Vaja 5: Mere podobnosti, sledenje objektov

Janez Perš

Laboratorij za strojni vid
Fakulteta za elektrotehniko, Univerza v Ljubljani
e-mail: janez.pers@fe.uni-lj.si

Povzetek

V okviru te vaje boste spoznali nekaj mer podobnosti in njihovo praktično uporabo. Aplikacija, s katero boste te mere preizkusili in primerjali, bo sledenje netogih objektov (ljudi).

1 Detekcija objekta s pomočjo mer podobnosti (60%)

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

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

Posnetek že poznate, gre za pogled iz ptičje perspektive. Vaša naloga je, da napišete program, ki naj deluje na naslednji način:

- Prikaže vam prvo sliko iz posnetka, na kateri označite enega od obeh igralcev. Izbiro igralca prepuščamo vam, zavedajte pa se, da bo vaša izbira vplivala na uspešnost metode. Igralca označite tako, da z dvema klikoma označite njegov očrtani pravokotnik.
- Program na podlagi izseka slike, ki ste ga označili, izračuna *značilnico*, ki jo boste uporabili za iskanje označenega objekta (igralca) na tej sliki, in na vseh ostalih slikah iz posnetka. Tip značilnice je podrobneje opisan v nadaljevanju naloge.
- Program obdela vse preostale slike iz posnetka tako, da na vsaki sliki s pomočjo izbrane značilnice poišče igralca, ki ste ga označili. Vsako od slik, na kateri je označena pozicija igralca, algoritem shrani v izhodno AVI datoteko, tako da boste lahko s predvajanjem AVI datoteke opazovali njegovo delovanje in ga tudi demonstrirali asistentu.

1.1 Mere podobnosti in predlagane značilnice

Nalogo rešite na dva načina, z uporabo dveh tipov značilnic, in dveh mer podobnosti.

Najprej izvedite primerjanje s predlogo, pri kateri uporabite preprosto Evklidsko razdaljo (L_2 razdaljo) kot mero podobnosti. Da bo naloga lažja, lahko najprej tako predlogo (izrezan del slike, na katerem je igralec, to je obenem tudi vaša značilnica) kot tudi vsako od slik, pretvorite iz formata RGB v sivinski format - v sliko svetlosti. Lahko pa delate tudi v RGB prostoru, pri čemer vsoto pod korenom Evklidske razdalje preprosto razširite na vrednosti iz vseh treh RGB kanalov.

Drug način je naslednji: vašo predlogo (izrezan del slike, ki ste ga določili na začetku) pretvorite v (ustrezno normiran) histogram (histogram je v tem primeru vaša značilnica, ki ima v tej aplikaciji določene prednosti pred preprosto predlogo – katere?). Nato izvedite iskanje po vsaki od slik iz posnetka s pomočjo mere podobnosti Hi-kvadrat. To pomeni, da morate izračunati histogram področja slike, na katerem izvajate primerjavo z vašo predlogo, in šele nato izračunati razliko histogramov. Da bo naloga preprostejša, pretvorite tako vašo predlogo, kot obdelovano sliko, v enokanalni format. To je lahko kar sivinska slika (slika svetlosti), ali pa katera 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!).

1.2 Nekaj navodil za uspešno izvedbo naloge

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 dva igralca, posnetek pa ima 25 slik na sekundo. Iz tega je očitno, da premik med prejšnjo in naslednjo lokacijo igralca ne bo ravno velik. V takšnem primeru 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, in ga določite 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 namreč algoritem "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).

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.

2 Sprotno učenje (40%)

Pri izvedbi te naloge se bo hitro izkazalo, da je izgled igralca na takšnem posnetku zelo spremenljiv. Zato algoritem predelajte tako, da se na vsaki sliki 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?