Detecció de Matrícules

Table of Contents

CODI

Imatges Inicials

```
close all; clc
w = warning ('off','all');

f=dir(['cars' '/*.jpg']);
files={f.name};
numcotxes = numel(files);
names = convertCharsToStrings(files);
for k=1:numcotxes
    names(k) = erase(names(k),".jpg");
    names(k) = erase(names(k),".JPG");
    names(k) = erase(names(k),"_");
end
im_ors=cell(1,numcotxes);
for k=1:numcotxes
  im_ors{k}=imread(files{k});
end
```

Detecció de Potencials Matrícules

```
% Aquesta funció detecta aquelles regions de la imatge que són potencials
% matrícules

function immat = F_PotencialsMatricules(imor)

    % Passem a Blanc i negre i obtenim edges
    imgray = rgb2gray(imor);
    imbin = imbinarize(imgray);
    imedges = edge(imgray, 'prewitt');
    imedges(200,:) = 0;

    % Seleccionem les zones tancades
```

```
ee = strel('rectangle',[5,10]);
immat = imdilate(imedges,ee);
immat = imfill(immat,"holes");
ee = strel('line',20,0);
immat = imerode(immat,ee);
ee = strel('line',10,90);
immat = imerode(immat,ee);

% Eliminem els bordes de les zones massa grans
ee = strel('rectangle',[200,200]);
imgrans = imopen(immat,ee);
imgrans = imreconstruct(imgrans,immat);
imgransbordes = imdilate(imgrans,strel('disk',10));
imgransbordes = imgransbordes - imgrans;
imedges = logical(imedges - (imedges .* imgransbordes));

% Repetim selecció de zones tancades
ee = strel('rectangle',[5,10]);
immat = imdilate(imedges,ee);
immat = imfill(immat,"holes");
ee = strel('line',20,0);
immat = imerode(immat,ee);
ee = strel('line',10,90);
immat = imerode(immat,ee);

% Eliminem les línies verticals o horitzontals molt finetes
immat = imopen(immat,strel("rectangle",[15,1]));
immat = imopen(immat,strel("rectangle",[1,15]));

% Separem blobs propers
td = bwdist(~immat);
td = imhmax(td,4);
segm = watershed(-td);
immat = ~(~immat | (segm == 0));

% Eliminem les zones no rectangulars
ee = strel('rectangle',[15,50]);
imnorectangles = imopen(immat,ee);
imnorectangles = imreconstruct(imnorectangles,immat);

immat = logical(immat .* imnorectangles);

% Eliminem les zones sense línies verticals o horitzontals
ee = strel('rectangle',[5,1]);
imlines_v = imopen(imedges,ee);
imlines = imreconstruct(imlines_v,immat);
ee = strel('rectangle',[1,5]);
imlines_h = imopen(imedges,ee);
imlines = imreconstruct(imlines_h,imlines);

immat = logical(immat .* imlines);

% Eliminem els blobs en contacte amb els bordes
[n, m] = size(imbin);
```

```
    imbordes = logical(zeros(n,m));
    imbordes(1:3,:) = 1;
    imbordes(n-2:n,:) = 1;
    imbordes(:,1:3) = 1;
    imbordes(:,m-2:m) = 1;
    imtouching = imreconstruct(imbordes,immat);

    immat = logical(immat - imtouching);
end
```

Eliminat de no-matricules

```
% Eliminem moltes matricules fals positives que hem detectat
% en el pas anterior

function immat = F_EliminarNoMatricules(imor,immator)
    [n, m] = size(immator);
    immat = logical(zeros(n,m));

    Iprops=regionprops(immator,'BoundingBox','Area', 'Image');
    count = numel(Iprops);

     for i=1:count
        boundingBox=Iprops(i).BoundingBox;
        region = imcrop(rgb2gray(imor),boundingBox);

        ridncalv = @ridncalv;
        regbin = ~imbinarize(region,ridncalv(region));

        regionmat = imcrop(immator,boundingBox);
        regbin = regionmat.*regbin;

        % Eliminem les regions amb < 5 regions connexes,
        % després de filtrar estructures petites
        ee = strel("rectangle",[7,1]);
        regero = imerode(regbin,ee);
        regrec2 = imreconstruct(regero,regbin);

        ee = strel("rectangle",[1,2]);
        regero = imerode(regrec2,ee);
        regrec2 = imreconstruct(regero,regrec2);

        regrec2 = imclose(regrec2,strel("rectangle",[2,1]));
        regrec2 = imclose(regrec2,strel("rectangle",[1,2]));

        regrec2 = padarray(regrec2,[1,1],1);
        cc = bwconncomp(regrec2);

        % Eliminem les regions amb proporcions inadequades
        [n2, m2] = size(regbin);
        ratio = abs(n2/m2 - 110/520)*100;
```

```
        %figure, imshow(regrec2),title(["cc", cc.NumObjects, "ratio", ratio]);

        % Afegim la potencial matricula
        if cc.NumObjects > 4 && ratio < 35
            X = boundingBox(1); Y = boundingBox(2);
            W = boundingBox(3); H = boundingBox(4);
            immat(Y:Y+H, X:X+W) = regionmat;
        end

    end
end
```

Binaritzat Riddler i Calvard

```
%Mètode iteratiu de Riddler i Calvard
function treshold = ridncalv(im)
    % El llindar inicial és la mitjana de valors de grisos
    currentT = mean(im(:));
    upT = im(im > currentT);
    belowT = im(im <= currentT);
    umean = sum(upT)/size(upT,1);
    bmean = sum(belowT)/size(belowT,1);
    % El llindar posterior és la mitjana de la suma dels valors que estan
    % per sobre del llindar inicial i de la suma dels valors que estan per
    % sota de llindar inicial
    nextT = (umean+bmean)/2;
    maxIter = 1000;
    i = 1;
    error = 0.0001;
    % Iterem fins obtenir llindar que compleixi amb el marge de error
    while (abs(nextT - currentT) > error) && (i < maxIter)
        currentT = nextT;
        upT = im(im > currentT);
        belowT = im(im <= currentT);
        umean = sum(upT)/size(upT,1);
        bmean = sum(belowT)/size(belowT,1);
        nextT = (umean+bmean)/2;
        i = i + 1;
    end
    treshold = currentT/256;

end
```

RESULTATS

En vermell i verd es troben remarcades les matrícules detectades en la primera funció. En verd hem ressaltat els candidats que han sobreviscut a la segona funció

```
im_mat=cell(1,numcotxes);
```

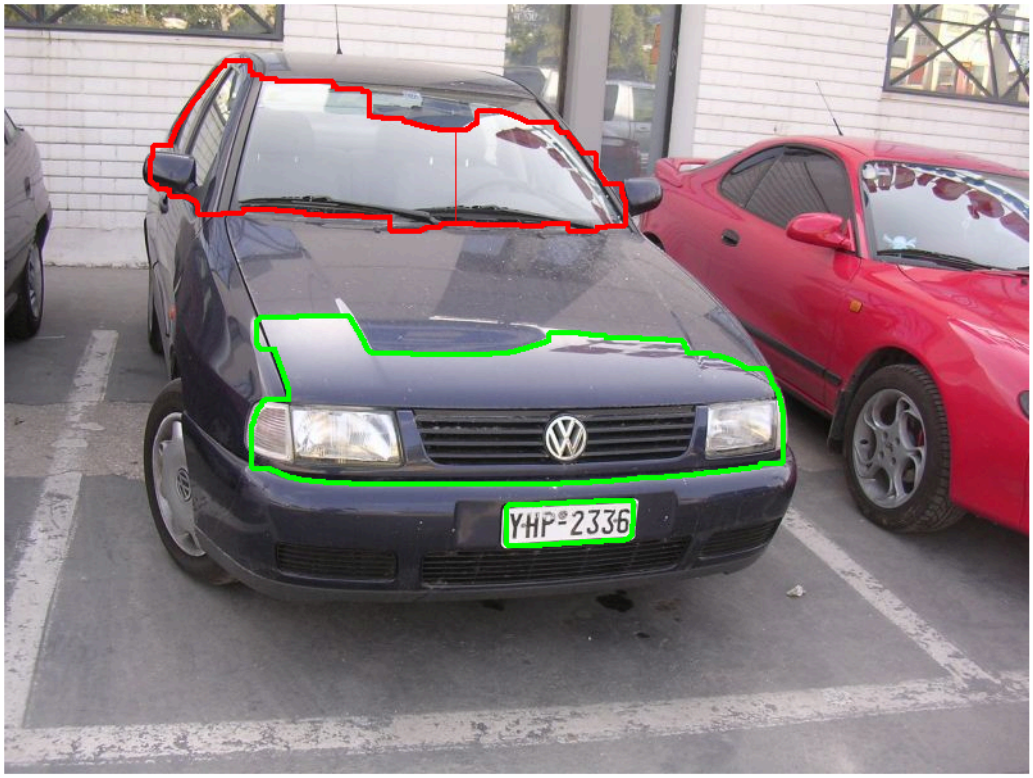
```
for k=1:numcotxes
    im_mat{k}=F_PotencialsMatricules(im_ors{k});
end

im_mat2=cell(1,numcotxes);
for k=1:numcotxes
    im_mat2{k}=F_EliminarNoMatricules(im_ors{k},im_mat{k});
end

for k=1:numcotxes
    imres = im_ors{k};

    immat2 = imdilate(im_mat{k},strel("disk",5)) - im_mat{k};
    imres(:,:,3) = imres(:,:,3) .* uint8(~immat2);
    imres(:,:,2) = imres(:,:,2) .* uint8(~immat2);
    imres(:,:,1) = imres(:,:,1) + uint8(immat2)*256;

    immat2 = imdilate(im_mat2{k},strel("disk",5)) - im_mat2{k};
    imres(:,:,3) = imres(:,:,3) .* uint8(~immat2);
    imres(:,:,1) = imres(:,:,1) .* uint8(~immat2);
    imres(:,:,2) = imres(:,:,2) + uint8(immat2)*256;

    figure, imshow(imres), title(names(k));
end
```

DSCN0410

DSCN0412

DSCN0413

DSCN0414



DSCN0415

DSCN0416

DSCN0418

IMG0378

IMG0379

IMG0380

IMG0381

IMG0382

IMG0383

IMG0384

IMG0385

IMG0386

IMG0387

IMG0388

IMG0389

IMG0392

IMG0393

IMG0394

IMG0395

IMG0396

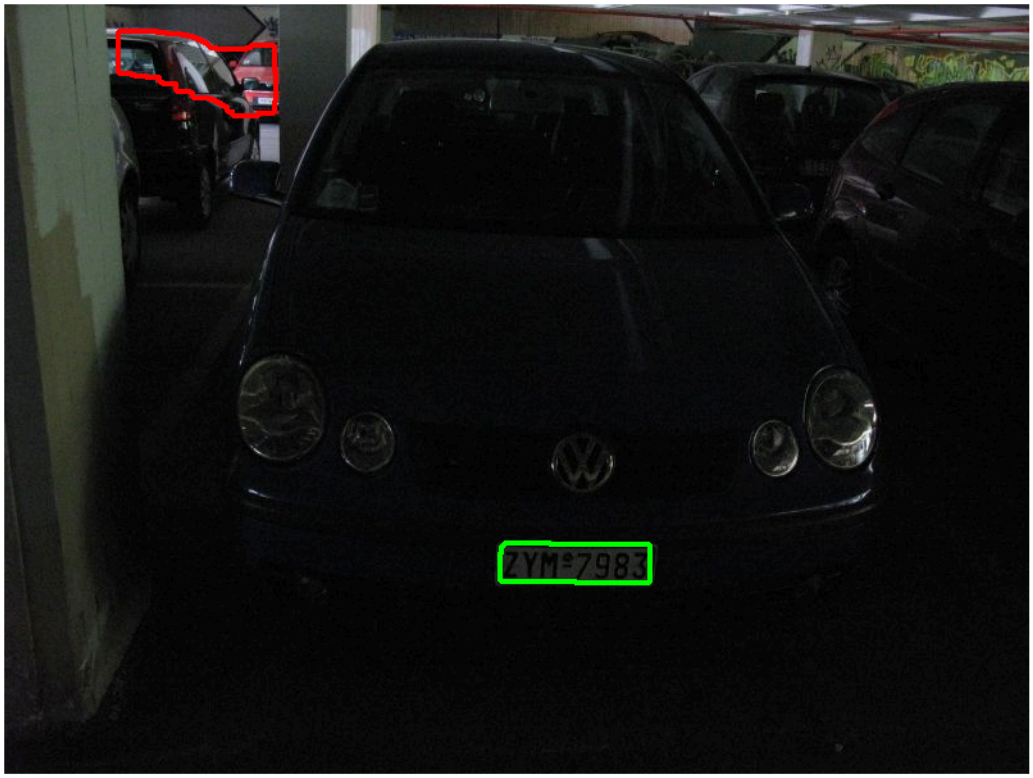
IMG0414

IMG0415

IMG0420

IMG0421

IMG0443



IMG0446

IMG0447

IMG0448

IMG0449

IMG0450

IMG0452

IMG0453

IMG0454

IMG0455

IMG0456

IMG0457

IMG0460

IMG0461

IMG0462

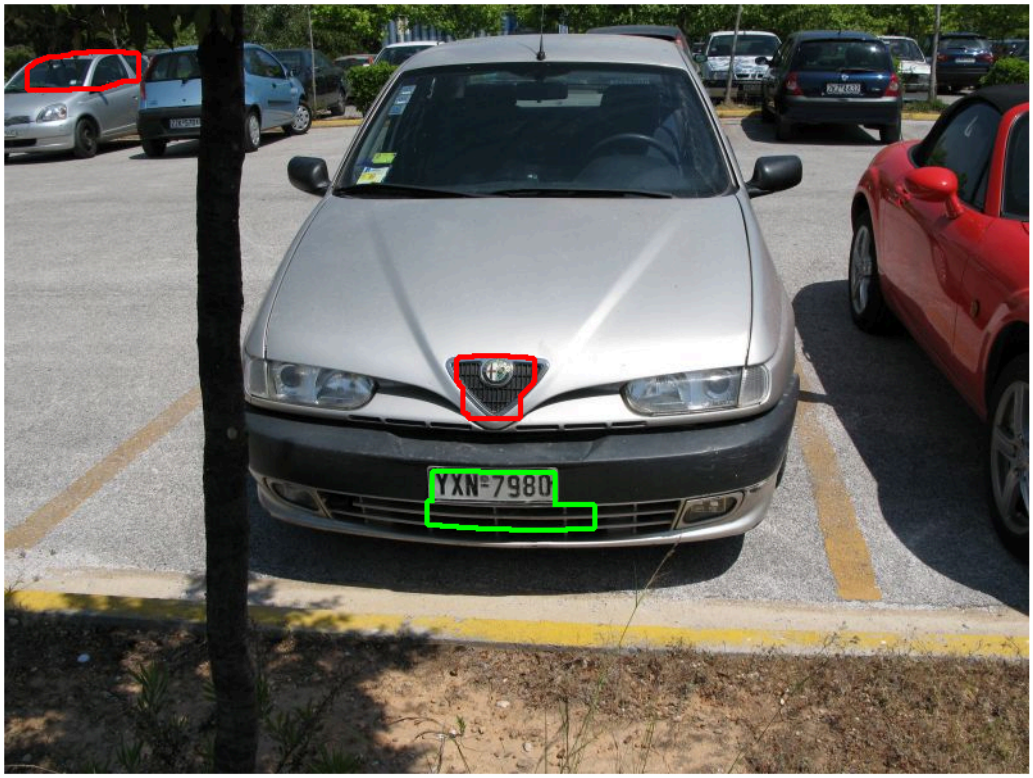
IMG0463

IMG0464

IMG0465

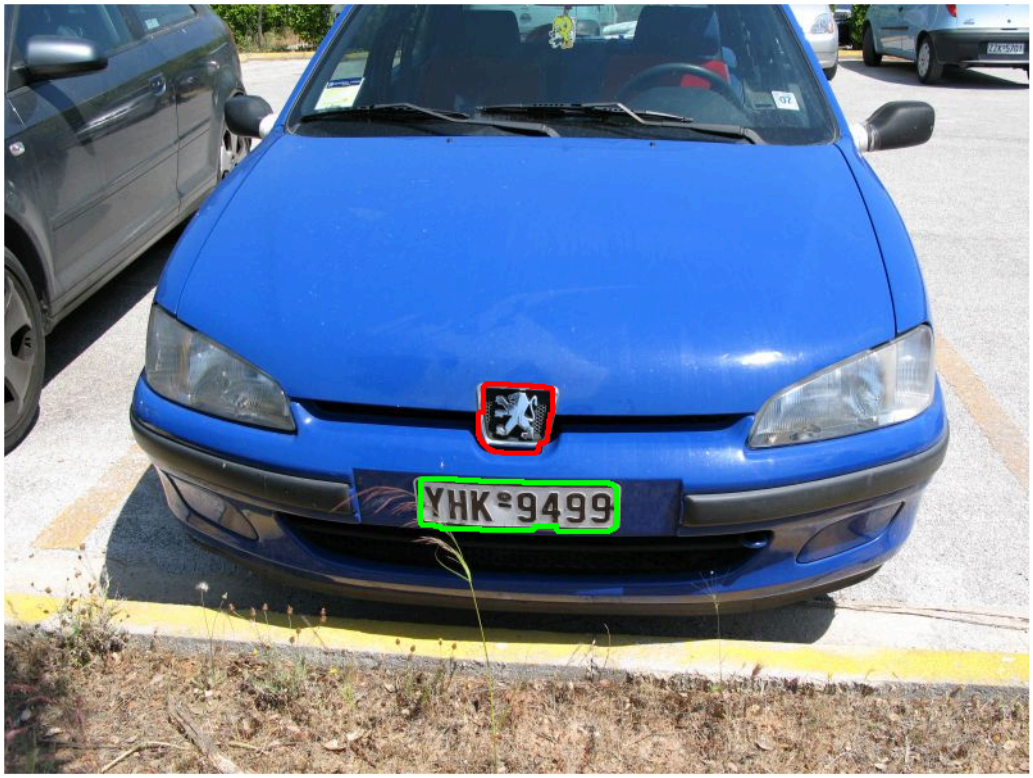
IMG0466

IMG0467



IMG0468

IMG0469



IMG0470

IMG0471

IMG0472

IMG0473

IMG0474

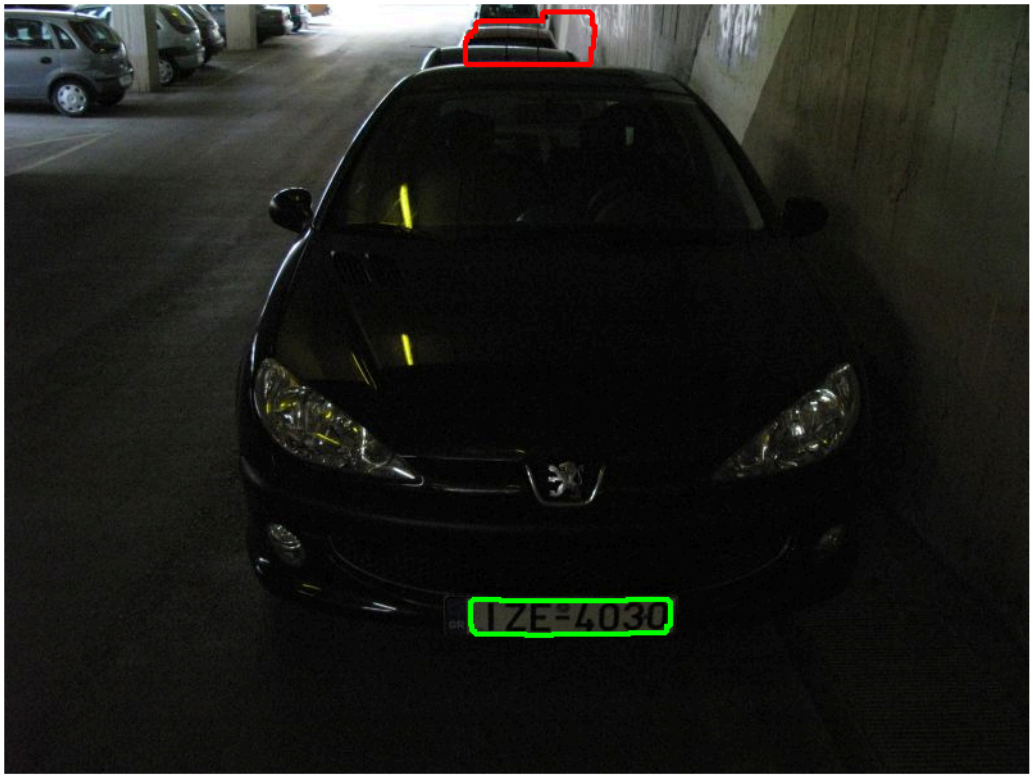
IMG0475

IMG0476

IMG0477

IMG0478

IMG0479



IMG0480

IMG0481

IMG0482

IMG0483

IMG0484

Published with MATLAB® R2022b