Cvičení z předmětu Biometrie
Úloha 3: Rozpoznávání duhovky

Eduard Bakštein, eduard.bakstein@fel.cvut.cz

3. ledna 2013

Úvod

Rozpoznávání duhovky je vysoce přesná biometrická metoda s celou řadou praktických, využívaná na letištních kontrolách, v přístupových systémech i humanitárních misích. Mezi její hlavní výhody patří robustnost a stabilita duhovky po celý život jedince, stejně jako možnost odečítat duhovku z větší vzdálenosti. Základními prvky systému rozpoznávání duhovky jsou snímání, segmentace (nalezení oblasti duhovky), rozbalení ("unwrapping") a kódování.

Cílem úlohy je vyzkoušet v praxi metody, používané pro rozpoznávání duhovky. Úloha je složena z následujících částí:

1. Implementace kódu pro segmentaci oblasti duhovky pomocí Houghovy transformace
2. Přiřazení zadaných obrázků oka k irisCodes v databázi
3. Bonusová úloha: zpracování obrázků vlastní duhovky, útoky na systém.

Řešení bude probíhat formou implementace požadovaných funkcí do existujícího Iris Toolboxu, který je ke stažení na stránkách cvičení. Jedním z výstupů úlohy bude vámi zpracovaný stručný protokol, do kterého vložíte zejména ty části které jsou vyžadovány v textu zadání. Jednotlivé body stručně okomentujte (především pro vaší vlastní orientaci)

Předpokládá se samostatná práce na úloze. Cvičení využívá zdrojové kódy Libora Maška z University of Western Australia.

Úloha 3, Duhovka - extrakce příznaků a porovnávání irisCode (20 bodů)

Pokyny k odevzdání

Úloha se odevzdává na posledním cvičení bloku cvičícímu. Stručně popíšete zdrojový kód Vaší implementace a předvedete funkci na zadaných testovacích obrázcích. Grafy a obrázky z jednotlivých kroků zpracování se odevzdávají v krátkém reportu se stručným popisem a zodpovězením

otázek, položených v zadání. K započtení bodů za úlohu je také nutno odeslat zdrojový kód (pouze zazipovaný adresář ToDo) do odevzdávacího systému (deadline v den cvičení do půlnoci).

1 Segmentace oblasti duhovky pomocí Houghovy transformace

K úspěšnému zakódování informace, obsažené ve struktuře duhovky, je nezbytné nalézt a oddělit (segmentovat) oblast duhovky od okolí (bělma, víčka atd.). Nejjednodušším způsobem je aproximace oblasti duhovky jako mezikruží. K nalezení oblasti duhovky pak můžeme použít Houghovu transformaci (HT) pro kruh (Viz přednáška 9). Při této metodě se nejprve pomocí hranového detektoru naleznou v obrázku hrany a následně se každý nalezený hranový bod projektuje do prostoru parametrů. Projekce pak probíhá přičtením pevně dané konstanty na všechna místa prostoru parametrů, kterými prochází pomyslná kružnice se středem ve zvoleném hranovém bodě a s daným poloměrem. Za předpokladu známého poloměru se v bodě středu hledaného kruhu v prostoru parametrů vyskytne maximum. Postup pro neznámý poloměr je obdobný: provede se projekce pro hodnoty poloměru ve zvoleném rozsahu a následně se hledá maximální hodnota přes celý prostor parametrů.

Parametrický prostor

Mějme obrázek velikosti x krát y pixelů. Hledáme-li kružnici o neznámém poloměru v rozsahu $r \in R, R = \{r_{min}, r_{min} + 1, ..., r_{max}\}$ se středem v libovolném bodě obrázku, bude naším výchozím parametrickým prostorem nulová matice o rozměrech $(y + 2 \cdot r_{max}) \times (x + 2 \cdot r_{max}) \times |R|$, kde $|R|$ značí kardinalitu množiny R, zde tedy počet pozorovaných poloměrů[1].

Do jednotlivých vrstev parametrického prostoru pak vepisujeme kružnice o poloměru odpovídajícím dané vrstvě se středy postupně ve všech hranových bodech. Při vykreslování kružnice můžete vyjít z analytické rovnice pro kružnici

$$(x - a)^2 + (y - b)^2 = r^2, \tag{1}$$

z níž si vyjádříte jednu ze souřadnic a přes druhou můžete tudíž snadno iterovat. Nejjednodušším způsobem vytvoření souřadnic na kružnici je funkce *circlecoords*, obsažená v iris toolboxu, která vrací přímo souřadnice bodů na křužnici o zadaných parametrech (tady se může hodit funkce *sub2ind*[2], pozor také na konvenci [*radeksloupec*] vs. [*xy*])). Další možností je pak např. využití goniometrických funkcí.

[1] Použijete-li pro vytvoření níže uvedenou funkci *circlecoords*, která dokáže omezit body na zadaný rozsah, není rozšíření parametrického prostoru v rozměru $x - y$ potřeba a parametrický prostor bude mít rozměr $y \times x \times |R|$

[2] Pokud v Matlabu zadáte jako indexy matice dva vektory, bude použit jejich kartézský součin, nikoliv jednotlié páry [řádek sloupec]! Funkce *sub2ind*() převede takovéto páry na lineární indexy, které lze již použít jako vektor. Př: $a = magic(3); r = 1 : 3; c = r; a(r, c) =?; a(sub2ind(size(a), r, c)) =?$

Stručný popis použití iris toolboxu naleznete na konci tohoto zadání v příloze. Kostry funkcí, které máte za úkol implementovat, naleznete v adresáři ToDo.

Úkol:

1. Implemetujte funkci

   ```
   [circlePupil, circleIris] = findIrisAnnulus(uint8 iris_image),
   ```

   založenou na Houghově transformaci (HT). *circlePupil* a *circleIris* jsou vektory parametrů nalezených kružnic [x, y, poloměr] pro zornici a duhovku. Rozsahy parametrů HT pro prohledávání vhodně zvolte tak, aby nebyla nutná jejich úprava pro jednotlivé Vám přidělené obrázky.

2. Funkci využijte spolu se zbytkem Iris Toolboxu pro zakódování zadaných obrázků duhovky.

3. Obrázky mezivýsledků (detekce hran, nalezené kružnice, parametrický prostor) přidejte do zprávy.

Poté, co implementujete funkci FindIrisAnnulus, můžete spustit toolbox na libovolný Vámi zvolený obrázek (stačí změnit název ukázkového obrázku v *iris_demo.m* na cestu k vybranému souboru). Iris toolbox, konkrétně obalová funkce *createIrisTemplate* provede rozbalení vámi segmentovaného mezikruží do pseudopolárních souřadnic a zakódování duhovky pomocí kvantizace fáze Gaborových filtrů (viz přednášky).

Poznámky k implementaci:

- Pro **detekci hran** použijte funkci $edge(I,'canny', thres)$. Funkce nejprve vypočítá gradientní pole obrázku a následně vyprahuje za použití parametru $thres$. Experimentujte s prahem, snadno dosáhnete odstranění části nadbytečných hranových bodů, čímž můžete významně urychlit a zpřesnit výpočet.

- Pro odstranění vysokofrekvenčních složek obrázku, které vedou k velkému počtu neužitečných hranových bodů (řasy apod). Výsledek můžete vylepšit předzpracováním obrázku ještě před hranovou detekcí, a to např. úpravou jasových hodnot (úpravou jasu a kontrastu) nebo **gaussovskou filtrací** ("rozmazáním"). Nejprve vytvoříte masku filtru pomocí $fspecial$, kterou následně aplikujete na obrázek:

  ```
  G = fspecial('gaussian',[a a],sigma);
  Ig = imfilter(image,G);
  ```

  Proměnná a určuje velikost vytvořené masky, sigma směrodatnou odchylky vypočítaného 2D gaussovského rozdělení.

- Vyzkoušejte, zda lépe funguje Vaše detekce pro vnější nebo vnitřní hranu duhovky a tuto detekci provádějte jako první. Nalezené parametry pak můžete s výhodou použít pro omezení druhé vyhledávané kružnice (např: střed kruhu ohraničujícího duhovku zvenčí by měl ležet uvnitř nalezené zornice apod.) Může se hodit pro vnitřní a vnější hranu použít různá nastavení předzpracování obrázku.

- Každé rozumné omezení počtu prohledávaných parametrů razantně snižuje dobu, potřebnou pro výpočet.

2 Nalezení zadaných obrázků v databázi

Tato část úlohy simuluje proces identifikace na základě duhovky tak, jak probíhá v reálném prostředí. Neznámé osobě jsou nejprve sejmuty fotografie duhovky (zde: zadané obrázky), které jsou zakódovány shodným způsobem, jako zbytek databáze. Následně se v databázi vyhledává uložený vzorek, který je právě sejmutému vzorku nejpodobnější. Tímto způsobem je v databázi nalezen nejpravděpodobnější subjekt.

K porovnání dvou irisCodes se používá Hammingovy vzdálenosti (*Hamming distance - HD*), normované počtem překrývajících se použitelných bitů v maskách obou Iris-kódů:

$$HD = \frac{\parallel (codeA \otimes codeB) \cap maskA \cap maskB \parallel}{\parallel maskA \cap maskB \parallel}, \tag{2}$$

kde $\otimes$ je logický *XOR* operátor, $\cap$ logický *AND* a $\parallel$ *norm* operátor[3] - počet jedničkových bitů v daném řetězci. CodeA a codeB jsou porovnávané iris kódy a maskA a maskB příslušné šumové masky (zde 1 je užitečný signál, 0 šum. *Pozor: v Iris toolboxu je konvence opačná!*)

Aby se předešlo nežádoucímu vlivu rotace duhovky (naklopení hlavy) na výsledek porovnání, používá se vzájemný bitový posun iris kódů vůči sobě ve zvoleném rozsahu a výpočet Hammingovy vzdálenosti pro každý posun. Jako výsledek porovnání se pak bere nejlepší shoda - tedy nejmenší dosažená vzdálenost.

Úkol:

1. Implementujte funkci

   ```
   HD = irisHammingDistance(codeA,codeB, maskA, maskB)
   ```

   pro porovnání dvou iris kódů na základě vzorce (2). Implementujte vzájemný bitový posun iris kódů, rozsah posunů volte ± 12 bitů[4], hodit se může funkce $circshift()$.

[3] Pozor: *norm* operátor zde operuje v binárním oboru - vrací tedy počet jedniček - narozdíl od Matlabovské funkce *norm()*, která vrací vždy euklidovskou vzdálenost od počátku a je tedy pro tyto účely nevhodná! Účelu poslouží funkce *sum()*.

[4] Uvědomte si, kolik bitů zároveň je zakódováno během kvantování fáze (viz přednášky), upravte podle toho krok posunu a ušetřete si podstatnou část porovnání!

2. Projděte databázi uložených iris kódu (*database.mat*) a najděte takovou osobu, od které Vám přidělený obrázek duhovky nejpravděpodobněji pochází. Nejnižší nalezené hodnoty Hammingových vzdáleností pro jednotlivé, Vám zadané obrázky, vložte do zprávy, stejnětak jako informaci, které osoby (a oči) jste na základě obrázků dohledali. K načtení a procházení databáze můžete pro usnadnění použít připravený skript *walkDatabase.m*

3. Projděte databázi ještě jednou a nyní porovnejte všechny záznamy se všemi, přičemž rozdělte ty, u kterých se porovnávaly iris kódy stejného oka (=stejné oko stejné osoby) a ostatní. Obě sady hodnot vykreslete do společného histogramu. Jaké vycházejí Hammingovy vzdálenosti pro porovnání stejných a různých očí. Kam byste umístili rozhodovací práh (zvolte)? Jaký by měl tento práh FAR a FRR? A Co nám výsledný graf vypovídá o naší metodě? Graf s krátkým komentářem a odpověďmi na otázky zaneste do zprávy.

3 Bonusové úlohy

V rámci těchto nepovinných úloh máte možnost využít Vámi implementovaných funkcí. Jejich zpracování nezabere příliš času a umožní Vám pochopit a prakticky si vyzkoušet další aspekty rozpoznávání duhovky.

Bonus: zpracování vlastních obrázků duhovky

- Pomocí kamery pro snímání duhovky vyfotografujte vlastní duhovky obou očí.

- Parametry iris toolboxu upravte tak, aby dobře vyhovovaly ke zpracování obrázků použité kamery. Pravděpodobně bude třeba upravit parametr *irisConfig.loNoiseThreshold* v *iris_demo.m*, určující práh pro řasy (oblasti tmavší než zadaný práh jsou označeny jako šum). Aktuální hodnota prahu je nastavena pro výchozí databázi.

- Pořízené snímky převeďte na iris kódy a uložte do vlastní databáze. Jaká je hammingova vzdálenost mezi fotografiemi obou Vašich očí? Jaký je vztah této hodnoty k Vámi zvolenému prahu z předchozího úkolu a k hodnotám porovnání stejných a rozdílných očí? Výsledky uveďte do zprávy a komentujte.

Bonus: Útok na systém pomocí vytištěných snímků duhovky

K útoku použijeme běžné fotografie duhovky.

- Poskytnuté fotografie duhovky, vytištěné na laserové tiskárně i ty, vzniklé klasickým fotografickým osvitem, nasnímejte iris kamerou.

- Snímky, sejmuté kamerou, segmentujte, převeďte na iris kódy a porovnejte se zadanými iris kódy, vzniklými přímo sejmutím oka

kamerou. Jak se liší, jaká je jejich hammingova vzdálenost? Byl by takový systém úspěšný?

- Vyzkoušejte detekci tohoto útoku pomocí 2D Fourierovy transformace: porovnejte spektra přímého a podvrženého obrázku. Jak se tato spektra liší? Je nějaký rozdíl mezi obrázkem z laserové tiskárny a fotografickým osvitem? Jaký? Jak by bylo možné útok detekovat a který z obrázků ke pro detekci obtížnější?

Závěr

Blahopřejeme, právě jste naimplementovali významnou část systému pro rozpoznávání duhovky. Vyzkoušeli jste, co obnáší předzpracování a segmentace a prozkoumali různé vlastnosti porovnávání pomocí normované Hammingovy vzdálenosti. Zobrazili jste histogramy pro porovnání mezi obrázky stejných a různých očí a zhodnotili statisticky kvality Vašeho systému. Ten funguje obdobně, jako systémy po celém světě, používané na letištích a bezpečnostních kontrolách. Navíc doufáme, že až si budete chtít vzpomenout, "jak jsem to vlastně dělal", pomůže vám výsledný protokol.

Poděkování

Vytvoření této úlohy bylo podpořeno Ministerstvem Školství, Mládeže a Tělovýchovy grantem FRVŠ 2529/2012.

Příloha: Stručný přehled Iris Toolboxu

- Představu o funkci toolboxu si uděláte spuštěním *iris_demo.m*. Při spuštění dema se zobrazí hodnoty předem předpočítané a uložené v cache.
- Hlavní funkcí, zodpovědnou za extrakci příznaků, je *createiristemplate.m*
- Pro nastavení parametrů a přidání adresářů do PATH matlabu slouží *iris_init.m*. Zde lze nastavit mj. hodnotu prahu pro vlastní obrázky duhovky.
- Kostry funkcí, které máte za úkol naimplementovat, najdete v adresáři *ToDo*. Sem ukládejte veškeré kódy, které během práce na úloze vytvoříte.
- Vám zadané obrázky naleznete v adresáři *Images*.

Struktura dat

Zakódované obrázky a příslušné masky se nacházejí v souboru *database.mat*, obsahujícím stejnojmenné pole struktur - celkem 30 pro 30 osob, uložených v databázi. Každá osoba pak obsahuje pole struktur R a L s různým počtem uložených kódů. Ty jsou v polích *template*, příslušné šumové masky v poli *mask*.

Příklad: k iris kódu, odpovídajícímu 3. obrázku pro levé oko osoby č. 12 se dostanete přes `database(12).R(3).template`, k příslušné masce pak přes `database(12).R(3).mask`. Procházení databáze je pro vás však připraveno ve funkci *walkDatabase.m*