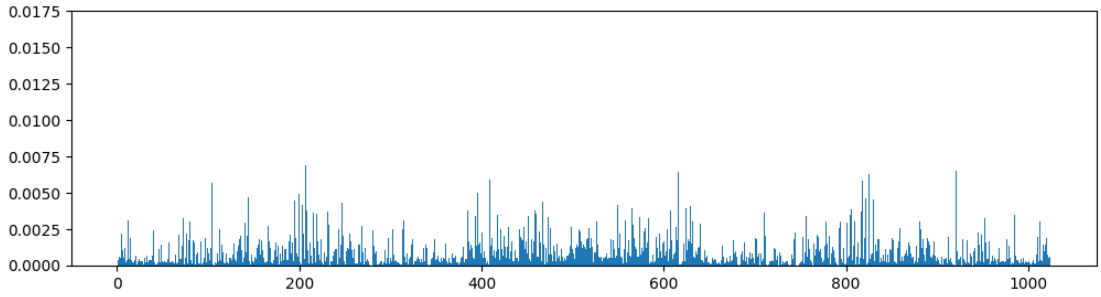
(d) (e) (f)

Obrázok 7.10. Na obrázku sa nachádzajú z pôvodných odtlačkov (a), (b) a morphovaného odtlačku (c), extrahované BSIF deskriptory (d), (e) a (f), filtrom s veľkosťou okna 8×8 a bitovou dĺžkou 6.

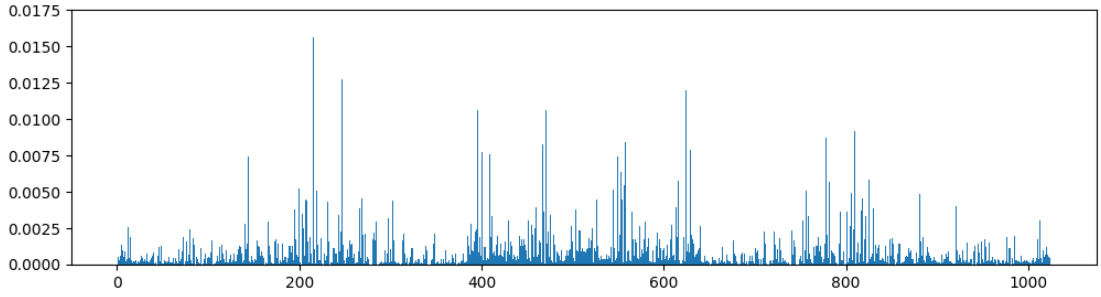
V rámci generovania BSIF deskriptorov boli zároveň generované aj modely týchto dát, histogramy. Modelovaný však nie je celý obrázok, ale len jeho segmentovaná časť. Bol kladený veľký dôraz na odstránenie všetkého pozadia a aj okrajových častí odtlačkov prstov, so snahou sa učiť len vlastnosti reznej línie a detekovať morphované odtlačky len na základe tejto informácie. Okrajové časti odtlačkov boli odstraňované z dôvodu artefaktov zanechaných morphingom – hranaté a rozmazané okraje. Histogramom nie sú modelované početnosti deskriptorov v obrázku, ale pravdepodobnosť výskytu jednotlivých deskriptorov. Ukážku histogramov popisujúcich BSIF deskriptory z obrázku 7.10 je možné vidieť na obrázku 7.11.



(a)



(b)



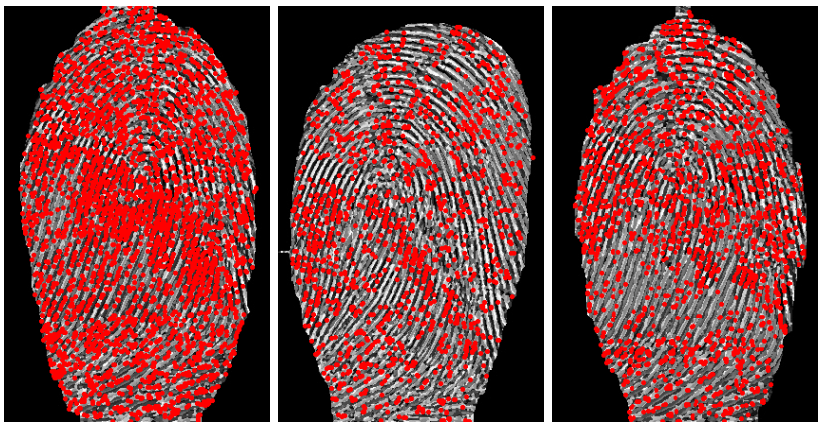
(c)

Obrázok 7.11. Ukážka modelovania BSIF deskriptorov z obrázkov 7.10d, 7.10e a 7.10f histogramami (a), (b) a (c), pre každé zvlášť. Histogramy neobsahujú deskriptory pozadia, len deskriptory popredia (viď algoritmus 2). Binárne deskriptory v histogramoch (horizontálna os) sú transformované do dekadických hodnôt, kvôli prehľadnejšej vizualizácií. Vertikálne os znázorňuje pravdepodobnosť výskytu jednotlivých deskriptorov.

Z histogramov 7.11 tiež vyplýva, že deskriptory morphovaného odtlačku nie sú moc ustálené a niektoré z nich majú výrazne väčší pravdepodobnostný výskyt než deskriptory v histogramoch pôvodných odtlačkov prstov. Táto skutočnosť bude pravdepodobne kľúčová v rozhodovacej logike SVM. V tabuľke 7.6 je možné vidieť prahové hodnoty pravdepodobností výskytu t_{prob} deskriptoru a priemerný počet kľúčových deskriptorov na odtlačok v jednotlivých dátových sadách. Prahy boli odhadnuté prostredníctvom implementácie z kapitoly **??** a extraktory (veľkosť okna a bitová dĺžka) boli zvolené na základe celkového vyhodnotenia v nasledujúcej kapitole v tabuľke 7.8a (najlepší natrénovaný model pre jednotlivé dátové sady). Na obrázku **??** je možné vidieť vizualizáciu kľúčových deskriptorov (deskriptor, ktorých pravdepodobnosť výskytu prekročili t_{prob}). Prehľad použitých parametrov je možné vidieť v tabuľke 7.7.

	Základný morphing					Adaptívny morphing				
	w_{size}	b_l	t_{prob}	m_{orig}	m_{morph}	w_{size}	b_l	t_{prob}	m_{orig}	m_{morph}
Bergdata	16×16	9	0,00064	341,95	367,59	17×17	10	0,00009	892,21	915,36
Sagem MSO	11×11	8	0,00127	193,25	207,66	17×17	11	0,00015	1418,86	1580,73
SecuGen	15×15	6	0,00201	54,92	58,48	5×5	6	0,00236	53,11	56,7
Syntetické	3×3	1	0,39147	2,0	1,0	3×3	5	0,39147	2,0	1,0

Tabuľka 7.6: V tabuľke sa nachádzajú informácie o kľúčových deskriptoroch pre jednotlivé sady odtlačkov prstov. w_{size} a b_l predstavujú konfiguráciu extraktoru vo forme veľkosti okna a počtu okien, t_{prob} je minimálna pravdepodobnosť výskytu deskriptoru a m_{orig} a m_{morph} sú priemerné počty kľúčových deskriptorov na odtlačok.



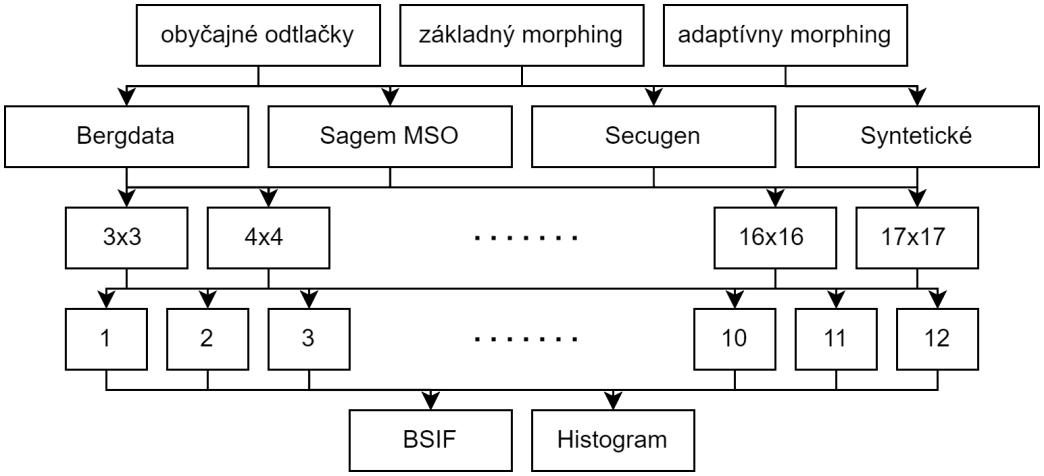
(a) (b) (c)

Obrázok 7.12. Vizualizácia kľúčových deskriptorov na pôvodných odtlačkov (a) a (b) a morphovanom odtlačku (c).

Parameter	**Hodnota**
max_window_size	17
max_bit_length	12

Tabuľka 7.7: Parametre potrebné k extrakcii deskriptorov BSIF algoritmom z kapitoly 6.4.

V rámci odhadu BSIF deskriptorov s oknami maximálneho rozsahu `max_window_size` a maximálnym počtom okien `max_bit_length` nastavených na hodnoty z tabuľky 7.7 bolo vygenerovaných 2016 databáz BSIF deskriptorov a 2016 databáz histogramových modelov z odhadnutých deskriptorov. Z dôvodu veľkého počtu rôznych kategórií nebude prehľad všetkých databáz znázornený v tabuľke, ale len slovne a diagramom priblížená architektúra. Ako je vidieť na diagrame 7.13, databáze sú generované pre originálne odtlačky a aj pre morphované oboma metódami, pre každý typ senzoru zvlášť (12 skupín). Pre každú z týchto skupín existujú podskupiny pre filtre každej veľkosti okna (168 podskupín). A nakoniec sa pre každú podskupinu generujú BSIF deskriptory na základe bitových dĺžok filtrov, čo vytvorí 12 databáz pre každú podskupinu, a teda dokopy 2016 databáz deskriptorov BSIF. Dáta týchto databáz sú následne modelované histogrammi.



Obrázok 7.13. 32 nezávislých komponent o veľkosti 32×32 odhadnutých z 256 dekorelovaných komponentov na obrázku 7.6.

7.4 Trénovanie klasifikátorov SVM

Táto kapitola obsahuje vyhodnotenie posledného kroku k uskutočneniu detekcie morphovaných odtlačkov prstov (implementácia z kapitoly 6.3). Najprv sa načítajú databáze vygenerované predošlým krokom a spárujú sa databáze histogramov originálnych odtlačkov prstov s databázami morphovaných odtlačkov, pričom musia byť dodržané nasledujúce podmienky (bližší popis v kapitole 5.3.1):

- Veľkosti okien filtrov, z ktorých histogramy vznikli, musia byť rovnaké.
- Bitové dĺžky filtrov, z ktorých histogramy vznikli, musia byť rovnaké.
- Párujú sa vždy histogramy originálnych odtlačkov s morphovanými odtlačkami oddelene pre obe morphovacie metódy.

Histogramom z párovaných databáz sú následne priradené triedy, či sa jedná o morphovaný odtlačok alebo originálny. V jednotlivých pároch sa dáta premiešajú (kvôli optimálnejšiemu učeniu vlastností). Na základe parametru `split` sú pomiešané dáta v pároch rozdelené na trénovacie a testovacie. Pre každý typ odtlačku generovaný každou morphovacou metódou teda vznikne 336 skupín trénovacích dát tvorených rôznymi typmi BSIF

deskriptorov (veľkosti okien a bitové dĺžky, jedna skupina predstavuje deskriptory extrahované konkrétnou veľkosťou okna a bitovou dĺžkou, napr. 17×17 a 10). Na trénovacích dátach prebieha učenie SVM, ktoré po ukončení je testované na testovacích (nevidených) dátach. Výsledok presnosti spolu s ďalšími metrikami sa ukladajú k modelu. Z týchto metrík prebieha vyhodnotenie najoptimálnejších modelov detekcie morphovaných odtlačkov. Pre každý typ odtlačku je nájdený taký model z 336 natrénovaných modelov používajúci určitý typ BSIF deskriptorov, ktorý má najlepšiu odozvu na testovacích dátach. Vo výsledku je z experimentov získaných osem najoptimálnejších modelov detekujúcich jednotlivé typy odtlačkov generované oboma morphovacími metódami. Experimenty boli rožšírené aj pre rôzne kernelové funkcie, určené k transformácií dát. Celý proces trénovania teda prebieha viac krát pre rôzne funkcie a 8 optimálnych modelov je odhadovaných pre každú z použitých kernelových funkcií. Prehľad natrénovaných SVM modelov je možné vidieť v tabuľke 7.8.

Polynomická	Základný morphing			Adaptívny morphing		
	`window_size`	`bit_length`	Presnosť (%)	`window_size`	`bit_length`	Presnosť (%)
Bergdata	16x16	9	97,12	17x17	10	98,85
Sagem MSO	11x11	8	98,49	17x17	11	98,49
SecuGen	5x5	6	94,97	5x5	6	94,97
Syntetické	3x3	1	100	3x3	1	100

(a)

Lineárna	Základný morphing			Adaptívny morphing		
	`window_size`	`bit_length`	Presnosť (%)	`window_size`	`bit_length`	Presnosť (%)
Bergdata	8x8	12	75,28	14x14	3	82,75
Sagem MSO	8x8	2	66,33	14x14	4	67,33
SecuGen	8x8	2	68,84	13x13	3	53,26
Syntetické	3x3	2	100	3x3	2	100

(b)

Radiálna bázová	Základný morphing			Adaptívny morphing		
	`window_size`	`bit_length`	Presnosť (%)	`window_size`	`bit_length`	Presnosť (%)
Bergdata	14x14	10	95,40	17x17	12	96,55
Sagem MSO	11x11	12	97,48	5x5	6	96,48
SecuGen	5x5	6	91,95	6x6	9	91,45
Syntetické	3x3	1	100	3x3	2	100

(c)

Sigmoidná	Základný morphing			Adaptívny morphing		
	`window_size`	`bit_length`	Presnosť (%)	`window_size`	`bit_length`	Presnosť (%)
Bergdata	12x12	10	82,75	17x17	12	83,90
Sagem MSO	17x17	11	92,46	17x17	12	92,46
SecuGen	14x14	10	80,90	7x7	11	80,40
Syntetické	4x4	4	100	4x4	4	99,69

(d)

Tabuľka 7.8: Tabuľky znázorňujúca najpresnejšie natrénované SVM modely používajúce rôzne kernelové funkcie. Kernelová funkcia, ktorú SVM použilo, je v každej z tabuliek uvedená v ľavom hornom rohu. Pre každý model je tiež uvedné, aký typ BSIF deskriptorov, bol použitý k učeniu vo forme veľkosti `window_size` a počtu okien `bit_length` filtra.

Z tabuľky 7.8 je zrejmé, že najpresnejšie detektory morphovaných odtlačkov prstov sú práve SVM modely s polynomickou kernelovou funkciou a chvalitebnou presnosťou minimálne 95 %. Naopak najhoršie dopadli modely s linéarnou kernelovou funkciou, čo znamená, že extrahované BSIF deskriptory z dát nie sú lineárne separovateľné. Ďalej je z tabuľky vidieť, že morphing vykonávaný na syntetických odtlačkoch prstov je najviac náchylný na odhalanie, čo sa podarilo skoro vo všetkých prípadoch s presnosťou 100 %. Nastavenie parametrov k uskutočneniu učenia detekcie morphovaných odtlačkov prstov je možné vidieť v tabuľke 7.9.

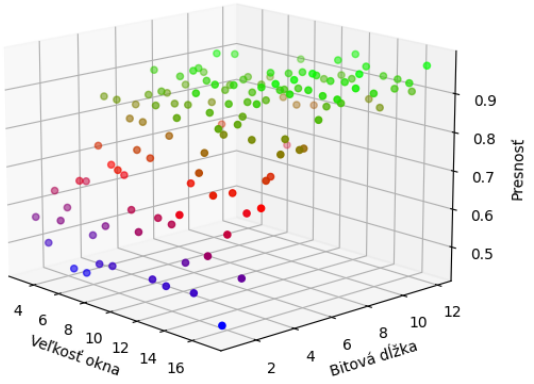
Parameter	**Hodnota**
window_range	3 - 17
bit_length_range	1 - 12
svm_kernel	linear, sigmoid, poly, rbf
split	0,7

Tabuľka 7.9: Parametre potrebné k vygenerovaniu deskriptorov BSIF algoritmom z kapitoly 6.4.

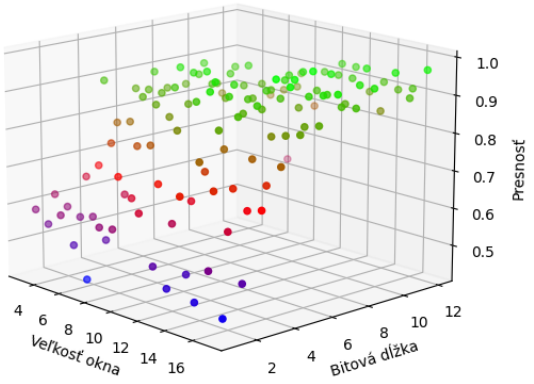
Analýza výsledkov tiež ukázala, že hoci je adaptívna metóda morphingu z mojej projektovej praxe [5] efektívnejšia vo vytváraní silnejších dvojitých identít, na znemožnenie detekcie pomocou extrakcie deskriptorov, odhadnutými nezávislými komponentami, nemá žiadny vplyv. Ďalej je v tabuľke 7.10 bližšie rozobraná presnosť najúspešnejších modelov s polynomickou kernelovou funkciou za pomoci metrík precision a recall. Precision metrika hodnotí podiel správne pozitívne označených dát oproti všetkým pozitívne označeným dátam a recall metrika hodnotí počet správne pozitívne označených dát oproti reálnym pozitívnym dátam. Pre jednoduchosť je možné precision interpretovať aj ako „Koľko rozhodnutí detektorom je relevantných?“ a recall ako „Koľko relevantných dát je detekovaných?“. Príkladom je extraktor s oknom 17×17 a dĺžkou 10. S presnosťou 100 % sú všetky označené obyčajné odtlačky v skutočnosti obyčajné, ale len 98 % obyčajných odtlačkov prstov z celkového počtu obyčajných odtlačkov bolo rozpoznaných. Rovnako pre prípad s morphovanými odtlačkami, kde precision hovorí, že len 98 % z označených morphovaných odtlačkov sú v skutočnosti morphované, ale zároveň v rámci metriky recall, boli odhalené všetky morphované odtlačky prstov. Na obrázku 7.14 je možné vidieť celý priebeh tréningu SVM modelov s polynomickou kernelovou funkciou vzhľadom na konfiguráciu extraktorov a presnosť modelu.

		Základný morphing		Adaptívny morphing	
		precision (%)	**recall** (%)	**precision** (%)	**recall** (%)
Bergdata	obyčajný	98	97	100	98
	morphovaný	97	98	98	100
Sagem MSO	obyčajný	97	100	98	99
	morphovaný	100	97	99	98
SecuGen	obyčajný	95	95	94	96
	morphovaný	95	95	96	94
Syntetické	obyčajný	100	100	100	100
	morphovaný	100	100	100	100

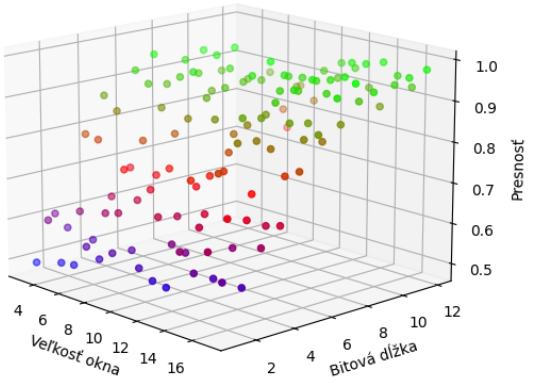
Tabuľka 7.10: V tabuľke sa nachádza bližší popis presnosti najoptimálnejších SVM modelov s polynomickou kernelovou funkciou, pomocou metrík precision a recall.



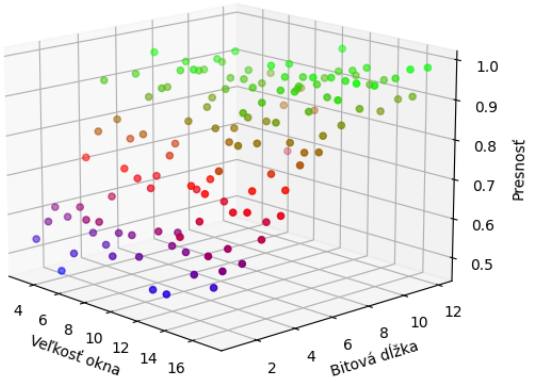
(a) Základný morphing, Bergdata.



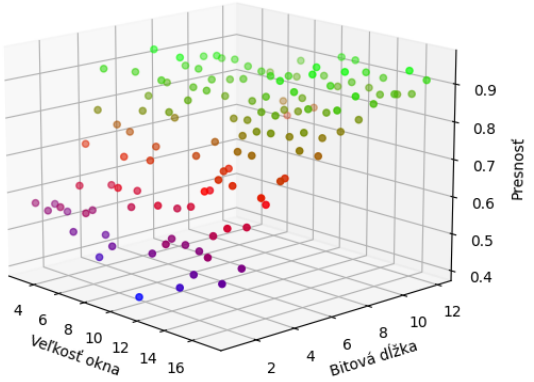
(b) Adaptívny morphing, Bergdata.



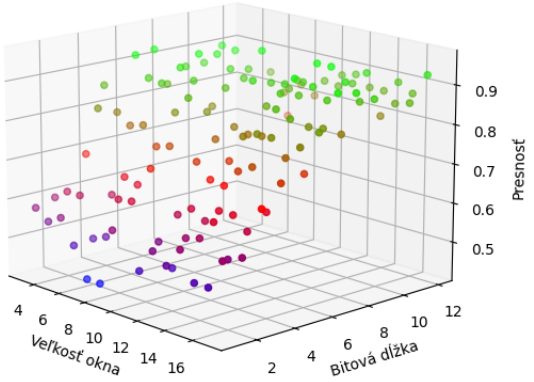
(c) Základný morphing, Sagem MSO.



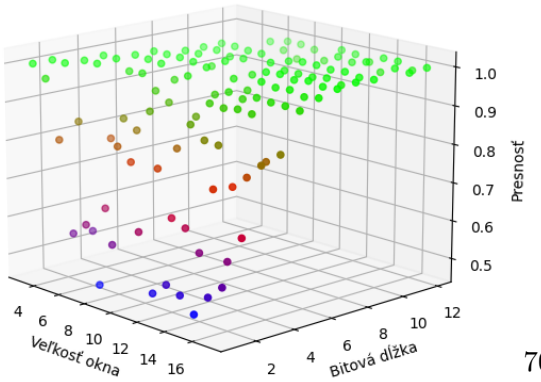
(d) Adaptívny morphing, Sagem MSO.



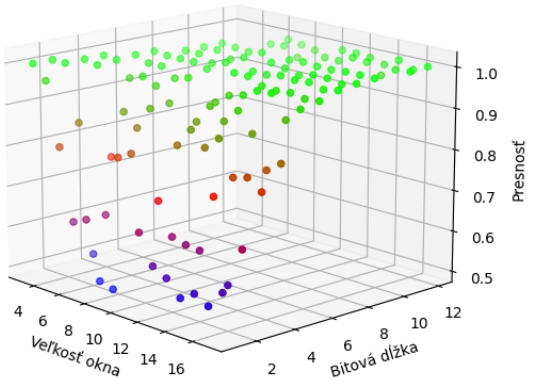
(e) Základný morphing, SecuGen.



(f) Adaptívny morphing, SecuGen.



(g) Základný morphing, syntetické.



(h) Adaptívny morphing, syntetické.

Obrázok 7.14. Vizualizácia priebehu trénovania SVM modelov s polynomickou kernelovou funkciou.

Kapitola 8

Záver

Cieľom diplomovej práce bolo uskutočniť detekciu morphovaných odtlačkov prstov. K dosiahnutiu tohto cieľa bolo v prvom rade potrebné vygenerovať dátovú sadu morphovaných odtlačkov prstov, z ktorých všetky morphované odtlačky vyhodnotené pomocou Minutia Cylinder-Code bez dvojitej identity, boli odstránené. Morphing prebiehal na základe morphovacej metódy rezom spracovanej v mojej bakalárskej práci a jej modifikácií z mojej projektovej praxe. Následne bolo potrebné dátovú sadu premiešať s normálnymi odtlačkami prstov a každému z nich priradiť informáciu o aký typ sa jedná (morphovaný/normálny).

Detekcia prebieha prostredníctvom deskriptorov textúr. Morphovaný odtlačok alebo normálny odtlačok je považovaný za stochastickú textúru, na ktorej je vykonaná štatistická analýza binárnych vlastností textúr BSIF. K extrakcií týchto vlastností z odtlačkov sú využité filtre optimálne odhadnuté na prírodných scénach tak, že každý filter má optimálnu odozvu na inej nezávislej komponente týchto prírodných scén. Filtrami sú extrahované binárne deskriptory obrazu, kde každý pixel predstavuje binárny vektor určitej bitovej dĺžky. Textúry transformované do priestoru vlastností, sú následne prevedené do histogramovej podoby. Histogram deskriptorov nemodeluje klasickú početnosť výskytov, ale pre efektívnejšiu analýzu modeluje pravdepodobnosť výskytu daného deskriptoru v textúre. Dátová sada podobných informácií je následne predaná klasifikačnému algoritmu, SVM.

Bolo natrénovaných mnoho SVM modelov na histogramoch s rôznymi nastaveniami BSIF extraktorov (veľkosť filtru, bitová dĺžka). V rámci hľadania najrobustnejšieho detektora, boli do tréningu modelov zaradené aj experimenty s rôznymi kernelovými funkciami. Testovanie natrénovaých detektorov morphingu odtlačkov prstov prebiehalo na syntetických odtlačkoch a odtlačkoch získaných z troch rôznych senzorov: Bergdata, Sagem MSO a SecuGen. Ako najoptimálnejší detektor sa ukázal SVM s polynomickou kernelovou funkciou s presnosťou detekcie základného morphingu 97,12 % na dátovej sade zo sensoru Bergdata, 98,97 % na Sagem MSO, 94,97 % na SecuGen a 100 % na syntetických odtlačkoch. Na adaptívnej metóde morphingu z mojej projektovej praxe bola presnosť detekcie 98,85 % na dátovej sade zo sensoru Bergdata, 98,49 % na Sagem MSO, 94,97 % na SecuGen a 100 % na syntetických odtlačkoch. Testy tiež ukázali, že deskriptory morphingu syntetických odtlačkov a deskryptory obyčajných odtlačkov sú lineárne separovateľné (lineárny SVM nadobudol presnosť detekcie 100 %).

K najoptimálnejším extraktorom bola taktiež odhadnutá aj prahová hodnota pravdepodobnosti výskytu deskriptoru v textúre, čím sa zabezpečilo pri použití extraktorov získanie okrem deskritorov textúry, aj ich kľúčových bodov. Adaptívna metóda morphingu z mojej projektovej praxe nepreukázala výrazný vplyv na znemožnenie detekcie morphovaných odtlačkov prstov, aj napriek tomu, že generuje silnejšie dvojité identity.

Literatúra

[1] Lee H.C. and Gaensslen R.E. *Advances in Fingerprint Technology.* London: CRC Press, 2 edition, 2001.

[2] Kanich O. *Fingerprint damage simulation: a simulation of fingerprint distortion, damaged sensor, pressure and moisture.* Saarbrücken: Lambert academic publishing, 2 edition, 2014.

[3] Gomez-Barrero M., Rathgeb C., Scherhag U., and Busch C. Is your biometric system robust to morphing attacks? *5th International Workshop on Biometrics and Forensics (IWBF)*, 2017.

[4] Denis Dovičic. Morphing otisků prstů, 2020.

[5] Denis Dovičic. Využití f1 skóre při morphingu otisků prstů, 2021.

[6] Ferrara M., Cappelli R., and Maltoni D. On the feacibility of creating double-identity fingerprints. *IEEE Transactions on Information Forensics and Security*, 2017.

[7] Tilborg H. *Encyclopedia of Cryptography and Security.* Boston, MA: Springer US, 2011.

[8] Maltoni D. *Handbook of fingerprint recognition.* London : Springer, 2009.

[9] Dudáková Ľ. Daktyloskopické stopy a jejich využití v praxi. Master's thesis, Univerzita Tomáše Bati ve Zlíně, 2013.

[10] Jain A.K., Ross A.A., and Nandakumar K. *Introduction to biometrics.* New York: Springer, 2011.

[11] Šišková J. Pojem a podstata daktyloskopie. Master's thesis, Univerzita Karlova v Praze, 2010.

[12] Li S.Z. and Jain A.K. *Encyclopedia of Biometrics.* New York, NY: Springer, 2015.

[13] Chowdhury C. and Saha B. Efficient fingerprint matching based upon minutiae extraction. *International Journal of Advanced Computer Research*, 2015.

[14] Henry E.R. *Classification and uses of finger prints.* London: H.M. Stationery Office, 1905.

[15] Cao K., Pang L., Liang J., and Tian J. Fingerprint classification by a hierarchical classifier. *Pattern Recognition*, 2013.

[16] Thakkar D. Minutiae based extraction in fingerprint recognition. https://www.bayometric.com/minutiae-based-extraction-fingerprint-recognition/. [Online; navštívené 24.05.2020].

[17] Xiao Q. and Raafat H. Fingerprint image postprocessing: A combined statistical and structural approach. *Pattern Recognition*, 1991.

[18] Othman A. and Ross A. On mixing fingerprints. *IEEE Transactions on Information Forensics and Security*, 2013.

[19] Armi L. and Fekri-Ershad S. Texture image analysis and texture classification methods - a review. 2019.

[20] *Handbook Of Pattern Recognition And Computer Vision (4th Edition)*. World Scientific Publishing Company, Singapore, 2009.

[21] van der Maaten L. and Postma E. Texton-based texture classification. 01 2007.

[22] Kumar V.V., Raju U.S.N., Premchand P., and Suresh A. Skeleton primitive extraction method on textures with different nonlinear wavelets. *Journal of computer science*, 4(7):591–599, 2008.

[23] Ilea D., Ghita O., and Whelan P. Evaluation of local orientation for texture classification. volume 1, pages 357–364, 01 2008.

[24] Szeliski R. *Computer Vision: Algorithms and Applications*. Texts in Computer Science. Springer International Publishing, 2022.

[25] Ojala T., Pietikainen M., and Maenpaa T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[26] Ahonen T., Rahtu E., Ojansivu V., and Heikkila J. Recognition of blurred faces using local phase quantization. In *2008 19th International Conference on Pattern Recognition*, pages 1–4. IEEE, 2008.

[27] Ojansivu V. and Heikkilä J. Blur insensitive texture classification using local phase quantization. In *Image and Signal Processing*, Lecture Notes in Computer Science, pages 236–243. Springer Berlin Heidelberg, Berlin, Heidelberg.

[28] Chengsheng Y., Sun X., and Rui L. Fingerprint liveness detection based on multi-scale lpq and pca. *China communications*, 13(7):60–65, 2016.

[29] Alemán-Flores M. and Alvarez L. Texture classification through multiscale orientation histogram analysis. volume 2695, pages 479–493, 06 2003.

[30] Marimon D. and Ebrahimi T. Orientation histogram-based matching for region tracking, 2017.

[31] Wieclaw L. Gradient based fingerprint orientation field estimation. *Journal of Medical Informatics and Technologies*, 2013.

[32] Cappelli R., Ferrara M., and Maio D. A fast and accurate palmprint recognition system based on minutiae. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2012.

[33] Cao M., Ming D., Xu L., Fang J., Liu L., Ling X., and Ma W. Frequency spectrum-based optimal texture window size selection for high spatial resolution remote sensing image analysis. *Journal of Spectroscopy*, 2019:4970376, Sep 2019.

[34] Tiecheng S., Hongliang L, Bing Z., and Gabbouj M. Texture classification using joint statistical representation in space-frequency domain with local quantized patterns. In *2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 886–889. IEEE, 2014.

[35] Hong L., Wan Y., and Jain A. Fingerprint image enhancement: Algorithm and performance evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1998.

[36] Hyvrinen A., Hurri J., and Hoyer P.O. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision.* Springer Publishing Company, Incorporated, 1st edition, 2009.

[37] Bro R. and Smilde A.K. Principal component analysis. *Analytical Methods*, 6:2812–2831, 2014.

[38] Hyvärinen A. and Oja E. Independent component analysis: algorithms and applications. *Neural Networks*, 13(4):411–430, 2000.

[39] Shah C.A., Arora M.K., Robila S.A., and Varshney P.K. Ica mixture model based unsupervised classification of hyperspectral imagery. In *Applied Imagery Pattern Recognition Workshop, 2002. Proceedings*, pages 29–35. IEEE, 2002.

[40] Loncaric S. A survey of shape analysis techniques. *Pattern recognition*, 31(8):983–1001, 1998.

[41] Zhang T.Y. and Suen C.Y. A fast parrallel algorithm for thinning digital patterns. *Communications of the ACM*, 1984.

[42] Yung-Sheng C. and Wen-Hsing H. A modified fast parrallel algorithm for thinning digital patterns. *Pattern Recognition Letters*, 1988.

[43] Scheunders P., Livens S., Wouwer G., Vautrot P., and Dyck D. Wavelet-based texture analysis. *International Journal Computer Science and Information Management*, 1, 02 2000.

[44] Daugman J.G. Complete discrete 2-d gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(7):1169–1179, 1988.

[45] Jain A.K. and Farrokhnia F. Unsupervised texture segmentation using gabor filters. In *1990 IEEE International Conference on Systems, Man, and Cybernetics Conference Proceedings*, pages 14–19, 1990.

[46] Cappelli R., Lumini A., Maio D., and Maltoni D. Fingerprint image reconstruction from standard templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007.

[47] Kecman V. *Support Vector Machines – An Introduction*, volume 177, pages 605–605. 05 2005.

[48] Kannala J. and Rahtu E. Bsif: Binarized statistical image features. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 1363–1366, 2012.

[49] Ghiani L., Hadid A., Marcialis G.L., et al. Fingerprint liveness detection using binarized statistical image features. In *2013 IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, pages 1–6, 2013.

[50] Raghavendra R., Raja K., and Busch C. Detecting morphed face images. 09 2016.

[51] Cappelli R., Ferrara M., Maltoni D., and Maio D. *MCC Software Development Kit (SDK) - Documentation*. University of Bologna, 2015.

[52] Cappelli R., Ferrara M., and Maltoni D. Minutia cylinder-code: A new representation and matching technique for fingerprint recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[53] The Qt Company Ltd. Qt 5.14. https://doc.qt.io/qt-5/index.html. [Online; navštívené 24.05.2020].

Príloha A

Obsah priloženého DVD

xdovic01.pdf - elektronická verzia textu tejto práce.

xdovic01.zip - zdrojové súbory textu tejto práce.

morphing_detection.zip - zdrojové súbory implementovanej aplikácie.

Results.zip - výsledky vyhodnotenia, najpresnejšie modely a textový súbor s logom experimentov trénovania.