Vaja 5: Transformacije slike, mere podobnosti, sledenje objektov

Janez Perš, Marija Ivanovska

Laboratorij za strojno inteligenco
Fakulteta za elektrotehniko, Univerza v Ljubljani
e-mail: janez.pers@fe.uni-lj.si, marija.ivanovska@fe.uni-lj.si

Povzetek

V okviru prvega dela te vaje boste opazovali efekt podvzorčenja slike ob uporabi različnih metod interpolacije. V nadaljevanju boste spoznali praktično uporabo dveh mer podobnosti pri sledenju netogih objektov (ljudi).

1 Podvzorčenje slik (0.5 točk)

Pri tej nalogi boste ponovno uporabili posnetek iz prve vaje - izsek iz filma Forrest Gump, ki ga lahko prenesete iz naslednjega spletnega naslova:

http://vision.fe.uni-lj.si/classes/RV/vaje/vaja1/Holywood2-t00427-rgb.avi

Vaša naloga je, da analizirate rezultate treh različnih metod interpolacije ob pomanjšanju velikosti slik in sicer interpolacije po metodi najbližjega soseda (nearest neighbor), bilinearne interpolacije brez filtriranja ("antialiasing off") ter bilinearne interpolacije z vklopljenim predfiltriranjem. Pri obdelavi posnetka uporabite matlabovo funkcijo `imresize`, ki vam omogoča, da sami določite katero metodo želite uporabiti. Oglejte si dokumentacijo omenjene funkcije in poskusite ugotoviti razliko med naštetimi metodami.

Za začetek z uporabo prve metode zmanjšajte ločljivost posnetka enkrat na 1/2, drugič na 1/4 in tretjič na 1/8. Vse tri posnetke shranite lokalno, na svoj računalnik. Postopek ponovite še z drugo in tretjo metodo. Rezultat naj bo torej 9 pomanjšanih posnetkov. Primerjajte kakovost slike pomanjšanih posnetkov enake resolucije.

2 Mere podobnosti - splošen opis naloge

Pri tej vaji boste uporabljali posnetek igre squasha, ki ga, tako kot pri prejšnji vaji, prenesete z naslednjega naslova:

http://vision.fe.uni-lj.si/classes/RV/vaje/vaja4/squash.avi

Vaša naloga je, da napišete program, ki naj deluje na naslednji način:

- Prikaže vam prvo sliko iz posnetka, na kateri z dvema klikoma definirate pravokotnik, ki vsebuje slikovne elemente enega od obeh igralcev. Izbiro igralca in velikosti pravokotnika prepuščamo vam, zavedajte pa se, da bo vaša izbira vplivala na uspešnost metode.

- Program na podlagi označenega izseka slike izračuna značilnico, ki jo boste uporabili kot referenco za iskanje označenega objekta (igralca) na vseh slikah iz posnetka. Tip značilnice je podrobneje opisan v nadaljevanju naloge.
- Program obdela vse preostale slike iz posnetka tako, da s pomočjo izračunane značilnice iz prve slike poišče lokacijo igralca, ki ste ga označili. Na vsaki od slik označite najdene lokacije iglaca, ki mu sledite in tako označene slike shranite v novo AVI datoteko, tako da boste lahko s predvajanjem AVI datoteke opazovali delovanje algoritma in ga tudi demonstrirali asistentu.

Zavedajte se, da je iskanje objekta na celotni sliki razmeroma počasno opravilo (če ima slika na primer 384×288 slikovnih elementov, morate načeloma opraviti več kot 110.000 primerjav na vseh možnih lokacijah, kjer bi se objekt lahko pojavil), zato upoštevajte specifiko problema, ki ga rešujete. Opazujemo namreč posnetek, ki ima 25 slik na sekundo. Iz tega je očitno, da bo premik lokacije igralcev med prejšnjo in naslednjo sliko posnetka relativno majhen. V takšnih primerih se splača uporabiti model gibanja igralca, ki je lahko tudi zelo preprost - na primer model ničtega reda, ki predpostavlja, da se igralec iz slike v sliko ne premakne. To pomeni, da boste pri obdelavi vsake od slik izhajali iz pozicije igralca na prejšnji sliki in izvajali iskanje samo v določeni bližnji okolici. Velikost te okolice naj predstavlja parameter ϵ, katerega vrednost določite eksperimentalno s poskušanjem na nekaj slikah.

Ko boste algoritem preizkušali, se zavedajte, da takšne lokalne metode sledenja niso zelo primerne za dolgotrajno sledenje, prej ali slej bo algoritem namreč "izgubil" pravo pozicijo igralca. Število slik, po katerem se to zgodi, je dejansko mera uspešnosti določenega algoritma (in njegovih komponent, torej izbrane značilnice in mere podobnosti).

Nasvet: Ker je tudi iskanje v lokalni okolici lahko počasno, razvijajte algoritem tako, da na začetku obdelujete le nekaj zaporednih slik, šele ko ste z delovanjem zadovoljni, ga poženite na celotni sekvenci.

Opisano nalogo rešite na dva načina, z uporabo dveh mer podobnosti.

2.1 1. način: Neposredno primerjanje s predlogo, Evklidska razdalja (2.5 točk)

Najprej izvedite primerjanje s predlogo (izsek iz prve slike posnetka, ki predstavlja označenega igralca in je v tem primeru tudi vaša značilnica), pri kateri uporabite preprosto Evklidsko razdaljo (L_2 razdaljo) kot mero podobnosti. Da bo naloga lažja, lahko naredite pretvorbo posnetka in značilnice iz RGB v sivinski format. Lahko delate tudi v RGB prostoru, pri čemer vsoto pod korenom Evklidske razdalje preprosto razširite na vrednosti iz vseh treh RGB kanalov. V obeh primerih naj bi se igralec nahajal na lokaciji, kjer je Evklidska razdalja najmanjša.

2.2 2. način: Primerjava lokalnih histogramov, razdalja hi-kvadrat (2.5 točk)

V algoritmu iz prvega dela te naloge zamenjajte Evklidsko razdaljo z mero podobnosti χ^2. Za začetek izračunajte histogram izseka igralca in ga ustrezno normirajte. Ta histogram bo v tem primeru vaša značilnica, ki ima v tej aplikaciji določene prednosti pred preprosto predlogo – katere? Z uporabo tako določene značilnice s pomočjo mere podobnosti χ^2 izvedite iskanje po vsaki od slik iz posnetka. To pomeni, da morate za vsako področje trenutne slike, na katerem izvajate primerjavo z vašo predlogo, najprej določiti histogram področja in šele nato izračunati razdaljo med novodoločenim histogramom in histogramom izseka igralca (vašo značilnico). Razdaljo izračunajte kot vsoto kvadratov razlik med vrednostmi binov prvega in vrednosti binov drugega histograma.

Da bo naloga lažja, lahko tudi v tem delu delate v sivinskem prostoru (s sivinskimi histogrami) ali pa uporabite katero od drugih predstavitev (pomislite na kanale slike, pretvorjene v HSV in to, da vam prepuščamo možnost, da izberete igralca, ki ga bo algoritem iskal na posnetku!).

3 Sprotno učenje (2 točki)

Pri izvedbi te naloge se bo hitro izkazalo, da je izgled igralca na takšnem posnetku zelo spremenljiv, zato algoritem predelajte tako, da se značilnica sproti posodablja (algoritem se na vsaki sliki posnetka znova "nauči" izgleda igralca, tako kot se je "naučil" izgleda na prvi sliki, ko ste ga označili ročno). Kako dela takšen algoritem? Bolje ali slabše?