Universitat Politècnica de Catalunya

Visió per Computador
Grau en Enginyeria Informàtica

EXERCICI 4 DE LABORATORI

WHERE´S ODLAW?

Autor:
Daniel DONATE

Professor:
Manel FRIGOLA
Q2 Curs 2020-2021

16 de març del 2021

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

Enunciat de l'exercici

En aquesta quarta sessió de laboratori volem desenvolupar una senzilla aplicació de VC, amb la finalitat de trobar a l'Odlaw, un dels personatges de l'entranyable saga *"Where's Wally?"*, del dibuixant britànic Martin Handford. La *Figura 1* mostra l'escenari base sobre el qual treballarem.

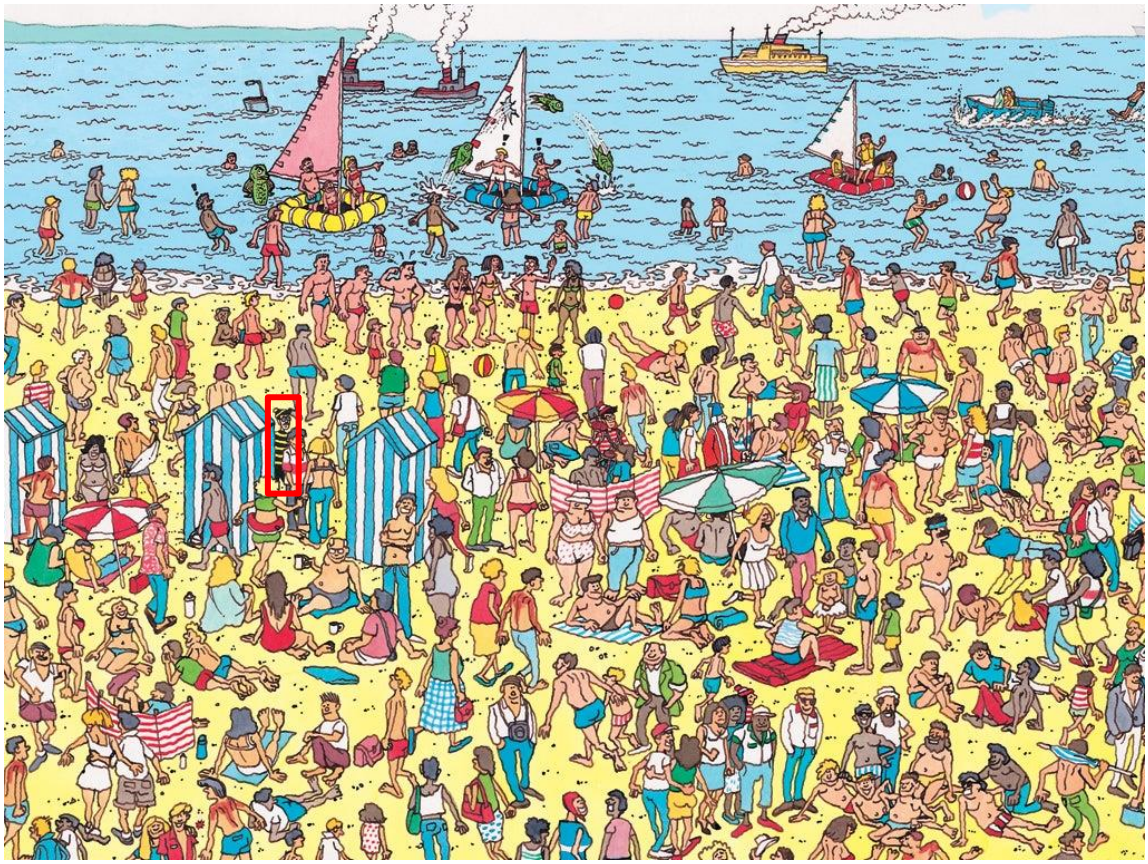

Figura 1: Escenari d'exemple extret de "Where's Wally?". L'Odlaw s'ha indicat amb un rectangle vermell

Cal notar que els personatges de la saga es distingeixen per endur una vestimenta característica. L'Odlaw —antagonista de Wally—, per exemple, sempre va vestit amb un jersei de ratlles horitzontals negres i grogues, uns texans negres, olleres i un gorro i sabates també de ratlles negres i grogues. Així que podem utilitzar els nostres coneixements de processament morfològic per identificar, mitjançant una espècie de mapa tèrmic, les zones de la imatge on hi ha franges negres i grogues. Aquesta és la idea essencial de l'algoritme que desenvoluparem a continuació.

Construcció d'un algoritme per trobar l'Odlaw

1. Llegint la imatge

Primer, llegim la imatge RGB i extraiem els seus components R, G i B per després poder-la filtrar i convertir-la a un espai de colors on sigui *còmode* detectar el color groc, com l'espai HSV. Recordem (ja havíem parlat d'aquest model a la sessió 2) que en l'espai HSV, la tonalitat es representa com un angle entre 0 i 360°, de manera que cadascun dels valors es correspon a un color. El groc, per exemple, té un angle d'aproximadament 55°.

```
I = imread('Wally.png');
R = I(:,:,1);
G = I(:,:,2);
B = I(:,:,3);
```

2. Eliminant el soroll de la imatge i extraient-ne el *Hue*

Aplicarem un filtre de mediana a cada canal de color per eliminar el soroll de la imatge. Recordem que el filtre de mediana calcula el valor d'un píxel de la imatge ordenant els valors de cada píxel de menor a major en el veïnatge que definim i seleccionant el valor en la posició intermèdia. Observem que, a mesura que augmentem la mida del veïnatge, perdem detalls de l'escena. Notem que el propòsit de l'algoritme és detectar un cert patró de línies horitzontals, així que té sentit que definim un veïnat unidimensional. En MATLAB, farem servir un veïnat $[m, n]$ amb una sola fila i un nombre de columnes no excessivament més gran que el nombre de píxels que fa d'ample una samarreta de l'escena. El veïnat usat en aquest codi ha estat: $[1,3]$.

```
% Apliquem filtrat a cada component RGB
Rm1 = medfilt2(R,[1 3]);
Gm1 = medfilt2(G,[1 3]);
Bm1 = medfilt2(B,[1 3]);
% Tornem a composar la imatge original
I(:,:,1) = Rm1; I(:,:,2) = Gm1; I(:,:,3) = Bm1;
% Convertim a espai HSV i extraiem el Hue
HSV = rgb2hsv(I);
Hm = HSV(:,:,1);
```

3. Detectant el color groc de la imatge

A continuació extraurem les zones grogues de la imatge. Com havíem anunciat anteriorment, el groc té un valor d'uns 55° en l'espai HSV, de manera que podem extreure les regions grogues de l'escena mitjançant la següent sentència lògica (notem l'ús de 5° de tolerància):

```
GROC = (Hm >= ((55-5)/360) & Hm <= ((55+5)/360));
```

Una inspecció del resultat ens confirma que, efectivament, estem extraient les regions grogues de la imatge. Podem veure que la major part de l'escena és sorra, per tant abunda el color groc.

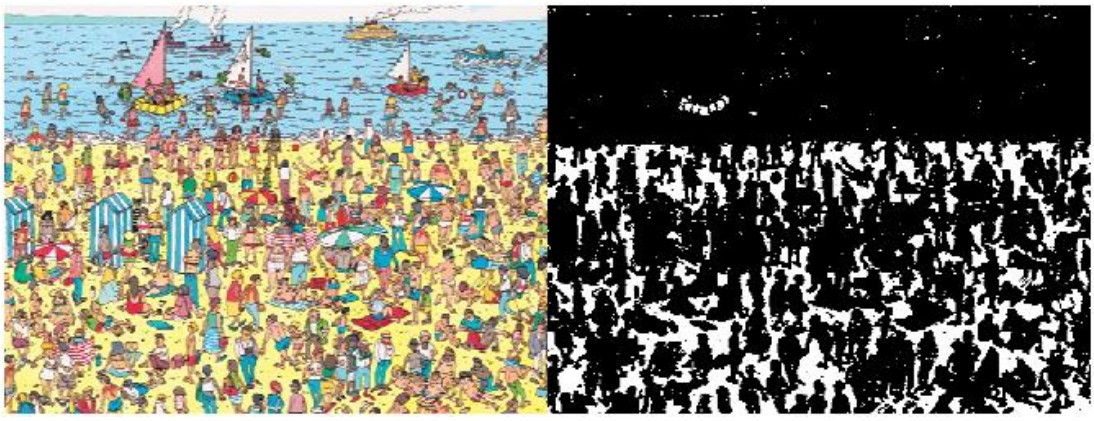

Figura 2: Imatge original filtrada (esquerra) junt amb la imatge de grocs a partir de la component Hue (dreta)

4. Detectant el color negre de la imatge

Per extreure les regions negres de l'escena, simplement seleccionarem els píxels de la imatge I que tinguin valors que estiguin per sota d'un cert llindar, que hem fixat en 50 (recordem que els valors dels components van des de 0 fins a 255), per a tots tres canals de color: R, G i B.

Com s'intueix a partir de la imatge original, les regions negres del dibuix són molt menys abundants que les grogues, i així es pot observar a la *Figura 3*, on mostrem la imatge $NEGRE$.

```
[f,c] = size(Hm);
NEGRE = zeros(f,c);
for i=1:f
    for j=1:c
        NEGRE(i,j) = ((I(i,j,1) < 50) & (I(i,j,2) < 50) & (I(i,j,3) < 50));
    end
end
```

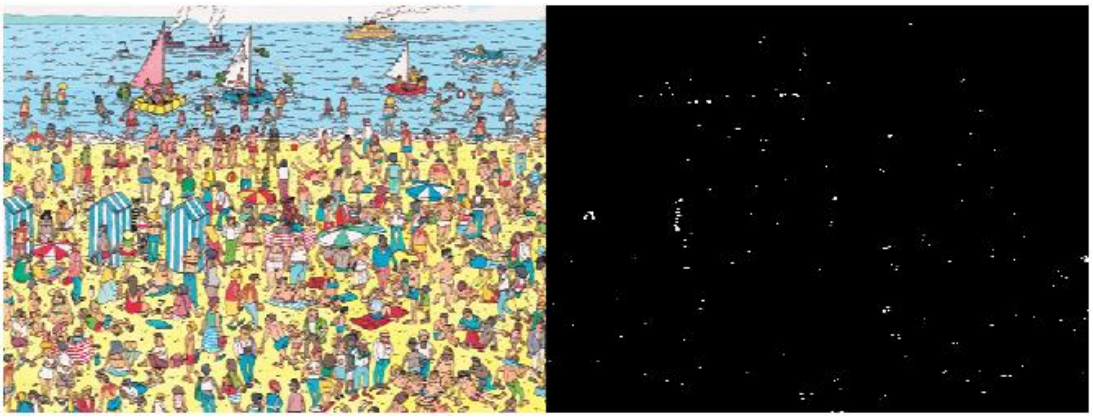

Figura 3: Imatge original (esquerra) junt amb la imatge de negres a partir dels components RGB baixos (dreta)

5. Detectant patrons de línies grogues horitzontals

Aquí és on comencem a fer servir processament morfològic. Per detectar línies horitzontals grogues podem fer servir una operació de *open* amb un element estructurant unidimensional, seguint la tònica del veïnat que s'ha fet servir amb el filtrat de soroll. Recordem que l'*opening* consisteix en realitzar una dilatació δ d'una erosió ε amb un mateix element estructurant: $\gamma_{SE3} = \delta_{SE3}|\varepsilon_{SE3}(GROC)|$, que podem implementar en MATLAB amb aquestes dues línies:

```
SE3 = strel('line',3,0);
GROCm = imopen(GROC,SE3);
```

De manera que farem servir un element estructurant $SE3$, que serà una recta horitzontal de mida 3 i inclinació 0, per a què així només es mantinguin els components grocs de la imatge que representin línies horitzontals d'almenys 3 píxels de longitud, com es mostra a la *Figura 4*.

Figura 4: Resultat de fer un open sobre GROC amb SE3. Notem la prevalença de línies horitzontals

6. Detectant patrons de línies negres horitzontals

Apliquem el mateix procediment mostrat al pas anterior, però sobre la matriu $NEGRE$ (*Fig. 5*).

```
NEGREm = imopen(NEGRE,SE3);
```

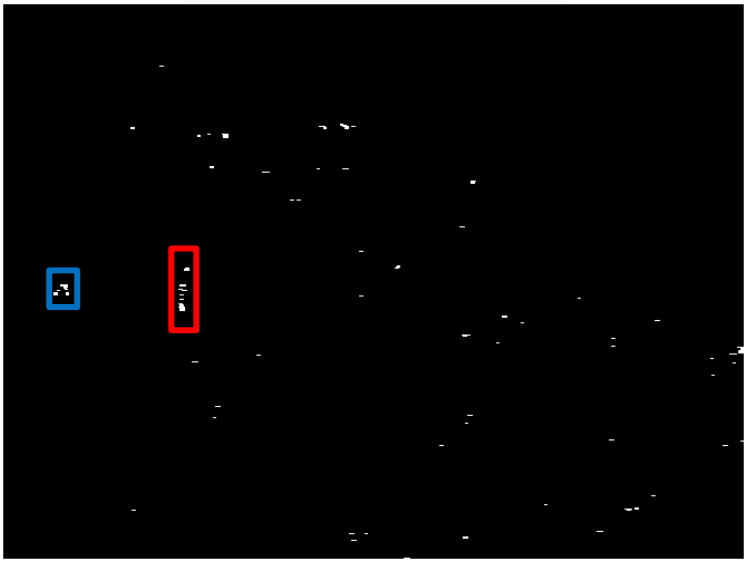

Figura 5: Resultat de fer un open sobre NEGRE amb SE3. Si ens fixem ja podem intuir on és l'Odlaw (en vermell)

7. Ajuntant línies properes i eliminant línies aïllades

A partir de la *Figura 5* es fàcil veure el que volem fer. La nostra intenció és, d'una banda, *ajuntar* les línies negres i grogues per a què es superposin entre si, i així poder extreure la intersecció d'ambdues regions de línies, el que ens indicarà on és el jersei de l'Odlaw. D'altra banda, volem eliminar les línies aïllades de la imatge, doncs no desitgem mostrar a la solució fragments de l'escena on hi ha elements grocs i negres irrellevants per al problema com, per exemple, el cabell fosc de la dona en *bikini* blanc de l'esquerra de la imatge (ubicació marcada en blau a la *Fig. 5*).
Hi ha diferents maneres de complir aquests dos propòsits. El que jo he fet (personalment crec que és bastant poc elegant) és utilitzar, primer, un element estructurant *disk* de radi 3 per fer una operació *imclose* i així tancar les línies horitzontals que es troben a prop. Després, faig una operació d'erosió (aquí és on s'aprecia la falta d'elegància...) per eliminar les línies aïllades de l'escena, aquesta vegada mitjançant un element estructurant *disk*, però de radi 1.
El resultat d'aplicar aquestes dues operacions sobre les imatges de les figures anteriors es mostra a continuació. L'efecte de l'eliminació de zones aïllades és molt més visible en la regió fosca, ja que les regions grogues són molt més denses, però tot i així encara queda algun *residu*.

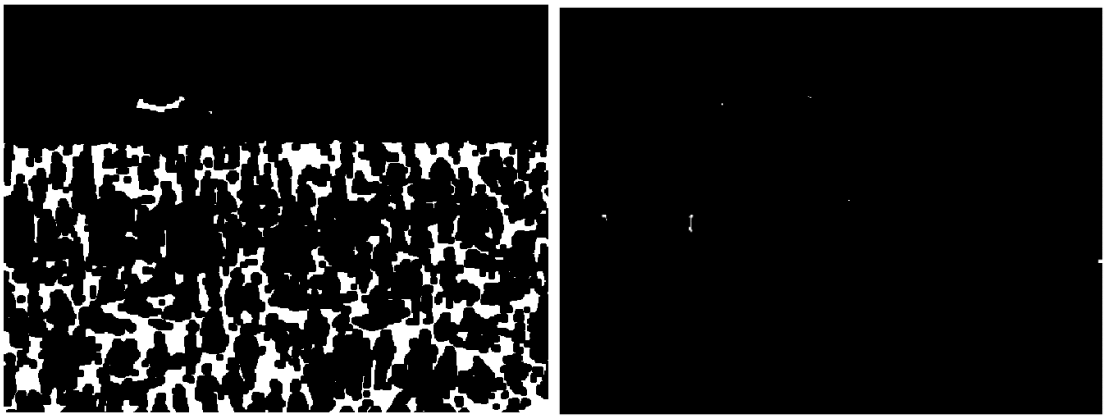

Figura 6: Resultat del processat morfològic de GROC (esquerra) i resultat del proc. morfològic de NEGRE (dreta)

```
GROCm = imclose(GROCm,strel('disk',3));
GROCm = imerode(GROCm,strel('disk',1));
NEGREm = imclose(NEGREm,strel('disk',3));
NEGREm = imerode(NEGREm,strel('disk',1));
```

8. Combinant els dos resultats i mostrant el resultat final

Si el que hem fet fins ara és (més o menys) correcte, la intersecció de les dues imatges de la *Figura 6* mitjançant una *and* lògica hauria de produir una única regió il·luminada de blanc, que permetés ubicar el jersei de l'Odlaw. Això és, efectivament, el que obtenim quan fem la intersecció de les imatges que hem processat. Amb el propòsit de mostrar en vermell els píxels de la imatge resultant d'aquesta intersecció, RES, sobre la imatge I_gray —convertida a escala de grisos— guardarem en un vector, pos, la posició dels píxels il·luminats de RES, com es mostra a continuació (notem que el resultat s'ha dilatat per incrementar la seva visibilitat).

```
RES = GROCm&NEGREm;
RESm = imdilate(RES,strel('disk',2));
imshow(RESm)
k = 1;
for i=1:f
    for j=1:c
        if RESm(i,j) == 1
            pos(k,1) = j;
            pos(k,2) = i;
            k = k + 1;
        end
    end
end
```

Finalment, farem ús de la funció *insertMaker*, de MATLAB, per marcar en color vermell els píxels guardats a pos, com es suggereix a continuació.

```
Result = insertMarker(I_gray,pos,'square','Size',1,'Color',"red");
```

El resultat d'aplicar aquest algoritme a la imatge inicial es mostra a la *Figura 7-1*. Com podem veure, hem trobat l'Odlaw! La *Figura 7-2* mostra una execució de l'algoritme amb altra imatge.

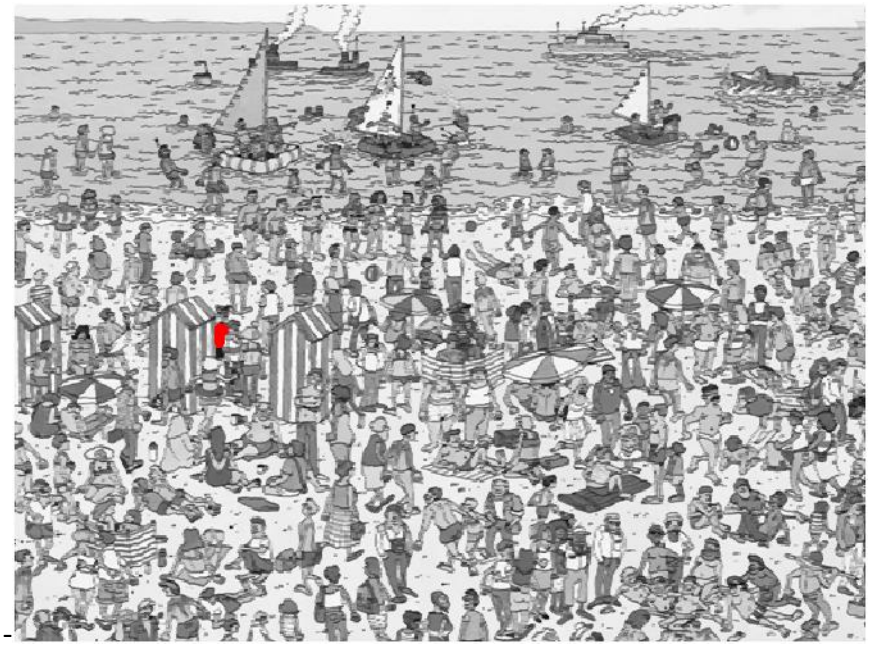

Figura 7-1: Resultat de l'algoritme sobre la imatge inicial, en escala de grisos. Com veiem, hem trobat l'Odlaw!

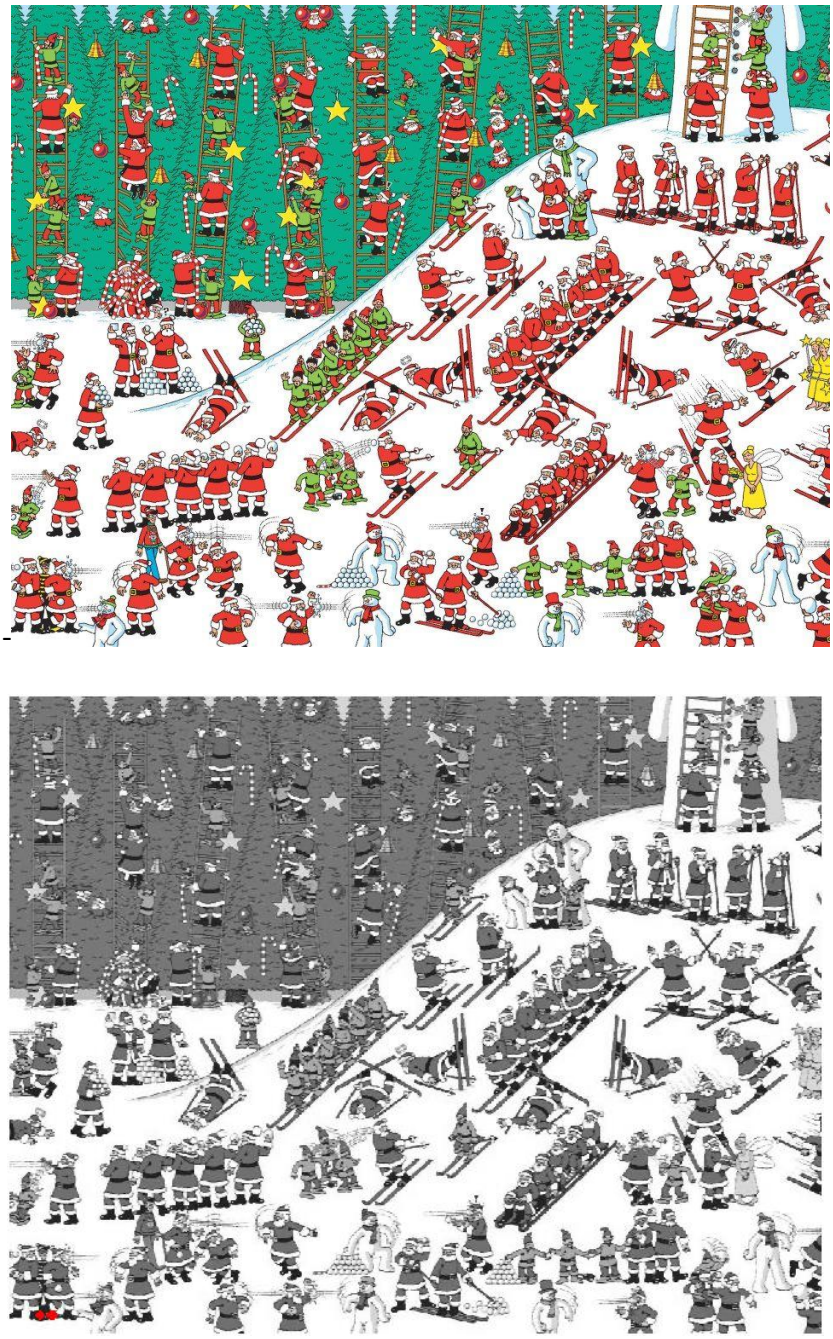

Figura 7-2: Altra imatge de la saga "Where's Wally?" (amunt) junt amb el resultat de l'algoritme sobre la nova imatge, en escala de grisos (avall). Notem que aquesta vegada el que detectem són les sabates de l'Odlaw, ja que són més visibles en aquesta escena que el seu jersei, i també són, lògicament, negres i grogues!