Corner Detection

Daniel Barmaimon
Visual Perception - Vibot Master Degree
University of Girona

Abstract

The main purpose of this lab is to understand how to detect a corner and the problems related with the accuracy and the time to get this points.

1. Introduction

For artificial vision systems the corners are points specially important. They have a greater variation of contrast in its neighbourhood in all directions, so they could be detected from several locations for different cameras. It is quite important in cases of image reconstruction where the knowledge of these representative points could lead into very accurate models. In this lab the idea is two analyse Harris corner detection method, understand its approximation and compare with a more accurate method (non-approximated). Let's consider a small window to study the image, as it is possible to see at the Fig. 1

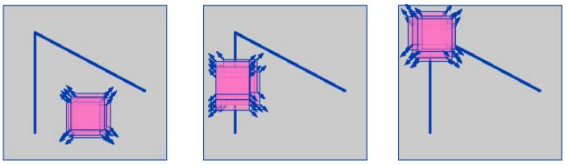

Figure 1: Flat region, edge and corner

If the windows is shifted into a flat region, small changes will be computed. If the windows is shifted through an edge, big changes will be observed in the perpendicular direction with respect to the edge. In the case of a corner, it will be a great change in all directions.

The change of intensity in an image when a windows is moved over it could be expressed mathematically with Moverec's corner detector as follows.

$$E_{x,y} = \sum_{u,v} w_{u,v} \mid I_{x+u,y+v} - I_{x,y} \mid^2 \tag{1}$$

where w is a rectangular window, and shifts could only be applied over x and y in horizontal and vertical directions.
Harris detected some issues in the application of this formula, so he solved it in a practical way. First, the approximation of the first gradients, reducing the form of E to a quadratic form.

$$E_{x,y} = Ax^2 + 2Cxy + By^2 \tag{2}$$

Later the use of a circular filter as a window (such as Gaussian), reduce substantially the noise in the image.

$$w_{u,v} = exp - (u^2 + v^2)/2\sigma^2 \tag{3}$$

And finally the reformulation of the corner measure, to be sure it will find it correctly, studding autocorrelation to check if is a flat region, an edge or a corner.

$$E(x,y) = (x,y)M(x,y)^T \tag{4}$$

where

$$M = \begin{bmatrix} A & C \\ C & B \end{bmatrix} \tag{5}$$

The equations for the trace and the determinant of a 2x2 matrix are as follows.

$$Tr(M) = A + B \tag{6}$$

$$Det(M) = AB - C^2 \tag{7}$$

As the eigenvalues should be calculated and it is computationally expensive, Harris avoided it by the following simplification.

$$R = Det - kTr^2 \tag{8}$$

2. Lab exercises

2.1. Compute the matrix E

For each pixel it will contain the smaller eigenvalue of M.

```
[n,m]=size(im);
Aux =ones(3);
Ix2w = conv2(Ix2, Aux, 'same'); % Summatory of
Iy2w = conv2(Iy2, Aux, 'same'); % numbers using
Ixyw = conv2(Ixy, Aux, 'same'); % mask of ones
E = zeros(n,m);
tic;
for i=1:n
    for j=1:m
       M = [Ix2w(i,j) Ixyw(i,j);
            Ixyw(i,j) Iy2w(i,j)];
       eigenvalues = eig(M);
       E(i,j)= min(eigenvalues);
    end
end
hold on;
time1 = toc;
subplot(1,2,1);
imshow(mat2gray(E));
title('Using matrix E');
```

2.2. Compute the matrix R

The matrix R is an approximation of M, avoiding the computation of the determinant for each pixel, with two simple assumptions. It reduces the computational time considerably at it will be demonstrated.

```
R = zeros(n,m);
k = 0.04;
tic;
R=((Ix2w.*Iy2w)-(Ixyw.*Ixyw))-k*(Ix2w+Iy2w).^2;
time2 = toc;
subplot(1,2,2);
imshow(mat2gray(R));
title('Using matrix R');
```

And if the two obtained images are shown, the result is as the one in Figure 2 The time computed for

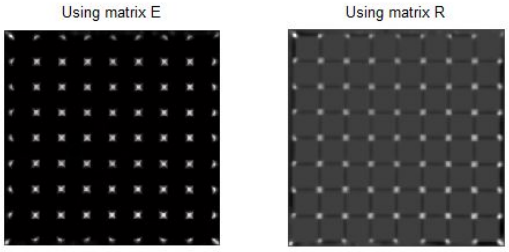

Figure 2: Images obtained for matrices E and R

the calculation of the image using the matrix E was *0.5451 s.*, and the time computed using matrix R was *0.00048932 s.*. It has to be considered that the size of the image to be analysed was 256x255 pixels, so the difference could be even greater in bigger size images.

2.3. Find 81 most salient points

In this part, the most salient points for the two cases expressed above will be shown over the images. At the beginning it was expected to obtain the points for each corner in the chessboard. The results for the two methods are displayed below, in Fig. 3

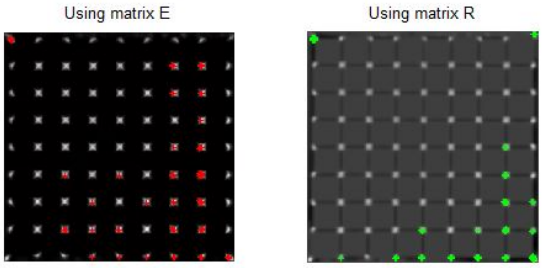

Figure 3: 81 most salient points using E and R

The points are really close to each other and that is due to the intensity of the light over the image that generates little differences between the pixels intensity. To avoid this effect it is needed to remove all pixels that are not a maximum for a given neighbourhood.

2.4. Non-maximum suppression

It was required to create a function to remove all points around the maximum, leaving only the corners. The function could be found below.

```
function[points] = nonMaxSup(E,sortedV,n,m,wSize)
points = zeros(2,81);
counter = 1;
iter = 1;
A1 = E;
border = (wSize-1)/2;
while (counter<82)
    maxValue=find((A1==sortedV(iter,1)),1,'first');
    if (isempty(sortedV(maxValue))== 0)
        [x, y]=ind2sub([n,m],maxValue);
        points(:,counter)=[y,x];
        minX = max(1, x - border);
        maxX = min(n, x + border);
        minY = max(1, y - border);
        maxY = min(m, y + border);
        A1(minX:maxX, minY:maxY)=0;
        counter = counter +1;
    end
    iter = iter +1;
end
end
```

The function has as input parameters the image with the differences of intensities, the values of the indexes of each pixel (vectorized and sorted by intensity), size of the image (to recover the position of the index) and

size of the window to be consider. The idea is to find the maximum values an store them into an array. Once this is found the neighbours will be set to zero and it will be search next maximum over the indexes of the image.

Results could be observed in the figure 4

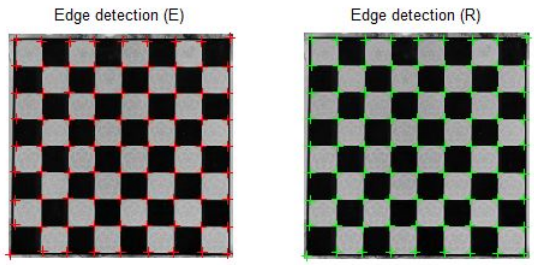

Figure 4: Corner detection using (E) and (R)

2.5. Subpixel Accuracy

The corners are not perfectly located at its place, but it is possible to measure in a more accurate way where is the maximum exactly located. This function has to fit in the best possible way with the points given by the image. The easiest function to fit with the data is a paraboloid, having the origin at the maximum value. Least squares distances or genetic algorithm could be implemented to find this function.

References

[1] 'A combined corner edge detector' - Chris Harris, Mike Stephens - Plessey Research Roke Manor, UK, 1988

[2] 'Visual Perception, Class notes', Rafael Garcia Campos, Joaqun Salvi Mas, Universitat de Girona, 2014