Cvičení z předmětu Biometrie
Porovnávání otisků prstů

P. Vostatek

November 9, 2011

1 Úloha 3, Otisk prstu - srovnávání otisků (15 bodů)

Termín odevzdání: 16. 11. 2011, 11:00 (odevzdání bude probíhat osobně na posledním cvičení bloku).

Na coursewaru jsou k dispozici zdrojové kódy k úloze Porovnávání otisků, stahněte si je, rozbalte a nastavte si rozbalený adresář jako aktuální v Matlabu. Porovnávání otisků prstů bude mít 2 hlavní části:

1. Porovnání pomocí markantů
2. Porovnání pomocí FingerCodu (Gaborova filtrace)

Mezi zdrojovými kódy cvicení je opět připraven skript main_batch.m který shrnuje všechny části včetně obrazového výstupu.

Hlavním úkolem cvičení bude naprogramování 2 funkcí pro porovnání (matching) otisku prstu s databází obrázků. Otisky vstupují do funkcí už vzájemně zarovnané podle jejich jádra. Po implementaci kódů bude za úkol zhodnotit použitelnost uvedených algoritmů pro porovnání otisků.

Update 9. 11. 2011: Ke 3. cvčení jsou přiložené pomocné kódy (viz program cvičení na cw).
ukazka.m slouží jako ukázka porovnání dvojice otisků, které si odpovídají a dvojice otisků, které si neodpovídají, má podobný obsah jako main_batch. Skript vypíše ohodnocení podobnosti (skóre) pro obě dvojice a obě metody porovnání.
fingercode_score.m slouží k výpočtu skóre z otiskových fingerCodů (rovnice [1]) po vytvoření příznakových vektorů.
show_align.m slouží k vykreslení dvou koster otisků přes sebe.
Také jsou přibaleny implementované funkce *match* a *fingercode_creation* ve formě p-souborů, což umožňuje jejich spuštění, ale ne editaci. Můžete na nich vyzkoušet funkci a správné výstupy.

Požadavky:

1. Implementujte funkci "match" pro výpočet shody mezi zarovnanými markanty mezi vstupním obrázkem a vzorem v databázi.
Syntaxe je následující:

`[matchingScore, nbmatch, inputmatch, dbmatch] = match(mAi1, mAi2);`

Vstupem jsou pole s vyznačenými markanty pro vzor (*mAi1*) a porovnávaný otisk (*mAi2*). Tato pole jsou přímo výstupem funkce `findminutia`, přičemž *mAi2* musí být zarovnány s *mAi1* pomocí funkce `align2`.
Update 9. 11. 2011: Výstupy jsou:
matchingScore - ohodnocení podobnosti dvou otisků. Výpočet je níže.
nbmatch - počet dvojic markantů.
inputmatch - pole stejné velikost jako mAi2, kde jsou vyznačeny markanty stejným způsobem jako v mAi2, ale pouze ty, které byly zpárovány.
dbmatch - stejně jako inputmatch, ale pro otisk 1.
Ukázky jsou v `main_batch`. Velikosti polí jsou stejné jako vstupních obrázků.

Algoritmus: Vstupem jsou dvě sady markantů (bodů) v rovině: `mAi1` a `mAi2` a práh pro vzdálenost `d`. Vstupní počty markantů jsou: `|mAi1|`, resp. `|mAi2|`, počet párů markantů iniciovaný `nbmatch = 0` a celková vzdálenost `dist = 0`.

• Pro každý bod z `mAi2` nalezněte nejbližší bod v `mAi1`. Pokud jsou markanty vzdáleny méně, než `d`, potom je označte za pár a vyjměte z `mAi1` i `mAi2`. Za každý nalezený pár `nbmatch + 1`

• Po vyčerpání všech markantů z `mAi2` možných spárovat vypočtěte $matchingScore = \frac{2 \cdot nbmatch}{|mAi1|+|mAi2|}$, což je výsledné ohodnocení přiřazení otisků. Platí, že čim vyšší matchingScore, tím spíš patří patří otisky stejnému původci (inteval je $< 0, 1 >$).

2. Implementujte funkci "fingercode_creation":

`[fingercode Fmasked] = fingercode_creation(imOriginal, Gfilt, core, maskSize, dia)`,

imOriginal je vstupni obrazek pro vypocet, *Gfilt* je sada k Gaborových filtrů vytvořená pomocí `GaborFilter_creation`, *core* jsou souřadnice jádra otisku, *maskSize* je velikost bloku pro filtraci obrazu a *dia* jsou velikosti vnějšího a vnitřního poloměru ořezu okolo jádra.
Update (9. 11. 2011: Výstupy:
fingercode - vektor velikosti [1 x N], kde N záleží na velikosti mezikruží (viz dále).
Fmasked - blokový obraz s vyříznutým mezikružím (Figure 1, vpravo) pro každý Gaborův filtr. Velikost [M x N x O]. M x N je velikost blokového obrazu a O je počet Gaborových filtrů. **Výstup Fmasked není nutné implementovat.**

Pomocí sady Gaborových fitrů s různým natočením filtrujte otisk prstu. Vznikne k filtrovaných obrazů. Každý filtrovaný obrázek zpracujte po blocích N velikosti n x n bez překryvu. Hodnotu pro každý blok spočítejte jako $f(N) = |mean(N) - std(N)|$, kde std značí směrodatnou

odchylku, mean střední hodnotu. Z každého bloku vznikne následně jeden pixel. Blokový obraz poté vyřízněte maskou pouze okolo jádra otisku, jak ukazuje Figure 1.

Figure 1: Postup při vytváření FingerCodu

Následný příznakový vektor f_1 (odpovídající vzoru, resp. f_2 odpovídající vstupnímu obrazu) vznikne seřazením hodnot ležících v mezikruží (nevynulované hodnoty Figure 1 vpravo) z každého z k obrazů a jejich seřazením za sebe. Porovnání 2 otisků poté proveďte jako $matchingScoreGabor = mean(abs(f_1 - f_2))$[1]. Čím nižší je výsledná hodnota, tím lépe si otisky odpovídají.

2 Výsledky

Po implementaci funkcí vyzkoušejte jejich funkčnost na přiřazování otisků v přiložené databázi. Nastavte hodnoty pro získání segmentace a ostatních výstupů z minulého cvičení, které vám subjektivně připadaly ideální. Poté otestujte, zda je signifikantní rozdíl mezi hodnotami výsledného hodnocení u odpovídajících dvojic otisků od dvojic, které neodpovídají stejnému prstu. Zkuste kombinovat výstup markatového přiřazování a ohodnocení pomocí FingerCodu. Zkuste vymyslet příznak, který by nejlépe oddělil odpovídající otisky od neodpovídajících dvojic. Správnost zarovnání otisků u odpovídajících si dvojic kontrolujte vizuálně pomocí zarovnaných koster.

Dvě výše uvedené funkce zabalte do zipu a nahrajte do coursewaru. Odevzdání bude probíhat ústně na posledním cvičení bloku.