VISIÓ PER

COMPUTADOR

Informe de laboratori 7

Marc Cervilla Rovira

14/04/2021

**Introducció:**

En aquesta sessió utilitzarem la segmentació per a realitzar una aplicació. La aplicació consistirà en un sistema semi-automàtic en que nosaltres li passarem una imatge amb qualsevol tipus d’objecte o d’animal i aquesta ens segmentarà l’objecte, diferenciant-lo del fons.

**Implementació:**

Per a realitzar aquest treball he seguit els passos orientatius de la metodologia que veurem a continuació junt amb el meu codi.

En primer lloc obrim la imatge, comprovem que es RGB mirant el size dins de Matlab o mirant la imatge.

I = imread('flor.png');

imshow(I);

En el meu cas utilitzo la imatge següent:



A continuació, utilitzem la funció getrect per a obtenir un rectangle, que utilitzarem per a marcar manualment on es troba l’objecte (més tard explicarem per a que l’utilitzem)

rect = getrect;

Seguidament, obtenim la imatge en format HSV, i preparem la taula en la que aplicarem l’algorisme de kmeans. Per a fer això, primer he separat les diferents components en 3 matrius diferents i el Hue l’he dividit en les seves coordenades circulars x,y en comptes de l’angle, aplicant les funcions de sinus i cosinus.

HSV = rgb2hsv(I);

[f, c, d] = size(HSV);

H = HSV(:,:,1)\*2\*pi;

Hx = cos(H);

Hy = sin(H);

S = HSV(:,:,2);

V = HSV(:,:,3);

\*f i c son les files i les columnes de la imatge original, que ens serviran durant tot el programa.

També he utilitzat la funció reshape, per a convertir les matrius en un vector tan llarg com els píxels amb el seu respectiu valor.

Hx = reshape(Hx,[f\*c,1]);

Hy = reshape(Hy,[f\*c,1]);

S = reshape(S,[f\*c,1]);

V = reshape(V,[f\*c,1]);

Un cop tenim aquests vectors, els podem concatenar fàcilment per poder aplicar kmeans a la matriu resultant:

O = [Hx Hy S V];

k=20;

C = kmeans(O,k);

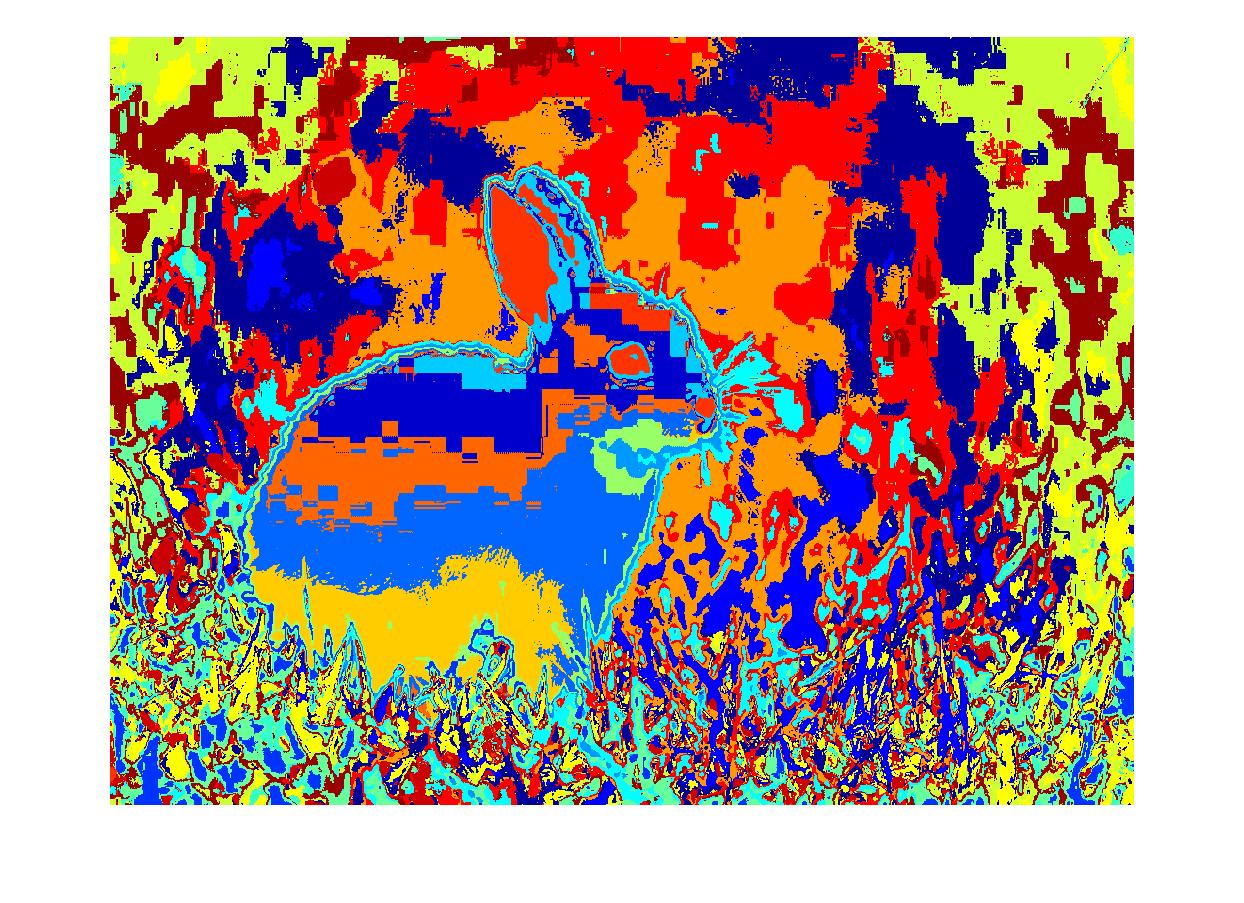
Podem observar el resultat de l’algorisme mirant la imatge només pintant els colors dels diferents grups a partir del següent fragment de codi.

IC = reshape(C,f,c);

rgb = label2rgb(IC);

imshow(rgb);

I el resultat amb k=20 seria algo semblant a això(varia lleument en cada execució):



Ara la idea seria mirar quins colors cauen a dins del rectangle que hem generat al principi i quins a fora. Construirem una imatge binària que actuarà com a màscara, en la que els píxels de dins del rectangle estaran a 1 i els de fora a 0.

En la variable rect teniem representada la posició, l'amplada i l'alçada de la següent forma: *[x,y,amplada,alçada]*, per tant, podem treure els quatre vèrtex del rectangle i dibuixar-lo amb la funció *poly2mask* com veiem a continuació.

x = [rect(1), rect(1)+rect(3),rect(1)+rect(3), rect(1)];

y = [rect(2), rect(2), rect(2)+rect(4), rect(2)+rect(4)];

MASK = poly2mask(x,y,f,c);

A continuació, construim un vector VEC\_H, el qual ens indicarà si un color cau dins del rectangle que hem generat abans o no.

VEC\_H = [C, MASK(:)];

Per a cada color, contarem quants pixels cauen a fora del rectangle, en l’array Hist0, i quants a dins, en Hist1. Per fer-ho, utilitzo un bucle en el que recorro tot el vector VEC\_H, accedeixo a la informació de cada píxel i li sumo 1 als hist. segons esta o no dins del rectangle (que aquesta informació la tenim en la segona columna del array).

Hist0 = zeros(1,k);

Hist1 = zeros(1,k);

for count = 1:f\*c

pos = VEC\_H(count,1);

res = VEC\_H(count,2);

if res

Hist1(pos) = Hist1(pos)+ 1;

else

Hist0(pos) = Hist0(pos)+ 1;

end

end

Llavors, decidim quins son els colors que voldrem pintar. Per fer-ho, comprovem el nombre d’aparicions de cada color a dins i fora del rectangle i, si tenim més dins del rectangle, vol dir que el voldrem pintar.

RES = Hist1 > Hist0;

Un cop decidits quins colors pintarem, torno a realitzar un bucle per decidir els píxels que pintarem. És a dir, mirarem per cada píxel, quin color té i, si és dels colors que hem escollit abans, el pintarem.

Com que ens queda un vector vertical amb el nombre de píxels, torno a utilitzar la funció reshape per a tornar-lo al tamany de la imatge original.

M = zeros(f\*c,1);

for count = 1:f\*c

pos = VEC\_H(count,1);

if RES(pos)

M(count)=1;

end

end

M = reshape(M,[f,c]);

Això ens genera la següent imatge:



Finalment, apliquem tècniques de filtratge per a millorar els resultats, com per exemple, eliminant elements petits del fons amb un close o amb el filtre de mitjana:

%disk = strel('disk',5);

%M = imclose(M,disk);

Resultat:



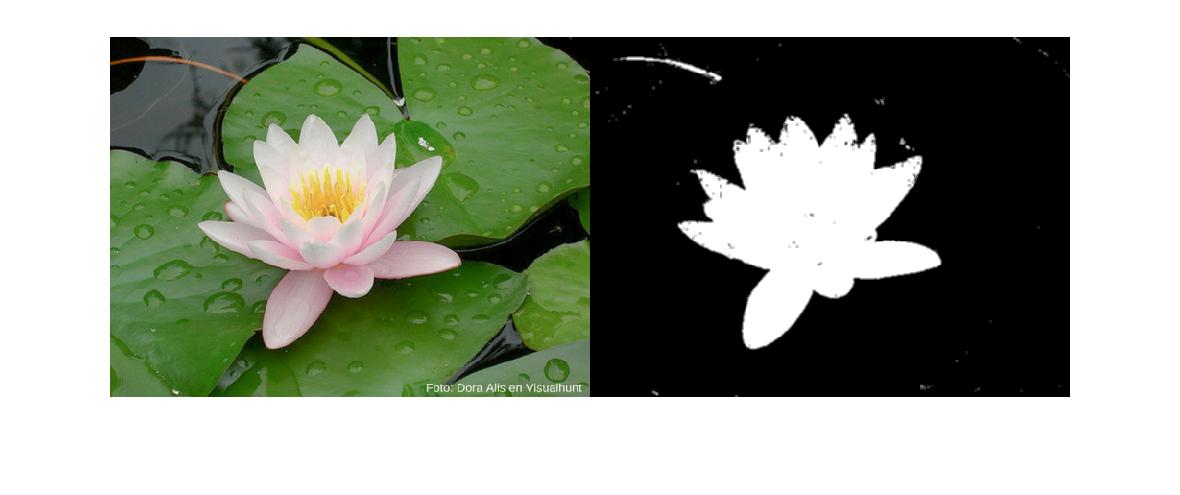
filt = fspecial('average', 3);

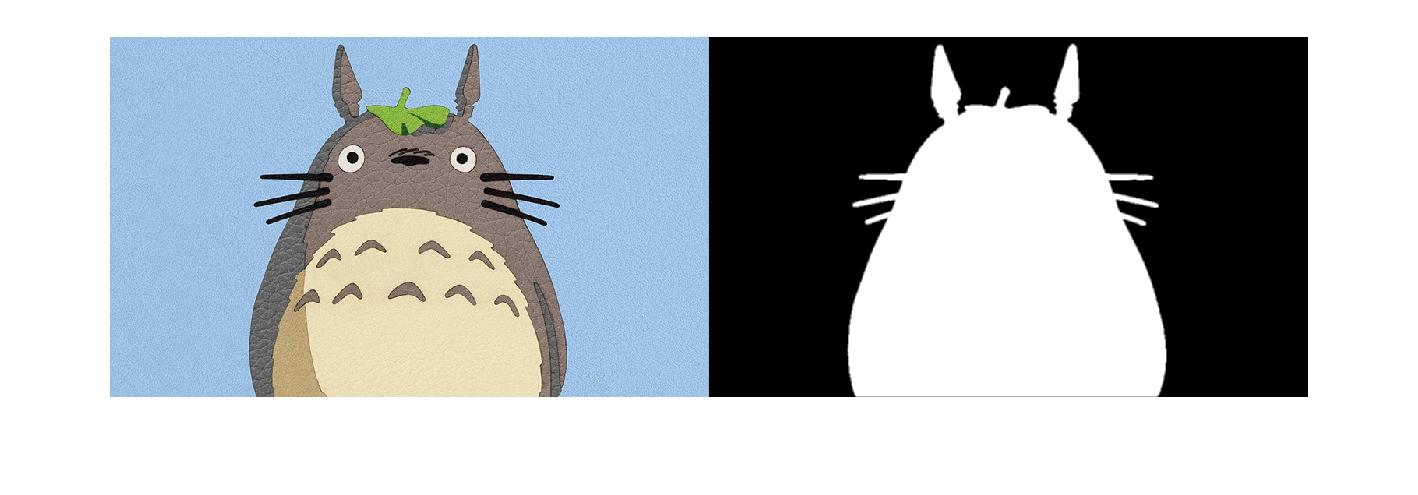
M = imfilter(M,filt);

Resultat:



He provat de fer el mateix procediment per a altres imatges i el resultats son prou bons com es mostra a continuació.





Un cop hem aconseguit aquesta segmentació, se m’ha acudit generar el contorn de la imatge, per tal de resseguir la figura de l’objecte en la imatge principal. He pensat que podria fer això utilitzant un erode i restant-lo de la imatge original:

disk = strel('disk',5);

M2 = imerode(M,disk);

MT = M-M2;

Els resultats aplicant el contorn són aquests:





