VISIÓ PER COMPUTADOR: EXERCICI 6

En aquest exercici dissenyarem un programa per treballar sobre d'una imatge capturada amb un telèfon mòbil, que serà un mètode per capturar imatges la il·luminació del qual no està gaire controlada. Treballarem binaritzacions globals i locals, així com segmentació per agrupament de píxels en imatges un cop ja binaritzades.

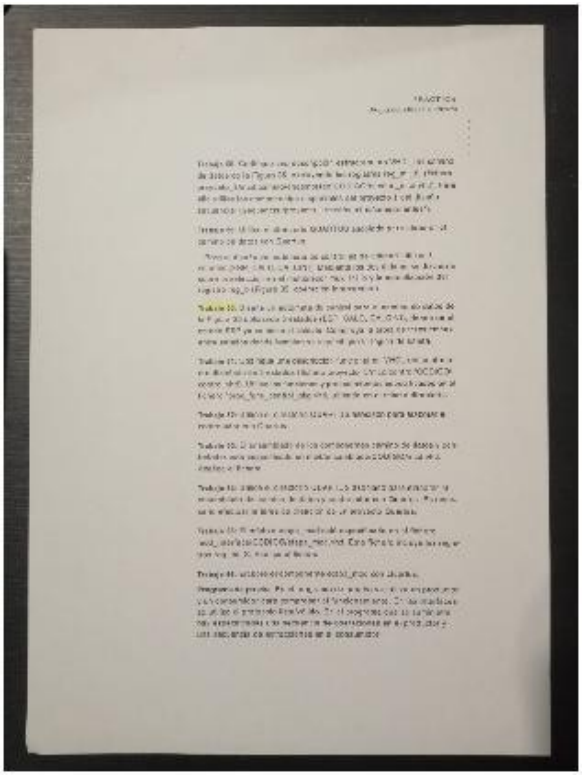

Figura 1: Imatge de la pàgina capturada amb la càmara del mòbil.

En primer lloc, generem un histograma de la imatge capturada amb el telèfon mòbil per estudiar quin seria el llindar més adequat per binaritzar-la:

```
I = imread('imatge.jpg');
h = imhist(I);
plot(h);
```

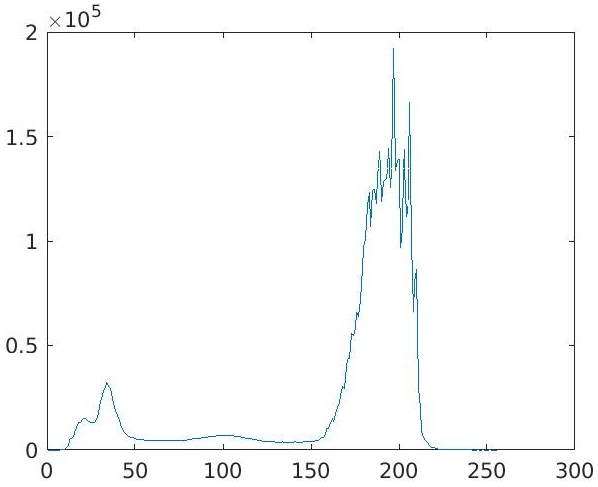

Figura 2: Histograma resultant.

Tal com es pot observar a la figura 2, tenim alguns valors a menys de 50 i moltíssims valors al voltant dels 200. Una possibilitat podria ser **un valor equidistant (al voltant de 100)**, tot i que vistos els pesos d'una àrea i l'altre, el més raonable seria pensar que **un valor més a prop dels 200,** que no pas dels 50, seria el més adequat.

Tenint de referència que el llindar més adequat seria tenir **prop del 80% dels píxels a blanc,** comprovarem per un cas i l'altre tot comparant el percentatge i la qualitat de la imatge binaritzada resultants.

```
% Per binaritzar la imatge primer la passem a
% imatge en escala de grisos:
Gray = rgb2gray(I);

BW = Gray > X; % Sent X: 100 o 150.

% Tenim nnz que és el nombre de píxels blancs i
% numel el nombre de píxels totals:
RatioBlanc = nnz(BW) / numel(BW);
```

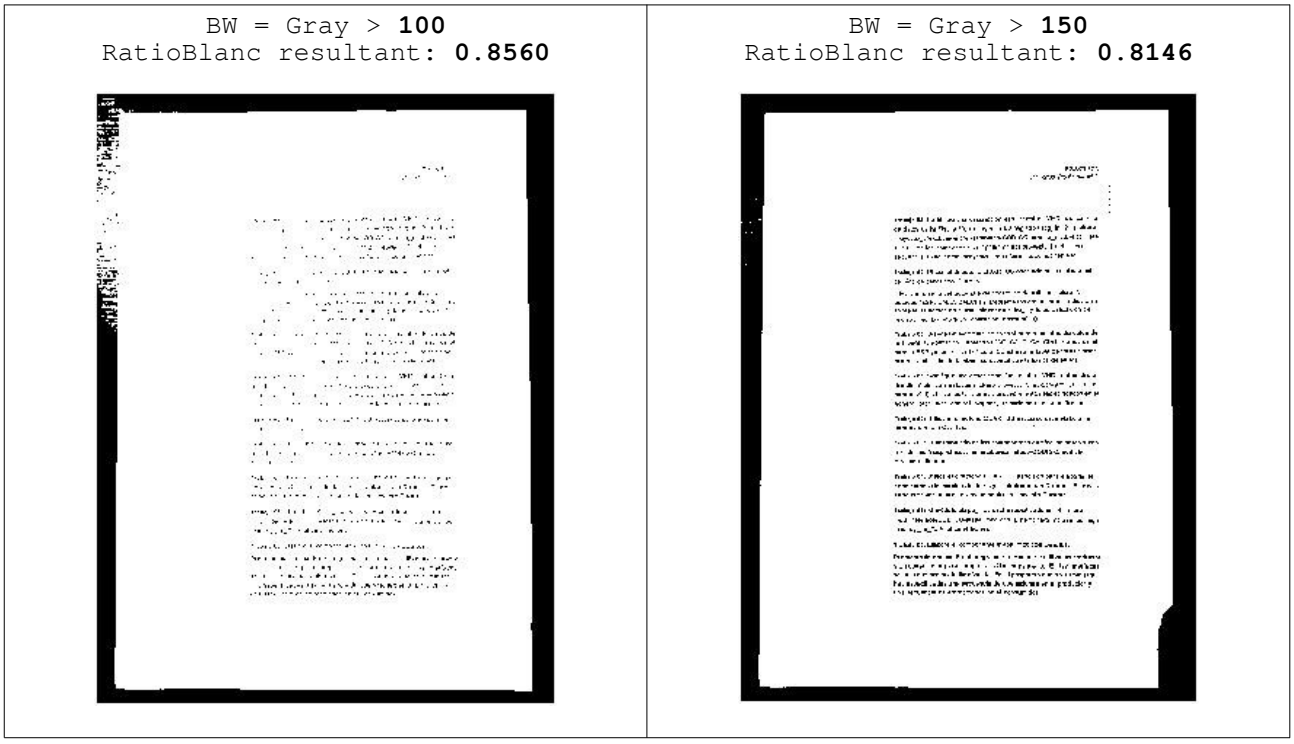

Figura 3: Comparació dels resultats per uns llindars de binarització de 100 i de 150.

Seguidament, vistos els resultats de la figura 3, **elegim un llindar de binarització de 150** perquè ens acosta més al valor del 80% dels píxels blancs respecte del total,de la mateixa manera que ens garanteix una millor qualitat d'imatge. Procedirem ara a retallar la imatge:

```
[rows, columns] = find(BW);
row1 = min(rows);
row2 = max(rows);
col1 = min(columns);
col2 = max(columns);
BW2 = BW(row1:row2, col1:col2);
```

A continuació, emprant la comanda `colfilt` farem una binarització local, utilitzant una finestra de 25x8 que anirà lliscant al llarg de la imatge aplicant la funció `binaritzLocal` per cadascuna de les finestres. S'han establert 25 píxels d'alçada perquè corresponen amb l'espai d'una línia incloent les separacions

verticals, que a més és múltiple amb l'espai buit a sobre del text. La matexa regla s'aplica amb els 8 píxels d'amplada, però en aquest cas pel que fa a un sol caràcter.

```
binLocal = colfilt(Gray,[25 8],'sliding', @binaritzLocal);

% La funció computa la mitjana de la finestra
% i posa a 1 els píxels que són 8 nivells de
% gris per sobre de la mitjana, amb aquest
% valor s'han obtingut els millors resultats:
function[y] = binaritzLocal(x)
      [f c] = size(x);
      mitjana = mean(x, 'all');
      xx = x(ceil(f/2),:);
      y = (xx(:,:) > mitjana + 8);
end
```

En total, s'observen 4013 *objectes* amb la funció `bwconncomp` fruit de la binarització local. Eliminem també els píxels salvatges.

```
maj_bin = bwmorph(binLocal, 'majority');

labeledImage = bwconncomp(maj_bin);
```

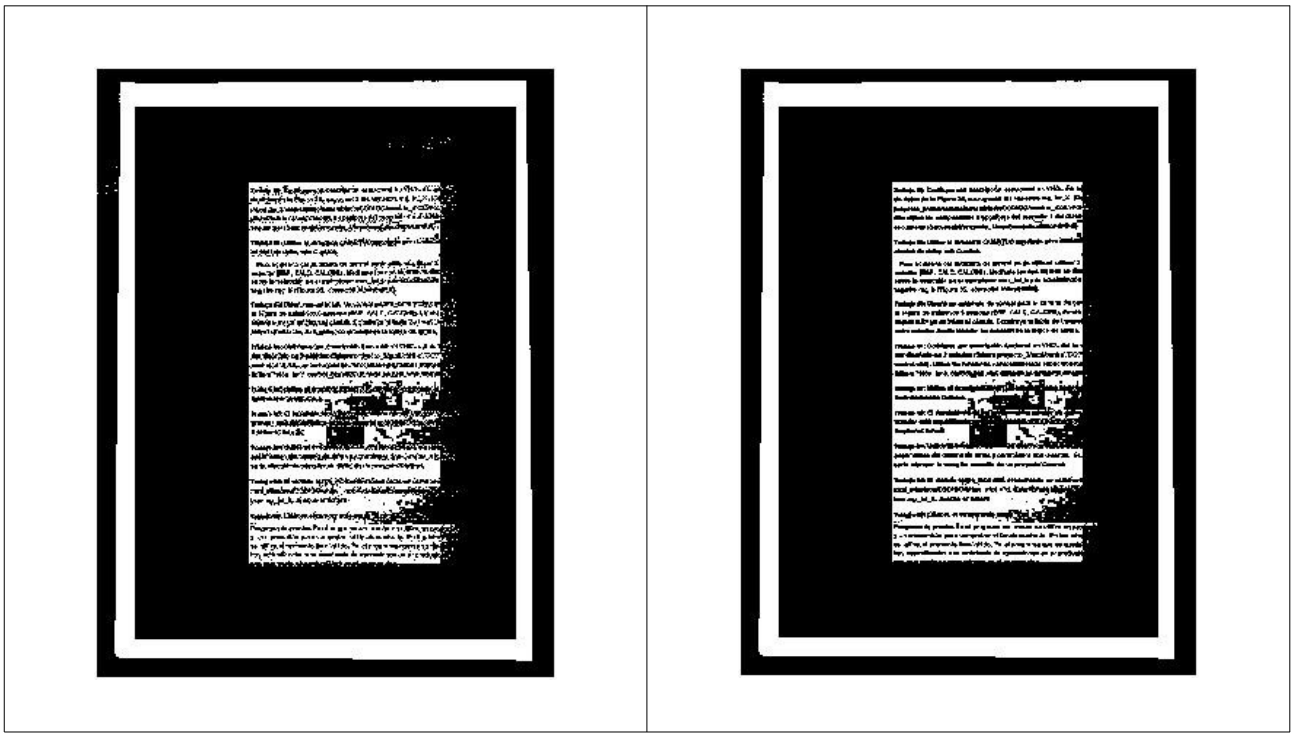

Figura 4: Comparació de la imatge binaritzada localment (esquerra) i de la versió un cop eliminats la major part dels píxels salvatges (dreta).

Finalment, s'intenta emmarcar en vermell els caràcters detectats:

```
labeledImage = bwconncomp(maj_bin);
measurements = regionprops(labeledImage,'BoundingBox');

for k = 1 : length(measurements)
      bb = measurements(k).BoundingBox;
      rectangle('Position',
      [bb(1),bb(2),bb(3),bb(4)],'EdgeColor','red','LineWidth',1 );
end
```

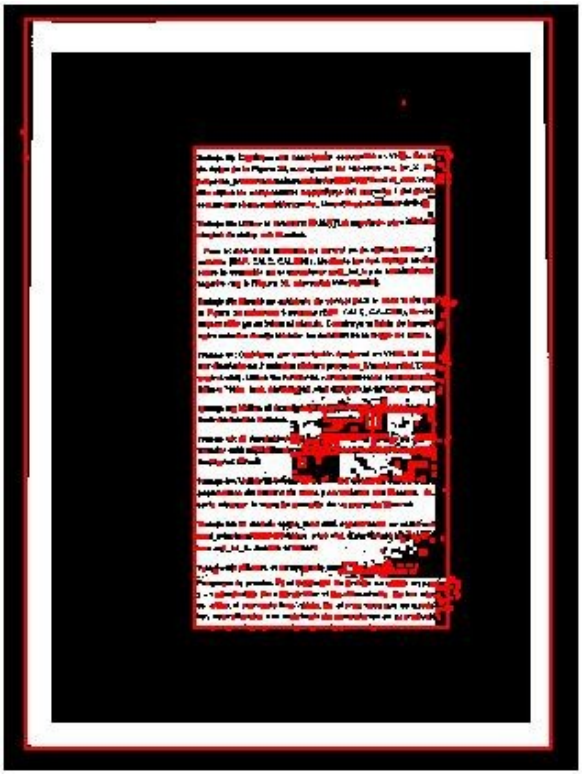

Figura 5: Resultat final d'emmarcar els caràcters detectats, moltes lletres no han estat correctament detectades i, en canvi, s'han detectat objectes que no són lletres.

S'ha provat fent servir una finestra de 50x8 (és a dir, dues files de text d'alçada), però el resultat ha estat pràcticament idèntic.