UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Uroš Klopčič

Sistem za verifikacijo osebe na podlagi prstnega odtisa

DIPLOMSKO DELO
NA UNIVERZITETNEM ŠTUDIJU

Mentor: doc. dr. Peter Peer

Ljubljana, 2009

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01561/2009
Datum: 05.04.2009

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **UROŠ KLOPČIČ**

Naslov: **SISTEM ZA VERIFIKACIJO OSEBE NA PODLAGI PRSTNEGA ODTISA**
FINGER-PRINT BASED HUMAN VERIFICATION SYSTEM

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Strokovnjaki s področja biometrije so si edini, da šarenica daje najzaneslivejše rezultate, prstni odtis ne zaostaja veliko, obrazi pa obetajo zelo veliko. Preučite področje razpoznavanja oseb na podlagi prstnh odtisov. Nato razvijte sistem razpoznave. Po korakih pojasnite celoten proces ter izpostavite učinkovitost, prednosti in slabosti vsakega koraka. Sistem testirajte nad ustrezno zbirko slik.

Mentor:
doc. dr. Peter Peer

Dekan:
prof. dr. Franc Solina

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Namesto te strani **vstavite** original izdane teme diplomskega dela s podpisom mentorja in dekana ter žigom fakultete, ki ga diplomant dvigne v študentskem referatu, preden odda izdelek v vezavo!

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani/-a Uroš Klopčič,

z vpisno številko 63030107,

sem avtor/-ica diplomskega dela z naslovom:

Sistem za verifikacijo osebe na podlagi prstnega odtisa

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom doc. dr. Peter Peer
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 03.10.2009 Podpis avtorja/-ice:

Zahvala

Na tem mestu bi se zahvalil mentorju doc. dr. Petru Peeru, za pomoč, nasvete ter predloge, prof. dr. Nežki Mramor Kosta za trud in nasvete pri reševanju matematičnih problemov ter dekletu in staršem, ki so mi ves čas stali ob strani.

Kazalo

Povzetek

V tem delu je predstavljena metoda za verifikacijo oseb na podlagi prstnega odtisa. Ker smo želeli čim bolj enostavno in modularno zasnovo, se metoda deli na več korakov. Kot vhod sprejme metoda sivinsko sliko prstnega odtisa, na kateri se najprej izvede segmentacija, kjer se ozadje loči od površine, na kateri se nahaja prstni odtis. Nato sledi izračun orientacijskega polja prstnega odtisa s pomočjo gradientne metode, za tem pa še izračun lokalne frekvence grebenov. Obe vrednosti uporabimo v koraku izboljšanja kvalitete slike prstnega odtisa kot vhod v Gaborjeve filtre. Nad izboljšano sliko izvedemo binarizacijo, kateri sledi tanjšanje grebenov. V slednjem koraku s pomočjo 21 pravil za tanjšanje prstnih odtisov ali simbolov ter 4 pravil za tanjšanje diagonalnih črt zožamo grebene na širino enega slikovnega elementa. S tem pridobimo skelet prstnega odtisa, nad katerim izvedemo iskanje značilk. Zaključke grebenov odkrijemo s štetjem števila križišč, razcepe pa odkrijemo z uporabo 24 mask. V koraku klasifikacije s pomočjo polja ukrivljenosti odkrijemo singularne točke, s katerimi določimo tip prstnega odtisa ter lokacijo referenčne točke. Ko določimo še usmerjenost referenčne točke pa v koraku primerjanja pretvorimo značilke v polarni prostor in damo končno oceno o enakosti dveh prstnih odtisov.

Algoritem je bil preizkušen na štirih testnih množicah, uporabljenih na tekmovanju FVC 2002. Vsaka testna množica je vsebovala 800 slik prstnih odtisov različnih kvalitet in velikosti. Primerjava med rezultati pridobljenimi z opisanim algoritmom in rezultati iz tekmovanja FVC 2002 so pokazala, da algoritem daje zadovoljive rezultate, vendar potrebuje še nekaj izboljšanj. Največ problemov povzročajo slike prstnih odtisov slabe kvalitete, kjer algoritem težko določi potek grebenov.

Izdelano aplikacijo smo uspešno povezali tudi s čitalcem prstnih odtisov. Rezultati so pokazali, da aplikacija še ni primerna za okolja, kjer je zahtevana 100% zanesljivost, potrebne pa bi bile tudi izboljšave v času procesiranja.

Ključne besede:

računalniški vid, prstni odtisi, verifikacija, samodejno razpoznavanje, čitalec prstnih odtisov

Abstract

The diploma thesis presents an algorithm for verification based on fingerprints. In order to achieve a simple and modular design, the algorithm is divided into number of steps. As an input, the algorithm takes greyscale fingerprint images. First, segmentation is performed where the background is separated from the area which represents the fingerprint. This is followed by the calculation of orientation field of the fingerprint using the gradient method and local frequency estimation. Both values are used as an input to Gabor filters in the step of improving the quality of fingerprint images. The enhanced image is then binarized, which is followed by the thinning procedure. In the latter step, the skeleton of the fingerprint is obtained with the help of 21 rules for the thinning of the lines and 4 rules for the thinning of the diagonal lines, where every ridge is one pixel wide. In the step of minutiae extraction, ridge endings are discovered using the method of crossing number, and ridge bifurcations are extracted using 24 masks. In the classification step, singular points are obtained by using the curvature map. With the help of singular points we determine the type of the fingerprint and the location of the reference point. Finally, the reference point orientation is calculated. With respect to the corresponding reference point all minutae are converted to polar coordinate system and a final assessment of equality of two fingerprints is given.

The algorithm has been tested on four test sets used at the FVC 2002 competition. Each set contained 800 fingerprint images of different qualities and sizes. Comparison between the results obtained using the described algorithm and results from the competition FVC 2002 showed that the algorithm gives satisfactory results, but needs some improvements. Most problems appear at images of poor quality, where it is difficult to determine the course of the ridges.

We successfully connected the application with the fingerprint reader. The results indicated that the application is not yet suitable for the environments where 100% reliability is necessary, further improvements are needed in the processing time as well.

Key words:

computer vision, fingerprints, verification, automated recognition, fingerprint reader

Poglavje 1

Uvod

„Ali je oseba tista, za katero se izdaja?“

To je vprašanje, ki si ga danes vse bolj zastavljamo. Ugotavljanje identitete posameznika je bil in je še vedno velik izziv, s katerim se ljudje srečujemo. Dostop v poslovno prostore, dvig denarja na bankomatu, e-trgovanje, pa tudi samo prijavljanje v računalniško omrežje so le nekateri primeri, ki so odvisni od identifikacije posameznika. Pri tem se večinoma uporabljajo gesla, vstopne kartice ter PIN kode. Čeprav so taki pristopi danes zelo razširjeni in samoumevni, vsebujejo tudi varnostne pomankljivosti. Vstopne kartice si lahko lastniki delijo, podvržene so tudi kraji, nič manj imuna niso gesla in PIN kode.

Zato so se že dokaj zgodaj začeli pojavljati tudi drugi, bolj zanesljivi načini ugotavljanja identitete. Nekaj več kot stoletje nazaj je Alphonse Bertillon prvi odkril in prakticiral idejo meritev telesnih značilnosti (glave, telesa, brazgotin, tudi tetovaž) za reševanje zločinov. Še preden je ideja dodobra zaživela, jo je zasenčilo odkritje unikatnosti prstnih odtisov, ki je bilo veliko bolj pomembno in precej bolj praktično.

Identifikacija oseb s pomočjo prstnih odtisov je le ena od metod iz širše skupine biometričnih metod, pri katerih poteka identifikacija na podlagi analize bioloških značilnosti osebe. Sem spadajo še identifikacija oseb na osnovi razpoznavanja glasu, šarenice, obraza itd. Vse te metode označimo s pojmom biometrija.

1.1 Biometrija

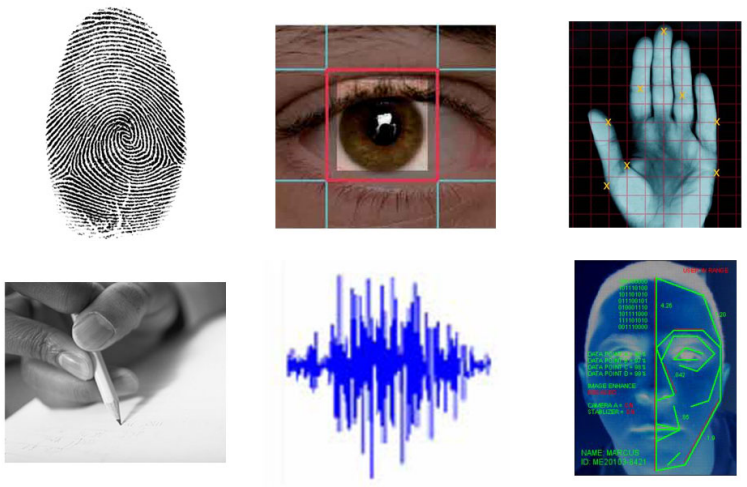
Biometrične značilnosti lahko ločimo v dve kategoriji (slika 1.1):

- fizične (telesne) značilnosti: prstni odtis, šarenica, geometrija dlani, poteze obraza, DNK itd.

- značilnosti obnašanja: motorične značilnosti, ritem tipkanja, tudi glas in podpis itd.

Za biometrične vzorce velja izkustveno (empirično) dejstvo, da so unikatni. Zato v primerjavi z ostalimi postopki identifikacije nudijo nekatere bistvene prednosti. Za gesla je znano, da se lahko odkrijejo s pomočjo t.i. napadov na silo (angl. *brute-force*) ali napadov s slovarji (angl. *dictionary attack*). Biometrični sistemi pa zahtevajo fizično prisotnost osebe in so zato imuni na take napade. Fizične značilnosti ter značilnosti obnašanja se tudi ne morejo prenašati na druge osebe, s čimer preprečimo klasične zlorabe, kot so poneverba, posojanje, kraja itd. Dodatna prednost pa je tudi, da so biometrične značilnosti vedno prisotne in jih ni potrebno nositi s seboj. Zato jih tudi ne moremo izgubiti ali pozabiti, hkrati pa je onemogočena tudi kraja.

Vseeno pa je možno biometrične sisteme tudi zlorabiti, saj ni težko pridobiti prstnega odtisa določene osebe (npr. s kozarca), zlorabljena oseba pa biometričnega vzorca ne more spremeniti. To dejstvo omejuje tudi tiste, ki bi radi zaradi kakršnegakoli razloga spremenili identiteto.



Slika 1.1: Različni biometrični principi: prstni odtis, šarenica, geometrija dlani, podpis, glas, poteze obraza

1.2 Identifikacija oseb na podlagi prstnega odtisa

1.2.1 Zgodovina

Že arheologi so našli veliko število zgodovinskih artefaktov, na katerih so bili vgravirani oziroma naslikani prstni odtisi. Čeprav ta odkritja kažejo na to, da so se takratna ljudstva zavedala unikatnosti prstnih odtisov, je malo verjetno, da so jih tudi uporabljala za dokazovanje identite.

Šele proti koncu 17. stoletja pa so se začela prva znanstvena raziskovanja. Leta 1684 je angleški anatomist Nehemiah Grew objavil delo, v katerem je opisal grebene, brazde in strukturo por prstnega odtisa. Kasneje se je s prstnimi odtisi začelo ukvarjati vse več raziskovalcev. Tako je leta 1788 Nemec Johann Christoph Mayer v svojem delu natančno opisal vzorce, ki se pojavijo na prstnem odtisu, zbral in kategoriziral pa je tudi več pomembnih značilnosti grebenov. Leta 1809 je Anglež Thomas Bewick začel uporabljati prstni odtis kot blagovno znamko. Odtisi, vgravirani v les, so bili tako natančni, da se Angleževo dejanje šteje kot pomemben mejnik v zgodovini raziskovanja prstnega odtisa.

Češki filozof in anatomist Johannes Purkinje Evangelista je bil prvi, ki je predlagal grupiranje prstnih odtisov v več razredov. V svojem doktorskem zagovoru je leta 1823 predlagal devet razredov (lok, šotorast lok, zanka ter šest različnih tipov spiral), od katerih se še danes uporabljajo prvi trije, ostale tipe spiral pa se ponavadi zarad enostavnosti združi v en razred.

Prvi, ki je uporabil prstne odtise za identifikacijo v praksi, pa je bil William James Herschel. V 19. stoletju je služboval kot britanski oficir v Indiji. Z uporabo prstnih odtisov je želel preprečiti pogoste goljufije pri podpisu pogodb, hkrati pa se je izognil tudi velikemu problemu prebivalstva takratne Indije – nepismenosti. Kasneje je spoznal, da bi lahko prstni odtis služil kot splošno sredstvo za identifikacijo oseb, vendar je njegovo spoznanje v domovini naletelo na gluha ušesa.

Večji preboj se je zgodil proti koncu 19. stoletja. Henry Fauld je bil škotski zdravnik, ki je deloval v bolnišnici na Japonskem. V tistem času je začel zbirati in analizirati prstne odtise. Prišel je do spoznanja, da se tipi prstnih odtisov ne spreminjajo ter da so sami prstni odtisi med seboj zelo različni. Kasneje je Fauld na to temo objavil članek v reviji *Nature*, napisal pa je tudi pismo Charlesu Darwinu, v katerem je razložil svoja spoznanja. Ker je bil Darwin takrat že prestar, da bi sodeloval pri raziskovanjih, je to pismo posredoval svojemu bratrancu, antropologu Francis Galtonu, ki je prvi pričel znanstveno preučevati prstne odtise.

Francis Galton je leta 1892 objavil knjigo o prstnih odtisih, v kateri je dokazal, da se prstni odtisi človeka ne spremenijo od rojstva do smrti in da ne obstajata dva prstna odtisa, ki bi bila enaka. Bil je tudi prvi, ki je poimenoval značilke grebenov (konec grebena, razcep, zanka, otoček). Zato se danes v njegovo čast te značilke imenujejo Galtonove značilnosti.

Z vsemi takratnimi raziskavami so lahko raziskovalci dokaj hitro in uspešno določili, ali sta dva prstna odtisa enaka. Kadar pa je bilo treba primerjati prstni odtis s sto ali več drugimi prstnimi odtisi, pa se je pojavil problem množičnosti. Tega problema se je zavedal Edward Henry, zato je vpeljal zanimiv problem indeksacije prstnih odtisov [21]. Vsaki osebi je vzel prstne odtise vseh desetih prstov, vsak prst označil s tipom vzorca in iz teh oznak zgradil označbo, pod katero je shranil zapis o osebi. Takšen sistem ima na voljo 1.024 možnih označb, zato pri iskanju pregledamo povprečno 1/1.024 označb. Leta 1901 je Scotland Yard prvi vpeljal tak sistem indeksacije, kasneje pa so ga začele uporabljati tudi druge države po svetu.

1.2.2 Osnovna dejstva in definicije

Prstni odtisi se formirajo v stalno obliko pri zarodku starem sedem mesecev, njihova zgradba pa se ne spremeni skozi celo življenje, če ne pride do kakšnih trajnih poškodb. Zaradi omenjene lastnosti so prstni odtisi zelo privlačna biometrika. Kot smo že omenili, individualnost prstnih odtisov ni dokazano dejstvo, ampak zgolj empirična ugotovitev. Do sedaj še ni bilo odkritih dveh osebkov, ki bi imela popolnoma enake prstne odtise. Tudi pri dvojčkih, kjer je možnost ponovitve največja, niso uspeli odkriti enakosti. Pri avtomatskih identifikacijskih sistemih je zgodba seveda drugačna, saj imamo opravka z napravami, ki niso odporne na šum in napake. Pri teh lahko pogosto pride do izgube razločevalne informacije in ugotovitve ujemanja dveh prstnih odtisov različnih osebkov (slika 1.2).

Načini razpoznavanja

Pri razpoznavanju se pojavljata dva osnovna problema:

- Verifikacija (angl. *verification*): Ali je oseba res tista, za katero se izdaja? Potrebna je primerjava biometričnih lastnosti osebe z že shranjenimi lastnostmi osebe v sistemu. Izvede se t.i. ena-proti-ena (angl. *one-to-one*) primerjava, s katero potrdimo oziroma zavrnemo avtentičnost osebe.
- Identifikacija (angl. *identification*): Kdo je ta oseba? Biometrične lastnosti osebe moramo primerjati z vsemi vzorci (angl. *template*) v sistemu.

Izvede se t.i. ena-proti-mnogo (angl. *one-to-many*) primerjava, ki da kot rezultat eno ali več identitet, ki ustrezajo lastnostim osebe, v nasprotnem primeru pa iskanje identitete spodleti.

Slika 1.2: Štirje različni zajemi istega prstnega odtisa

Na problem verifikacije lahko naletimo pri prijavi v računalniško omrežje ali operacijski sistem, kjer moramo najprej vnesti uporabniško ime in potem še geslo. Z uporabo prstnih odtisov bi odpadla potreba po vpisovanju gesla. Ker najprej vnesemo uporabniško ime, bi se izvedla le primerjava med vhodnim prstnim odtisom in vzorcem.

S problemom identifikacije pa se največkrat soočijo policisti. Primerjava prstnega odtisa z mesta zločina z vsemi prstnimi odtisi v podatkovni zbirki jih ponavadi pripelje do odkritja zločinca.

Vrednotenje razpoznavanja

Za razliko od gesel in kriptografskih ključev, naletimo pri biometričnih sistemih na velike razlike med posameznimi zajemi vzorcev. V primeru prstnih odtisov se pogosto pojavlja šum, ki je posledica ureznin, opeklin ali kakšnih drugih poškodb. Pri fizičnem kontaku s čitalcem prstnih odtisov lahko pride do prevelike oziroma premajhne jakosti pritiska, težave povzroča tudi umazanija ali vlaga.

Ker pri primerjanju dveh različnih prstnih odtisov istega osebka težko pričakujemo popolno ujemanje, prihaja do dveh vrst napak:

- Napačno ujemanje (angl. *false match*): ujemanje dveh vzorcev različnih osebkov. Od tod izhaja verjetnost napačnega ujemanja (angl. *false*

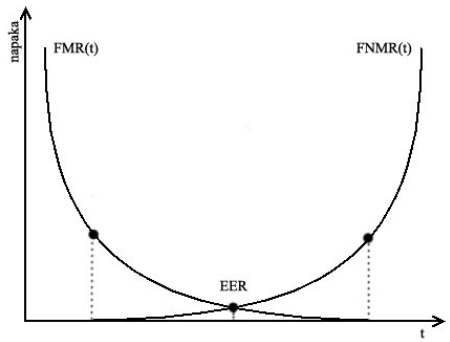
match rate – FMR):

$$FMR = \frac{\textit{število nepravilno sprejetih}}{\textit{število vseh poskusov}} \cdot 100 \ \ [\%] \tag{1.1}$$

- Napačno ne-ujemanje (angl. *false non-match*): ne-ujemanje dveh vzorcev istega osebka. Od tod izhaja verjetnost napačnega ne-ujemanja (angl. *false non-match rate – FNMR*):

$$FNMR = \frac{\textit{število nepravilnih zavrnitev}}{\textit{število vseh poskusov}} \cdot 100 \ \ [\%] \tag{1.2}$$

Ti dve razmerji sta odvisni od parametra sistema t, ki mu pravimo tudi prag (angl. *treshold*). Če t zmanjšamo in s tem naredimo sistem bolj toleranten, se *FMR(t)* poveča. Obratno velja, če t povečamo, naredimo sistem bolj varen, takrat se poveča *FNMR(t)*. Sistem ima običajno podano krivuljo operacijske značilnosti sprejemnika (angl. *receiver operating characteristic - ROC*), ki prikazuje *FMR* in *FNMR* v odvisnosti od t (slika 1.3). Pri nekem parametru t velja: *FMR(t) = FNMR(t)*, takrat je verjetnost obeh napak izenačena in enaka *EER* (angl. *equal error rate*).



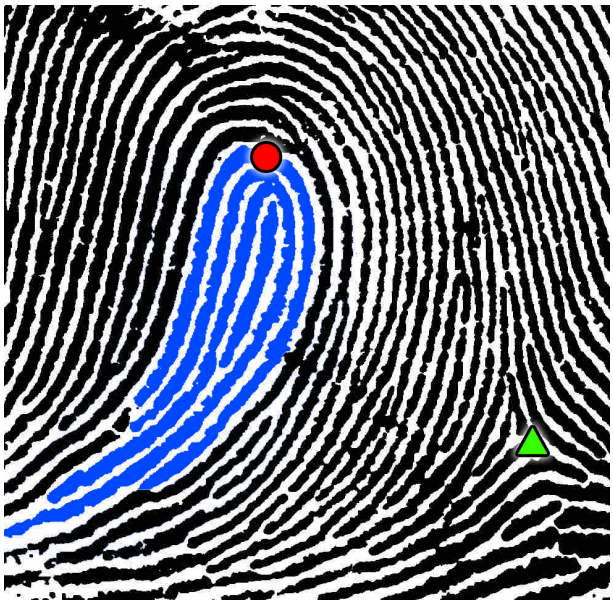
Slika 1.3: ROC krivulja

1.2.3 Zgradba prstnega odtisa

Poglavitna značilnost prstnega odtisa so grebeni in doline, ki so paralelno razporejeni po prstu. Skupaj predstavljajo bogato strukturno informacijo pri obdelavi prstnega odtisa. Glede na ločljivost zajemanja, ločimo značilke prstnega odtisa v tri kategorije [15].

Globalne značilke

Na globalni ravni tvorijo grebeni različne vzorce. Vzorec določajo singularne točke, imenovane jedra in delte, ki predstavljajo kontrolne točke, okoli katerih so ovite črte grebenov. Jedro (angl. *core*) odtisa je definirano kot center najbolj severne zanke. Vzorec zanke se pojavi pri prstnih odtisih, kjer grebeni vstopijo na eni strani, se ukrivijo in izstopijo na isti strani. Jedra najdemo tudi v spiralah, kjer grebeni tvorijo krožnice. Delta pa je področje odtisa, kjer grebeni tvorijo trikotno strukturo. Za boljšo predstavo sta jedro in delta prikazana tudi na sliki 1.4. Omenjene točke in groba oblika, ki jo tvorijo črte grebenov, so običajno osnova za klasifikacijo in indeksiranje prstnih odtisov, medtem ko samo poznavanje le-teh ni zadostno za učinkovito implementacijo ujemanja.



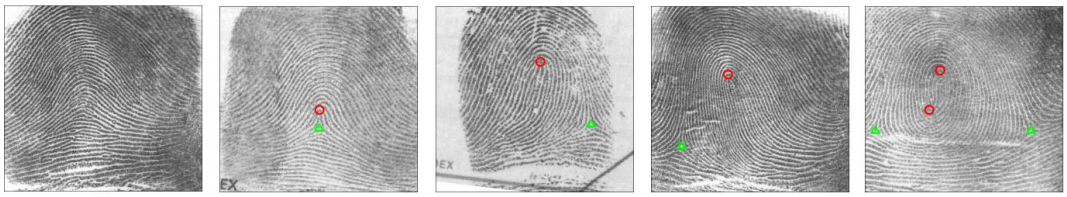
Slika 1.4: Primer zanke (modri grebeni), jedra (rdeč krog) in delte (zelen trikotnik)

Prvi, ki je razdelil oblike prstnih odtisov na devet razredov (lok, špičasti lok, zanka in šest tipov spiral) je bil Johannes Purkinje. Tudi Francis Galton in Henry Faulds sta se zavedala problema klasifikacije zato sta poskušala najti rešitev s pomočjo razdelitve prstnih odtisov v tri razrede (lok - A, zanka - L, spirala - W), kjer se označbe razredov vseh desetih prstov zapišejo v niz znakov (npr. LLAAW, LWAWL). Dobljen niz znakov lahko enostavno sortiramo po

abecedi. Kasneje je Henry predrugačil Galtonov sistem z definiranjem nekaterih novih razredov. Galton-Henryev sistem klasifikacije je prevzela večina držav. Na sliki 1.5 je prikazanih pet najbolj uporabljenih razredov Galton-Henryevega sistema:

- Lok (angl. *arch*): ima grebene, ki vstopijo na eni strani, tvorijo rahlo izboklino in izstopijo na drugi strani. Nimajo ne zank ne delt.
- Šotorast lok (angl. *tented arch*): je podoben običajnemu loku, le da je vsaj en greben zelo ukrivljen in vsebuje zanko ter delto.
- Leva zanka (angl. *left loop*): eden ali več grebenov vstopi z leve strani, se ukrivi in izstopi spet na levi strani. Prisotni sta singularni točki zanka in delta.
- Desna zanka (angl. *right loop*): podobno kot leva, le da grebeni vstopajo in izstopajo z desne.
- Spirala (angl. *whorl*): vsebuje vsaj en greben, ki potuje celotnih 360° okrog centra. Vsebuje dve zanki in dve delti. Lahko je sestavljena tudi iz dveh zank (leve in desne), takrat ji pravimo tudi dvojna zanka.

Naj še omenimo, da med globalne značilke poleg omenjenih sodijo še zunanja oblika odtisa, orientacijska slika in frekvenčna slika.



Slika 1.5: Najpogostejši vzorci prstnih odtisov: (a) lok (b) šotorast lok (c) leva zanka (d) desna zanka (e) spirala; jedra najbolj severne zanke so označena z rdečim krogcem, delte pa z zelenim trikotnikom

Lokalne značilke

Na lokalni ravni je bilo identificiranih 150 različnih značilnosti grebenov (angl. *minutiae* - iz latinske besede *minutia*, kar pomeni malost). Te značilnosti so neenakomerno razporejene po prstnem odtisu. Večina jih je odvisna od

kvalitete prstnega odtisa in so zato redko prisotne. Najbolj pogosti in največkrat uporabljeni značilki sta zaključek grebena (angl. *ridge termination*) in razcep grebena (angl. *ridge bifurcation*). Ti dve sta v splošnem stabilni in robustni glede na stanje vtisa prstnega odtisa, vendar njihovo iskanje predstavlja precejšen problem pri odtisih slabše kvalitete. Ostale pomembnejše značilke so še jezero (angl. *lake*), točka ali otok (angl. *island*), špica - kratek rob (angl. *spur*) in križišče (angl. *crossover*). Nekatere naštete značilke so prikazane na sliki 1.6.

Slika 1.6: Pogostejše značilke prstnih odtisov

Z uporabo lokalnih značilk je relativno enostavno primerjati dva prstna odtisa, kadar sta zajeti sliki dobre kvalitete. Večino detajlov, ki se nahajajo na enem prstnem odtisu, moramo najti tudi na drugem prstnem odtisu. Detajli se morajo ujemati v tipu značilke, relativni lokaciji in usmeritvi.

Do problema pride pri tistih slikah prstih odtisov, kjer je prišlo do kakršnekoli deformacije ali šuma. Posledično pride do zamika detajlov, ali pa nekaterih detajlov sploh ne moremo razbrati s slike (pri zajemu prstnih odtisov s pomočjo črnila lahko pride do izgube detajlov ob preveliki količini črnila, pri optičnih senzorjih pa je problem umazanija).

Mikro značilke

Na mikro ravni opazujemo notranje podrobnosti grebena, ki jih predstavljajo znojne pore (angl. *sweat pores*), katerih število, lega in oblika so zelo razločujoče. Takšno iskanje značilk je možno le v kvalitetnih slikah z visoko ločljivostjo (1000 dpi), kar je običajno nekoliko manj praktično za večino aplikacij.

1.3 Čitalci prstnih odtisov

V preteklosti se je prstne odtise pridobivalo s pritiskom prsta, namočenega v črnilo, na bel papir. Današnji verifikacijski (identifikacijski) sistemi pa uporabljajo neposredni zajem (angl. *live-scan*), kjer pridobimo digitalno sliko prstnega odtisa v realnem času. V nadaljevanju bomo predstavili štiri najpogosteje uporabljene tipe čitalcev z neposrednim zajemom.

1.3.1 Optični čitalci

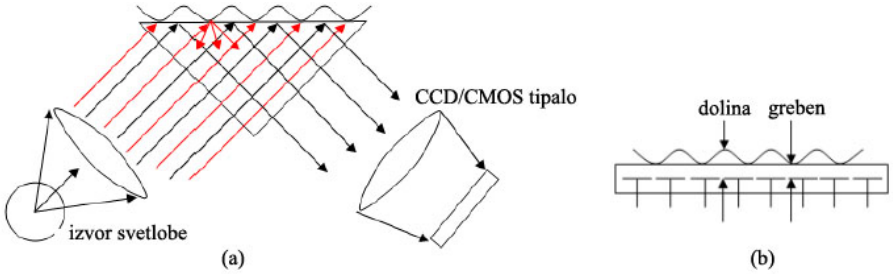
So najstarejši in najpogosteje uporabljeni digitalni čitalci na trgu [7]. Za pretvorbo slike prstnega odtisa v digitalni signal uporablja CCD ali CMOS tipalo. So razmeroma poceni in zajemajo slike z ločljivostmi do 500 dpi. Večina čitalcev temelji na tehniki notranje reflekcije (angl. *FTIR - Frustrated Total Internal Reflection*). Omenjena tehnika deluje tako, da prst v celoti položimo na prizmo, na katero usmerimo svetlobo. Tam, kjer se grebeni dotikajo prizme, se svetloba razprši in ne doseže tipala CCD, kar se odraža na sliki kot črna barva. Pri dolinah pa se žarki zaradi pojava interne refleksije v večini odbijejo nazaj na tipalo, kar se odraža na sliki kot bela barva. Princip delovanja FTIR vidimo na sliki 1.7(a).

Kvaliteta slik je odvisna od vlažnosti oziroma suhosti prsta. Problem predstavljajo tudi ostanki prstnega odtisa od prejšnih vzorčenj. Optični čitalci so tudi eni večjih na trgu. Ker zaznavajo obliko prstnega odtisa v treh dimenzijah, jih je težko pretentati s slikami prstnih odtisov.

1.3.2 Kapacitativni čitalci

Kapacitativni čitalci (slika 1.7(b)) so sestavljeni iz silicija, ki vsebuje polje kapacitativnih senzorjev [7]. Ko se s prstom dotaknemo površine senzorja, se med površino kože ter senzorji vzpostavi majhen električni naboj. Jakost tega je na področjih, kjer so prisotni grebeni, manjša kakor na področjih, kjer so prisotne doline. Z opiranjem na to razliko zna čitalec ustvariti 8 bitno črno belo sliko prstnega odtisa.

Slabost kapacitativnih čitalcev je, da naboj med prstom in senzorjem zelo hitro upade, zato mora biti zaščitna plast čim bolj tanka. Posledica tega je tudi občutljivost na statično elektriko, kar lahko privede do napak. Čitalci se slabo obnesejo pri vlažnih in mastnih prstnih, vendar pa so zelo dobri pri prepoznavanju ponarejenih odtisov. Zaradi majhne velikosti večina elektronskih naprav z identifikacijo preko prstnega odtisa uporablja kapacitativne čitalce.



Slika 1.7: (a) Čitalec z uporabo tehnike FTIR in (b) kapacitativni čitalec

1.3.3 Ultrazvočni čitalci

Ultrazvočni čitalci naj bi bili najnatančnejši izmed vseh čitalcev [7]. Temeljijo na pošiljanju ultrazvočnih singalov proti prstu in sprejemanju odbojnih signalov. Preko teh se nato izračuna oddaljenost posamezne točke in posledično prstni odtis. Ultrazvočni signali lahko prodrejo tudi skozi umazanijo in oljne madeže, kar je njihova velika prednost. Zato ponavadi z ultrazvočnimi čitalci pridobimo slike prstnih odtisov zelo dobre kvalitete.

Slaba stran takih čitalcev je, da vsebujejo mehanske dele, so veliki in dragi. Ponavadi se uporabljajo pri policijskem delu za identifikacijo zločincev.

1.3.4 Termični čitalci

Ti čitalci so ponavadi narejeni iz piroelektričnih materialov (angl. *pyro-electric*). Piro izhaja iz grške abecede in pomeni ogenj, toplota. Čitalci namreč ob stiku s prstom zaradi toplote slednjega proizvedejo majhen električni tok. Preko tega lahko razberemo strukturo prstnega odtisa, saj je temperaturna razlika med grebeni prstnega odtisa, ki so v stiku s površino čitalca, in zračnih žepov v dolinah, dovolj velika.

Pri termičnih čitalcih, ki zajamejo celoten prstni odtis, pride do problema, ko se prst dotakne površine čitalca. Takrat v trenutku pride do ravnovesja v temperaturi kože, s čimer se informacija o grebenih in dolinah izgubi. Zato so termični čitalci ponavadi izdelani na poteg (angl. *sweep*). Tak čitalec mora biti sposoben zajeti toliko prekrivajočih se slik prstnega odtisa, da lahko potem sestavi celotno sliko. Dobro se odrežejo tudi pri vlažnih, mastnih in umazanih prstih [18].

1.4 Opis problema

Sistem za verifikacijo oseb na podlagi prstnega odtisa temelji običajno na treh korakih:

- zajem podatkov: s pomočjo čitalca prstnega odtisa,
- iskanje značilk: izdelava digitalnega zapisa vzorca,
- odločanje: primerjava pridobljenega vzorca z vzorci v podatkovni zbirki.

Podrobnejši opis teh korakov in podkorakov sledi v nadaljevanju [12].

1.4.1 Zajem prstnega odtisa

Čeprav so bili prvi elektronski čitalci predstavljeni že več kot 30 let nazaj, se danes še mnogokrat uporablja pridobivanje prstnih odtisov s pomočjo črnila. Pri tej metodi poznamo dva načina: z vrtenjem (angl. *Rolled Ink Fingerprint*) ter na pritisk (angl. *Dab Ink Fingerprint*). Pri prvem na praznem papirju zavrtimo v črnilo namočen prst od enega roba nohta do drugega, s čimer zajamemo skoraj vse grebene na prstu, toda odtis je velikokrat deformiran. Hkrati je postopek okoren, počasen ter zahteva izkušenost. Pri drugemu v črnilo namočen prst enostavno pritisnemo na bel papir. Pridobljena površina je sicer manjša, vendar pa se izognemo popačenosti. Kasneje obe sliki enostavno pretvorimo v digitalno obliko s pomočjo optičnega čitalca.

Za nas bolj zanimiv način je neposredni zajem. Prstni odtis lahko zajamemo s pomočjo čitalca prstnih odtisov. Tako pridobljena slika je velikokrat bolj kvalitetna, kot če bi uporabili črnilo. Velika prednost pa je tudi čas pridobitve slike, saj današnji čitalci zajamejo sliko v nekaj sekundah.

Naj omenimo še, da je zajem prstnega odtisa eden od najpomembnejših korakov pri obdelavi, saj je vsak nadaljni korak odvisen od kvalitete slike prstnega odtisa [22].

1.4.2 Segmentacija

Pri segmentaciji (angl. *segmentation*) poskušamo ločiti prstni odtis od ozadja. Tako ločevanje je smiselno, saj se skušamo izogniti procesiranju predelov slike, ki za nas niso zanimivi ali pa vsebujejo preveliko šuma. S tem povečamo možnost uspešnega primerjanja ter hkrati skrajšamo čas procesiranja.

Ker so prstni odtisi progastega vzorca, loči odtis od ozadja prisotnost črtastega in orientiranega vzorca, medtem ko je ozadje izotropno (brez dominantne orientacije). Če bi bilo ozadje vedno konstantno in svetlejše od površine odtisa, bi segmentacijo omogočal že preprost pristop na osnovi intenzitete. Ker pa je v praksi vedno prisoten šum, potrebujemo kompleksnejše tehnike.

1.4.3 Izboljšanje kvalitete prstnega odtisa

Učinkovitost algoritma za iskanje značilk je močno odvisna od kvalitete slike prstnega odtisa. V idealni sliki prstnega odtisa je iskanje značilk relativno preprosto, v praksi pa imamo opravka s slikami prstnih odtisov slabše kvalitete. Težavo največkrat predstavljajo ureznine, praske, pa tudi preveč suhi ali vlažni prsti. Do popačenj pride tudi zaradi prevelike jakosti pritiska na čitalec. Ločimo dobre, srednje dobre in slabe prstne odtise, ki lahko vsebujejo naslednje tipe nepravilnosti:

- prekinitve grebenov,
- vzporedni grebeni niso dovolj ločeni med seboj zaradi prisotnosti šuma,
- ureznine, gube, modrice.

Omenjene nepravilnosti lahko v nadaljnemu procesiranju vodijo do določitve napačnih značilk, zgrešitve pravih značilk in napačne določitev lokacije značilk. Da bi se v čim večji meri temu izognili, je potrebno uporabiti algoritem za izboljšanje kvalitete. Kvaliteto prstnega odtisa lahko določimo za posamezne regije. Tako ločimo:

- Dobro definirane regije, kjer so grebeni jasno ločeni drug od drugega.
- Obnovljive regije, kjer so grebeni pokvarjeni z manjšim številom razpok, gub, madežev ipd. vendar so še vedno vidni, informacijo za rekonstrukcijo pa dobimo iz sosednih regij.
- Neobnovljive regije, kjer so grebeni pokvarjeni s preveč šuma in motenj, da bi bile rekonstrukcije s pomočjo sosednih regij uspešne.

Cilj algoritma za izboljšanje kvalitete (angl. *enhancement*) je, da popravi strukturo obnovljivih regij in kot možnost označi neobnovljive regije kot prešumne za nadaljne procesiranje.

1.4.4 Binarizacija

Z binarizacijo dosežemo zmanjšanje bitne globine črno-bele slike iz prejšnega koraka na samo 1 bit na slikovni element (angl. *pixel*). Najpreprostejši algoritem za binarizacijo uporablja globalni prag t. Točke, s stopnjo sivine manjše od t postavi na 0, ostale pa na 1. V splošnem en sam prag ne zadostuje, ker so lahko različne regije slike izpostavljene različnim kontrastom in intenzitetam. Tehnika lokalnega praga spreminja t lokalno tako, da se prilagaja povprečni lokalni inteziteti, kar izboljša rezultate binarizacije.

1.4.5 Iskanje značilk

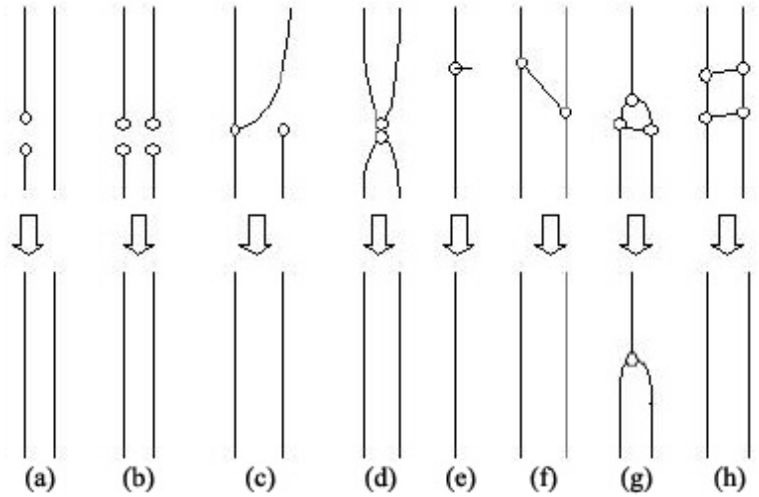
Mnogo današnjih sistemov temelji na primerjanju s pomočjo značilk, zato je zanesljivost koraka iskanja značilk zelo pomembna. Metoda iskanja značilk zahteva za vhod binarno sliko prstnega odtisa, ki jo dobi v prejšnem koraku. Samo iskanje se ponavadi izvede po tanjšanju binarne slike, s čimer pridobimo skelet (angl. *skeleton*) prstnega odtisa. To pomeni, da so vsi grebeni široki natanko 1 slikovni element.

Iz skeleta odtisa pridobljene značilke pa niso vse verodostojne. Mnogo jih je nastalo zaradi nepravilnosti, kot so špice, jezera, prekinjeni grebeni itd. Zato se poskušamo s postopkom naknadne obdelave takih (angl. *postprocessing*) znebiti. Veliko nepravilnih značilk se pojavi na robovih prstnega odtisa, saj je tam pri zajemanju slike največ napak. Zato se na robu prstnega odtisa določi območje, kjer značilk ne iščemo. Ostale nepravilnosti, ki se lahko pojavijo, pa lahko vidimo na sliki 1.8.

1.4.6 Klasifikacija

Podobno kot pri ročni identifikaciji oseb, se lahko klasifikacija uporabi tudi pri samodejni identifikaciji oseb. Še posebej dobrodošla je v sistemih, ki hranijo sto ali več tisoč prstnih odtisov. S klasifikacijo se močno zoža izbor kandidatov in s tem izboljša tudi čas procesiranja.

Klasifikacija prstnih odtisov je težavna naloga predvsem zaradi majhne medrazredne variabilnosti in velike intra-razredne variabilnosti, kar pomeni, da so si prstni odtisi istega razreda lahko zelo različni in da so si prstni odtisi različnih razredov lahko precej podobni. Poleg tega klasifikacijo dodatno otežuje še šum. Skoraj vse metode temeljijo na eni ali več naslednjih značilnosti: pretok grebenov, orientacijska slika, singularne točke in odzivi gaborjevih filtrov.



Slika 1.8: Primeri napačnih struktur podrobnosti in pripadajoči popravki: (a) prekinitev, (b) dvojna prekinitev, (c) prekinitev in združitev, (d) združitev, (e) špica, (f) most, (g) trikotnik, (h) lestev

1.4.7 Primerjanje

Algoritmi za ujemanje prstnih odtisov primerjajo dva prstna odtisa in vrnejo rezultat. Rezultat je lahko v obliki stopnje ujemanja (med 0 in 1) ali v obliki binarne odločitve (se ujema/se ne ujema). Prstni odtis pridobljen med registracijo imenujemo vzorec T (angl. *template*). Prstni odtis, za katerega želimo ugotoviti ujemanje, pa imenujemo vhod (angl. *input*) I.

Ujemanje prstnih odtisov je precej težka naloga predvsem zaradi velikih razlik med različnimi odtisi istega prsta. Glavni razlogi za te razlike so naslednji:

- Premik: Isti prst je lahko postavljen na različna mesta senzorja pri različnih poizvedbah.
- Rotacija: Isti prst je lahko rotiran za različne kote glede na površino senzorja pri različnih poizvedbah.
- Delno prekrivanje: Zaradi rotacije in premika pride do tega, da nekateri deli prstnega odtisa niso zajeti, kar lahko privede do manjšega prekrivanja med zajeto površino in vzorcem prstnega odtisa.
- Ne-linearne distorzije: Se pojavijo zaradi elastičnosti prsta ob pritisku na senzor. Ob pritisku na senzor nepravokotne sile, ki delujejo na prst

povzročijo premike, ki imajo za posledico raztezanje ali krčenje zajete slike prstnega odtisa.

- Pritisk in stanje kože: Stik med prstom in površino senzorja ni konstanten zaradi pritiska, suhe kože, potu, kožnih bolezni itd. Kot posledica je zajet prstni odtis šumen, stopnja šuma pa je odvisna od naštetih faktorjev in je precej različna pri različnih zajetjih istega prsta.
- Šum: Nastane predvsem zaradi stanja senzorja.
- Napake pri iskanju značilk: So pogoste in precej variabilne in lahko nastanejo v vsakem koraku algoritma.

1.5 Cilji

Prstni odtisi so bili v začetku 20. stoletja sprejeti kot zanesljivo orodje za identifikacijo oseb. Od takrat so postali de-facto avtentikacijski standard za uporabo v boju proti kriminalu, vse bolj se uveljavljajo tudi na civilnem področju. S svojo razširjenostjo, unikatnostjo in nespremenljivostjo imajo veliko prednosti pred ostalimi tradicionalnimi tehnikami. Prav tako pa so današnji čitalci prstnih odtisov hitri in dajo veliko bolj kvalitetne slike kakor postopki pridobivanja odtisov s črnilom.

Z uporabo čitalca ter računalniše opreme želimo izdelati sistem za verifikacijo oseb na podlagi prstnih odtisov, ki bi imel naslednje lastnosti:

- robustnost glede kakovosti vhodne slike,
- odpornost na rotacijske in translacijske zamike,
- neodvisnost od ločljivosti slike.

Pri implementaciji postopka smo uporabljali slike prstnih odtisov, ki so bile uporabljene na tekmovanju *Fingerprint Verification Competition* [29] leta 2002, krajše FVC 2002. Uporabljene so bile štiri različne podatkovne zbirke (DB1, DB2, DB3 in DB4), ki so bile zajete na naslednji način:

- DB1: optični čitalec "TouchView II"od podjetja Identix,
- DB2: optični čitalec "FX2000"od podjetja Biometrika,
- DB3: kapacitativni čitalec "100 SC"od podjetja Precise Biometrics,

- DB4: sintetično generirani prstni odtisi - SFinGe [30].

Vsaka zbirka sestoji iz 110 prstnih odtisov, vsak prstni odtis pa je vzorčen z 8 slikami, kar skupaj znese 880 slik na zbirko. Za nastavljanje parametrov za vsako zbirko smo uporabili učno množico (prstni odtisi od 101 do 110), v kateri je bilo skupaj 80 prstnih odtisov. Za namene testiranje smo uporabili testno množico (prstni odtisi od 1 do 100), ki je vsebovala 800 slik.

Lastnosti podatkovnih zbirk so podane v tabeli 1.1:

	tip čitalca	**velikost slike**	**testna množica**	**učna množica**	**ločljivost**
DB1	optični	388×374	100×8	10×8	500 dpi
DB2	optični	296×560	100×8	10×8	569 dpi
DB3	kapacitativni	300×300	100×8	10×8	500 dpi
DB4	SFinGe v2.51	288×384	100×8	10×8	okoli 500 dpi

Tabela 1.1: Lastnosti podatkovnih zbirk uporabljenih pri razvoju in testiranju aplikacije

1.5.1 Uporabljena orodja

Aplikacijo smo izdelali v računalniškemu jeziku C# 3.5. Vsak korak algoritma smo izdelali v svoji knjižnici DLL. Testiranje je potekalo na računalniku z enojedrnim procesorjem AMD Sempron 2000MHz in 1500MB pomnilnika.

Poglavje 2

Sorodne rešitve

V tem poglavju bomo pogledali tri sorodne rešitve, ki temeljijo na tehnologiji prstnih odtisov.

2.1 JetFlash 220 - USB ključ s čitalcem prstnih odtisov

Preizkusili smo USB ključ podjetja Transcend [34], ki nudi zaščito podatkov z vgrajenim čitalcem prstnih odtisov. Ob prvem zagonu nam program, ki je shranjen na ločeni particiji na ključku, ponudi začetno nastavitev aplikacije. Najprej je potrebno vpisati geslo, kasneje pa sledi registracija kateregakoli prsta na obeh rokah (lahko jih uporabimo tudi več). Pri registraciji je potrebno prst potegniti čez čitalec tolikokrat, dokler programu ne uspe sestaviti slike prsta. V primeru, da nam v šestih poskusih ne uspe, je potrebno postopek ponoviti. Po začetnih nastavitvah nam program omogoča zaščito podatkov na računalniku, ključu ali kakšnem drugem zapisljivem mediju. Potrditev zaščite overimo z geslom ali s prstnim odtisom kateregakoli registriranega prsta. Podatki se zakriptirajo z 256 bitnim algoritmom AES in jih lahko odpremo šele s ponovnim overjanjem preko čitalca ali gesla.

V samih navodilih in specifikacijah nismo našli nobenih podatkov o čitalcu, tudi slik registriranih prstnih odtisov ni bilo mogoče preveriti. Zato bomo v tem delu podali le naše ugotovitve. Čitalec spada v skupino termičnih čitalcev in je drsnega tipa, saj velikost USB ključa preprečuje vgradnjo večjih čitalcev. Delovanje je dokaj hitro in zanesljivo. Povprečno branje in primerjanje odtisa traja manj kot sekundo, pa tudi pretentati nam ga ni uspelo z drugimi odtisi. V primeru prehitrega potega čez čitalec nas program opozori, da ni uspel

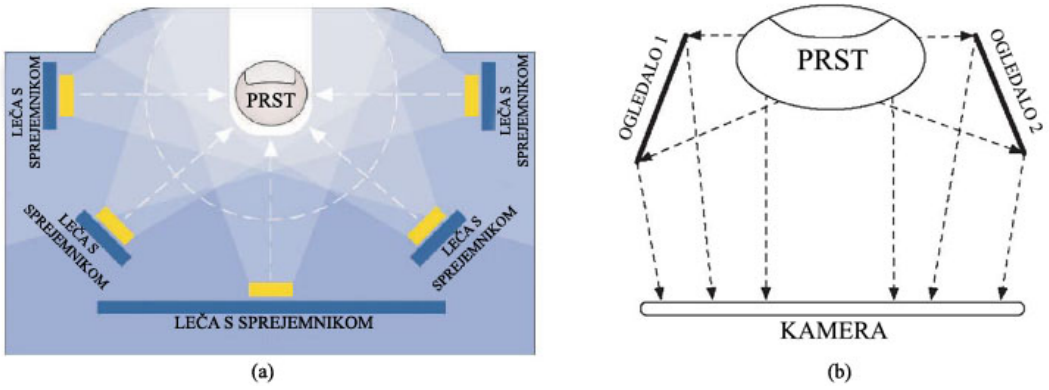
pridobiti celotne slike. Preizkusili smo tudi odpornost na rotacijo in ugotovili, da se pojavijo problemi pri zamiku prsta za 30 stopinj in več.

Kakšna večja primerjanja z našim pristopom niso bila mogoča, saj nam ni uspelo dobiti podatkov o samem postopku procesiranja prstnega odtisa.

2.2 TBS - Touchless Biometric Systems

Medtem, ko večina današnjih rešitev temelji na obdelavi 2D slike prstnega odtisa, je podjetje TBS ponudilo rešitev, kjer stik prsta s površino čitalca ni več potreben. Za pridobivanje prstnega odtisa se namreč uporablja naprava, ki analizira in pridobi prstni odtis v 3D okolju. TBS je sploh prvo podjetje na svetu, ki je ponudilo rešitev na osnovi 3D prstnega odtisa.

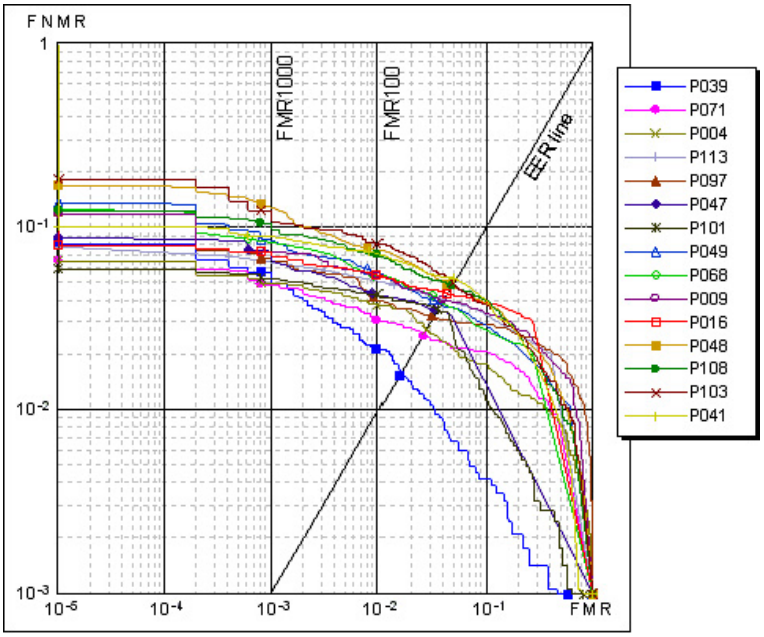
Naprave s 3D branjem prstnega odtisa uporabljajo eno ali več kamer za pridobitev odtisa (slika 2.1). Glavna prednost te tehnologije se zato odraža v kvaliteti prstnega odtisa, saj bomo deformacije, popačenja in šum zaradi čitalca na takih slikah težko našli. Tudi do problemov z ostanki prejšnih vzorčenj tu ne pride. Kot rezultat dobimo 3D model prstnega odtisa, ki nudi veliko več podrobnosti in površine prstnega odtisa, kakor običajne metode na dotik oziroma poteg [6]. Seveda pa je potrebno v primeru, da bi hoteli združljivost s čitalci, kjer prst zavrtimo od enega roba nohta do drugega, v procesiranje vključiti še dodaten korak, ki razvije 3D sliko modela v 2D predstavitev prstnega odtisa.



Slika 2.1: Prikaz 3D zajema prstnega odtisa: (a) rešitev z uporabo petih kamer (b) rešitev z uporabo ene kamere in dveh dodatnih ogledal

Tehnologija je v porastu, razlog za majhno prisotnost na trgu pa je cena. Proizvajalci klestijo proizvodne stroške z uporabo manjšega števila kamer, v

nekaterih primerih celo z uporabo samo ene. Seveda pa to vpliva tudi na velikost zajete površine prstnega odtisa.



Slika 2.2: ROC krivulja algoritma ElasticMatch

TBS nudi trenutno eno napravo za 3D branje – BioGuard Surround. Ločljivost slik pridobljena s to napravo je 500 dpi. Čas zajema slike znaša manj kot eno sekundo, čas identifikacije pa manj kot dve sekundi. Pri identifikaciji uporabljajo rešitev podjetja *Innovatrics* [35]. Kot smo lahko razbrali iz skopega opisa, uporabljajo algoritem, ki je zelo robusten in v koraku izboljšanja kvalitete poskuša odstraniti vse znake šuma. Primerjanje temelji tako kot pri našem algoritmu na lokalnih značilkah, vendar uporabljajo lastno razvit primerjalni algoritem poimenovan *ElasticMatch*. Za slednjega trdijo, da je zmožen izvesti 1.500.000 primerjanj v sekundi, vendar ni znano na kakšni strojni opremi. O natančnosti algoritma pa lahko sklepamo iz slike 2.2, ki prikazuje ROC krivuljo za podatkovno zbirko DB2 iz tekmovanja FVC 2004. Na tej zbirki je dosegel algoritem ElasticMatch najnižjo EER stopnjo, sicer pa je skupaj na tekmovanju dosegel dve zlati in eno bronasto medaljo.

2.3 Printrak Biometrics Identification Solution (BIS)

V tem delu si bomo na splošno pogledali AFIS (angl. *Automated Fingerprint Identification System*), ki ga uporablja tako FBI kot tudi New Scotland Yard, in nenazadnje tudi mnoge druge inštitucije po svetu. Sam sistem je bil razvit v podjetju Printrak v začetku sedemdesetih let, leta 2000 pa je prišel pod okrilje Motorole. Od leta 2009 poteka razvoj pod okriljem skupine Sagem.

BIS je zgrajen na storitveno orientirani arhitekturi, uporablja pa komercialno strojno opremo. Zaradi tega je zelo fleksibilen in zaradi svoje modularne zasnove enostaven za nadgradnjo. Poleg tega je zmožen tudi izmenjave podatkov z drugimi ANSI oziroma ISO kompatibilnimi sistemi, kar mu zagotavlja veliko interoperabilnost. BIS se ne omejuje le na prstne odtise, omogoča podporo tudi odtisom rok, obraznim potezam, šarenici, podpisom ter drugim dokumentom.

Sistem omogoča primerjavo prstnih odtisov, dobljenih s črnilom, kakor tudi odtisov, pridobljenih s čitalcem. Podpira slike ločljivosti 1000 dpi in 500 dpi. Glavna prednost pa je povezljivost. Na področju industrije podpira standarde ANSI/NIST, ANSI/INCITS, ISO in Prüm, na področju informacijske tehnologije pa podpira standarde XML, HTTP, WSQ, JPEG2000, SOA/ESB in Java. Zaradi tega je zmožen izmenjave podatkov in povezav s podobnimi AFIS sistemi po svetu.

Zaradi skrivanja podrobnosti pred konkurenco nismo o samem sistemu našli nobenega podatka o uporabljenih metodah pri obdelavi prstnih odtisov. Nekaj vzporednic lahko povlečemo s podobnim produktom Motorole, sistemom Omnitrak. Če predpostavimo, da uporabljata podobne rešitve, lahko nekaj informacij o natančnosti sistema pridobimo na podlagi testiranja [32], ki ga je izvedel ameriški NIST (angl. *National Institute of Standards and Technology*). Tu je Omnitrak zasedel mesto tik pod stopničkami s povprečno vrednostjo FNMR 0.01% in FMR 0.001%. Če velja podobno tudi za BIS, lahko zaključimo, da gre za superioren sistem, ki nudi zanesljivost in natančnost na zelo visokem nivoju.

2.4 Špica - Time & Space

Špica je vodilno podjetje v Sloveniji na področju kontrole pristopa, registracije delovnega časa in mobilnih rešitev. Že pred nekaj leti so spoznali potencial biometrične identifikacije, zato imajo danes z njo že veliko izkušenj in lepo

število referenc.

Z Ministrstvom za notranje zadeve je Špica sodelovala pri uvedbi biometričnih potnih listov, s čimer je Slovenija izpolnila določila uredbe EU, po kateri morajo novi potni listi vsebovati tudi biometrijo dveh prstnih odtisov. Špica je vsem upravnim enotam dobavila čitalce prstnih odtisov *Dermalog ZF1*. Slednji zajame sliko prstnega odtisa velikosti 320×480 slikovnih elementov z ločljivostjo 500 dpi. Čitalec ustreza standardom EFTS/F in BSI TR-PDÜ, ki se nanašajo na kakovost zajetih slik.

Špica ponuja tudi rešitev Time & Space, s katero omogoča sledenje delovnega časa, natančen obračun ter pripravo podatkov, ki jih je mogoče prenesti v različne programe za obračun plač ali druge HR sisteme. Time & Space združuje različne platforme, naprave in tehnološke rešitve, že več let pa vgrajujejo tudi rešitve na osnovi prstnih odtisov. Pri tem uporabljajo čitalce vodilnega svetovnega proizvajalca *Bioscrypt*. Če definiramo natačneje, se uporabljata čitalca *Bioscrypt V-Pass* ter *V-Smart*. Prvi nudi identifikacijo brez uporabe kartice ali PIN kode, ali verifikacijo (potreben dodatni identifikator). Kapaciteta čitalca znaša do 500 vzorcev v načinu identifikacije in do 4.000 vzorcev v načinu verifikacije. Namenjen je samostojnim rešitvam kontrole pristopa. Čitalec *V-Smart* pa ima vgrajen še *Mifare* čitalec pametnih kartic (angl. *smartcard*). Uporablja se za „vzorec na kartici“ (angl. *Template-On-Card*) rešitve. Čitalci uporabljajo za obdelavo prstnih odtisov Bioscryptovo programsko opremo *VeriAdmin*. O natančnosti nam dosti pove podatek, da se je njihova rešitev na treh zaporednih tekmovanjih FVC (2000, 2002, 2004) uvrstila na prvo mesto.

2.5 Fingerprint Verification Competition 2002

Pri testiranju naše aplikacije smo se opirali na podatkovne zbirke, uporabljene na tekmovanju FVC 2002. Tekmovanje se je prvič odvijalo v letu 2000, zaradi velikega zanimanja in interesa pa se je poleg leta 2002 ponovilo še leta 2004 in 2006. Tekmovanje je odprtega tipa, sprejemale so se prijave tako iz akademske sfere kot tudi industrijske. Cilj tekmovanja je primerjati najnovejše algoritme in ponuditi pregled nad razvojem in napredkom tehnologije prepoznavanja prstnih odtisov.

Vsak udeleženec tekmovanja je moral oddati dve izvršilni datoteki, eno za registracijo in drugo za primerjavo prstnih odtisov. Osnutek datotek je bil na voljo tekmovalcem, ti pa so ju potem priredili svojim algoritmom. Pri testiranju se je za vsak algoritem in za vsako podatkovno zbirko merilo 13

parametrov, omenili bomo le najpomembnejše:

- EER,
- FMR100: najnižji FNMR, kadar je $FMR \leq 1\%$,
- FMR1000: najnižji FNMR, kadar je $FMR \leq 0,1\%$,
- povprečen čas registracije,
- povprečen čas primerjave.

Testiranje se je izvedlo nad štirimi različnimi podatkovnimi zbirkami s po 880 prstnimi odtisi. Vsaka zbirka je bila pridobljena z drugim čitalcem, zadnja pa je vsebovala programsko ustvarjene prstne odtise, zato se zbirke razlikujejo po kvaliteti in velikosti. Vsak tekmovalec je imel na voljo 80 prstnih odtisov iz vsake zbirke, ki so mu služili za nastavljanje parametrov. Za potrebe testiranja se je uporabilo ostalih 800 prstnih odtisov. Testiranje se je izvedlo na sistemu Pentium III 933MHz z operacijskim sistemom Windows 2000.

Na tekmovanju je sodelovalo 31 tekmovalcev, od tega 21 iz industrijskega področja, 6 iz akademskega področja ter 4 iz drugih področij. Pri zbirkah DB1, DB3 in DB4 je najboljši rezultat dosegljo podjetje Bioscrypt Inc., pri zbirki DB2 pa podjetje, ki je ostalo anonimno. V povprečju je torej podjetje Bioscrypt Inc. zasedlo prvo mesto. Za njihov algoritem znaša EER 0,19%, FMR100 0,15%, FMR1000 0,28%, povprečen čas registracije 0,11 sekunde in povprečen čas primerjave 1,97 sekunde. Da gre za odličen algoritem, je podjetje dokazalo na tekmovanju FVC 2004, kjer je v odprti kategoriji (brez omejitev) zasedlo prvo mesto, v lahki kategoriji (omejitve pomnilnika, časa procesiranja ter velikosti vzorca) pa četrto.

S testiranjem našega algoritma nad zbirkami iz tekmovanja FVC 2004 in primerjanjem rezultatov smo lahko dobili odgovor, kako dobro se naš algoritem obnese v primerjavi z ostalimi pomembnimi igralci na področju prstnih odtisov. Velja pa, da je to le okvirna primerjava, saj tekmovanje ni mišljeno kot uradno ocenjevanje uspešnosti sistemov. Zato tudi zajete slike ne ustrezajo predpisanim standardom, poleg tega pa se uporabljajo tudi slike, ki niso bile zajete v resničnem okolju. Kljub temu je bilo za tekmovanje vsako leto več zanimanja, vendar žal nismo zasledili nobene informacije o morebitnem nadaljevanju.

Poglavje 3

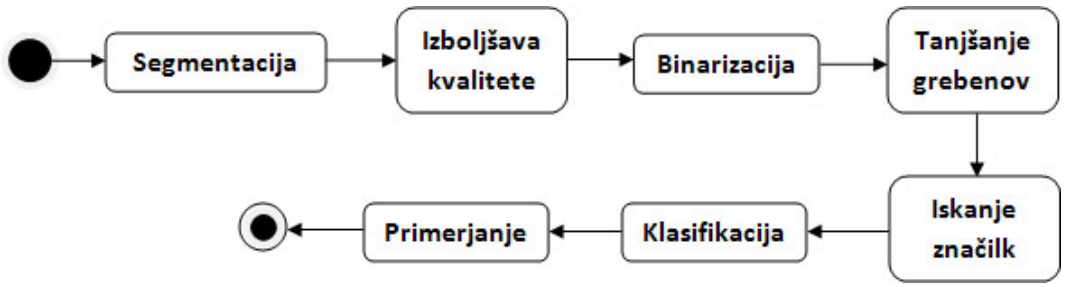
Algoritem za verifikacijo oseb na podlagi prstnega odtisa

V tem poglavju je opisan postopek, ki smo ga uporabili pri vzpostavitvi sistema za verifikacijo oseb na podlagi prstnega odtisa. Predstavljeni in opisani so posamezni koraki postopka ter parametri, ki smo jih uporabili.

3.1 Koraki algoritma

Postopek smo razdelili na korake, ki so prikazani na sliki 3.1. Ker vsak korak predstavlja zaokoroženo celoto, smo tudi pri izdelavi sistema za vsak korak izdelali lastno knjižnico DLL. Z modularno arhitekturo smo zagotovili enostavnost pri morebitnih kasnejših nadgradnjah.

Vhod v algoritem je slika v formatu *TIF*. Barvna paleta je črnobela oziroma sivinska z barvno globino 8 bitov na slikovni element.



Slika 3.1: Koraki algoritma

3.2 Segmentacija

Pri segmentaciji poskušamo čim bolje ločiti področje, kjer se nahaja prstni odtis, od ozadja. S tem se izognemo nadaljnemu procesiranju šumnih regij prstnega odtisa, kar bi vplivalo na kočni rezultat. Ker sestoji prstni odtis iz progastega vzorca, na ozadju pa se lahko pojavlja šum, enostaven pristop ni mogoč.

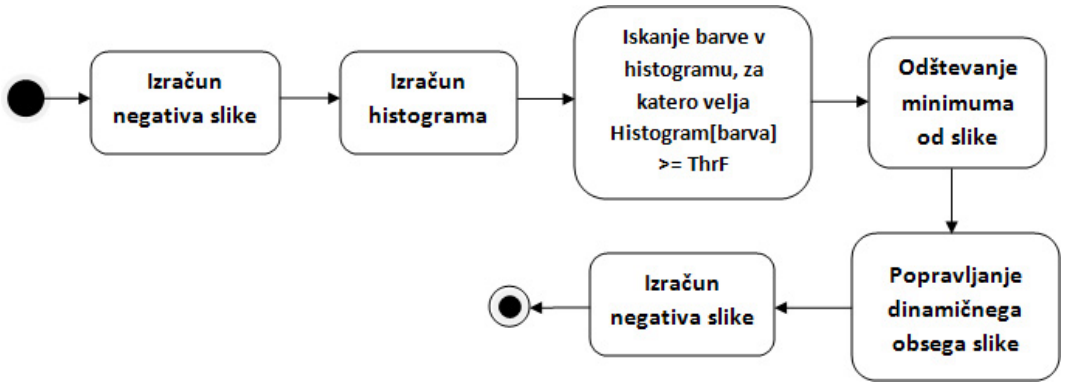
Odločili smo se za pristop, ki temelji na sledečih korakih [8]:

- odštevanje vrednosti, ki predstavlja ozadje,
- segmentacija.

Rezultat segmentacije je polje velikost M × N, kjer je M širina, N pa višina slike. V polju je za vsak element slike shranjena vrednost 0 ali 1. Prva pomeni, da element pripada ozadju slike, druga pa ospredju.

3.2.1 Odštevanje vrednosti, ki predstavlja ozadje

Namen tega koraka je določiti vrednost, ki predstavlja ozadje slike in jo odšteti od vrednosti vsakega slikovnega elementa na sliki prstnega odtisa. Poleg tega pa je cilj tudi povečati dinamično razdaljo med grebeni in dolinami (slika 3.2).



Slika 3.2: Prikaz korakov odštevanja ozadja sliki prstnega odtisa

Opis korakov:

1. Izračun negativa slike (neg_I) iz originalne slike (I):

$$neg\_I(i,j) = 255 - I(i,j) \tag{3.1}$$

Enačba 3.1 se izvede za vsak slikovni element (i,j) na sliki I. V našem primeru smo uporabili knjižnico *Aforge .NET* [31], ki preko objekta *Invert* nudi API klic *ApplyInPlace(Bitmap b)*, ki vrne negativ slike.

2. Izračun histograma iz negativa slike (*neg_I*) - tudi tu smo si pomagali s knjižnico *Aforge .NET*, ki za te namene nudi objekt *ImageStatistics*.

3. Iskanje barve v histogramu, ki predstavlja ozadje - v histogramu poiščemo tisto barvo, katere količina je večja ali enaka pragu *ThrF*. Pri razvoju aplikacije smo imeli parameter nastavljen na 1000, bolj natančno pa smo ga določili pri kasnejšem testiranju.

4. Odštevanje minimuma od slike:

$$neg\_subtracted(i,j) = neg\_I(i,j) - minimum \tag{3.2}$$

Barvo, dobljeno v prejšnem koraku, odštejemo od vsakega slikovnega elementa slike *neg_I*.

5. Popravljanje dinamičnega obsega slike

$$neg\_streched(i,j) = \frac{neg\_subtracted(i,j)}{\max\limits_{i,j}(neg\_subtracted(i,j))} \cdot 255 \tag{3.3}$$

Za vsak slikovni element izračunamo novo vrednost z enačbo 3.3, s čimer povečamo dinamično razdaljo med grebeni in dolinami.

6. Izračun negativa slike *BS_image* iz slike *neg_streched*

$$BS\_image(i,j) = 255 - neg\_streched(i,j) \tag{3.4}$$

Končni rezultat si lahko ogledamo na sliki 3.3.

3.2.2 Segmentacija

Pri segmentaciji ne operiramo nad celotno sliko naenkrat, temveč se izvajajo operacije nad bloki velikosti W × W. Ko s trenutnim blokom zaključimo, se prestavimo na sosednji blok. Za poenostavitev operacij obrežemo sliko tako, da sta višina ter širina slike deljivi z vrednostjo W. S tem sicer izgubimo *0,5 × W × (širina + višina - W)* informacije, kar je v povprečju manj kot 3% površine, vendar to ne vpliva na rezultat nadaljnega procesiranja.

V nadaljevanju postopamo na naslednji način:

Slika 3.3: Rezultati odštevanja vrednosti ozadja od slike: (a) original (b) negativ (c) ozadje odšteto (d) popravljen dinamičen obseg slike

1. Trenutni blok označimo z 1.

2. Za blok B izračunamo srednjo vrednost (angl. *mean*):

$$mean(B) = \frac{1}{W^2} \sum_{i=0}^{W-1} \sum_{j=0}^{W-1} B(i,j), \tag{3.5}$$

   Če je večja od praga *ThrM*, označimo blok z 0 in gremo na korak 6.

3. Za blok izračunamo varianco (angl. *variance*):

$$variance(B) = \frac{1}{W^2} \sum_{i=0}^{W-1} \sum_{j=0}^{W-1} (I(i,j) - mean(I))^2, \tag{3.6}$$

Če je manjša od praga *ThrV*, označimo blok z 0 in gremo na korak 6.

4. Za blok s središčem v (i,j) izračunamo gradient:

$$G(i,j) = \frac{1}{W^2} \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} |I(u+1, v+1) - I(u,v)|, \tag{3.7}$$

Gradient se v tem primeru računa le v diagonalni smeri in ne v smereh x in y posebej, saj je dovolj že informacija, ali prehod iz beline v sivino obstaja, ne pa tudi smer prehoda. Če je gradient manjši od praga *ThrG*, označimo blok z 0 in gremo na korak 6.

5. Za blok s središčem v (i,j) izračunamo območje interesa (angl. *Region of Interest – ROI*).

$$ROI(i,j) = \sqrt{\frac{1}{W^2} \cdot \frac{(V_x(i,j)^2 + V_y(i,j)^2)}{V_e(i,j)}}, \tag{3.8}$$

kjer velja

$$V_x(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} 2 \cdot \partial_x(u,v) \cdot \partial_y(u,v), \tag{3.9}$$

$$V_y(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} (\partial_x^2(u,v) - \partial_y^2(u,v)), \tag{3.10}$$

$$V_e(i,j) = \sum_{u=i-\frac{W}{2}}^{i+\frac{W}{2}} \sum_{v=j-\frac{W}{2}}^{j+\frac{W}{2}} (\partial_x^2(u,v) + \partial_y^2(u,v)) \tag{3.11}$$

Vrednost ROI poda stopnjo gotovosti, ki nam glede na prag pove, ali slikovni element pripada ozadju ali ne. ∂_x in ∂_y sta jakosti gradienta v x in y smeri. Za njun izračun se uporablja Sobelov operator s 3×3 okolico (slika 3.4) ter konvolucijski maski (slika 3.5) [16].

a_0	a_1	a_2
a_7	$[i,j]$	a_3
a_6	a_5	a_4

Slika 3.4: Označevanje sosednjih slikovnih elementov pri uporabi Sobelovega operatorja

-1	0	1
-2	0	2
-1	0	1

1	2	1
0	0	0
-1	-2	-1

Slika 3.5: Konvolucijski maski za izračun Sobelovega operatorja v x in y smeri

Kadar računamo gradient za točko I(i,j) (kjer je i $\in$ [1, M] in j $\in$ [1, N] pri velikosti slike M $\times$ N slikovnih elementov), lahko ∂_x in ∂_y dobimo z uporabo enačb 3.12 in 3.13, ki ju izpeljemo s pomočjo konvolucijskih mask (slika 3.5).

$$\partial_x = (a_2 + 2a_3 + a_4) - (a_0 + 2a_7 + a_6), \tag{3.12}$$

$$\partial_y = (a_0 + 2a_1 + a_2) - (a_6 + 2a_5 + a_4) \tag{3.13}$$

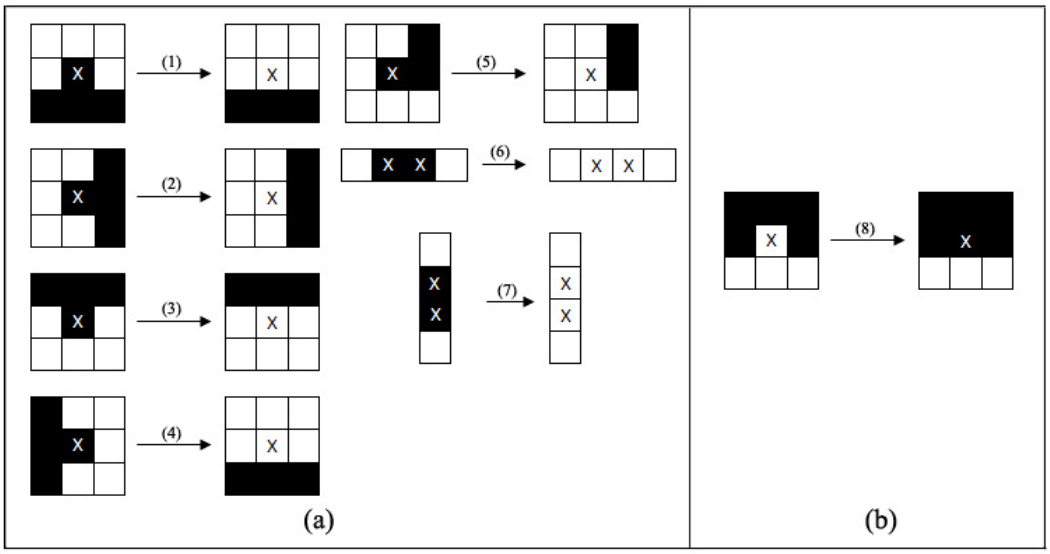
Ko izračunamo oceno ROI, jo primerjamo s pragom $ThrR$ in v primeru, da je vrednost ROI manjša, označimo blok z 0.

6. Po izvedbi opisanih korakov nad vsemi bloki, predstavljajo bloki, označeni z 0, ozadje, ostali pa področje prstnega odtisa. Vrednosti parametrov se nastavijo tako, da dajo najboljše rezultate pri izbrani zbirki prstnih odtisov. Ker je bilo pri nastavljanju parametrov kar nekaj poizkušanja, smo sam postopek nastavljanja in vrednosti parametrov predstavili skupaj z rezultati.

Naknadna obdelava

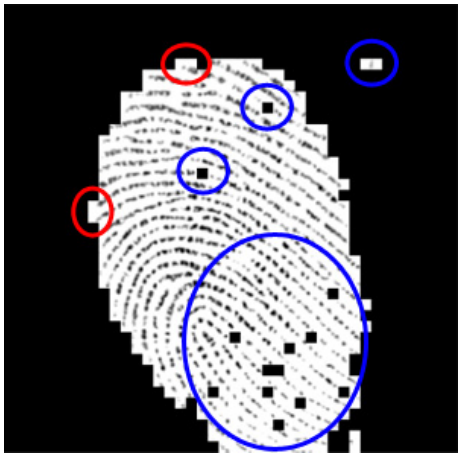
Pri segmentaciji lahko pride pri nekaterih blokih do napak. Blok, ki se nahaja na področju prstnega odtisa (ospredju), je lahko označen za ozadje, lahko pa je blok, ki sicer pripada ozadju, označen za področje prstnega odtisa. Take bloke je potrebno v naknadni obdelavi prepoznati in ustrezno popraviti.

Napačno označeni bloki so relativno lahko prepoznavni, saj se ponavadi nahajajo v okolici, kjer prevladuje le ena označba – ali večina blokov pripada ozadju ali pa prstnemu odtisu. Glede na okolico jih lahko s pomočjo hevrističnih pravil na sliki 3.6 ustrezno kvalificiramo. Pregledovanje blokov poteka od zgoraj navzdol, od leve proti desni. Za vsak blok se preveri, če ustreza kakšnemu pogoju in se ga temu primerno ustrezno popravi.



Slika 3.6: Hevristična pravila uporabljena pri naknadni obdelavi: (a) odstranjevanje blokov, označenih za ospredje, (b) odstranjevanje blokov, označenih za ozadje

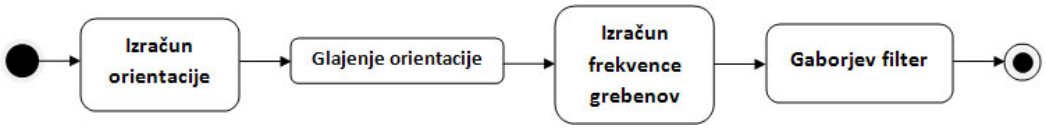
Najprej poskušamo odstraniti bloke, ki so napačno označeni za ospredje. Pri tem upoštevamo pogoje, ki so prikazani na sliki 3.6(a). Tu zahtevajo več pojasnil zadnja tri pravila. Pravilo 5 se uporabi takrat, kadar je trenutni blok označen kot ospredje ter ima manj kot tri sosednje bloke označene za ospredje, pravili 6 in 7 pa uporabimo na robu prstnega odtisa (slikq 3.7). Zatem sledi odstranjevanje blokov, ki so napačno označeni za ozadje. Pri tem se zanašamo na naslednje pravilo (slika 3.6): kadar je trenutni blok označen kot ozadje in ima več kot štiri sosednje bloke označene za ospredje, potem tudi trenutni blok označimo za ospredje. Sledi še ponovno preverjanje blokov z uporabo pravil 6 in 7, potem pa je področje prstnega odtisa dokončno določeno.



Slika 3.7: Primer uporabe hevrističnih pravil: z modrimi krogi je prikazana uporaba pravila 5, z rdečimi pa pravil 6 in 7

3.3 Izboljšava kvalitete slike prstnega odtisa

Pri prepoznavanju prstnih odtisov je zelo pomembna predhodna obdelava zajete slike prstnega odtisa, saj slaba kvaliteta slabo vpliva na postopke iskanja značilk, to pa v nadaljevanju privede do napačnih rezultatov. Zajeta slika vsebuje sivinske odtenke, kjer so grebeni le nekoliko temnejši od dolin. Poleg slabega kontrasta in različnih sivinskih odtenkov grebenov pa so na sliki prisotne tudi napake, ki so nastale zaradi različnih vzrokov pri zajemanju (šum, smeti, napake zajemanja, premajhen, prevelik ali neenakomeren pritisk prsta na čitalec, vlažnost itd.) ali mehanske poškodbe samega prsta (urez, praska itd). Zaradi posledic, ki jih te nepravilnosti povzročajo pri iskanju značilk, potrebujemo algoritem za izboljšanje kvalitete slike [10]. Koraki algoritma so prikazani na sliki 3.8.



Slika 3.8: Prikaz korakov izboljšave kvalitete slike prstnega odtisa

Vhod v algoritem je sivinska slika ter polje, ki je rezultat koraka segmentacije. Izhod iz algoritma pa je izboljšana slika prstnega odtisa, ki jo je potrebno v koraku binarizacije pretvoriti v binarno obliko.

3.3.1 Izračun orientacije prstnega odtisa

Lokalna orientacija grebenov (angl. *local ridge orientation*) v točki (x, y) je kot $\theta(x, y)$, pri katerem črte grebenov prstnega odtisa v povprečju prečkajo poljubno majhno okolico s središčem v (x, y) [15]. Izhod algoritma je orientacijska ali smerna slika **O**. To je matrika, katere elementi kodirajo lokalno orientacijo grebenov prstnega odtisa (slika 3.9).

Za izračun orientacije smo uporabili metodo gradientov. Tu velja opozoriti, da je izračunana orientacija pravokotna na usmeritev grebenov. Končno orientacijo dobimo tako, da prišetejmo vrednost $\pi/2$. Procesiranje poteka po sledečih korakih:

1. Sivinsko sliko razdelimo na bloke velikosti W × W slikovnih elementov. Pri tem bloke, ki se nahajajo na ozadju, izpustimo iz nadaljnega procesiranja. Velikost bloka W ni nujno enaka velikosti bloka W pri koraku segmentacije.

2. Za vsak slikovni element v bloku izračunamo gradient ∂_x ter ∂_y z uporabo Sobelovega operatorja (enačbi 3.12 in 3.13).

3. Izračunamo lokalno usmerjenost za vsako točko:

$$\theta(i, j) = \frac{1}{2} \arctan \frac{2C}{A - B} \tag{3.14}$$

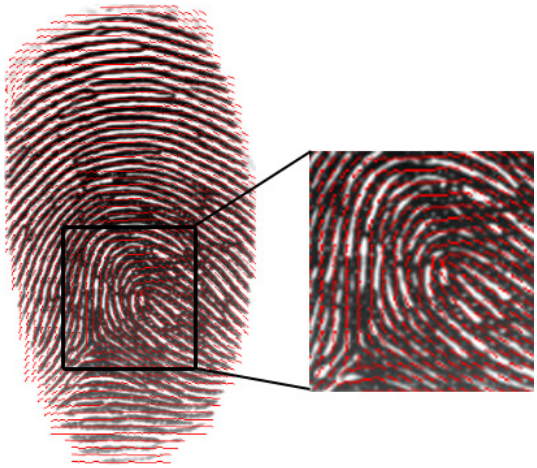
kjer je:

$$A = \sum_{(u,v) \in W \times W} \partial_x^2(u, v), \tag{3.15}$$

$$B = \sum_{(u,v) \in W \times W} \partial_y^2(u, v), \tag{3.16}$$

$$C = \sum_{(u,v) \in W \times W} \partial_x(u, v) \partial_y(u, v) \tag{3.17}$$

Na sliki 3.9 lahko vidimo rezultat izračuna orientacijske slike. Orientacijska slika pa lahko vsebuje tudi kakšne nezanesljive orientacije, ki so posledica ureznin, prask ali kakšnih drugih poškodb. Zato izračunu orientacije vedno sledi tudi glajenje orientacije prstnega odtisa.



Slika 3.9: Orientacijska slika

3.3.2 Glajenje orientacije prstnega odtisa

Dobljena množica lokalnih usmeritev je dostikrat neuporabna, saj vsebuje preveč šuma. Zato nad množico izvedemo glajenje na osnovi gostote enakih usmeritev. Pri tem je potrebno paziti na velikost okolice, na podlagi katere se izvede glajenje. Če je okolica premajhna, bo odprava velikega šuma neuspešna, obratno pa bodo imela področja z veliko šuma preveč vpliva na končno orientacijo. Še več, orientacija v zelo ukrivljenih delih prstnega odtisa bi se zaradi prevelike okolice popačila.

Zato smo se odločili, da uporabimo metodo, ki velikost okolice pametno prilagaja [19], s čimer se ohrani orientacija v delih z veliko ukrivljenostjo, obenem pa tudi dobro odpravlja šum. Da bi vedeli, kako dobro se lokalna orientacija ujema z orientacijo okolice, vpeljemo oceno **doslednost orientacije** (angl. *consistency orientation*). V okolici $\Omega(s)$ vsakega bloka, kjer s pomeni velikost okolice, definiramo doslednost orientacije kot:

$$Cons(s) = \frac{\sqrt{(\sum_{(i,j)\in\Omega(s)} \cos(2\theta(i,j)))^2}}{O(s)} \tag{3.18}$$

kjer je O(s) število orientacij θ(i, j) v Ω(s). Če vse orientacije kažejo v isti smeri, vrne *Cons(s)* največjo vrednost. V primeru, da so orientacije porazdeljene enakomerno v smereh $[-\frac{\pi}{2}, \frac{\pi}{2}]$, vrne *Cons(s)* najmanjšo vrednost 0.

Osnovna ideja izbrane metode je, da se velikost okolice glajenja določa glede na doslednost orientacije. Prstni odtis je sestavljen iz vzporednih grebenov in temu primerno se lokalna orientacija spreminja počasi, z izjemo v delih visoke

ukrivljenosti (okolica singularnih točk). Velike spremembe v orientaciji se nahajajo tudi v predelih, kjer je prisoten šum. Razlika med šumnimi regijami in regijami visoke ukrivljenosti je ta, da se doslednost orientacije povečuje z večanjem okolice glajenja pri šumnih regijah, medtem ko je pri regijah visoke ukrivljenosti vedno enaka oziroma indiferentna glede na velikost okolice. Glede na to ugotovitev se le šumne regije gladijo z večjo okolico, medtem ko se manjša okolica uporablja tam, kjer je šum majhen, ali pa ga ni, s čimer se ohranja lokalnost orientacije. Vpliv šuma se zmanjšuje tudi tako, da primerjamo doslednost orientacije večje in manjše okolice. V primeru, da ima večja okolica tudi večjo doslednost, potem je orientacija v večji okolici tudi bolj zanesljiva kot na manjši okolici. Zato za vsak blok izračunamo $Cons(s)$ in njegovo orientacijo na podlagi zunanjih $8 \cdot s$ blokov iz njegove $(2 \cdot s+1) \times (2 \cdot s+1)$ velike okolice. Koraki algoritma so:

1. Izračunaj enotski vektor z dvakratno orientacijo: $[\cos(2\theta(i,j)), \sin(2\theta(i,j))]$. Nastavi s = 1 in uporabi vrednosti enotskega vektorja pri izračunu $Cons(s)$ po enačbi 3.18, pri čemer je $\Omega(s)$ 8 zunanjih blokov iz okolice 3×3.

2. Povečaj s za 1. Izračunaj *Cons(s)* in $\Omega(s)$. $\Omega(s)$ je $8 \cdot s$ zunanjih blokov iz okolice $(2 \cdot s + 1) \times (2 \cdot s + 1)$.

3. $Cons(s)$ doseže vrednosti na intervalu $[0, 1]$. Če je $Cons(s)$ manjši od praga (0,5 v našem primeru) ali pa manjši od $Cons(s-1)$, potem pojdi na korak 2; na korak 2 se vračaš, dokler ne dosežeš maksimuma vrnitev (5 v našem primeru).

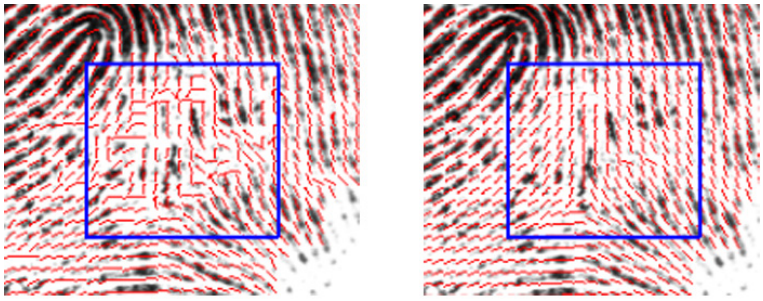
4. Če je s enak maksimumu, nastavi $\Omega(s)$ na okolico velikosti 3×3 blokov, sicer pojdi na korak 5.

5. Izračunaj zglajeno orientacijo po sledeči enačbi:

$$\overline{\theta}(i_c, j_c) = \frac{1}{2} \arctan\left(\frac{\sum_{(i,j)\in\Omega(s)} \sin(2\theta(i,j))}{\sum_{(i,j)\in\Omega(s)} \cos(2\theta(i,j))}\right) \tag{3.19}$$

   kjer sta i_c in j_c koordinati točke, ki predstavlja središče trenutno obravnavanega bloka.

Iz zgornjih korakov razberemo, da v primeru, kadar je $Cons(s)$ večji od praga in $Cons(s-1)$, potem je ocenjena orientacija na podlagi $\Omega(s)$ zanesljiva. Če pa je $Cons(s)$ za vse s ($s = 1, 2, 3, 4$) manjši od praga, potem je $\Omega(s)$ po vsej verjetnosti regija z veliko ukrivljenostjo, zato okolico $\Omega(s)$ zmanjšamo na najmanjšo velikost $\Omega(1)$. Na sliki 3.10 si lahko ogledamo rezultat glajenja orientacijske slike.



Slika 3.10: Orientacijska slika (a) pred in (b) po glajenju

3.3.3 Izračun lokalne frekvence grebenov

Lokalna frekvenca grebena (angl. *local ridge frequency*) $f(i,j)$ v točki $[i,j]$ je inverz števila grebenov na enoto dolžine vzdolž hipotetičnega segmenta s središčem v točki $[i,j]$, ki je pravokoten na lokalno orientacijo grebena $\theta(i,j)$ (slika 3.11). Če frekvence ocenimo na diskretnih razdaljah in uredimo v matriko, lahko definiramo frekvenčno sliko $\mathbf{F}$.

Za izračun lokalne frekvence grebenov potrebujemo na vhodu sivinsko sliko prstnega odtisa ter orientacijsko sliko, na izhodu pa dobimo frekvenčno sliko $\mathbf{F}$.

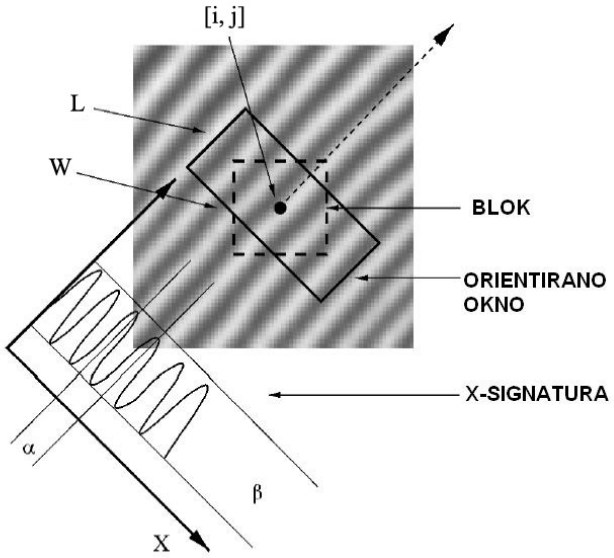
Lokalno frekvenco grebenov lahko ocenimo tako, da preštejemo povprečno število točk med dvema zaporednima vrhoma stopnje sivine v smeri normale na lokalno orientacijo grebena [10]. Frekvenco izračunamo po sledečih korakih:

1. Sliko razdelimo na bloke velikosti $W \times W$ slikovnih elementov. Sami smo izbrali velikost 7×7, saj je ponujala najboljše razmerje med hitrostjo in natančnostjo.

2. Za vsak blok s središčem v (i, j) definiramo orientirano okno velikosti $L \times W$ (pri nas 32×16) v grebenskem koordinatnem sistemu tako, da je y os okna poravnana z lokalno orientacijo grebena (slika 3.11).

3. Za vsak blok s središčem v (i, j) izračunamo *x-signaturo* stopnje sivine $X(0)$, $X(1)$, ... $X(L-1)$. *X-signatura* predstavlja sinusoidno krivuljo, kjer je α razdalja med dvema vrhovoma, β pa višina krivulje. Enačbe, potrebne za izračun *x-signature*, so sledeče:

$$X(k) = \frac{1}{W} \sum_{d=0}^{W-1} G(u,v), \quad k = 0, 1, ..., L-1, \tag{3.20}$$



Slika 3.11: Orientirano okno in *x-signatura*

$$u = i + (d - \frac{W}{2}) \cos O(i,j) + (k - \frac{L}{2}) \sin O(i,j), \tag{3.21}$$

$$v = j + (d - \frac{W}{2}) \sin O(i,j) + (\frac{L}{2} - k) \cos O(i,j) \tag{3.22}$$

kjer je $O(i,j)$ orientacija bloka s središčem v točki (i,j).

Če v oknu ni značilk ali singularnih točk, potem *x-signatura* tvori sinusoidno krivuljo, ki ima enako frekvenco kakor grebeni in doline v orientiranemu oknu. Zato lahko frekvenco le-teh dobimo iz *x-signature*. Če je $\tau(i,j)$ povprečno število slikovnih elementov med dvem vrhovoma v *x-signaturi*, potem je frekvenca $\Omega(i,j) = 1 \ / \ \tau(i,j)$. Če v *x-signaturi* ni zaporednih vrhov, potem frekvenca privzame vrednost -1, s čimer jo ločimo od ostalih vrednosti.

4. Za večino slik prstnih odtisov se frekvenca grebenov in dolin v okolici nahaja v nekem določenem območju. Za sliko z ločljivostjo 500 dpi je to območje definirano na intervalu $[\frac{1}{3}, \frac{1}{25}]$ [10]. Torej v primeru, da je izračunana frekvenca izven tega območja, se le-ta nastavi na -1. S tem vemo, da frekvence za ta blok ni možno pridobiti.

5. Za vse tiste bloke, katerih frekvenca je neznana, je potrebno nastaviti frekvenco glede na sosednje bloke. Pri tem se kot kandidate izbere tiste,

katerih frekvenca je višja od praga ϵ (v našem primeru 0,0001). Izmed teh se izbere tistega z najvišjo frekvenco.

6. Razdalje med grebeni se v okolici spreminjajo počasi, zato se za vsak blok izvede še povprečenje glede na okolico velikosti 5×5. Nova vrednost frekvence je enaka količniku vsote vseh frekvenc blokov iz izbrane okolice in števila le-teh blokov.

3.3.4 Filtriranje

Pri izboljšavi kvalitete slike smo se odločili za uporabo Gaborjevih filtrov [10], ki imajo frekvenčno in orientacijsko selektivne lastnosti. Ploščat, simetričen 2D Gaborjev filter v točki (x, y) ima obliko:

$$g(x, y, \theta, f) = e^{\{-\frac{1}{2}[\frac{x_\theta^2}{\sigma_x^2}+\frac{y_\theta^2}{\sigma_y^2}]\}} \cdot cos(2\pi f \cdot x_\theta), \tag{3.23}$$

Slika 3.12: Izboljšanje kvalitete prstnega odtisa z Gaborjevim filtrom

kjer je θ orientacija filtra, (x_θ, y_θ) so kordinate (x, y) po rotaciji za kot $(90° - \theta)$, f je frekvenca sinusoidne ploskve, σ_x in σ_y pa sta standardni deviaciji Gausove ovojnice vzdolž x in y osi. Da filtriramo sliko z Gaborjevim filtrom potrebujemo torej štiri parametre $(\theta, f, \sigma_x, \sigma_y)$. Frekvenco ter orientacijo torej že imamo,

potrebna je le še določitev σ_x in σ_y. Pri slednjih je potreben kompromis. Če želimo večjo odpornost na šum, izberemo višje vrednosti, kar lahko povzroči nastanek lažnih grebenov, nižje vrednosti pa tega ne povzročajo, so pa manj odporne na šum. S tema parametroma torej uravnavamo pasovno širino filtra. V tej diplomski nalogi smo za oba parametra uporabili vrednosti 4.0. Če smo izbrali manjšo vrednost, je algoritem veliko grebenov izpustil, če pa je bila vrednost prevelika, je pri nekaterih grebenih že prišlo do stika. Rezultat Gaborjevega filtra za vse točke slike prstnega odtisa zapišemo v novo sivinsko sliko, ki pa še ni popolnoma binarna. Rezultat filtriranje je viden na sliki 3.12.

3.4 Binarizacija

Pri binarizaciji izvedemo pretvorbo barvne globine slike prstnega odtisa iz 8 bitov na 1 bit na slikovni element. Pri našem pristopu smo uporabili filter *SiSThresholding* iz knjižnice *AForge .NET* [31]. Filter izračuna prag z uporabo metode enostavne statistike slike (angl. *Simple Image Statistics method*). Za vsak slikovni element se izvede naslednje:

1. Izračuna se dva gradienta

$$g_1 = |I(x+1, y) - I(x-1, y)| \text{ in } g_2 = |I(x, y+1) - I(x, y-1)|, \quad (3.24)$$

   kjer $I(x, y)$ pomeni vrednost slikovna elementa (x, y)

2. Izračuna se teža $T = \max\{g_1, g_2\}$

3. Skupna teža se posodobi: $weightTotal \mathrel{+}= T$

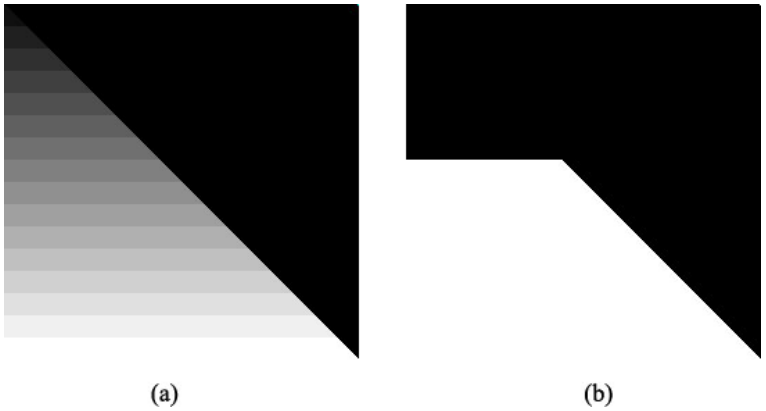
4. Skupna vrednost uteženih slikovnih elementov se posodobi: $total \mathrel{+}= T \cdot I(x, y)$

Prag se nato izračuna po enačbi:

$$prag = \frac{total}{weightTotal} \quad (3.25)$$

Na sliki 3.13 je podan primer binarizacije.

Uporaba knjižnice v programu zahteva dodajanje sklica (angl. *reference*) na *Aforge.Imaging.dll* datoteko. Nato se ustvari objekt *SISThreshold*, preko katerega se pokliče klic API *ApplyInPlace*, ki sprejme kot parameter 8 bitno sivinsko sliko.



Slika 3.13: Primer binarizacije: (a) original, (b) rezultat (izračunan prag znaša 127)

3.5 Tanjšanje grebenov

Tanjšanje grebenov je postopek, ki zmanjša debelino grebena na eno točko. Je zelo pomemben korak v sistemu AFIS, ker v veliki meri vpliva na natančnost algoritmov za iskanje podrobnosti, nepravilnosti pa negativno vplivajo na število najdenih podrobnosti. Zahteve za dober algoritem tanjšanja robov so:

1. Dobljeno ogrodje mora biti široko največ 1 točko brez kakršnihkoli nepravilnosti.

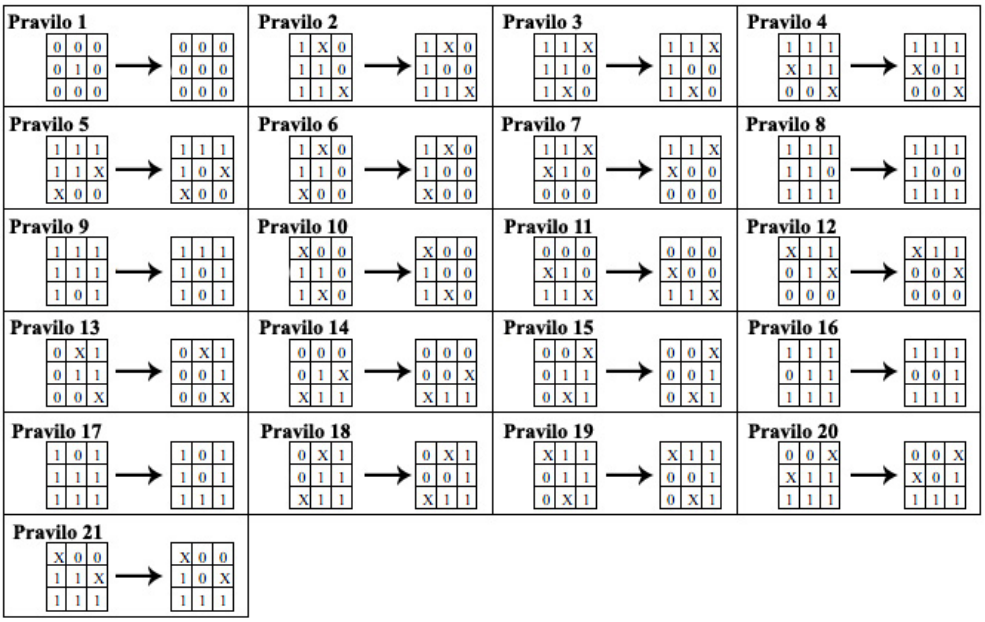
2. Vsak rob mora biti stanjšan proti svoji središčni točki.

3. Šum in samotne točke morajo biti odstranjene.

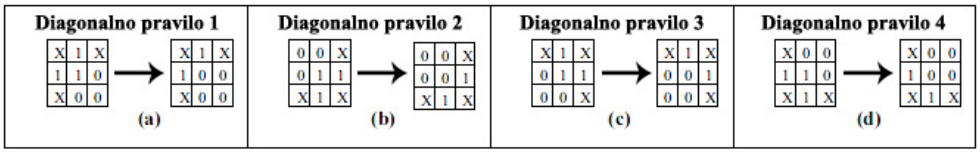
4. Po končanem procesu ni moč odstraniti nobene točke več.

Marsikomu se bo zdelo sporno odstranjevanje samotnih točk, toda v primeru prstnih odtisov se take točke obravnavajo kot šum in so zato odstranjene. Pravtako pa je veliko pravil pri tanjšanju črk in simbolov nepotrebnih pri prstnih odtisih.

Pri tanjšanju smo se odločili za metodo [20], ki ohranja topologijo ter ne povzroča nobenih nepravilnosti. Deluje tako, da prične odstranjevati slikovne elemente na zunanjih robovih grebenov dokler le-ti niso debeli zgolj 1 slikovni element. To dosežemo z uporabo 21 pravil na sliki 3.14, ki jih uporabimo v vsaki iteraciji nad vsakim slikovnim elementom. 21. pravilo služi odstranjevanju samotnih slikovnih elementov. Metoda vsebuje tudi 4 posebna pravila



Slika 3.14: 21 pravil za tanjšanje prstnih odtisov ali simbolov: 0 in 1 označujeta odsotnost oziroma prisotnost grebena, X pa označuje ali greben ali dolino



Slika 3.15: 4 pravila za tanjšanje diagonalnih črt

na sliki 3.15, ki se uporabljajo za tanjšanje diagonalnih črt. Pri tanjšanju slednjih se pogosto zgodi, da ostanejo široke več kot en slikovni element, zato jih je potrebno obravnavati posebej.

Pri ustvarjanju pravil za tanjšanje diagonalnih linij so bile upoštevane naslednje smernice:

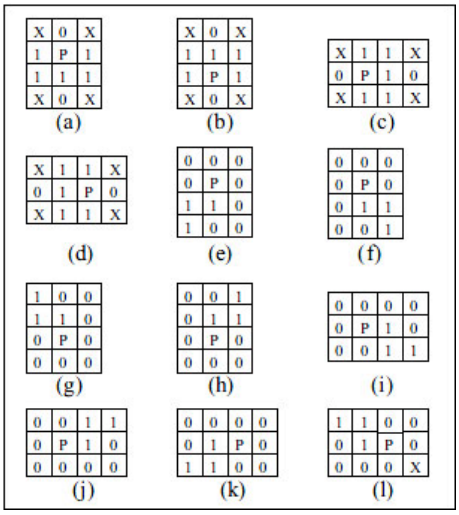
- Pri odstranjevanju kateregakoli dodatnega slikovnega elementa se ne smejo pojaviti nove nepravilnosti.
- Slikovni element, ki ga odstranjujemo, mora biti v središču 3×3 okolice.
- Odstranjevanje slikovnega elementa ne sme vplivati na razcep.

- Originalni kot diagonalne črte se ne sme spremeniti.

Pri tanjšanju obravnavamo poleg izbranega slikovnega elementa še vseh 8 njegovih sosedov. Število iteracij algoritma je odvisno od najdebelejšega grebena ali elementa na sliki.

3.5.1 Koraki tanjšanja

Vhod v algoritem je binarna slika prstnega odtisa, izhod pa binarna slika stanjšanega prstnega odtisa.



Slika 3.16: 12 pravil za pripravo na tanjšanje

Korak 1:

Izbrani korak ponavljamo dokler od ene do druge iteracije ni več sprememb. Za vsak slikovni element P na sliki ponavljaj:

1. Če ima **P** soseda v vertikalni smeri, pojdi na točko 2. Če ima P soseda v horizontalni smeri, pojdi na točko 3. Sicer pojdi na točko 6.

2. Če **P** ustreza pravilu (a) na sliki 3.16, potem prekini preverjanje za izbrani slikovni element, sicer preveri, če **P** ustreza pravilu (b). Če ustreza, zbriši **P** in prekini preverjanje za izbrani slikovni element, sicer pojdi na točko 4.

3. Če **P** ustreza pravilu (c) na sliki 3.16, potem prekini preverjanje za izbrani slikovni element, sicer preveri, če **P** ustreza pravilu (d). Če ustreza, zbriši **P** in prekini preverjanje za izbrani slikovni element, sicer pojdi na točko 5.

4. Če **P** ustreza praviloma (e) ali (f) na sliki 3.16, potem prekini preverjanje za izbrani slikovni element, sicer preveri, če **P** ustreza praviloma (g) ali (h). Če ustreza, prekini preverjanje za izbrani slikovni element, sicer pojdi na točko 6.

5. Če **P** ustreza praviloma (i) ali (j) na sliki 3.16, potem prekini preverjanje za izbrani slikovni element, sicer preveri, če **P** ustreza praviloma (k) ali (l). Če ustreza, prekini preverjanje za izbrani slikovni element, sicer pojdi na točko 6.

6. Uveljavi 21 pravil iz slike 3.14 in prekini preverjanje za izbrani slikovni element.

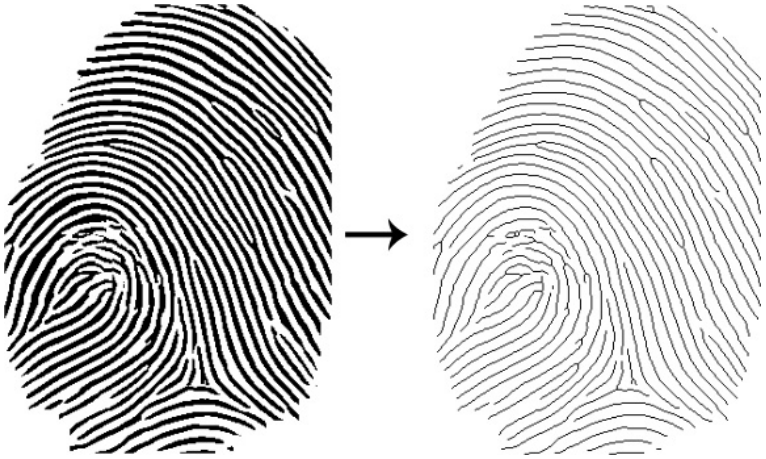
Korak 2:
Po uspešni izvedbi zgornjih korakov sledi uveljavljanje 4 diagonalnih pravil s slike 3.15 za vsak slikovni element. S tem se procedura tanjšanja zaključi.

Končen rezultat tanjšanja grebenov vidimo na sliki 3.17. Kot je razvidno, je algoritem uspešno opravil nalogo. Pravtako je algoritem tudi hiter, saj porabljen čas predstavlja dobre tri odstotke časa, ki ga porabi algoritem za verifikacijo prstnega odtisa.

3.6 Iskanje značilk

Naloga algoritma iskanja značilk je poiskati osnovne značilke: razcep (angl. *bifurcation*) in zaključek (angl. *ridge ending*). Kadar imamo na voljo skelet prstnega odtisa, je naloga razmeroma enostavna (slika 3.18). Pri iskanju zaključkov grebenov se opiramo na število križišč v trenutni točki (angl. *Crossing Number*) [25]. Število križišč $cn(\mathbf{p})$ točke $\mathbf{p}$ v binarni sliki je definirano kot polovica vsote razlik med pari sosednjih točk v okolici točke $\mathbf{p}$:

$$cn(\mathbf{p}) = \frac{1}{2}\sum_{i=1}^{8} |\mathrm{val}(\mathbf{p}_{\mathrm{i\ mod\ 8}}) - \mathrm{val}(\mathbf{p}_{\mathrm{i-1}})|, \tag{3.26}$$



Slika 3.17: Rezultat tanjšanja grebenov

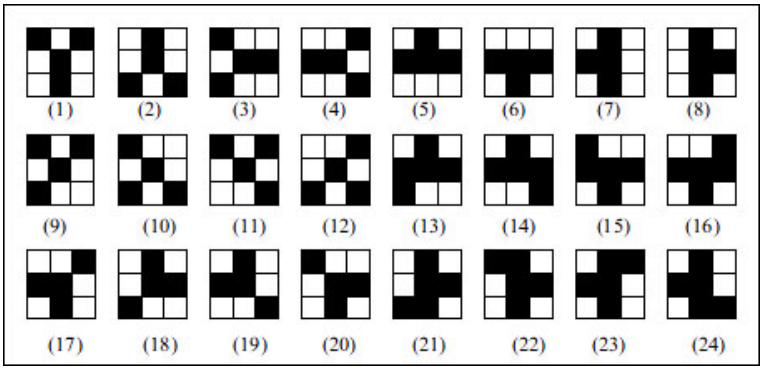
Slika 3.18: Koraki iskanja značilk

kjer so $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_7$ točke, ki pripadajo okolici točke $\mathbf{p}$, $\text{val}(\mathbf{p}) \in \{0, 1\}$ pa predstavlja vrednost točke. Za točko $\mathbf{p}$ z vrednostjo $\text{val}(\mathbf{p}) = 1$ velja, da je zaključek grebena, če je $cn(\mathbf{p}) = 1$.

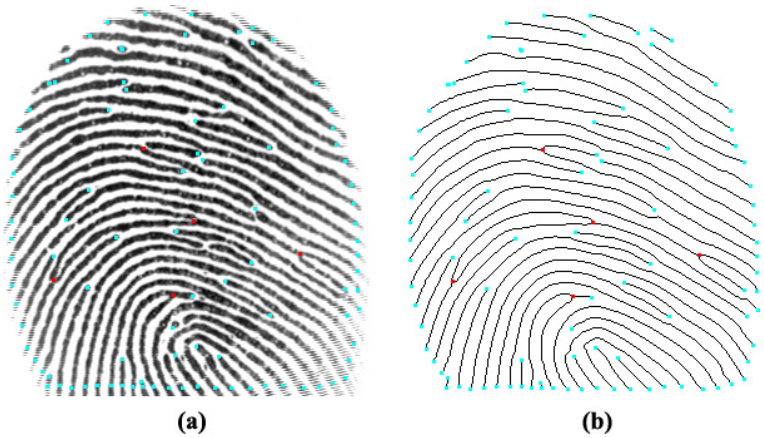
Pri iskanju razcepov grebenov pa si pomagamo z množico pravil. Za vsako točko se njena 3×3 okolica primerja s 24 možnimi maskami za racep na sliki 3.19. Če najdemo ujemanje, potem je ta kandidat dejansko razcep. Pogoji , da je slikovni element razcep, so sledeči [20]:

- V okolici slikovnega elementa se morajo vedno nahajati vsaj 3 slikovni elementi, ki predstavljajo greben in hkrati ne ležijo drug ob drugem (razen ob obravnavanem).
- Na maski je največje število dovoljenih slikovnih elementov, ki predstavljajo greben, enako 5.
- Središčni slikovni element je povezan z vsemi ostalimi na maski.

Na sliki 3.20 si lahko ogledamo rezultat iskanja značilk. Zaključki grebenov so označeni z modro barvo, razcepi pa z rdečo. Kot vidimo, je algoritem našel mnogo zaključkov grebenov na samem robu prstnega odtisa, ki bi jih želeli izpustiti. Prav tako se nekaj razcepov nahaja preveč blizu skupaj, kar je očitna napaka. Jasno je torej, da je potrebno dobljene značilke še naknadno obdelati.



Slika 3.19: Maske za vse možne oblike razcepa

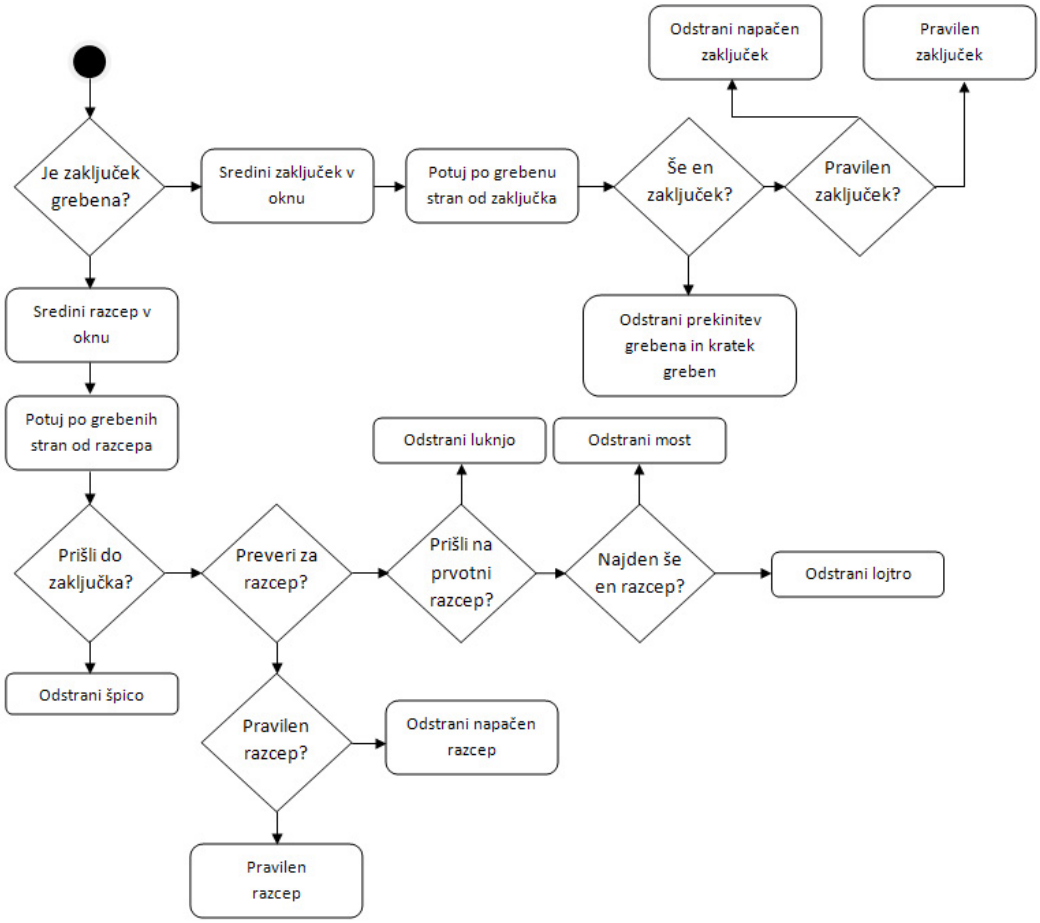


Slika 3.20: Najdene značilke, prikazane na (a) sivinski sliki in (b) sliki s skeletom prstnega odtisa

3.6.1 Naknadna obdelava

Število najdenih značilk je zelo pomembno pri iskanju podobnosti med dvema prstnima odtisoma. Ker je v povprečju število pravilnih značilk v vzorcu prstnega odtisa relativno visoko, nam izpad nekaj pravilnih značilk ne povzroča večjih preglavic. Problem nastane, če smo poleg pravilnih našli še kopico nepravilnih značilk. Posledica je manjša možnost ujemanja dveh enakih prstnih odtisov, počasnejše procesiranje ter večje zahteve po pomnilniškem prostoru. Kljub temu, da smo kvaliteto slike prstnega odtisa izboljšali in da smo iskali značilke le na področju, kjer se nahaja prstni odtis, algoritem še vedno najde večje število nepravilnih značilk (slika 1.8), ki niso primerne za nadaljno obdelavo. Zato je potrebno določiti kriterije, ki nepravilne značilke odpravijo

in s tem povečajo natančnost primerjanja prstnih odtisov [1].



Slika 3.21: Diagram poteka naknadne obdelave

Na sliki 3.21 je viden diagram poteka naknadne obdelave. Pri obdelavi vsake značilke vzamemo v precep še njeno $W \times W$ okolico, pri čemer je $W = 2d + 1$, kjer je d lokalna razdalja med grebeni (število slikovnih elementov). Lokalna razdalja med grebeni pomeni povprečno razdaljo med grebeni na kateremkoli delu slike prstnega odtisa in je zaokrožena na pozitivno celo število.

3.6.2 Razveljavitev značilk

Prvi korak pri naknadni obdelavi je določitev in odstranitev nepravilnih značilk. Za vsak tip nepravilne značilke smo ubrali svoj način odstranitve le-te.

Odstranitev nepravilnih značilk, ki se pojavijo zaradi špic, mostov ter lestev:

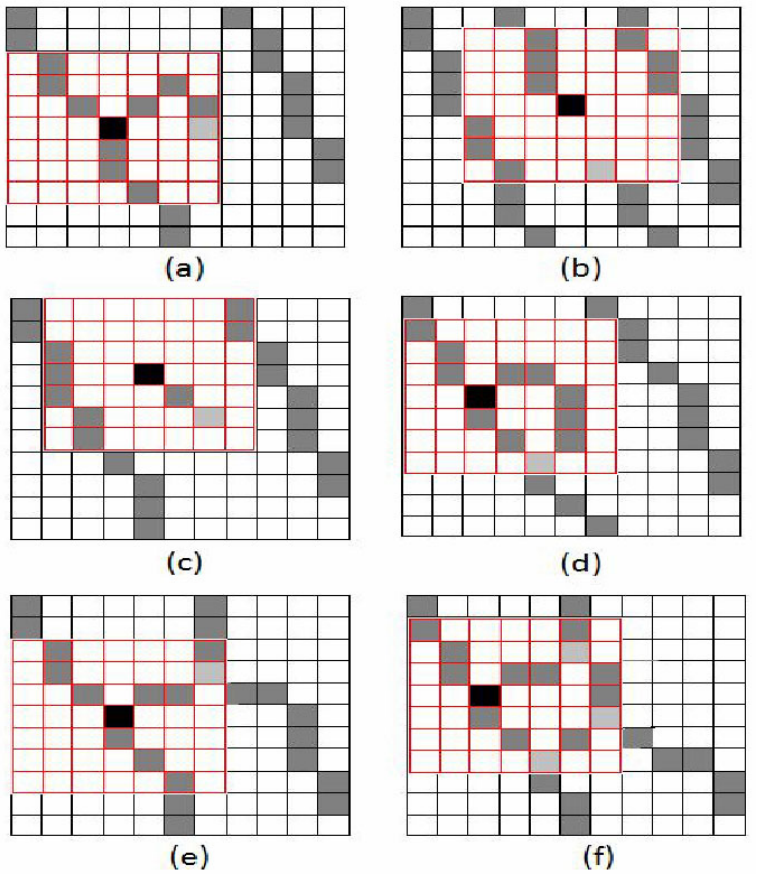
Za vsak razcep naredimo naslednje:

1. Določimo okno velikosti $W \times W$.
2. Razcep postavimo v središče okna.
3. Potujemo po vejah razcepa.
4. Če na kateri veji naletimo na zaključek grebena, potem odstranimo zaključek in razcep, saj sta posledica špice (slika 3.22(a)).
5. Če na kateri veji naletimo na rezcep grebena, potem odstranima oba razcepa, saj sta posledica mostu (slika 3.22(e)).
6. Če na kateri veji naletimo na dva razcepa grebena znotraj ali na robu okna, potem odstranimo vse tri razcepe, saj so posledica lestve (slika 3.22(f)).

Odstranitev nepravilnih značilk, ki se pojavijo zaradi lukenj/otokov:

Otoki so krožne poti, ki vsebujejo nepravilne razcepe. Za vsak razcep naredimo naslednje:

1. Določimo okno velikosti $W \times W$.
2. Razcep postavimo v središče okna.
3. Potujemo po vejah razcepa.
4. Če naletimo na dveh različnih vejah na enak razcep znotraj ali na robu okna, potem odstranimo oba razcepa, saj sta posledica luknje/otoka (slika 3.22(d)).



Slika 3.22: Odstranitev nepravilnih značilk z uporabo okna: (a) špice, (b) prekinitev grebena, (c) kratki grebeni, (d) luknja/otok, (e) most, (f) lestev

Odstranitev nepravilnih značilk, ki se pojavijo zaradi prekinitev grebena ter kratkih grebenov:

Za vsak zaključek naredimo naslednje:

1. Določimo okno velikosti $W \times W$.

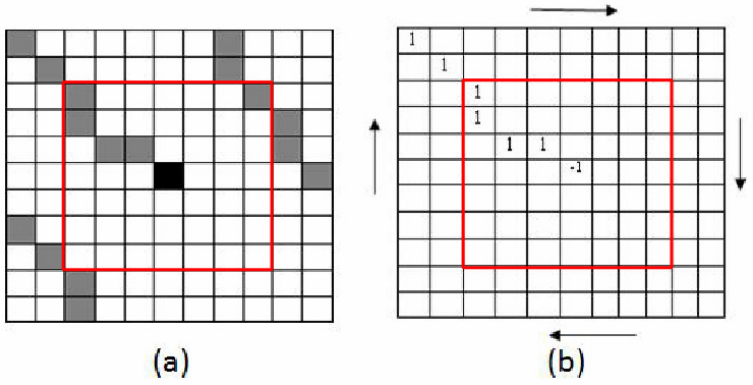
2. Zaključek postavimo v središče okna.

3. Če znotraj okna obstaja še kakšen zaključek, potem odstranimo oba, saj sta posledica prekinitve grebena (slika 3.22(b)).

4. Potujemo po veji stran od zaključka grebena.

5. Če naletimo na zaključek grebena, potem odstranimo oba zaključka, saj sta posledica kratkega grebena (slika 3.22(c)).

Na sliki 3.22 lahko vidimo proces razveljavitve značilk. Rdeče okno je že omenjena okolica, ki jo obravnavamo ob procesiranju vsake značilke ($W = 7$). Sivi slikovni elementi predstavljajo grebene, obravnavana značilka se nahaja v središču okna in je obarvana s črno barvo, nepravilne značilke pa so predstavljene s svetlo sivo barvo.



Slika 3.23: Ocenjevanje pravilnosti zaključka grebena: (a) zaključek grebena, (b) po določitvi vrednosti povezani veji

3.6.3 Potrjevanje značilk

Ko so nepravilne značilke odkrite in odpravljene, je potrebno ovrednotiti pravilnost preostalih značilk.

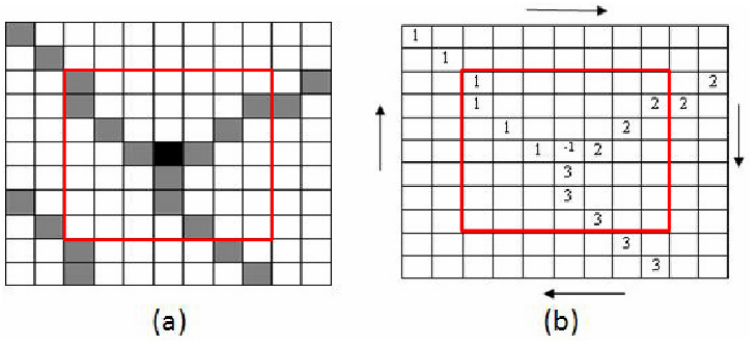
Ocena pravilnosti zaključka grebena

Za vsak zaključek naredimo naslednje:

- Določimo okno velikosti $W \times W$.
- Zaključek postavimo v središče okna (slika 3.23(a)).
- Vrednost središčnega slikovnega elementa, kjer se nahaja zaključek grebena, nastavimo na -1 (slika 3.23(b)).
- Potujemo po veji stran od zaključka grebena in vsakemu obiskanemu slikovnemu elementu nastavimo vrednost na 1 (slika 3.23(b)).

- Potujemo le po robu okna v smeri urinega kazalca.
- Med premikanjem po robu seštevamo število pojavitev prehodov iz 0 v 1.
- Če je seštevek enak 1, potem je izbrani zaključek grebena pravilen.

Slika 3.23 prikazuje, kako je ocenjena pravilnost zaključka grebena. Izbrani zaključek grebena je označen s črno barvo.



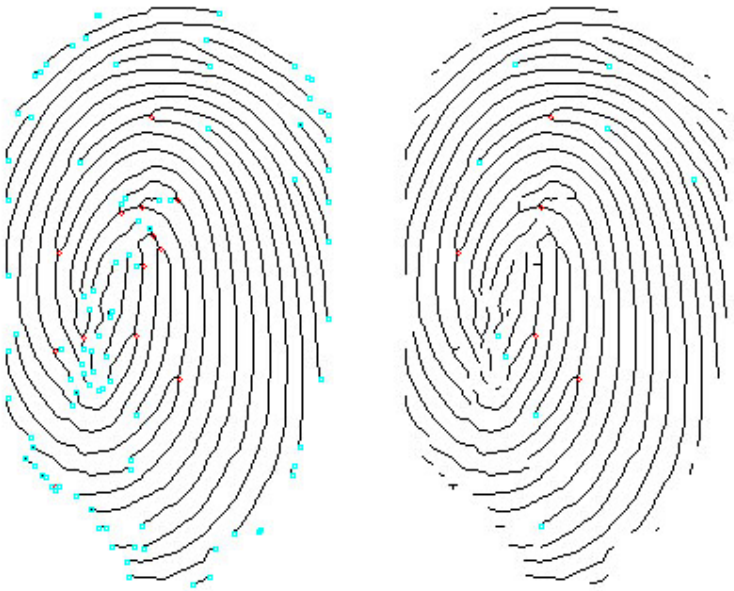
Slika 3.24: Ocenjevanje pravilnosti razcepa grebena, (a) razcep grebena, (b) po določitvi vrednosti povezanim vejam

Ocena pravilnosti razcepa grebena

Za vsak razcep naredimo naslednje:

- Določimo okno velikosti $W \times W$.
- Razcep postavimo v središče okna (slika 3.24(a)).
- Vrednost središčnega slikovnega elementa, kjer se nahaja razcep grebena, nastavimo na -1 (slika 3.24(b)).
- Potujemo po vejah stran od razcepa grebena in vsakemu obiskanemu slikovnemu elementu nastavimo določeno vrednost, prvemu 1, drugemu 2 in tretjemu 3 (slika 3.24(b)).
- Potujemo po robu okna v obratni smeri urinega kazalca.
- Med premikanjem po robu seštevamo število pojavitev prehodov iz 0 v 1, 0 v 2 ter 0 v 3.

- Če je seštevek enak 3, potem je izbrani razcep grebena pravilen, v nasprotnem primeru pa ga odstranimo.



Slika 3.25: Rezultat naknadne obdelave množice značilk: (a) pred naknadno obdelavo, (b) po naknadni obdelavi

Slika 3.24 prikazuje, kako je ocenjena pravilnost razcepa grebena. Izbrani razcep grebena je označen s črno barvo.

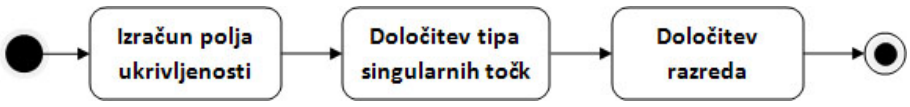
Ko dobimo končno množico značilk, je potrebno vsem značilkam določiti še orientacijo. Zaradi enostavnosti smo se odločili, da za orientacijo značilke vzamemo kar orientacijo bloka, v katerem se značilka nahaja. Načeloma pride tu do manjših odstopanj, ki pa na rezultat nimajo večjega vpliva.

Na sliki 3.25 lahko vidimo ogromno zmanjšanje števila značilk, s čimer smo močno vplivali na hitrost primerjanja, bolj pomembno pa je, da smo izboljšali natančnost primerjanja.

3.7 Klasifikacija

Klasifikacija prstnih odtisov v našem primeru niti ne bi bila potrebna, saj bo našo aplikacijo uporabljalo nekaj 10 uporabnikov, zaradi česar odpade potreba

po indeksaciji v podatkovni zbirki. Ob verifikaciji se bo preverilo ujemanje z vsemi prstnimi odtisi v zbirki, kar se na času procesiranja zaradi majhnega števila prstnih odtisov ne bo poznalo. Vendar pa smo klasifikacijo vseeno izvedli, saj je danes nepogrešljiv del sistemov za verifikacijo z uporabo prstnega odtisa.



Slika 3.26: Koraki klasifikacije

Čeprav je bilo za klasifikacijo razvitih precejšnje število algoritmov, se za to nalogo uporablja le majhno število značilk. Skoraj vse metode temeljijo na eni ali več naslednjih značilkah: pretok grebenov, orientacijska slika, singularne točke in odzivi gaborjevih filtrov. V naši aplikaciji smo se odločili, da bomo določili razred prstnega odtisa s pomočjo singularnih točk (jedro, delta) (slika 3.26). Za klasificiranje smo uporabili pet najbolj uporabljenih razredov Galton-Henryevega sistema: lok, šotorast lok, leva in desna zanka ter spirala [13].

Položaj singularnih točk smo izračunali s pomočjo polja ukrivljenosti (angl. *curvature map*) [17], tip singularne točke pa smo določili z uporabo metode *Poincare Index*. Korak kot vhod sprejme orientacijsko polje, vrne pa tip prstnega odtisa ter množico singularnih točk.

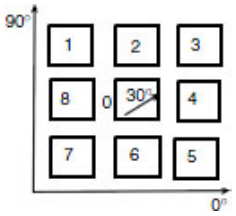
3.7.1 Izračun polja ukrivljenosti in določitev singularnih točk

Izračun polja ukrivljenosti smo izvedli nad bloki velikosti $W \times W$ – le-ta je enaka velikosti blokov pri izračunu orientacijske slike. Ukrivljenost bloka smo izračunali glede na ukrivljenost 8 sosednjih blokov v njegovi 3×3 okolici. Natančneje, ukrivljenost bloka je enaka polovici razlike ukrivljenosti blokov, na katere kaže smer obravnavanega bloka. Formula za izračun ukrivljenost B_k je enaka

$$B_k = \frac{|O(B_1) - O(B_2)|}{2}, \tag{3.27}$$

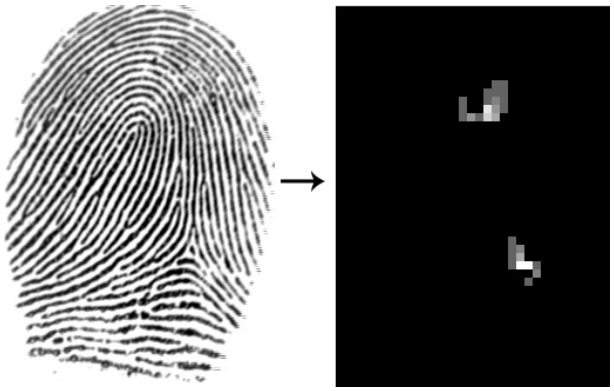
kjer sta B_1 in B_2 bloka, na katera kaže smer središčnega bloka. Če si ogledamo primer na sliki 3.27, so bloki P1 in P5, P2 in P6, P3 in P7, P4 in P8 razporejeni v smereh -45°, 90°, 45° in 0°. Smer središčnega bloka je 30°. Očitno je, da se med navedenimi štirimi smermi najbolj prilagaja smer 45°, saj

najmanj odstopa od orientacije bloka (+15°). Zato je ukrivljenost središčnega bloka enaka polovici razlike ukrivljenosti bloka P3 ter P7.



Slika 3.27: Primer izračuna ukrivljenost

Na sliki 3.28 lahko vidimo, da se polje ukrivljenosti izračuna zanesljivo in natančno. Tam, kjer je površina polja bele barve, se nahajajo največje ukrivljenosti. To so področja, kjer ležijo jedra in delte.



Slika 3.28: Polje ukrivljenosti

Ko je polje ukrivljenosti določeno, s spodnjim postopkom izračunamo položaj singularnih točk:

1. Določimo prag ukrivljenosti T_c ter prag razdalje T_d. V našem primeru smo T_c po nekaj preizkusih nastavili na 0,35, s čimer smo izločili točke z majhno ukrivljenostjo, hkrati pa obdržali še zadostno število točk z visoko ukrivljenostjo. T_d smo nastavili na 4, saj pričakujemo, da bodo singularne točke narazen vsaj 4 bloke velikosti 7×7.

2. Označimo s S_c množico singularnih točk, katerih ukrivljenost je večja od T_c.

3. Izberimo katerakoli dva bloka B_i in B_j iz množice S_c. Če je razdalja med njima manjša od T_d, se blok z manjšo ukrivljenostjo odstrani iz množice S_c.

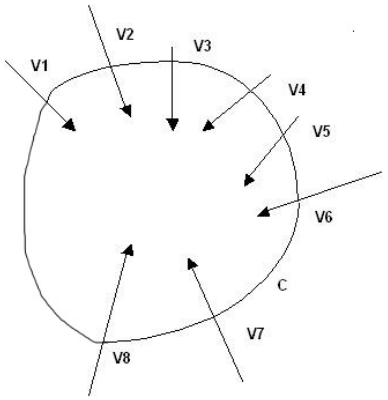
4. Pojdi na korak 3, dokler velikost množice S_c ni stalna.

5. Množica S_c vsebuje resnične singularne točke.

Velikost množice S_c je torej enaka številu singularnih točk, ki smo jih našli. Z uporabo pragov T_c in T_d smo zahtevali, da sta dve singularni točki z visoko ukrivljenostjo dovolj narazen. Za prstne odtise tipa lok je znano, da ne vsebujejo izrazitih singularnih točk. Temu se z našim postopkom izognemo tako, da za jedro vzamemo blok z najvišjo ukrivljenostjo. Glede na testiranja se metoda izkaže za učinkovito.

3.7.2 Določitev tipa singularne točke z uporabo Poincare Indeks metode

Poincare Indeks je enostavna in praktična metoda [23]. Imamo polje osmih vektorjev $\mathbf{G}$ in krivuljo C znotraj polja. Poincare indeks $P_{\mathbf{G},C}$ je definiran kot skupna rotacija vektorjev polja $\mathbf{G}$ vzdolž krivulje C (slika 3.29).



Slika 3.29: Poincare indeks: krivulja C, ki jo tvorijo vektorji polja $\mathbf{G}$

Za vsak blok dobljen v poglavju 3.7.1 izračunamo $P_{\mathbf{G},C}$ po naslednjem postopku:

- Krivulja C je zaprta pot, definirana kot urejeno zaporedje elementov orientacijske slike O tako, da je izbrani blok v notranjosti.

- $P_{\mathbf{G},C}$ izračunamo z algebraičnim seštevanjem razlik orientacij sosednjih elementov krivulje:

$$P_{\mathbf{G},C} = \frac{1}{2\pi} \sum_{k=0}^{N-1} \Delta(k) \tag{3.28}$$

kjer so $\Delta(k)$ razlike orientacijske slike O, smer uporabljenih elementov v sliki O pa je določena tako, da je absolutna vrednost kota med dvema zaporednima sosedoma manjša od 90°:

$$\Delta(k) = \begin{cases} \delta(k), & |\delta(k)| < \frac{\pi}{2} \\ \delta(k) + \pi, & \delta(k) \leq -\frac{\pi}{2} \\ \delta(k) - \pi, & \delta(k) \geq \frac{\pi}{2} \end{cases} \tag{3.29}$$

$$\delta(k) = \theta(x_{(k+1) \bmod \mathrm{N}}, y_{(k+1) \bmod \mathrm{N}}) - \theta(x_k, y_k) \tag{3.30}$$

Tip singularne točke je določen z naslednjimi vrednostmi:

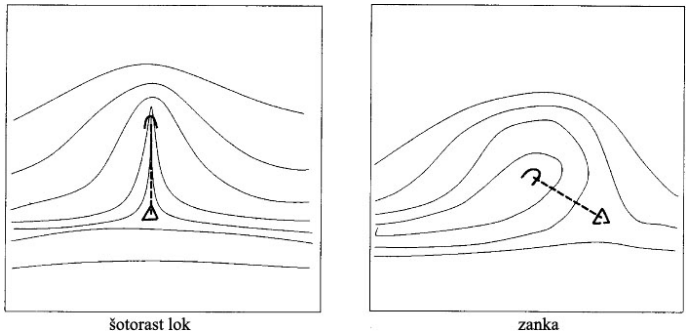
- $P_{\mathbf{G},C} = \frac{1}{2}$, blok vsebuje jedro.
- $P_{\mathbf{G},C} = -\frac{1}{2}$, blok vsebuje delto.
- Blok ne vsebuje singularne točke.

3.7.3 Določitev razreda

S pridobljenimi singularnimi točkami ter njihovimi tipi lahko določimo, v kateri razred spada prstni odtis. Klasifikacijo opravimo glede na število jeder (N_c) ter delt (N_d) [14]:

1. Če je $N_c = 2$, potem je prstni odtis tipa spirala.
2. Če je $N_c = 0$ in $N_d = 0$, potem je prstni odtis tipa lok.
3. Če je $N_c = 1$ in $N_d = 1$, potem je prstni odtis lahko tipa šotorast lok ali zanka. Potrebna je nadaljna analiza. Med jedrom in delto potegnemo ravno črto. Pri šotorastem loku je črta vzporedna z lokalnimi orientacijami, medtem ko pri zanki seka lokalne orientacije (slika 3.30). Naj bo λ kot, ki ga tvori daljica med jedrom in delto in η_1, η_2, ..., η_n koti lokalnih usmeritev vzdolž omenjene daljice. Če je povprečna vsota (formula 3.31)

$$\sum_{i=1}^{n} \sin(\eta_i - \lambda) \tag{3.31}$$



Slika 3.30: Ločevanje med šotorastim lokom ter zanko

manjša od pragu (v našem primeru smo prag nastavili na 0,3), potem je prstni odtis tipa šotorast lok, sicer je tipa zanka. Leve zanke ločimo od desnih na naslednji način (slika 3.31). Kadar sledimo grebenom stran od jedra, potem ostaja v levi zanki delta na desni, v desni zanki ostaja na levi. Povedano drugače, če označimo jedro s **C** in delto z **D** (slika 3.31) ter začnemo slediti grebenom iz **C** do robne točke **B**, potem lahko klasificiramo prstni odtis kot desno zanko, če je razlika

$$(B_x - C_x)(D_y - C_y) - (B_y - C_y)(D_x - C_x) \tag{3.32}$$

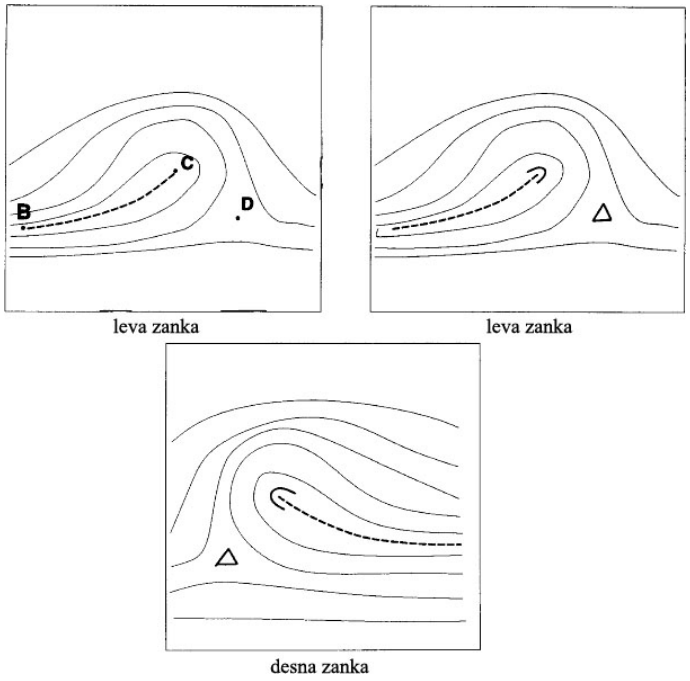
večja od nič, sicer pa kot levo zanko (indeksa x in y označujeta x in y koordinati točk).

4. Sicer tipa prstnega odtisa ni mogoče določiti.

3.8 Primerjanje

Primerjanje prstnih odtisov je zadnji korak pri sistemih za verifikacijo na podlagi prstnih odtisov, ki da odgovor na to, ali se vhodni prstni odtis ujema s katerim od prstnih odtisov, shranjenih v podatkovni zbirki. Je eden od ključnih členov v sistemih AFIS, kvalitetna izvedba pa močno vpliva na rezultat. Dandanes je znanih več načinov primerjanja prstnih odtisov, od katerih nekateri temeljijo na grafih, drugi na strukturi grebenov. Od slednjih se največkrat uporablja primerjanje na podlagi značilk, saj je v primerjavi z ostalimi načini zelo robustno, enostavno in učinkovito. Tipični koraki primerjanja na podlagi značilk so prikazani na sliki 3.32.

Pri primerjanju na podlagi značilk, se prstni odtisi ujemajo takrat, kadar obstaja potrebno število značilk, ki se ujemajo v tipu, lokaciji ter usmerjenosti.



Slika 3.31: Ločevanje med desno in levo zanko

Slika 3.32: Koraki primerjanja

V primeru, da bi pri zajemu prstnih odtisov zmeraj dobili skoraj enako (poravnano) sliko, potem bi bila primerjava enostavna. V praksi pa prihaja pri zajemu prstnih odtisov do rotacij in translacij, zato je potrebno značilke predstaviti v nekem koordinatnem sistemu, ki bi odpravil to slabost. Možnosti sta dve:

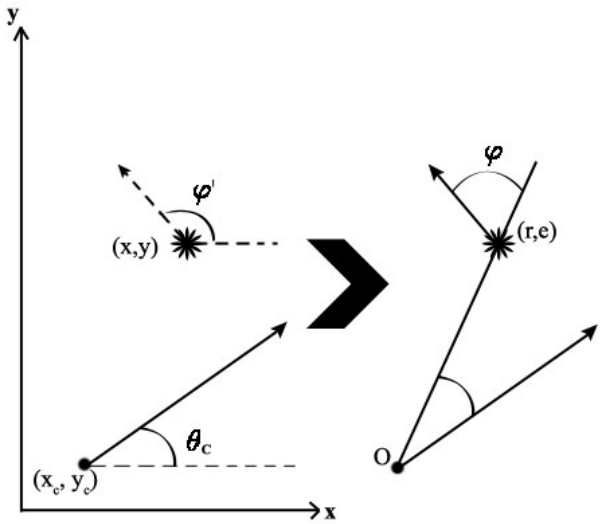
- Izpeljava značilk v kartezičnem prostoru
- Izpeljava značilk v polarnem prostoru

V dvodimenzionalnem kartezičnem prostoru bi lahko podali lokacijo značilk z vektorjema $\vec{\mathbf{X}}$ in $\vec{\mathbf{Y}}$, kjer bi $\vec{\mathbf{X}}$ podajal razdaljo med koordinatnim središčem in izbrano točko v x smeri, $\vec{\mathbf{Y}}$ pa v y smeri. V tem primeru bi morali določiti

tudi središče koordinatnega sistema. Ergonomsko oblikovani senzorji pri zajemu slike prstnega odtisa zagotavljajo določeno ponovljivost postavitve prsta v nekih tolerancah, zato lahko v tem primeru postavimo navidezno koordinatno središče tam, kjer se nahaja središče zajetega vzorca, torej na polovici x in y smeri. V primeru prstnih odtisov, ki vsebujejo spirale in zanke, bi lahko določili referenčno točko v jedru oziroma sredini dveh jeder. Pravtako bi v primeru, kadar bi poznali singularne točke, z uporabo ergonomsko oblikovanih senzorjev dokaj lahko določili tudi usmerjenost. Vseeno je odpornost zajema prstnih odtisov na rotacijo in translacijo težko zagotoviti, zato je primerjanje prstnih odtisov v karteziˇcnem prostoru zapleteno in zahtevno. Zato veliko aplikacij raje uporablja izpeljavo značilk v polarnem prostoru [24].

3.8.1 Izpeljava značilk v polarnem prostoru

Značilke so v dvodimenzionalnem polarnem prostoru podane s trojko $((\vec{r}, e), \varphi)$ (slika 3.33). $\vec{r}$ predstavlja razdaljo med središčem koordinatnega sistema ter značilko (enačba (3.33)), e pa kot med smerjo središča koordinatnega sistema in vektorjem $\vec{r}$ (enačba (3.34)). φ pa poda kot med vektorjem $\vec{r}$ in smerjo značilke (enačba (3.35)).



Slika 3.33: Pretvorba značilke v polarni sistem

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}, \tag{3.33}$$

$$e = \arctan(2(\frac{y - y_c}{x - x_c})) - \theta_c, \tag{3.34}$$

$$\varphi = \varphi' - e - \theta_c \tag{3.35}$$

Vse značilke v polarnih koordinatah lahko predstavimo z vektorjem značilk (angl. *feature vector*) (enačba (3.36)) P_i, kjer indeks i označuje trenutno izbrani prstni odtis. V P_i je vsaka značilka podana s četvorčkom ($\vec{r}$,e,s,φ). Tu smo poleg že prej opisanih značilnosti dodali še parameter s, ki poda tip značilke (razcep ali zaključek).

$$P_i = \{(\vec{r_{i0}}, e_{i0}, s_{i0}, \varphi_{i0})^T, ..., (\vec{r_{i(n-1)}}, e_{i(n-1)}, s_{i(n-1)}, \varphi_{i(n-1)})^T\}, \tag{3.36}$$

Da bi izbrane parametere lahko izračunali, moramo poznati še lokacijo referenčne točke ter njeno usmerjenost. Če smo v koraku klasifikacije uspešno pridobili singularne točke, potem lahko določimo lokacijo na naslednji način:

- Če je število jeder enako 1, potem vzamemo to jedro kot referenčno točko. To velja za prstne odtise tipa šotorast lok in oba tipa zank, pravtako pa smo v koraku klasifikacije določili jedro tudi za prstne odtise tipa lok.
- Če je število jeder enako 2, potem se referenčna točka nahaja na polovici daljice med jedroma. To velja za prstne odtise tipa spirala.

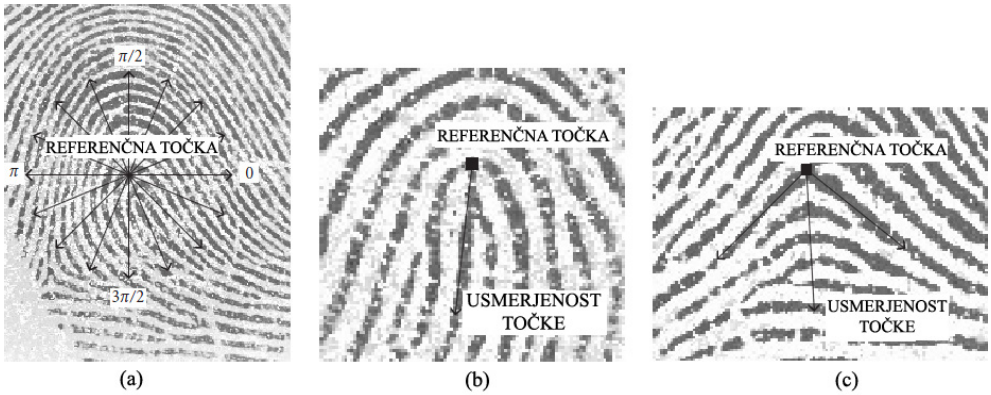
Določitev usmerjenosti referenčne točke

Določitev usmerjenosti referenčne točke je malo bolj zahtevna. V primeru, da je število jeder enako 2, je usmerjenost enaka usmerjenosti premice skozi obe jedri. Sicer pa je potrebno ubrati postopek opisan v nadaljevanju.

Usmerjenost referenčne točke more biti unikatna in konsistentna za vse tipe prstnih odtisov. Za izračun usmerjenosti smo izbrali postopek [19], ki definira 16 smeri izvirajočih v referenčni točki (slika 3.34(a)). Korak med smermi je enak $\frac{\pi}{8}$. Po analizi lokalnih usmeritev okoli vsake od 16 smeri, izbere za usmeritev referenčne točke tisto, ki se najbolj prilagaja lokalnim usmeritvam (slika 3.34(b)). Pri prstnih odtisih tipa lok, kjer referenčna točka ni pravo jedro, obstajata kot kandidatki dve smeri (slika 3.34(c)). Za rešitev se uporabi povprečna vrednost obeh smeri.

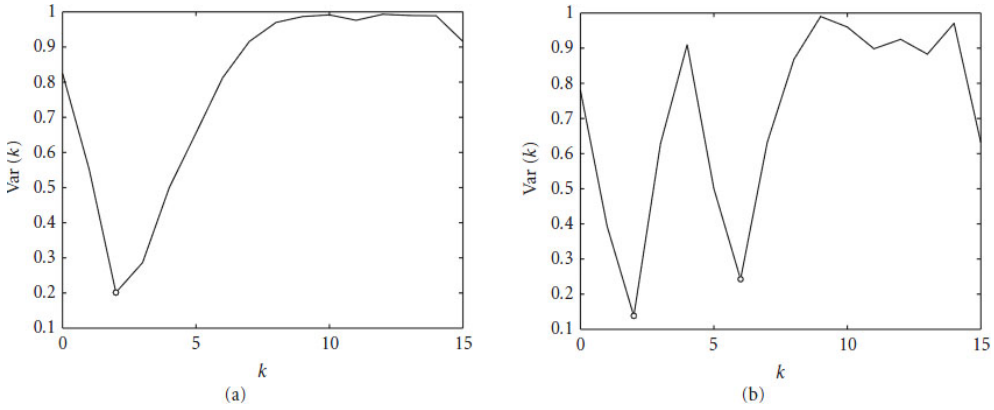
Z enačbo (3.37) za vsako od 16 smeri izračunamo, kako dobro se prilagaja lokalnim usmeritvam:

$$var(k) = \frac{1}{M} \sum_{(i,j)\in\Omega_k} |\sin(\theta(i,j) - \theta_k)|, \quad \theta_k = \frac{k\pi}{8}, \ \ k = 0, 1, ...15, \tag{3.37}$$



Slika 3.34: Izračun usmerjenost referenčne točke: (a) 16 smeri iz referenčne točke (b) usmerjenost jedra, (c) usmerjenost za prstne odtise tipa lok

kjer je Ω_k množica lokalnih usmeritev (označene z $\theta(i,j)$) vzdolž premice s smerjo θ_k, M pa število lokalnih usmeritev. $var(k)$ nam poda povprečno razliko med izbrano smerjo ter lokalnimi usmeritvami. Množico Ω_k si lahko predstavljamo kot pravokotnik lokalnih usmeritev, ki je z daljšim delom vzporeden s premico in je simetričen na premico. $var(k)$ zavzame vrednosti na intervalu $[0,1]$. Kadar je izbrana premica ortogonalna na vse lokalne usmeritve, zavzame vrednost 1, in 0, kadar je vzporedna. Zato bomo iskali tiste smeri, katerih vrednost $var(k)$ bo najmanjša. Za prstne odtise, ki imajo jedra, velja, da obstaja le en minimum (slika 3.35(a)), za prstni odtis tipa lok pa je značilno, da vsebuje $var(k)$ dva minimuma (slika 3.35(b)).



Slika 3.35: Graf $var(k)$ v odvisnosti od k: (a) en minimum – jedro, (b) dva minimuma – lok

Da bi učinkovito našli minimum $var(k)$, dinamično prilagajamo velikost množice Ω_k. Širino, ki je pravokotna na premico, nastavimo na fiksno velikost petih usmeritev. Dolžino Ω_k na začetku nastavimo na 4 (toliko blokov naj bi tudi znašala razdalja med dvema jedroma) in se prilagaja glede na vrednost $var(k)$. Postopek izračuna usmeritve poteka po naslednjih korakih:

1. Dolžina in širina množice Ω_k se nastavita na 4 in 5.

2. Izračunamo $var(k)$ za vseh 16 smeri z uporabo enačbe (3.37) in shranimo v $var(k_{min})$ minimum iz $var(k)$.

3. Izberemo tiste smeri θ_k, za katere velja $var(k) < var(k_{min}) + 0.1$. Če je takih smeri več kot 2 in z dolžino Ω_k nismo dosegli meja slike ali maksimuma (v našem primeru 15), povečamo dolžino in se vrnemo na korak 2.

4. Seštejemo usmeritve iz tiste množice Ω_k, za katero smo našli najmanjši $var(k)$. Če je takih množic več, vzamemo povprečje množic. Seštevek delimo s številom vseh usmeritev in tako smo dobili usmeritev referenčne točke.

Slika 3.36: Izračun usmerjenosti referenčne točke za različne tipe prstnih odtisov: (a) spirala, (b) lok, (c) leva zanka

3.8.2 Registracija

Pri registraciji vektorja značilk obdelanega prstnega odtisa ne primerjamo z ostalimi, temveč ga shranimo kot vzorec v podatkovno zbirko. V našem primeru smo za potrebe testiranje podatkovno zbirko nadomestili z ASCII datotekami. Vsak prstni odtis je torej imel še pripadajočo ASCII datoteko z enakim imenom ter končnico txt. Povprečna velikost takih datotek je bila reda do 2KB. Takšna rešitev je sicer enostavna za razvoj in testiranje sistema, za aplikativno rešitev pa je neprimerna. V takem primeru bi bilo bolje uporabiti podatkovno zbirko, ki bi skrbela za učinkovit zapis in hitro iskanje.

V vsako datoteko smo zapisali vektor značilk v naslednji obliki:

- polarni radij,
- polarni kot,
- tip značilke,
- smer značilke.

Zapise smo med seboj ločili z znakom #, pred zapisom pa smo jih uredili naraščajoče po velikosti polarnega kota. Pri branju zapisov iz datotek smo zmeraj prebrali štiri vrednosti skupaj, kar predstavlja štiri parametre ene značilke. Primer zapisa je viden na sliki 3.37.

```
83,1865#-3,4154#2#3,2902#128,6002#-3,2232#2#4,6442#169,7999#-3,2164#2#3,7265#154,
7288#-3,1906#2#4,2935#169,4521#-3,1559#3#4,5359#169,5936#-3,1135#3#4,2807#98,5089
#-3,1083#3#3,6707#107,5174#-2,9279#2#4,6607#65,3758#-2,809#2#4,6373#30,3645#-2,51
96#3#4,4667#67,5352#-2,4449#2#4,4933#86,2148#-2,3677#2#4,5241#34,8281#-2,0636#3#4
,1409#137,8876#-1,1372#3#2,1813#145,0414#-1,0292#2#1,9363#134,1827#0,0345#2#-0,06
73#137,2443#0,2192#3#-0,0674#123,065#0,2375#3#-0,2049#98,0051#0,535#3#-0,8728#52,
6972#1,5085#2#-1,7369#40,3609#1,7986#2#-2,167#
```

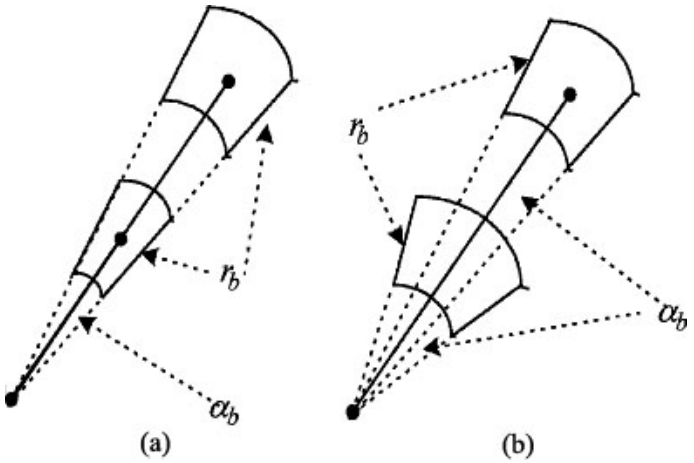
Slika 3.37: Zapis vektorja značilk v ASCII datoteki

3.8.3 Izvedba ujemanja

Ko imamo za prstni odtis, ki se preverja, zbrane vse značilke v vektorju značilk P_i, lahko izvedemo primerjanje z ostalimi prstnimi odtisi v podatkovni zbirki. Primerjanje med vhodnim vektorjem značilk $P_i = \{(r_{i0},e_{i0},s_{i0},\varphi_{i0})^T$, ..., $(r_{i(n-1)},e_{i(n-1)},s_{i(n-1)},\varphi_{i(n-1)})^T\}$ ter tistim iz podatkovne zbirke $W_j = \{(r_{j0},e_{j0},s_{j0},\varphi_{j0})^T$, ..., $(r_{j(n-1)},e_{j(n-1)},s_{j(n-1)},\varphi_{j(n-1)})^T\}$ poteka na naslednji način:

1. Za vsako značilko v vektorju značilk W_j preverimo ujemanje z vsako od značilk v vektorju značilk P_i. Preverjanje se izvaja nad naslednjimi lastnosti:
   - tip,
   - razdalja od referečne točke,
   - kot,
   - smer značilke.
2. Če je število enakih značilk večje od določene meje, potem smo našli enak prstni odtis in končamo proces.
3. Vzamemo naslednji vektor značilk W_j.

Primerjanje po tipu značilke je enostavno, pri ostalih primerjanjih pa je treba upoštevati tudi določeno toleranco. Toleranca pa ne sme biti prevelika, saj lahko pride do povečanja verjetnosti ujemanja neenakih prstnih odtisov. Zato smo empirično določili prag ujemanja značilk na podlagi smeri na $\pm\frac{\pi}{5}$, pri razdalji ter kotu pa smo ubrali metodo dinamične mejne škatle (angl. *Variable Bounding Box*)[9]. Središče škatle je značilka, ki se primerja, velikost škatle pa je določena dinamično glede na polarni radij značilke. To je ravno obratno kot pri fiksni mejni škatli, kjer je velikost škatle konstantna (slika 3.38).



Slika 3.38: Fiksna (a) ter dinamična (b) mejna škatla

Označimo velikost škatle s kotom α_b in radijem r_b, kot je to vidno na sliki 3.38. Z radijem določimi dolžino škatle, s kotom pa širino. Če se radij

značilke iz vzorca poveča, se bo to odražalo na večjemu r_b in manjšemu α_b, v nasprotnem primeru pa se bo α_b povečal in r_b zmanjšal. r_b in α_b izračunamo po naslednji enačbi:

$$r_b = \begin{cases} r_{bs}, & \frac{r}{a} < r_{bs} \\ \frac{r}{a}, & r_{bs} \leq \frac{r}{a} < r_{bl} \\ r_{bl}, & \frac{r}{a} \geq r_{bl} \end{cases}, \tag{3.38}$$

$$\alpha_b = \begin{cases} \alpha_{bs}, & \frac{b}{r^2} < \alpha_{bs} \\ \frac{b}{r^2}, & \alpha_{bs} \leq \frac{b}{r^2} < \alpha_{bl} \\ \alpha_{bl}, & \frac{b}{r^2} \geq \alpha_{bl} \end{cases} \tag{3.39}$$

kjer so α_{bs} in r_{bs} zgornje ter α_{bl} in r_{bl} spodnje meje α_b in r_b. a in b sta dve prednastavljeni konstanti, v našem primeru smo ju nastavili na 7 in 1000. α_{bs}, r_{bs}, α_{bl}, r_{bl}, a in b se spreminjajo empirično glede na velikost slike. r_{bs} in r_{bl} povečujemo, kadar je slika velika, saj lahko pride pri takih slikah zaradi elastičnosti prsta do velikih razlik v smeri radija. V našem primeru smo nastavili α_{bs} na 14 in α_{bl} na 21 pri zbirkah DB1, DB2 in DB4, pri zbirki DB3 pa smo ju nastavili na 7 in 14. Pri manjših slikah pa povečamo α_{bs} in α_{bl}, saj lahko že majhen šum povzroči veliko spremembo v kotu radija, medtem ko sama dolžina radija ostaja enaka. V tem primeru smo nastavili α_{bs} na 0,75 in α_{bl} na 1,5 pri velikih slikah iz množic DB1, DB2,DB4 in 1 ter 2 pri majhnih slikah iz množice DB3.

Z uporabo dinamične mejne škatle se spopademo z nelinearnimi deformacijami slike. Kadar je radij značilke majhen, lahko majhna deformacija povzroči veliko spremembo v kotu radija značilke, medtem ko na radij nima večjega vpliva. V tem primeru bi moral biti α_b večji in r_b manjši. Kadar pa je radij značilke velik, lahko majhna sprememba v kotu radija značilke povzroči veliko spremembo v lokaciji značilke. V tem primeru bi moral biti α_b manjši in r_b večji, saj se vse deformacije med referenčno točko ter značilko seštevajo, kar se pozna na radiju.

Na koncu je potrebna še odločitev. Recimo, da označujeta N_0 in M_0 število značilk v prvem in drugem prstnem odtisu, M_N pa število uspešnih ujemanj značilk. Kadar je $M_N/\sqrt{N_0 M_0}$ večji od praga β, potem smatramo obravnavana prstna odtisa za enaka. V našem primeru smo prag nastavili na 0.3. Če prag povečamo, povečamo število napačnih zavrnitev (FNMR), hkrati pa zmanjšamo število nepravilno sprejetih (FMR). Če nivo zmanjšamo, se število napačnih zavrnitev zmanjša, poveča pa se število nepravilno sprejetih.

Poglavje 4

Rezultati

Sistem smo testirali po modulih, tako kakor je bil zgrajen. Zato so tudi rezultati podani za vsak modul posebej. Kot smo že omenili, smo uporabili prstne odtise pridobljene s tremi različnimi čitalci, zadnja množica pa je vsebovala računalniško generirane prstne odtise. Za nastavitev parametrov smo pri vsaki podatkovni zbirki uporabili učno množico 80 prstnih odtisov, rezultati pa so odraz testiranj na testni množici 800 prstnih odtisov (glej poglavje 1.5).

4.1 Segmentacija

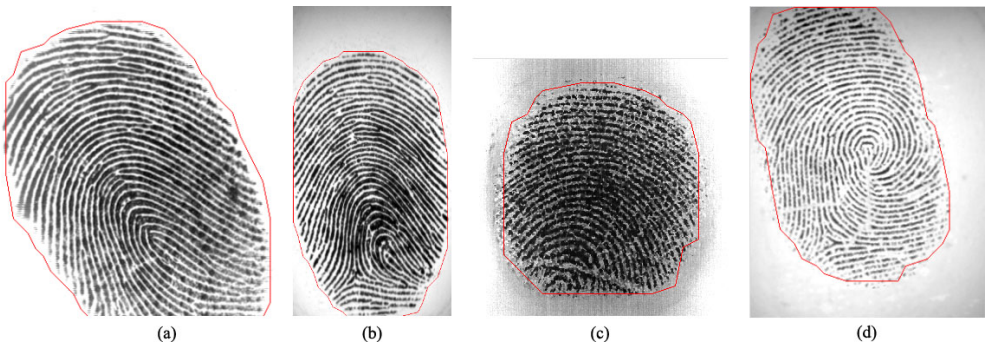
Pri testiranju segmentacije prstnega odtisa smo rezultate ocenili predvsem vizualno ter tudi podali število napak na posamezno množico. Naš algoritem pa smo tudi primerjali po času izvedbe z nekaterimi drugimi algoritmi. V obzir smo vzeli tudi čas procesiranja.

Zbirka	**ThrM**	**ThrV**	**ThrG**	**ThrR**	**ThrF**
DB1	252	0	0	0	1000
DB2	250	20	5	13	1000
DB3	250	450	5	30	800
DB4	240	150	0	10	1100

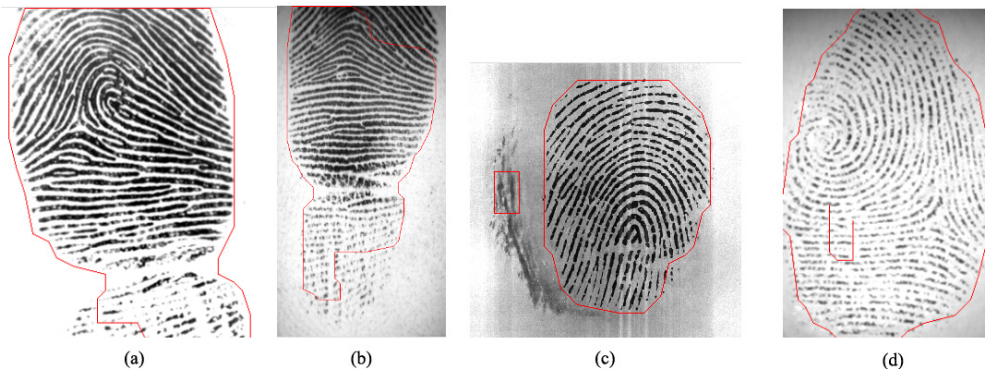
Tabela 4.1: Izbira parametrov za segmentacijo

Poglejmo si nastavljene parametre, prikazane v tabeli 4.1. Pri množici DB1 ležijo vsi prstni odtisi na beli podlagi, na kateri ni šuma. Tako smo lahko izluščili prstni odtis s slike, že samo z definiranjem srednje vrednosti, določitev

ostalih parametrov ni bila potrebna. Pri ostalih treh množicah pa je bilo na sliki več kontrasta, kar smo kompenzirali s povečanjem stopnje variance. Prag ThrF smo pri množici DB3 morali pomanjšati zaradi manjše slike, obratno pa smo ga pri množici DB4 povečali, saj je bilo na sliki več ozadja. Segmentacijo smo izvedli nad bloki velikosti 9×9. Na sliki 4.1 so primeri segmentacije nad prstnimi odtisi v učni množici, na sliki 4.2 pa so prikazane napake, ki so se zgodilo pri testiranju na testni množici. Take napake bi lahko odpravili s povečanjem pragov ThrV, ThrG ali ThrR, vendar bi s tem lahko ogrozili segmentacijo pri drugih prstnih odtisih, kjer bi lahko del prstnega odtisa označili za ozadje. Število podobnih napak smo za vsako množico podali v tabeli 4.2.



Slika 4.1: Primeri segmentacije na učnih množicah: (a) DB1, (b) DB2, (c) DB3, (d) DB4



Slika 4.2: Primeri napak segmentacije na testnih množicah: (a) DB1, (b) DB2, (c) DB3, (d) DB4

Izkaže se, da se največ napak zgodi pri množicah DB2 ter DB3 in sicer na področju, kjer prihaja do velikih kontrastnih razlik. Pri množici DB2 smo

Zbirka	**Št. napak**	**Delež** [%]
DB1	18	2,3
DB2	72	9
DB3	55	6,8
DB4	15	1,85

Tabela 4.2: Število napak pri segmentaciji glede na testno množico.

opazili tudi, da so se napake zgodile zaradi ostankov prejšnih prstnih odtisov na sliki, pri množici DB3 pa je prihajalo do napak zaradi šuma na robu prstnega odtisa. Tu smo naleteli tudi na primere, kjer sta bili za področje prstnega odtisa označeni dve ločeni regiji.

Poglejmo si še hitrost izvedbe segmentacije (tabela 4.3) v primerjavi z dvema algoritmoma, ki uporabljata linearni klasifikator.

Porabljen čas	**DB1**	**DB2**	**DB3**	**DB4**
naš algoritem (s)	0,065	0,104	0,042	0,061
algoritem v [5] (s)	0,018	0,019	0,015	0,016
algoritem v [3] (s)	0,125	0,145	0,094	0,110

Tabela 4.3: Čas porabljen za segmentacijo

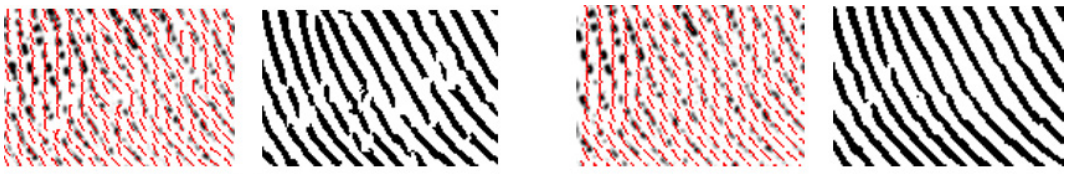
Izkaže se, da je naš pristop hiter, saj je zaostal le za algoritmom v [5]. Slednji je bil testiran na računalniku s procesorjem Pentium 4 2.4GHZ, medtem ko smo naš algoritem testirali na AMD Sempron 2GHZ. Zagotovo bi bila razlika še manjša, če bi uporabili enako strojno opremo.

Pokazali smo, da je naš pristop zanesljiv in hiter. Pri vsaki množici prstnih odtisov smo naleteli na nekaj napak, kar je bilo za pričakovati, saj so se slike prstnih odtisov zelo razlikovale. Vendar je bilo le nekaj napak takih, ki bi lahko znatno vplivale na končni rezultat. Tem bi se lahko izognili tako, da bi nastavili parametre še bolj restriktivno. Potrebno pa bi bilo tudi odpravljanje napak v primerih, kjer smo dobili za rezultat dve ločeni regiji prstnega odtisa. Ob predpostavki, da je manjša regija ponavadi šum, bi odstranitev slednje povečala natančnost segmentacije.

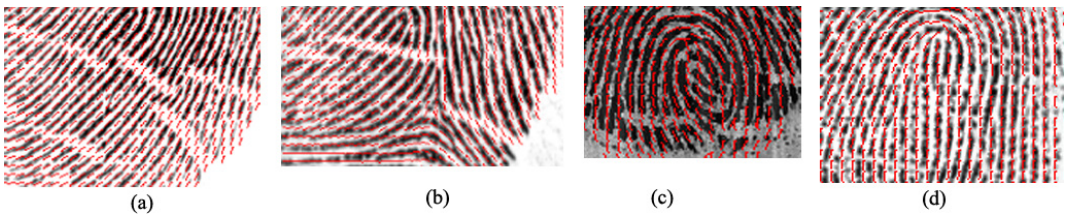
4.2 Izboljšava kvalitete slike prstnega odtisa

Pri koraku izboljšave kvalitete slike prstnega odtisa, smo najprej preverjali kvaliteto izvedbe izračuna orientacijskega polja. Kakor je navedeno v [4], pri ocenjevanju orientacijskega polja ni nekega objektivnega merila, zato je kvantitativno oceno težko podati. Zato smo se odločili, da bomo podali kakovostno oceno orientacijskega polja pri poljubnih prstnih odtisih. Poskusili bomo oceniti pravilnost orientacijskega polja za vsak razred prstnih odtisov ter za prstne odtise slabe kakovosti. Orientacijo smo računali z bloki velikosti 7×7, saj ponuja kompromis med hitrostjo in natančnostjo.

Poglejmo si najprej smiselnost uporabe glajenja orientacijskega polja na primeru prstnega odtisa. Kadar se glajenje ne uporablja, vsebuje orientacijsko polje odstopanja smeri na področjih, kjer je odtis prstnega odtisa slab (slika 4.3(a)). Če uporabimo tako orientacijsko polje za vhod v Gaborjev filter, pride na problematičnih področjih do nepravilnosti, kot so prekinjeni grebeni in pojav novih razcepov. Z uporabo glajenja pa se odstopanja v orientacijskem polju priredijo glede na okolico (slika 4.3(b)) in zato pri končni sliki ne pride do nepravilnosti.



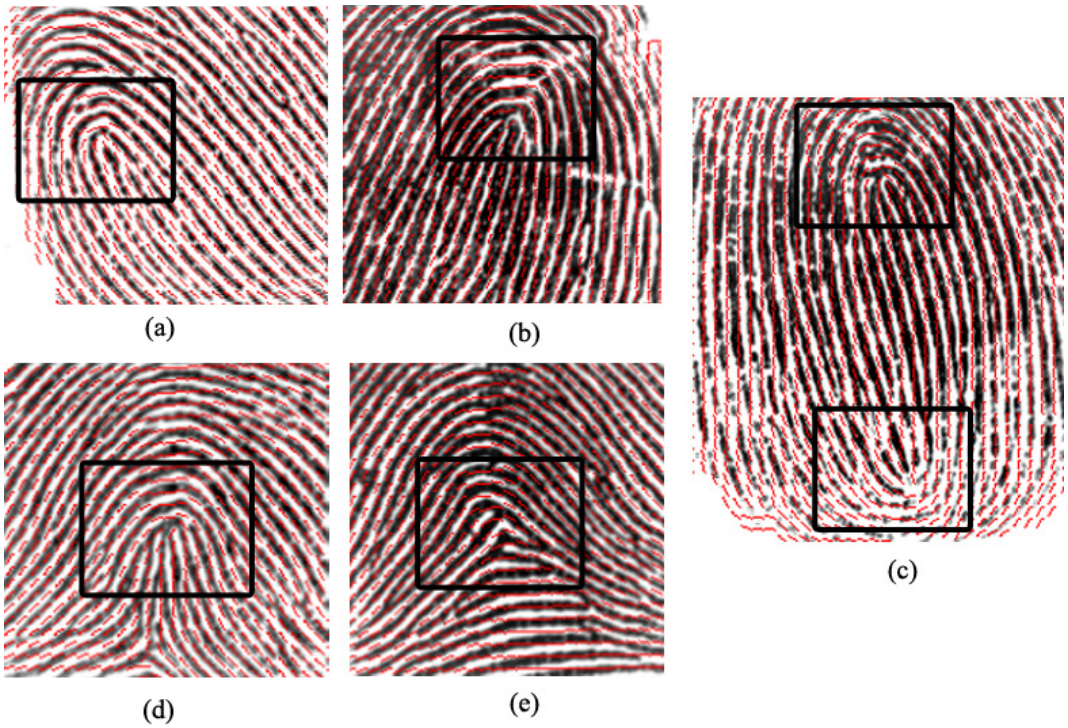
Slika 4.3: Primer orientacijskega polja (a) brez in (b) z glajenjem ter končnega rezultata po filtriranju



Slika 4.4: Primeri orientcijskega polja pri prstnih odtisih z urezninami: (a) DB1, (b) DB2, (c) DB3, (d) DB4

Testirali smo tudi, kako dobro se algoritem glajenja odreže pri prstnih odtisih z urezninami in poškodbami. Na sliki 4.4 so podani primeri poškodovanih

prstnih odtisov iz vseh štirih množic. Izkaže se, da algoritem zelo dobro prilagodi smer v šumnih regijah. Ker naj bi algoritem ohranjal smer v področjih velike ukrivljenosti, smo preverili tudi slednje.

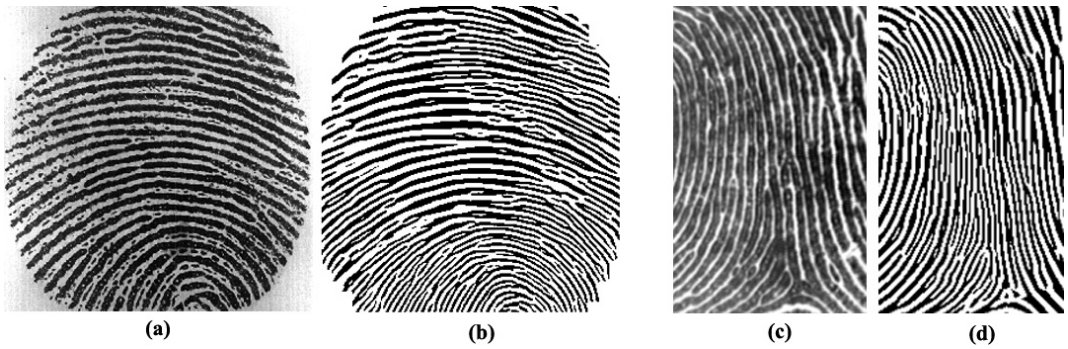


Slika 4.5: Primeri orientacijskih polj za (a) desna zanka, (b) leva zanka, (c) spirala, (d) šotorast lok, (e) lok

Na sliki 4.5 so prikazana orientacijska polja za različne tipe razredov prstnih odtisov. Pri vseh lahko opazimo, da se v okolici singularnih točk ukrivljenost področja ohranja in se ne prilagodi, kakor pri šumnih regijah. Ta lastnost pride predvsem prav pri iskanju jeder in delt, saj lokacija le-teh vpliva na natančnost primerjanja. Zaključimo lahko, da je izračun orientacijskega polja natančen in zanesljiv.

Nato smo preverili izračun lokalne frekvence grebenov. Velikost orientiranega okna smo nastavili na 32×16. Za to velikost smo se odločili zato, ker smo pri manjših velikostih že opazili anomalije v obliki novih grebenov. Če pa smo hoteli povečati velikost, pa je natančnost izračuna rastla počasneje kakor čas, potreben za izračun. Pri izračunu frekvence si nismo mogli pomagati z nobenim vizualnim izpisom frekvence, kjer bi s kombinacijo bele in črne barve

ponazorili visoko in nizko frekvenco grebenov. Zato smo si ogledali kar rezultat filtriranja z uporabo Gaborjevih filtrov. Opazili smo, da prihaja do napačnih izračunov frekvence pri vlažnih prstnih odtisih, saj je razmik med grebeni premajhen. Pri vseh učnih in testnih množicah je bilo takih prstnih odtisov manj kot 10%. Problemi pa so se pojavljali tudi tam, kjer so bile znojne pore na prstnih odtisih izrazito poudarjene, ali pa se je med grebeni nahajal šum. Slednje je bilo še najbolj opazno pri učni (in testni) množici DB3 (slika 4.6(a)). Posledica vseh omenjenih problemov je nastanek novih grebenov po uporabi Gaborjevih filtrov, kar vpliva na kasnejše odkritje nepravilnih značilk, to pa na sam rezultat primerjanja.



Slika 4.6: Primeri napak pri izračunu lokalne frekvence grebenov: (a) napake zaradi šuma med grebeni (DB3), (b) napake zaradi preveč vlažnega prsta (DB2)

Pri šumnih regijah in regijah, kjer je razmik med grebeni premajhen, izračunana lokalna frekvenca torej ne tvori pravilnega sinusoidnega signala. Pri našem algoritmu smo tako uporabili preprosto povprečje frekvence glede na sosednje bloke, kar zadostuje v večini primerov.

Na koncu smo testirali še filtriranje. Kot smo že v opisu postopka omenili, Gaborjevi filtri kot vhod sprejmejo štiri parametre, od katerih sta dva frekvenca ter orientacijsko polje. Od njiju je v veliki meri odvisen rezultat filtriranja. Na sliki 4.3(a) smo lahko videli, kako lahko napačna orientacija vpliva na izhod, na sliki 4.6 pa smo videli vpliv (napačno izračunane) frekvence na filtriranje. Da bi podali neko realno oceno, koliko filtriranje pripomore k boljšemu rezultatu, smo se odločili primerjati število pravilnih značilk, ki jih najdemo na originalni binarni sliki ter na izboljšani sliki prstnega odtisa (slika 4.7). Ker bi bilo testiranje nad celotno množico časovno potratno, smo se odločili izvesti primerjavo nad dvema prstnima odtisoma iz vsake testne množice. Prvi odtis je bil boljše, drugi pa slabše kvalitete.

Slika 4.7: Slika prstnega odtisa, (b) njegova binarna slika ter (c) njegova izboljšava

Cel postopek smo izvedli v naslednjih korakih:

1. Vhodno sliko smo:
   - V prvem primeru pretvorili v binarno,
   - V drugem primeru uporabili Gaborjeve filtre ter sliko nato pretvorili v binarno.
2. Grebene smo stanjšali na širino enega slikovnega elementa,
3. Poiskali smo vse značilke,
4. Primerjali smo število najdenih značilk s pravim številom značilk, obenem pa preverili, koliko je pravilnih in nepravilnih značilk.

V tabeli 4.4 so podani rezultati primerjav števila značilk iz originalne ter izboljšane slike. Po predvidevanjih se iskanje značilk na izboljšani sliki obnese veliko bolje, saj smo sposobni poiskati več pravilnih značilk, obenem pa jih tudi manj zgrešimo. V povprečju najdemo na binarni originalni sliki le 30,8% ter zgrešimo 69,2% pravilnih značilk, pri izboljšani sliki pa najdemo dobrih 67,4% ter zgrešimo 32,6%. S postopkom izboljšanja kvalitete slike prstnega odtisa torej najdemo okoli 30% več pravilnih značilk, kar je velik doprinos k celotni natančnosti sistema. Predvidevamo, da bi bili tudi rezultati nad celotno testno množico podobni.

Slika	Značilk	Binarna slika			Izboljšana slika		
		Najdene	Pravilne	Zgrešene	Najdene	Pravilne	Zgrešene
DB1_1	18	36	4	17	18	13	5
DB1_2	44	66	22	22	58	37	7
DB2_1	49	97	30	19	59	47	11
DB2_2	27	75	8	19	45	15	12
DB3_1	12	6	3	9	23	8	4
DB3_2	20	23	2	18	100	3	17
DB4_1	22	20	8	14	24	18	4
DB4_2	24	30	3	21	28	16	8

Tabela 4.4: Število značilk dobljenih z originalno in izboljšano sliko prstnega odtisa

Korak	Porabljen čas (ms)			
	DB1	DB2	DB3	DB4
izračun orientacije	79	112	46	56
glajenje orientacije	10	10	6	6
izračun frekvence	900	1097	543	524
filtriranje	1120	1356	714	701

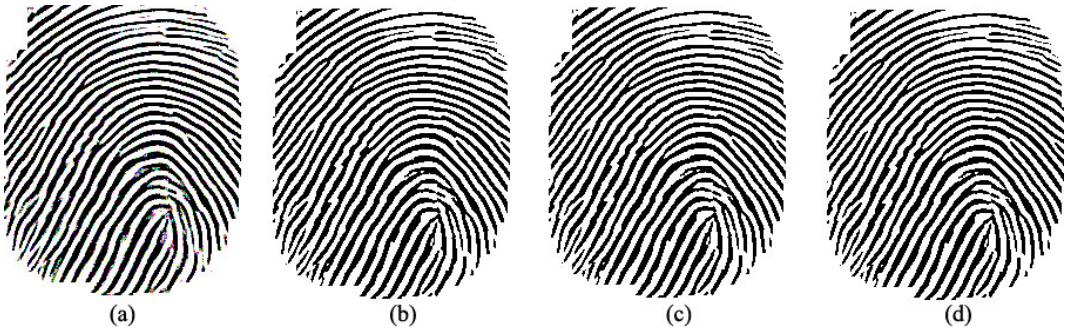
Tabela 4.5: Čas porabljen za izboljšanje kvalitete slike prstnega odtisa

Zato lahko zaključimo, da je postopek izboljšanja kvalitete nujen v našem sistemu, saj ciljamo k čim večji natančnosti. Hkrati pa je treba opozoriti, da je tudi najbolj procesorsko ter časovno obremenjujoč. S tem predstavlja ozko grlo sistema in je zato pospešitev postopka v prihodnosti ključnega pomena. Iz tabele 4.5 lahko razberemo, da sta izračun frekvence in filtriranje najbolj zahtevna postopka. Izračun frekvence smo poskušali pospešiti tudi z zmanjšanjem velikosti orientiranega okna, kar pa ni obrodilo sadov, saj smo s tem izgubili na natančnosti (pojavilo se je več fiktivnih grebenov).

4.3 Binarizacija

Korak izboljšanja kvalitete slike prstnega odtisa ponudi kot izhod sliko, ki je že skoraj v celoti binarna. Zato v koraku binarizacije nismo izdelali lastnega algoritma ali uporabljali zapletenejših postopkov, temveč smo uporabili

kar knjižnico *Aforge .NET*. Knjižnica ponuja več metod binarizacije, katerih lastnosti smo preverili na prstnemu odtisu na sliki 4.8.



Slika 4.8: Primeri binarizacij z uporabo knjižnice Aforge .NET: (a) vhodna slika, (b) SIS Thresholding, (c) Otsu Thresholding, (d) Iterative Thresholding

Kot smo omenili v poglavju 3.4, smo v našem sistemu uporabili način *SISThresholding*. Ker je pri vhodnih slikah potrebno pretvoriti le nekaj slikovnih elementov v binarno obliko, razlika v vseh treh metodah ne pride do izraza in bi pravtako lahko uporabili katerokoli od drugih dveh metod. Nobenih razlik nismo opazili tudi pri merjenju časa obdelave (tabela 4.6). Vse metode potrebujejo nad vsemi štirimi množicami prstnih odtisov povprečno manj kot 10 ms, kar je razumljivo glede na samo naravo problema.

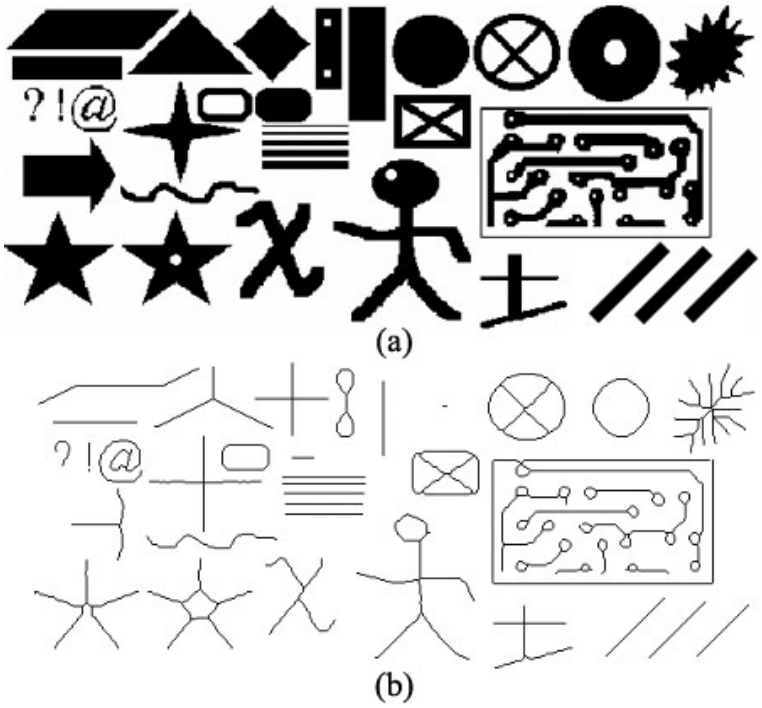
Pri binarizaciji nismo naleteli na težave, hkrati pa področje zaradi enostavnosti problema ne nudi veliko možnosti za izboljšave.

Korak	**Porabljen čas (ms)**			
	DB1	**DB2**	**DB3**	**DB4**
binarizacija	< 10	< 10	< 10	< 10

Tabela 4.6: Čas porabljen za binarizacijo

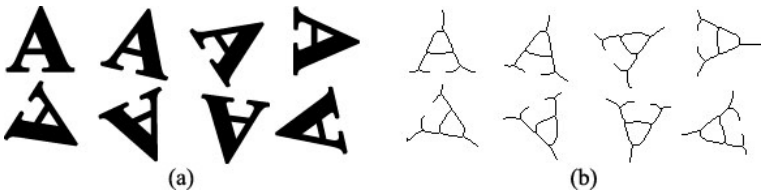
4.4 Tanjšanje grebenov

Pri koraku tanjšanja grebenov smo želeli izvedeti, ali algoritem stanjša elemente proti njihovemu središču ter s tem ohranja njihovo lego. Algoritem smo želeli najprej testirati nad sliko 4.9, ki vsebuje vzorce različnih oblik in debelin. Hkrati smo na sliko dodali v spodnji desni kot tudi diagonalne linije, da bi preverili tudi natančnost pravil za tanjšanje diagonal.



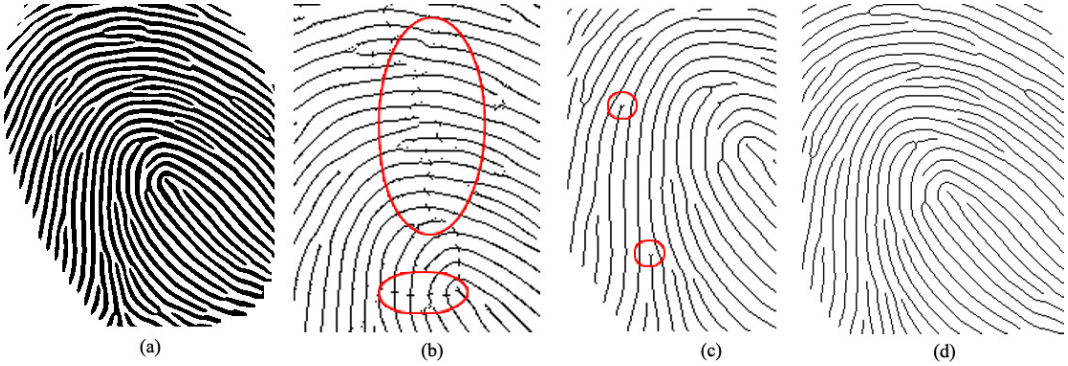
Slika 4.9: Primer tanjšanja različnih vzorcev: (a) vhodna slika, (b) rezultat

Rezultat pokaže, da algoritem pravilno stanjša elemente proti središču na širino enega slikovnega elementa. Poleg tega se uspešno odreže tudi pri diagonalnih črtah, kjer opazimo, da se kot črte ohranja. Opozoriti pa moramo tudi na 21 pravilo (slika 3.14). Z uporabo slednjega smo uspešno odpravili piko pod vprašajem. Nadalje smo testirali odpornost algoritma na rotacijo (slika 4.10).



Slika 4.10: Primer tanjšanja rotiranih elementov

Tudi pri temu testu se algoritem odreže odlično, saj vidimo, da kot elementa ne vpliva na končni rezultat. Na koncu pa smo testirali algoritem še nad binarno sliko prstnega odtisa. Za primerjavo smo vzeli dva algoritma iz knjižnice *Aforge .NET - SimpleSkelenization* in *FilterIterator*.



Slika 4.11: Primer tanjšanja prstnega odtisa: (a) vhodna slika, (b) Aforge .NET Simple Skelenization, (c) Aforge .NET FilterIterator, (d) algoritem, ki smo ga uporabili v naši aplikaciji

Na sliki 4.11 so prikazani rezultati vseh treh postopkov. Prvi postopek (slika 4.11(b)) se je odrezal najslabše, saj ni bil uspešen pri tanjšanju na širino enega slikovnega elementa, pojavile pa so se tudi špice pri vodoravnih ter navpičnih grebenih. Bolj uspešen je bil drugi postopek (slika 4.11(c)), kjer je rezultat zelo podoben rezultatu algoritma, ki smo ga uporabili mi (slika 4.11(d)) [20]. Vseeno so se pri nekaterih zaključkih grebenov pojavile nepravilnosti v obliki dveh špic. Zato lahko sklenemo, da je uporabljen algoritem boljši od ostalih dveh.

Algoritem	**Porabljen čas (ms)**			
	DB1	**DB2**	**DB3**	**DB4**
naš algoritem	166	193	76	98
Aforge .NET Simple Skelenization	< 10	< 10	< 10	< 10
Aforge .NET FilterIterator	538	798	358	436

Tabela 4.7: Čas porabljen za tanjšanje grebenov

V tabeli 4.7 si lahko ogledamo tudi potreben čas obdelave za vse tri postopke. Še najhitreje se odreže postopek *SimpleSkelenization*, vendar daje, kot smo omenili, tudi najslabše rezultate. Sledi algoritem, ki smo ga uporabili mi, najslabše pa se odreže postopek *FilterIterator*.

Spoznali smo, da je uporabljen postopek hiter in zelo uspešen pri tanjšanju. Vsi elementi so bili stanjšani na širino enega slikovnega elementa, obenem pa se je ohranila topologija (oblika) elementov. Zato lahko uporabljen postopek

uporabimo tako pri prstnih odtisih kot tudi pri drugih simbolih. Poleg tega je postopek odporen na rotacijo, odličen pa je tudi pri tanjšanju diagonal.

4.5 Iskanje značilk

Pri sistemih AFIS, ki temeljijo na primerjanju lege, tipa in smeri značilk, je pomembno število značilk ter njihova verodostojnost. Zato smo najprej preverili, koliko značilk poišče in zgreši naš algoritem. Za vsak prvi primerek prstnega odtisa iz testne množice smo ročno določili položaj značilk, kasneje pa ga primerjali z dobljenimi rezultati iz algoritma. Ugotovili smo, da z uporabo 24 pravil (slika 3.19) za odkrivanje razcepov grebenov uspešno najdemo vse razcepe, s štetjem križič pa najdemo tudi vse zaključke grebenov.

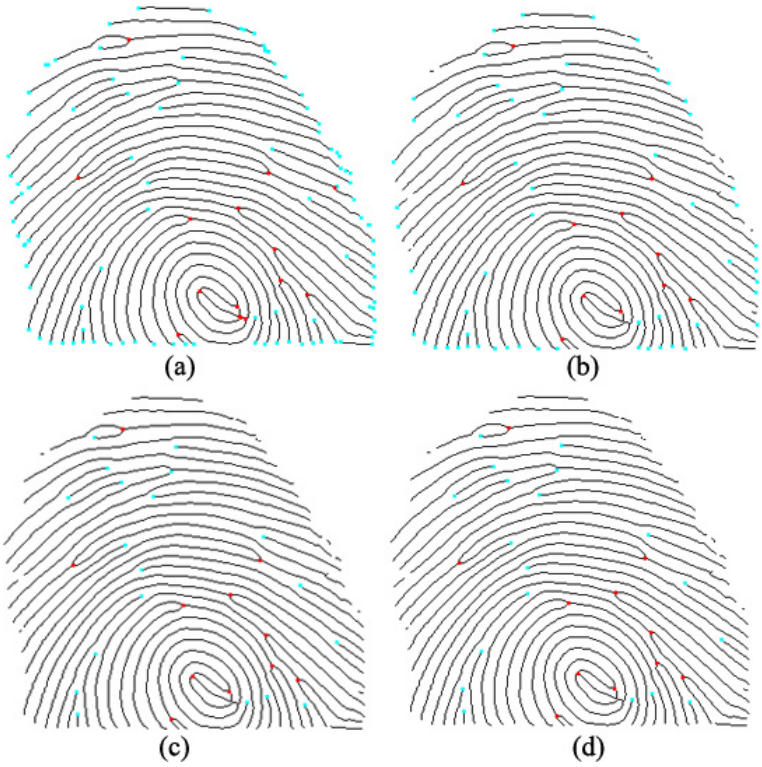
Vse značilke pa ne želimo v končni množici (npr. tiste na robu), zato smo želeli preveriti tudi, kako uspešno naš algoritem odstrani nepravilne značilke in kakšen odstotek natančnosti pridobimo s tem korakom. Naključno smo izbrali po dva prstna odtisa iz vsake testne množice in zapisali število značilk, ki jih dobimo na začetku in po naknadni obdelavi (tabela 4.8). Velikost okna, v katerem smo preverjali značilke, smo sprva nastavili na 11×11, vendar se je kasneje izkazalo, da že z velikostjo 7×7 uspešno odstranimo vse napačne značilke, zmanjšamo pa tudi možnost odstranitve pravih značilk.

Slika	**Na začetku**		**Po razveljavitvi značilk**		**Po odstranitvi robnih značilk**		**Po potrjevanju značilk**	
	Z	**R**	**Z**	**R**	**Z**	**R**	**Z**	**R**
DB1_1	97	37	74	24	25	22	25	21
DB1_2	115	31	68	22	20	18	20	18
DB2_1	130	66	99	37	42	37	42	37
DB2_2	126	14	93	10	43	10	43	9
DB3_1	183	35	31	9	16	4	16	2
DB3_2	99	31	50	12	14	10	14	9
DB4_1	96	12	65	9	14	5	14	4
DB4_2	126	12	54	8	21	8	21	8

Tabela 4.8: Število značilk pred in po naknadni obdelavi za 8 naključnih prstnih odtisov (**Z** = zaključek, **R** = razcep)

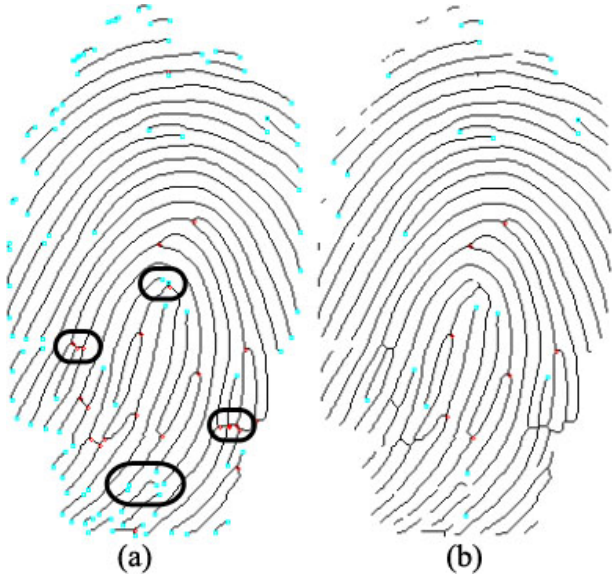
Že na začetku lahko opazimo, da se končno število značilk zelo razlikuje od prvotnega. To še posebej velja za zaključke, saj imamo na začetku označene

tudi vse zaključke, ki se nahajajo na robu prstnega odtisa in ki v bistvu sploh niso zaključki. Značilke odstranimo predvsem s prvima dvema korakoma, kjer odstranimo vse napačne strukture ter robne značilke. Pri vhodu v tretji korak je množica značilk več ali manj verodostojna, tako da se v zadnjem koraku velikost množice redko spremeni. Število značilk lahko spremljamo tudi na sliki 4.12.



Slika 4.12: Značilke (a) pred obdelavo (116), (b) po razveljavitvi (70), (c) po odstranitvi robnih značilk (30), (d) po potrjevanju značilk (30)

Na skeletu prstnega odtisa, kjer smo zaznali napačne strukture, smo preverili tudi, kako se algoritem spopade s temi problemi. Kot smo ugotovili (slika 4.13), je večina napačnih struktur pravilno odstranjenih. Problemi se pojavijo le tam, kjer so napačne značilke preveč oddaljene druga od druge. Takrat jih v oknu velikosti $W \times W$ ne moremo zajeti, zato jih ob preverjanju spregledamo. V primeru, da bi hoteli okno povečati, pa bi lahko prišlo do primera, ko bi odstranili pravilne značilke. Vseeno naj omenimo, da takih primerov ni veliko. Pojavijo se predvsem pri odtisih slabše kvalitete. Hkrati pa ostaja za primerjanje še dosti pravilnih značilk.



Slika 4.13: Odstranjevanje napačnih struktur

V tabeli 4.9 podajamo še kvantitativno oceno o odstranjenih napačnih strukturah za osem naključnih prstnih odtisov, ki smo jih uporabili v tabeli 4.8. Za vsak korak je podano še zmanjšanje zaključkov in razcepov v odstotkih. Iz tabele je razvidno, da se število zaključkov najbolj zmanjša ob preverjanju kratkih grebenov oziroma prekinitev grebenov. Le-to preverjanje odstrani tudi mnogo zaključkov na robu prstnega odtisa, kar vpliva na manjše število odstranjenih značilk pri koraku odstranitve robnih značilk. Če smo poskušali vrstni red teh dveh korakov zamenjati, se je število odstranjenih značilk zmanjšalo. Ko smo namreč odstranili robne značilke, so nekatere preostale značilke blizu roba ostale brez sosednjih značilk. Ko smo te značilke potem obravnavali, nismo v njihovi okolici našli nobene druge in smo jih zato smatrali za pravilne. Omenimo naj še število razcepov. To se najbolj zmanjša ob odstranitvi lestev, saj je bilo v tem primeru med prstnimi odtisi tudi nekaj takih, ki so bili slabše kvalitete. Pri takih odtisih so napačne strukture, kot so lestve, pogost pojav.

Pokazali smo, da lahko z našim algoritmom uspešno odkrijemo vse značilke, tako pravilne kot nepravilne. Z naknadno obdelavo nam je uspelo odstraniti večino napačnih struktur, ki so se pojavile zaradi vlažnih ali suhih prstov. S tem smo zelo vplivali na natačnost sistema, saj potrebujemo pri primerjavi le tiste značilke, ki so verodostojne. Sistem bi lahko še izboljšali, če bi pri preverjanju značilk uporabljali kombinacijo skeleta grebenov in dolin [25]. Pri primerjanju dveh soležečih zaključkov pa bi lahko kot dodatno možnost

Korak	Povprečno zaključkov	Povprečno razcepov	Zmanjšanje zaključkov (%)	Zmanjšanje razcepov (%)	Skupaj zmanjšanje (%)
iskanje značilk	121	30	-	-	-
špica	114	28	5,6	6,6	6,0
most	114	25	0,0	10,7	2,1
lestev	114	17	0,0	32	5,8
luknja / otok	114	16	0,0	5,9	0,8
kratek greben / prekinitev grebena	41	16	64,0	0,0	56,2
odstranitev robnih značilk	24	14	41,5	12,5	33,3
potrditev pravilnosti	24	13	0,0	7,1	2,6

Tabela 4.9: Delež napačnih struktur za osem naključnih prstnih odtisov

upoštevali tudi njuno smer.

Porabljen čas (ms)			
DB1	**DB2**	**DB3**	**DB4**
146	173	485	89

Tabela 4.10: Povprečen čas porabljen za iskanje značilk

Čas procesiranja algoritma je močno odvisen od kvalitete prstnega odtisa. Če je le-ta slaba, vsebuje prstni odtis mnogo nepravilnih grebenov in s tem tudi zaključkov in razcepov grebenov. Za iskanje značilk na prstnem odtisu slabe kvalitete je zato potrebno veliko več časa. Zato se naš algoritem, razumljivo, časovno najslabše izkaže pri prstnih odtisih iz testne množice DB3 (tabela 4.10), kjer smo imeli tudi največ težav z izboljšanjem kvalitete.

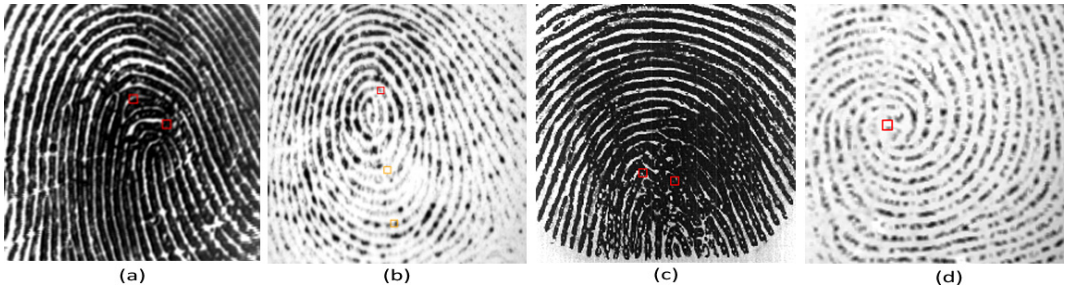
4.6 Klasifikacija

Pri klasifikaciji smo se najprej odločili preveriti, kako dobro se naš algoritem izkaže pri iskanju singularnih točk. Za vsako singularno točko smo preverili, ali je pravega tipa ter ali je njena lokacija pravilna. Pri slednji je bila uporabljena določena toleranca – če se je singularna točka nahajala za blok dolžine stran od ročno določene prave lokacije, smo jo še zmeraj šteli za pravilno. Omenjeni test smo zaradi kompleksnosti izvedli le na učnih množicah, saj smo morali za vsako sliko ročno določiti število singularnih točk in preveriti tiste, ki jih je našel naš algoritem.

Množica	Dejanske ST		Zaznane ST			
	Tip	**Število**	**Pravo jedro**	**Prava delta**	**Nepravo jedro**	**Neprava delta**
DB1	Jedro	96	94	-	6	0
	Delta	18	-	14	0	4
DB2	Jedro	104	101	-	2	0
	Delta	50	-	50	0	7
DB3	Jedro	88	81	-	11	1
	Delta	7	-	6	0	6
DB4	Jedro	88	82	-	3	0
	Delta	23	-	21	0	0

Tabela 4.11: Rezultat detekcije singularnih točk

Če pogledamo tabelo 4.11, lahko ocenimo, da je naš sistem dokaj natančen. V povprečju na 80 prstnih odtisov zgreši 4,5 jeder (4,7%) in 1,75 delt (9,7%) ter označi 4,75 napačnih jeder (4,9%) in 4,25 napačnih delt (23,6%). Do napačnih označitev pride predvsem v šumnih predelih, kjer je orientacija klub glajenju še vedno malo nesorazmerna glede na okolico. Tak primer se je zgodil pri množici DB3, kjer je algoritem napačno označil singularno točko tipa jedro za delto. Nekaterih singularnih točk pa naš algoritem ni našel. Zgrešil je tiste, ki so se nahajale na robu prstnega odtisa. V našem primeru smatramo orientacijo na robu prstnega odtisa za manj zanesljivo. Zato tudi vse singularne točke, ki se nahajajo manj kot 21 slikovnih elementov od roba, odstranimo iz množice kandidatov. Opazili smo tudi primere pri prstnih odtisih tipa šotorast lok, kjer se jedro ni nahajalo v vrhu krivulje. To je seveda napaka in bi lahko povzročala probleme pri klasifikaciji prstnega odtisa. Nekaj primerov napak si lahko ogledamo na sliki 4.14.



Slika 4.14: Primeri napak pri detekciji singularnih točk: (a) DB1, (b) DB2, (c) DB3, (d) DB4

Ko poznamo število in lokacijo singularnih točk, se lahko lotimo testiranja klasifikacije. Testiranje smo tokrat izvedli na testnih množicah. Rezultate za vsako množico smo zapisali v tabelah 4.12, 4.13, 4.14 in 4.15.

Pravi razred	**Dodeljen razred**				
	L	**Š**	**LZ**	**DZ**	**S**
L	**8**	0	0	0	0
Š	10	**7**	5	8	2
LZ	125	23	**92**	13	11
DZ	169	7	17	**90**	5
S	41	1	4	3	**159**

Tabela 4.12: Rezultati klasifikacije za testno množico DB1 z uporabo petih razredov, L – lok, Š – šotorast lok, LZ – leva zanka, DZ – desna zanka, S – spirala

Pri vseh štirih testnih množicah najprej opazimo slabe rezultate pri določanju razreda lok. V več kot polovici primerov smo prstne odtise, ki spadajo v razred levih in desnih zank, opredelili kot razred lok. Po preverjanju postopka smo ugotovili, da je do omenjene situacije prišlo večinoma pri tistih prstnih odtisi iz razreda zank, kjer določitev delte ni bila mogoča. To se zgodi takrat, ko se delta nahaja na skrajnem robu prstnega odtisa ali pa če delta sploh ni prisotna na odtisu. Po pregledu vseh množic smo ugotovili, da je bilo takih prstnih odtisov tipa leva ali desna zanka v testni množici DB1 53%, v DB2 57%, v DB3 92% in v DB4 75%. Rezultati so torej najslabši pri testni množici DB3, kar je razumljivo, saj je vsebovala slike najmanjših velikosti (300×300) v našem preizkusu. Na sliki 4.15 je nekaj prstnih odtisov iz množice DB3,

Pravi razred	Dodeljen razred				
	L	**Š**	**LZ**	**DZ**	**S**
L	**31**	4	4	4	5
Š	10	**22**	1	1	6
LZ	148	11	**65**	11	13
DZ	140	16	11	**68**	21
S	40	2	6	6	**154**

Tabela 4.13: Rezultati klasifikacije za testno množico DB2 z uporabo petih razredov

Pravi razred	Dodeljen razred				
	L	**Š**	**LZ**	**DZ**	**S**
L	**23**	0	1	0	0
Š	67	**1**	0	1	3
LZ	258	1	**5**	6	2
DZ	189	1	5	**19**	2
S	117	0	2	4	**93**

Tabela 4.14: Rezultati klasifikacije za testno množico DB3 z uporabo petih razredov

kjer smo imeli največ problemov. Za lok smo mnogokrat označili tudi prstne odtise, ki sicer spadajo pod šotorast lok ali spiralo. Tudi v tem primeru je prišlo do napake, kadar nismo našli vseh singularnih točk – delt pri šotorastem loku in enega od jeder pri spirali.

Kot šotorast lok smo označili zelo malo prstnih odtisov, prav zaradi problema, omenjenega v prejšnem stavku. Pogosto smo šotorast lok zamenjali za levo ali desno zanko. Vzrok za to smo odkrili že pri testu singularnih točk. Veliko jeder se pri šotorastem loku namreč ni nahajalo v konici krivulje. Do napake je prišlo, ko se je računala usklajenost lokalnih orientacij s smerjo od jedra do delte. Ker je bilo jedro zamaknjeno, je bila izračunana usklajenost večja od praga in prstni odtis je bil označen za zanko.

Dobre rezultate smo dosegli pri klasifikaciji zank pri katerih smo našli delto. Kjer delte nismo našli, je bil prstni odtis označen za levo ali desno zanko.

Še najbolje se je algoritem odrezal pri iskanju spiral. Večina napak se je zgodila pri prstnih odtisih, ki niso vsebovali celotne spirale. V takem primeru

Pravi razred	Dodeljen razred				
	L	**Š**	**LZ**	**DZ**	**S**
L	**24**	0	0	0	0
Š	5	**3**	0	0	0
LZ	246	0	**70**	1	3
DZ	184	0	3	**63**	6
S	8	2	6	4	**172**

Tabela 4.15: Rezultati klasifikacije za testno množico DB4 z uporabo petih razredov

Slika 4.15: Primeri prstnih odtisov iz testne množice DB3, kjer je klasifikacija otežena zaradi lokacije singularne točke na robu prstnega odtisa

je algoritem našel le eno jedro in je v večini primerov prstni odtis označil za lok.

Na koncu si poglejmo povprečno število napak glede na tip prstnega odtisa za vse testne množice (tabela 4.16). Tu lahko samo potrdimo, da je pri vseh testnih množicah do največ napak prišlo pri obeh zankah, sledi šotorast lok, najbolje pa se algoritem odreže pri loku ter spirali.

Stopnje napak za vse štiri testne množice so na voljo v tabeli 4.17. Pričakovano se največ napak zgodi v testni množici DB3, saj slike prstnih odtisov zaradi majhne velikosti pogosto ne vsebujejo dovolj singularnih točk za uspešno klasifikacijo. Pri ostalih testnih množicah pa v povprečju vsak drugi prstni odtis napačno klasificiramo. Rezultat ni odličen, je pa lahko dobra osnova za izboljšave. Naj spomnimo, da levji delež napak odpade na napačno klasificiranje zank zaradi manjkajočih delt. V tem primeru bi ločevanje med zankama in lokom lahko izvedli tako, da bi preverjali ukrivljenost v točki, kjer leži jedro. Za lok velja, da ima majhno ukrivljenost, medtem ko imata zanki veliko. Še vedno pa ne bi ločili med levo in desno zanko. Slednji problem pa bi lahko rešili

Razred	Povprečna stopnja napak [%]			
	DB1	DB2	DB3	DB4
L	0	35,4	4,2	0
ŠL	78,1	55	99	62,5
LZ	65,2	73,8	98,2	78,1
DZ	68,8	73,4	91,2	75,4
S	23,6	26	57	10,4

Tabela 4.16: Stopnja napak na testnih množicah za vse tipe prstnih odtisov

	DB1	DB2	DB3	DB4
stopnja napake	47,1%	52,7%	70%	45,3%

Tabela 4.17: Povprečna stopnja napak na testnih množicah

z novim razredom zanka, ki bi vseboval tako leve kot desne zanke. Natačnost takega sistema bi bila po vsej verjetnosti večja.

Časovna zahtevnost naše metode je majhna, saj opravi svoje delo za vse testne množice v manj kot 30 ms (tabela 4.18).

4.7 Primerjanje

Primerjanje je zadnji od korakov našega sistema, ki poda dokončno odločitev, komu pripada podani prstni odtis. Sam sistem smo izdelali kot verifikator. To pomeni, da smo zapisali v datoteko podatke o značilkah zajetega prstnega odtisa, potem pa smo preverjali istovetnost na ostalih prstnih odtisih. Vsak prstni odtis smo najprej primerjali z ostalimi 7 vzorci istega prstnega odtisa, nato pa še z vsemi ostalimi prstnimi odtisi. S tem smo prišli do ocen FNMR in FMR, ki nam nudita informacijo o natančnosti sistema. Pri tem velja še manjše opozorilo. Če smo primerjali prstni odtis h s prstnim odtisom g, potem

Porabljen čas (ms)			
DB1	DB2	DB3	DB4
18	24	15	15

Tabela 4.18: Čas porabljen za klasifikacijo

obratne primerjave g z h nismo izvedli, saj bi s tem dobili enak rezultat.

Da bi lahko oceni FNMR in FMR primerjali z ocenami algoritmov iz tekmovanja FVC 2002, smo uporabili enako število testov za vsako testno množico [29]:

$$\begin{aligned} ((8 \cdot 7)/2) \cdot 100 &= 2.800 \text{ pri FNMR} \\ ((100 \cdot 99)/2) &= 4.950 \text{ pri FMR} \end{aligned} \tag{4.1}$$

Pred izvedbo primerjanja na testni množici smo nastavili vse potrebne parametre glede na podatke iz učnih množic. Nato smo kot poseben test preizkusili, kaj se zgodi, če postavimo prag odločitve β na 0 (poglavje 3.8.3). V tem primeru bi pričakovali, da je FNMR enaka nič in FMR enaka 100. V tabeli 4.19 so podani rezultati.

Ocena	**Testne množice**			
	DB1	**DB2**	**DB3**	**DB4**
FNMR (%)	0	0	0	0
FMR (%)	100	100	100	100

Tabela 4.19: Stopnji FNMR in FMR pri pragu odločitve $\beta = 0$

Tako kot smo predvidevali, je naš sistem našel vse enake prstne odtise, hkrati pa ni zavrnil nobenega pri navzkrižnem primerjanju. Glede na testiranje v učnih množicah, smo ugotovili, da je naš sistem dokaj restriktiven pri lokaciji in smeri značilk, zato smo v naslednjem testu prag odločitve postavili na 30% vseh zadetkov. Stopnji FNMR in FMR za vse testne množice si lahko ogledamo v tabeli 4.20.

Ocena	**Testne množice**			
	DB1	**DB2**	**DB3**	**DB4**
FNMR (%)	89,3	88,6	91,2	81,3
FMR (%)	1,7	3,7	2,4	0,88

Tabela 4.20: Stopnji FNMR in FMR pri pragu odločitve $\beta = 0.3$

Kot vidimo iz rezultatov, je za vse tri množice značilna velika stopnja FNMR. Torej naš sistem zavrne veliko prstnih odtisov, ki so enaki vhodnemu. Na drugi strani je stopnja FMR zelo nizka, kar pomeni, da naš sistem napačno potrdi majhen odstotek prstnih odtisov, torej je dokaj zanesljiv. Glavni razlog

za te napake je napačna frekvenca na šumnih predelih, kar povzroči nastanek večjega števila napačnih značilk. Drugi razlog pa je napačna lokacija referenčne točke, saj se je dostikrat zgodilo, da se je referenčna točka pri dveh vzorcih istega prsta velikokrat nahajala na različnih mestih. Vse to je vplivalo na manjše število uspešnih primerjanj, zaradi česar prag odločitve ni bil presežen. V takem verifikacijskem sistemu bi moral uporabnik za uspešno preverjanje zagotoviti čim bolj kvaliteten prstni odtis, kjer bi bili razmiki med dolinami in grebeni jasno vidni. Za to bi moral paziti na pritisk in premike prstnega odtisa, singularne točke (predvsem jedra) pa bi se morale nahajati čim bolj v sredini slike prstnega odtisa. Ker smo hoteli narediti sistem manj restriktiven, smo poskušali zmanjšati prag na 0,2 (tabela 4.21).

Ocena	**Testne množice**			
	DB1	**DB2**	**DB3**	**DB4**
FNMR (%)	81,5	81,1	79,5	73,1
FMR (%)	10,7	12,4	11,1	6,0

Tabela 4.21: Stopnji FNMR in FMR pri pragu odločitve $\beta = 0,2$

Po pričakovanju se je z zmanjšanjem praga znižala stopnja FNMR, s čimer smo zmanjšali število napačnih zavrnitev. Na drugi strani pa se je za nekaj odstotokov povečala stopnja FMR. Kot vidimo, sta stopnji FNMR in FMR v močni medsebojni odvisnosti, zato ni mogoče spreminjati vrednosti ene stopnje, ne da bi s tem hkrati vplivali na drugo. Od vrste sistema pa je odvisno, kako bomo določili vrednost stopenj. Če želimo bolj zanesljiv sistem, je veliko število napačnih sprejemov popolnoma nesprejemljivo, zato poskušamo zmanjšati faktor FMR na minimum. S tem sicer povečamo število napačnih zavrnitev, kar pa je v tem primeru veliko manj problematično.

Pri nezahtevnih sistemih, kjer zanesljivost ni kritična, lahko v prid preprostejše uporabe zmanjšamo števila napačnih zavrnitev na minimum. S tem se lahko izognemo nadležnemu ponavljanju zajema prstnih odtisov. Ker največkrat taki sistemi obdelujejo manjše število prstnih odtisov, je verjetnost napake tudi pri večjem faktorju FMR majhna.

S tem ko imamo stopnji FNMR in FMR podani, še vedno ne vemo dosti o sistemu. Kajti če lahko prag dinamično prilagajamo, je še vedno nemogoče oceniti, ali je sistem z majhnim FNMR in veliki FMR boljši od sistema z velikim FNMR in majhnim FMR. Zato poskušamo na temu mestu določiti oceno EER, torej točko, kjer sta FNMR in FMR enaki. Le-ta nam lahko poda oceno natačnosti sistema, ki ni odvisna od praga [33]. S poizkušanjem različnih

pragov nam je uspelo najti EER za testne množice. Poleg tega smo za primerjavo izbrali najboljši in predzadnji algoritem izmed 31 algoritmov udeleženih na tekmovanju FVC 2002 [29] (tabela 4.22). Zadnjega algoritma nismo podajali, ker je imel stopnjo EER enako 50%. Na sliki 4.16 je podana še ROC krivulja, ki vsebuje še EER stopnji za izbrana algoritma iz FVC 2002. Za slednja pa smo za primerjavo podali še njuni ROC krivulji. Za boljše razumevanje grafov naj razložimo, da modra in zelena točka na sliki 4.16 označujeta točko na slikah 4.17 in 4.18, kjer se krivulji FMR in FNMR sekata.

EER (%)	**Testne množice**			
	DB1	**DB2**	**DB3**	**DB4**
naš algoritem	44,4	48,7	48,1	41,2
najboljši iz FVC 2002	0,10	0,17	0,37	0,10
predzadnji iz FVC 2002	35,0	35,2	42,3	43,96

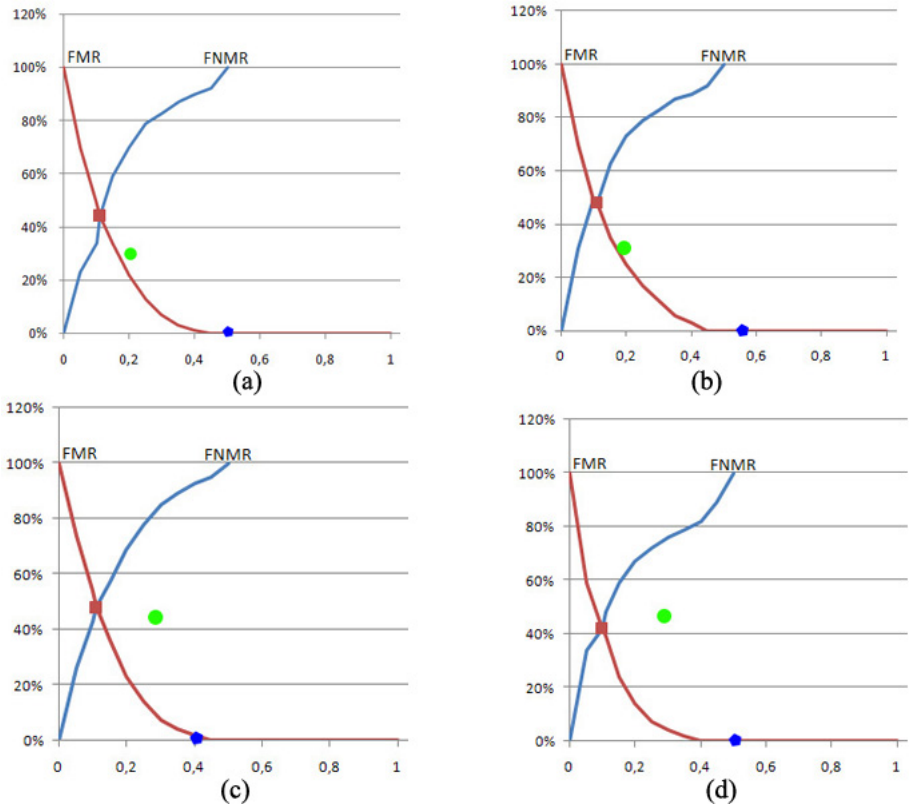
Tabela 4.22: Stopnja EER za vse štiri testne množice

Rezultati niso najboljši, saj je EER pri vseh testnih množicah, razen zadnji, blizu 50%. To pomeni, da v povprečju napačno zavrnemo in napačno potrdimo vsak drugi prstni odtis. Rezultati so veliko slabši od najboljšega algoritma iz tekmovanja FVC 2002, vendar pa ne zaostajajo dosti za predzadnjim algoritmom. Pri slednjem smo celo boljši pri množici DB4. Izkaže se namreč, da dosežemo najboljše rezultate na testnih množicah DB1 in DB4, saj vsebujeta bolj kvalitetne slike, kakor ostali dve množici. Očitno imajo torej slike slabe kvalitete velik vpliv na rezultat. Zato smo hoteli oceniti algoritem še na množici prstnih odtisov dobre kvalitete. V ta namen smo zbrali iz vsake testne množice 16 prstnih odtisov boljše kvalitete ter izračunali FNMR in FMR. Prag smo nastavili na 0,3.

Ocena	**Testne množice**			
	DB1	**DB2**	**DB3**	**DB4**
FNMR (%)	6,1	41,1	42,6	8,2
FMR (%)	0	0	0	0

Tabela 4.23: Stopnja EER za prstne odtise boljše kvalitete pri pragu odločitve $\beta = 0,3$

Naš algoritem se na prstnih odtisih dobre kvalitete torej odreže dobro (tabela 4.23). Že tako nizko stopnjo FMR iz tabele 4.20 smo uspeli znižati

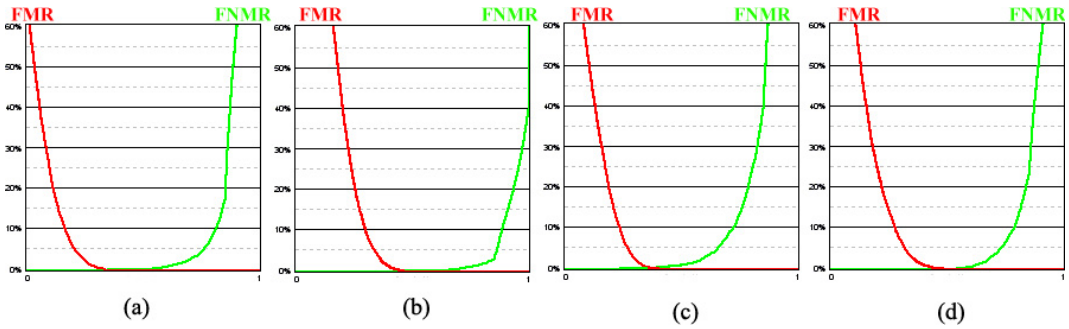


Slika 4.16: ROC krivulja za vse štiri testne množice, EER za naš algoritem je označen z rdečo točko, za prvi algoritem iz FVC2002 z modro točko, za zadnji pa z zeleno točko: (a) DB1, (b) DB2, (c) DB3, (d) DB4

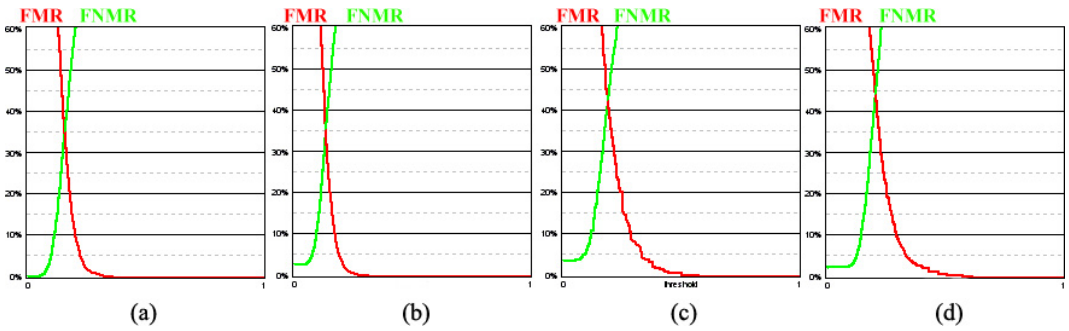
na 0, medtem ko se je stopnja FNMR pri vseh močno zmanjšala. Tak sistem bi bil primeren v manjših okoljih, obenem pa bi morali zagotavljati tudi kakovost prstnih odtisov pri zajemanju. S temi rezultati se uvrstimo v sredino algoritmov iz tekmovanja FVC 2002.

Za naš sistem smo tudi omenili, da je neobčutljiv na rotacije, translacije in skaliranje. Zato smo ga želeli preveriti tudi pri rotiranih prstnih odtisih. V ta namen smo naključni prstni odtis iz učne množice DB1 s pomočjo programske opreme zarotirali v treh korakih po 45 stopinj (slika 4.19). Na koncu smo dobili 4 vzorce, za katere smo poizkušali izračunati stopnjo FNMR. Prag odločitve smo nastavili na 0.3.

Izmerjena stopnja FNMR za rotirane prstne odtise je 0%, torej je sistem za vse prstne odtise potrdil ujemanje z ostalimi rotiranimi prstnimi odtisi. Zato lahko potrdimo, da je neobčutljivost na rotacije zagotovljena. Na koncu si



Slika 4.17: ROC krivulje za algoritem, ki je zasedel prvo mesto na tekmovanju FVC 2002: (a) DB1, (b) DB2, (c) DB3, (d) DB4



Slika 4.18: ROC krivulje za algoritem, ki je zasedel predzadnje mesto na tekmovanju FVC 2002: (a) DB1, (b) DB2, (c) DB3, (d) DB4

poglejmo še čas potreben za primerjavo dveh prstnih odtisov. V čas procesiranja je vključena pretvorba značilk iz kartezijskega prostora v polarni prostor, branje vektorja značilk iz ASCII datoteke ter sama primerjava. Porabljen čas je odvisen predvsem od števila značilk. Ker je slednje pri vseh prstnih odtisih majhno in redko presega 50, so tudi razlike med testnimi množicami neopazne. Povprečen čas za eno primerjavo znaša 15 ms. Čas za primerjavo 800 prstnih odtisov pa je znašal povprečno 1 sekundo. Z uporabo relacijske podatkovne baze bi lahko tudi slednjega zmanjšali.

Skupen čas procesiranja od segmentacije do primerjanja v povprečju znaša 3 sekunde. Odvisen je predvsem od velikosti slike in je pri majhnih slikah manjši od dveh sekund, pri večjih pa doseže tudi štiri sekunde. Za primerjavo, najboljše uvrščen algoritem iz tekmovanja FVC 2002 je dosegel čas 1,97 sekunde.

Rezultati primerjanja niso ravno zadovoljivi. V poglavju 3.3 smo omenili,

Slika 4.19: Zavrteni prstni odtisi naključnega prstnega odtisa

da prihaja pri izračunu frekvence do napak, kar se pri prstnem odtisu kaže v obliki fiktivnih grebenov. To posledično vpliva tudi na število najdenih značilk, zato se mnogokrat zgodi, da pri dveh prstnih odtisih ne presežemo praga β, ki označuje dva prstna odtisa za enaka. Z izboljšavo omenjenega problema bi tudi množica značilk vsebovala le dejanske značilke, kar bi doprineslo k natačnosti sistema.

Dodatno težavo povzroča zajem in kvaliteta slik. Pri spiralah se je dostikrat zgodilo, da je bilo pol prstnega odtisa odrezanega, zato določitev lokacije drugega jedra ni bila mogoča. Ker je bila s tem spremenjena tudi lokacija referenčne točke, je bilo nemogoče pričakovati pozitiven rezultat. Možna rešitev bi bila določitev meril pri zajemu prstnih odtisov, pri čemer bi sprejeli le prstne odtise z dovolj informacije.

4.8 Pomanjkljivosti ter nadaljne delo

Najpogostejši razlogi za slab rezultat so:

- napačno izračunana lokalna frekvenca grebenov na šumnih predelih,
- napačno določena lokacija referenčne točke.

Oba razloga sta več ali manj posledica slabše kvalitete slik (slika 4.2(c)). Prstne odtise slabše kvalitete bi lahko odstranili že pri registraciji. V koraku izboljšanja kvalitete bi lahko preverili intenziteto sivine posameznih področjih prstnega odtisa. V primeru, da bi bila intenziteta prevelika ali premajhna, bi dani prstni odtis zavrnili. S tem bi zagotovili, da so v sistemu shranjeni le dobri odtisi, kar bi znatno vplivalo na natančnost sistema.

Predlagali bi tudi odkrivanje šumnih področij na način, pri katerem bi preverili doslednost orientacije prstnega odtisa pred glajenjem. Če bi večina

lokalnih orientacij kazala v isto smer, potem bi lahko trdili, da je prstni odtis zadovoljive kvalitete. Če pa bi število lokalnih orientacij, ki niso v skladu z okolico, presegalo določen prag, bi lahko prstni odtis pri registraciji zavrnili.

Učinkovitost sistema bi najbolj povečali z izboljšanjem izračuna lokalne frekvence grebenov. V koraku za izboljšanje kvalitete prstnega odtisa je pri izračunu lokalne frekvence grebenov prihajalo do napak pri vlažnih prstnih odtisih, kjer grebeni odtisa tiščijo skupaj ter pri odtisih, ki so vsebovali izrazite znojne pore. Na takih področjih bi bilo potrebno frekvenco nastaviti glede na sosedne zanesljive vrednosti. V našem postopku smo uporabljali enostavno povprečenje frekvence za celotno frekvenčno sliko. Predvidevamo, da bi vstavljanje (angl. *interpolation*) z uporabo Gaussovega jedra [10] dalo tudi na takih prstnih odtisih boljše rezultate.

Dosti pozornosti bi morali poleg lokalne frekvence grebenov nameniti tudi izračunu lokacije singularnih točk. Pri klasifikaciji se je dostikrat zgodilo, da smo našli namesto enega jedra kar dva v neposredni bližini. Posledica tega je bila napačna lokacija referenčne točke, s čimer smo zapravili možnost uspešnega primerjanja. To se je dogajalo kljub temu, da smo za oceno verodostojnosti singularnih točk uporabljali še dodatno zglajeno orientacijsko polje. Na take primere smo naleteli pri prstnih odtisih slabe kvalitete, saj je orientacijsko polje kljub glajenju še vedno vsebovalo nekaj nepravilnosti. Za dokončno potrditev singularnih točk bi lahko uporabljali še tretje orientacijsko polje, ki bi bilo še dodatno zglajeno. Samemu iskanju referenčne točke pa bi se lahko tudi izognili, če bi izvedli primerjavo kar med značilkami skupaj z njihovo okolico [26].

Ne tako pomemben problem pri klasifikaciji je bilo ločevanje med zankama in lokom. Kadar prstni odtis tipa zanka ni vseboval delte, je naš algoritem predpostavil, da gre za prstni odtis tipa lok. Možna rešitev bi bila, da bi poleg singularnih točk spremljali še obliko grebenov okoli jedra [11]. Podoben problem se je pojavil pri spiralah, kjer je bilo na sliki zajeto le eno jedro. V tem primeru bi se lahko zgledovali po metodi [2], ki je sposobna v nekaterih primerih odkriti jedro tudi v primeru, kadar se nahaja izven slike.

Kot dodatno možnost pri koraku primerjanja bi predlagali še možnost preverjanja števila grebenov, ki jih seka navidezna črta od referenčne točke do trenutno izbrane značilke. S tem bi še dodatno omejili možnost primerjave dveh napačnih značilk in s tem uspešno doprinesli k učinkovitosti sistema.

Ko bi rešili problem s frekvenco in singularnimi točkami pa bi morali izboljšati tudi čas procesiranja prstnega odtisa. Željeno je, da je uporabnik verificiran v sekundi ali manj. Največ časa porabimo za izboljšanje kvalitete, zato je potrebno v tem segmentu pogledati tudi za alternativnimi rešitvami.

Pospešitev bi lahko dosegli tudi z uporabo programskega jezika C ali C++. Razlike med omenjenima jezikoma ter jezikom C# je ta, da se pri slednjem periodično izvaja pobiranje smeti (angl. *Garbage Collection*) – sproščanje pomnilnika objektov, ki niso več potrebni. Pri jeziku C ali C++ pa programer sam poskrbi za čiščenje, zato je ta pristop hitrejši. Pospešitev, ki bi jo s tem dosegli, je žal nemogoče napovedati. Še zadnji (manjši) korak k pospešitvi pa bi bila uporaba relacijske podatkovne baze, ki bi sama skrbela za čim hitrejši način vračanja ter zapisovanja podatkov.

Poglavje 5

Postavitev sistema

Našo aplikacijo smo poskušali povezati tudi z dejanskim čitalcem prstnih odtisov, s čimer bi vzpostavili pravi mali sistem za verifikacijo oseb na podlagi prstnega odtisa. Čitalec smo povezali z računalnikom preko USB vmesnika, za povezavo med našo aplikacijo ter čitalcem pa je skrbel programski vmesnik SDK (angl. *Software Development Kit*) proizvajalca čitalca. Z uporabo SDK smo iz čitalca pridobili sliko, ki smo jo poslali v naš sistem na obdelavo. Na koncu smo natančnost takega sistema tudi preverili.

5.1 Izbira čitalca

Naša aplikacija se je odrezala bolje pri slikah prstnih odtisov, ki so bile pridobljene z optičnimi čitalci, kakor pri tistih pridobljenih s kapacitativnim čitalcem. Zato smo se odločili, da se tudi pri izbiri osredotočimo na slednje. Optični čitalci v primerjavi s kapacitativnimi ponudijo večjo sliko prstnega odtisa, s čimer dobi aplikacija veliko več podatkov, kar posledično daje tudi boljše rezultate. Našo aplikacijo smo preizkusili tudi s slikami pridobljenimi iz termičnih čitalcev, vendar smo se zaradi slabših rezultatov odločili, da se tem čitalcem izognemo.

Ker je dandanes ponudba čitalcev zelo velika, smo podali nekaj parametrov, s katerimi bi zožali izbor:

- ločljivost vsaj 500 dpi,
- cena čitalca ne večja kot 200€,
- povezljivost programskih vmesnikov z .NET okoljem,

- velikost zajete slike vsaj 300 × 250 slikovnih elementov.

Če čitalec ni popolnoma ustrezal vsem kriterijem, nas je pa presenetil s katero drugo lastnostjo, smo ga vseeno dodali na seznam. Prednost so imeli tisti čitalci, pri katerih je bilo možno pridobiti SDK zastonj. V tabeli 5.1 je podan seznam čitalcev.

Podjetje	**Model**	**Ločljivost [dpi]**	**Velikost slike**	**Cena**
Biometrika	Fx2000	569	560 × 296	172€
DigitalPersona	U.are.U 4000B	512	500 × 550	63€
Identix	BioTouch 500	500	20mm×25mm	80€
CrossMatch	Verifier 300 LC 2.0	500	31mm×31mm	350€
Futronic	FS88	500	480 × 320	66€
SecuGen	Hamster Plus	500	260 × 300	56€

Tabela 5.1: Izbira čitalca prstnih odtisov; kjer ni bilo podatka o velikosti slike, je podana velikost senzorja

Pri modelu Biometrika Fx2000 nismo uspeli izvedeti cene programskega vmesnika SDK in opcije, ali podpira okolje .NET. Na seznam smo ga dodali predvsem zaradi velikosti slike prstnega odtisa.

Model DigitalPersona U.are.U 4000B nam je bil še najbolj všeč zaradi dobre ločljivosti in velikosti slike, hkrati pa je cena čitalca sprejemljiva. Cena SDK kompleta znaša 421€, v ponudbi pa je tudi malo omejen zastonj komplet One Touch for Windows SDK, ki bi ga lahko izrabili za zajem slike.

Model Identix BioTouch 500 ne podpira okolja .NET, kar bi morali rešiti s svojim vmesnikom. Ponuja pa dobre rešitve za svojo ceno. Cene programskega vmesnika nam ni uspelo izvedeti.

CrossMatch Verifier 300 LC 2.0 ima tudi odlične specifikacije, vendar pa nas od modela odvrača predvsem cena ter oblika (velikost).

Futronic FS88 za ceno 66€ nudi odlično ločljivost in velikost slike. Za 344€ nudi programski vmesnik SDK z mnogo funkcijami, možna pa je tudi izbira vmesnika SDK, ki je zastonj. Slaba lastnost je ta, da ponudi sliko prstnega odtisa v RAW formatu, pri čemer bi morali sami poskrbeti za pretvorbo v bitmap format.

Poleg DigitalPersona U.are.U 4000B nam je bil najbolj všeč še SecuGen Hamster Plus z dobrimi lastnostmi ter možnostjo zajema slike že ob dotiku prstnega odtisa brez posredovanja uporabnika. Ponuja tudi zastonj programski

vmesnik FDx SDK, s katerim je omogočen zajem slik. Oba omenjena čitalca smo preizkusili v povezavi z našo aplikacijo.

5.2 Rezultati

V tem delu predstavljamo dobljene rezultate pri testiranju čitalcev. Ker smo posamezne korake algoritma že dodobra stestirali, se bomo tu osredotočili na natačnost klasifikacije in predvsem primerjanja.

Testna množica je sestavljena iz petih oseb, pri čemer je bila vsaka vzorčena z osmimi odtisi istega prsta. Torej smo imeli skupaj 40 vzorcev.

5.2.1 DigitalPersona U.are.U 4000B

Klasifikacija

V množici vzorčenih prstnih odtisov smo imeli 2 odtisa tipa lok, 1 odtis tipa spirala, 1 odtis tipa desna zanka in 1 odtis tipa leva zanka. Ko smo preizkusili naš algoritem nad zbranimi prstnimi odtisi, nam je podal naslednje rezultate:

Pravi razred	**Dodeljen razred**				
	L	**Š**	**LZ**	**DZ**	**S**
L	**15**	0	1	0	2
Š	0	**0**	0	0	0
LZ	7	0	**0**	0	0
DZ	3	0	1	**4**	0
S	2	0	4	0	**2**

Tabela 5.2: Rezultati klasifikacije pri uporabi čitalca DigitalPersona U.are.U 4000B

Sistem se dobro obnese pri prstnih odtisih tipa lok, medtem ko je pri ostalih tipih nenatančen. Prstni odtis tipa leva zanka namreč ni vseboval delte, zato je naš algoritem ta vzorec odtisa prepoznal kot lok. Prstni odtis tipa spirala pa je bil zelo neopredeljiv tudi za ročno klasifikacijo, saj je imel tako značilnosti spirale kot desne zanke. Čas procesiranja klasifikacije posameznega prstnega odtisa je znašal manj kot 30 ms.

Zaključimo lahko, da se klasifikacija obnese podobno kakor pri vseh štirih testnih množicah iz tekmovanja FVC 2002. Še vedno ima velik vpliv na rezultat

kvaliteta prstnega odtisa ter količina informacij, ki jih prstni odtis nosi. Večjih razlik v koraku klasifikacije nismo opazili.

Primerjanje

Primerjanje smo izvedli tako, da smo prstni odtis, ki je bil vzorčen, prebrali preko čitalca ter poskušali najti zadetek v množici vzorcev. Nato smo na koncu izračunali oceno FNMR in FMR. Za izračun ocene FNMR je bilo izvedenih 34 primerjav, za FMR pa 150. Pri slednji smo namreč izvedli primerjanje prvega odtisa vsakega prsta z vsemi ostalimi odtisi, ki ne pripadajo trenutnemu prstu. Če tega ne bi storili, bi morali izvesti le 4 primerjave, kar pa ne bi dalo realne ocene FMR. V tabeli 5.3 so podani rezultati za primerjavo izvedeno pri pragu $\beta = 0,3$.

Prstni odtis	**Nepravilno zavrnjeni**	**Nepravilno sprejeti**
Odtis 1	3	1
Odtis 2	6	2
Odtis 3	2	1
Odtis 4	4	1
Odtis 5	6	3

Tabela 5.3: Rezultati klasifikacije pri uporabi čitalca DigitalPersona U.are.U 4000B

Tak sistem ima stopnjo EER 42%. Slabi rezultati so predvsem posledica napak pri izračunu orientacije, frekvence ter položaja in števila singularnih točk. Slike prstnih odtisov vseeno niso bile tako kvalitetne (slika 5.1), kot smo pričakovali. Zaradi velikih kontrastnih razlik na nekaterih mestih, izrazitih por in šuma, je prihajalo pri orientacijski sliki do nepravilnosti. Posledično je algoritem napačno ocenil tudi položaj singularnih točk, kar pa ima zelo velik vpliv na končno odločitev.

Razočarani smo bili tudi nad formatom slike prstnega odtisa, ki nam jo ponudi čitalec. Slika je namreč v obliki trapeza in ima na levi in desni strani ter na vrhu bele robove. Poleg tega pa čitalec odreže skoraj tretjino prstnega odtisa, četudi položimo cel prst na čitalec.

Naš sistem pa vseeno ni neuporaben. Možen scenarij uporabe je tak, kjer bi shranjevali v sistemu za vsak prstni odtis 10 različnih vzorcev. Prag odločitve bi povečali na tisto mejo, kjer bi bilo število nepravilno sprejetih prstnih odtisov (FMR) enako 0. Kadar bi se primerjal prstni odtis, ki bi bil shranjen v sistemu,

Slika 5.1: Primeri zajetih slik s čitalcem DigitalPersona U.are.U 4000B

bi bila možnost pozitivnega odgovora večja. V nasprotnem primeru pa bi zaradi visokega praga napačen prstni odtis redko pretental sistem.

Pri času procesiranja nismo pričakovali večjih razlik, kakor v poglavju 4.7. Tudi tu porabi algoritem v povprečju 15 ms za izvedbo ene primerjave. Za izvedbo celotnega postopka od segmentacije do primerjanja pa je algoritem potreboval v povprečju 4 sekunde, saj so bile slike velike 500×500 slikovnih elementov.

5.2.2 SecuGen Hamster Plus

Klasifikacija

V množici vzorčenih prstnih odtisov smo imeli tako kot pri čitalcu DigitalPersona U.are.U 4000B 2 odtisa tipa lok, 1 odtis tipa spirala, 1 odtis tipa desna zanka in 1 odtis tipa leva zanka. Ko smo preizkusili naš algoritem nad zbranimi prstnimi odtisi, nam je podal naslednje rezultate:

Pravi razred	**Dodeljen razred**				
	L	**Š**	**LZ**	**DZ**	**S**
L	**16**	0	0	0	0
Š	0	**0**	0	0	0
LZ	1	0	**7**	0	0
DZ	3	0	1	**4**	0
S	2	0	0	1	**5**

Tabela 5.4: Rezultati klasifikacije pri uporabi čitalca SecuGen Hamster Plus

Sistem je našel vse prstne odtise tipa lok. Šest prstnih odtisov je napačno označil za lok, ker smo prstne odtise zajemali zelo različno in nekateri zato

niso vsebovali delt. Problem smo imeli tudi pri prstnemu odtisu spirala, ki je imel značilnosti desne zanke. Čas procesiranja klasifikacije je znašal manj kot 30 ms.

Sistem se obnese bolje kot pri prvem čitalcu in pri vseh štirih testnih množicah iz tekmovanja FVC 2002. S čitalcem nam je namreč uspelo zajeti slike prstnih odtisov, ki so imele dosti velike razmike med grebeni, le-ti pa so bili tudi dosti bolj razločni kakor pri prvem čitalcu. Posledično je bila tudi izračunana orientacija bolj natančna in s tem klasifikacija. Vseeno pa se na to oceno ne moremo zanesti, saj je klasifikacija močno odvisna od zajema.

Primerjanje

Tudi pri drugem čitalcu smo ubrali enak pristop kakor pri prvem. V tabeli 5.5 so podani rezultati za primerjavo izvedeno pri pragu $\beta = 0,3$.

Prstni odtis	**Nepravilno zavrnjeni**	**Nepravilno sprejeti**
Odtis 1	3	2
Odtis 2	3	2
Odtis 3	0	1
Odtis 4	4	2
Odtis 5	5	0

Tabela 5.5: Rezultati klasifikacije pri uporabi čitalca SecuGen Hamster Plus

Za sistem s čitalcem SecuGen Hamster Plus znaša stopnja EER 40%. Visoka stopnja je posledica obdelave prstnih odtisov, kjer so bile singularne točke na robu, pa tudi pritisk prsta na čitalec nismo nadzorovali. Zato je prišlo do napak predvsem pri izračunu frekvence ter števila in lokacije singularnih točk. Slabo se naš sistem odreže tudi pri prstnem odtisu tipa spirala, kjer zaradi majhne medrazredne variabilnosti število jeder ni konstantno med zaporednimi zajemi. Posledica tega je, da se lokacija referenčne točke zelo spreminja. Zaradi takih zamikov primerjave istih prstnih odtisov ne uspejo.

Opazimo lahko, da se čitalec SecuGen Hamster Plus obnese bolje kot čitalec DigitalPersona U.are.U 4000B, kljub temu, da nudi manjšo velikost slik, ki znaša 260×300 slikovnih elementov. Hamster Plus za razliko od U.are.U 4000B namreč celotno sliko izkoristi za prikaz prstnega odtisa (slika 5.2). Z pazljivostjo pri zajemu, predvsem pri pritisku prsta na čitalec, pa nam je uspelo tudi pridobiti slike z dobro razločnimi grebeni in dolinami.

Slika 5.2: Primeri zajetih slik s čitalcem SecuGen Hamster Plus

Čas procesiranja je podoben kot pri prvem čitalcu in znaša povprečno 15 ms za izvedbo ene primerjave. Za izvedbo celotnega postopka od segmentacije do primerjanja pa je algoritem potreboval v povprečju 2,2 sekundi.

Poglavje 6

Zaključek

V tem delu je bil predstavljen algoritem za verifikacijo oseb na podlagi prstnega odtisa. Algoritem izvede segmentacijo prstnega odtisa, izboljšavo kvalitete slike prstnega odtisa, binarizacijo in tanjšanje grebenov, iskanje značilk, klasifikacijo ter na koncu primerjanje. Po uspešni izvedbi naštetih operacij nam algoritem poda odgovor, ali sta dva prstna odtisa enaka.

Aplikacija daje zadovoljive rezultate, vendar še vedno preslabe za uporabo v komercialne namene. Na slabe rezultate so vplivali predvsem prstni odtisi slabe kvalitete. Če lahko zagotovimo dobro kvaliteto prstnih odtisov, bomo z rezultati aplikacije več kot zadovoljni, kar smo tudi pokazali na izbrani množici prstnih odtisov zadovoljive kvalitete.

Največ pozornosti in nadaljnih izboljšav zahteva postopek izboljšanja kvalitete prstnega odtisa. Tu smo naleteli tudi na največ težav, saj nam je napačen izračun frekvence grebenov povzročal veliko preglavic. Hkrati pa je omenjeni korak tudi ozko grlo sistema, saj je procesorsko in časovno najbolj potraten. Ker je od kvalitete slike prstnega odtisa močno odvisna natačnost sistema, velja temu koraku v prihodnosti posvečati veliko pozornosti.

Aplikacija je bila razvita tudi z namenom vključitve v sistem, ki bi na podlagi prstnega odtisa preveril pravice uporabnika za vstop v laboratorij. Zato smo poskušali povezati aplikacijo tudi s čitalcem prstnih odtisov. Pri tem nismo imeli večjih težav, rezultate pa smo dosegli podobne kakor na testih.

Ker je samodejno razpoznavanje oseb na podlagi prstnih odtisov kompleksen problem, smo lahko z izdelanim sistemom zadovoljni, saj smo dosegli relativno ugodne rezultate. Pri izdelavi aplikacije smo se srečali s kopico problemov, ki smo jih dokaj uspešno rešili. Vsak korak aplikacije pa ponuja še veliko možnosti za izboljšanje, kar sistem tudi omogoča s svojo modularno zasnovo.

Slike

Tabele

Literatura

[1] M.U. Akram, A. Tariq, S.A. Khan, S. Nasir, Fingerprint image: pre- and post-processing, *International Journal of Biometrics 2008*, 1(1), str. 63-80, 2008

[2] V. Areekul, K. Suppasriwasuseth, S. Jirachawang, The New Focal Point Localization Algorithm for Fingerprint Registration, *Proceedings of the 18th International Conference on Pattern Recognition*, zbornik 04, str. 497-500, 2006

[3] A.M. Bazen, S.H.Gerez, Segmentation of fingerprint images, *ProRISC Workshop on Circuits*, str. 276-280, 2001

[4] A.M. Bazen, S.H.Gerez, Systematic methods for the computation of the directional fieldsand singular points of fingerprints, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), str. 905-919, 2002

[5] X. Chen, J. Tian, J. Cheng, X. Yang, Segmentation of Fingerprint Images Using Linear Classifier, *EURASIP Journal on Applied Signal Processing*, letnik 2004, Januar, str. 480-494, 2004

[6] Y. Chen, G.Parziale, E. Diaz-Santana, A.K. Jain, 3D Touchless Fingerprints: Compatibility With Legacy Rolled Images, *Biometrics Symposium: Special Session on Research at the Biometric Consortium Conference*, str. 1-6, Avgust 2006

[7] S.S. Chikkerur, *Online fingerprint verification system*, M.S.C. thesis, State University of New York at Buffalo, 2005

[8] M.M. Hadhoud, W.S. El Kilani, M.I. Samaan, An adaptive algorithm for fingerprints image enhancement using gabor filters, *ICCES '07. International Conference on Computer Engineering & Systems*, str. 227-236, November 2007

[9] Y. He, J. Tian, X. Luo, T. Zhang, Image enhancement and minutiae matching in fingerprint verification, Pattern Recognition Letters, 24(9-10), str. 1349-1360, 2003

[10] L. Hong, Y. Wan, A. Jain, Fingerprint image enhancement: algorithm and performance evaluation, *IEEE Transactions Pattern Analysis and Machine Intelligence*, 20(8), str. 777-789, 1998

[11] L. Hong, A. Jain, Classification of Fingerprint Images, *Proceedings of the 11th Scandinavian Conference on Image Analysis*, str. 7-11, 1999

[12] A. Jain, L. Hong, R. Bolle, On-line fingerprint verification, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9), str. 302-314, 1997

[13] D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar, Handbook of Fingerprint Recognition, 2003

[14] K. Karu, A.K. Jain, Fingerprint Classification, *Pattern Recognition*, 29 (3), str. 389-404, 1996

[15] J. Kovač, *Sistemi za avtomatsko identifikacijo prstnih odtisov*, seminarska naloga, 2004

[16] J. Qi, M. Xie, Segmentation of fingerprint images using the gradient vector field, *IEEE Conference on Cybernetics and Intelligent Systems*, str. 543-545, September 2008

[17] J. Qi, M. Xie, A Robust Algorithm for Fingerprint Singular Point Detection and Image Reference Direction Determination Based on the Analysis of Curvature Map, *IEEE Conference on Cybernetics and Intelligent Systems*, str. 1051-1054, 2008

[18] H. Lemme, Fingerprint as Password, *Elektor Electronics*, letnik 2005, Oktober, str. 20-24, oktober 2005

[19] M. Liu, X. Jiang, A.C. Kot, Fingerprint Reference Point Detection, *EURASIP Journal on Applied Signal Processing*, letnik 2005, Januar, str. 498-509, 2005

[20] P.M. Patil, S.R. Suralkar, F.B. Sheikh, Rotation Invariant Thinning Algorithm to Detect Ridge Bifurcations for Fingerprint Identification, *17th IEEE International Conference on Tools with Artificial Intelligence*, str. 641-649, 2005

[21] G. Praprotnik, *Samodejno razpoznavanje prstnih odtisov*, diplomsko delo, Fakulteta za elektrotehniko, november 2002

[22] E. Tabassi, C. Wilson, C. Watson, *Fingerprint Image Quality*, Tech. Rep. 7151, National Institute of Standards and Technology (NIST), Avgust 2004

[23] T. Tang, X. Wu, M. Xiang, An Improved Fingerprint Singular Point Detection Algorithm Based on Continuous Orientation Field, *International Symposium on Computer Science and Computational Technology ISCSCT*, zbornik 2, str. 454-457, 2008

[24] J. Zang, J. Yuan, F. Shi, S. Du, A Fingerprint Matching Algorithm of Minutia Based on Local Characteristic, *Fourth International Conference on Natural Computation*, zbornik 4, str. 13-17, 2008

[25] F. Zhao, X. Tang, Preprocessing and postprocessing for skeleton-based fingerprint minutiae extraction, *Pattern Recognition*, 40 (4), str. 1270-1281, 2007

[26] W. Zhong, X. Ning, C. Wei, A Fingerprint Matching Algorithm Based on Relative Topological Relationship Among Minutiae, *International Conference on Neural Networks and Signal Processing*, str. 225-228, 2008

[27] Biometrics:
http://en.wikipedia.org/wiki/Identification_of_human_individuals
(5.9.2009)

[28] History of Fingerprinting:
http://www.policensw.com/info/fingerprints/finger01.html (25.9.2009)

[29] Fingerprint Verification Competition 2002:
http://bias.csr.unibo.it/fvc2002/ (7.10.2009)

[30] SFinGe (**S**ynthetic **Fin**gerprint **Ge**nerator):
http://biolab.csr.unibo.it/research.asp?organize=Activities&select=&selObj=12&pathSubj=111 (5.9.2009)

[31] AForge .NET Framework:
http://www.aforgenet.com/framework/ (5.9.2009)

[32] Fingerprint Vendor Technology Evaluation:
http://fpvte.nist.gov/? (30.9.2009)

[33] About FMR, FNMR and ERR:
http://www.bioid.com/sdk/docs/About_EER.htm (5.9.2009)

[34] JetFlashR220 Fingerprint USB Hard Drive:
http://www.transcendusa.com/products/ModDetail.asp?ModNo=169
(5.9.2009)

[35] Innovatrics:
http://www.innovatrics.com (5.9.2009)