Computer Vision_Savitha San Markan

by SAVITHA SAN MARKAN

Submission date: 08-Feb-2023 11:48PM (UTC+0800)

Submission ID: 2009351853

File name: SAVITHA_SANMARKAN__REPORT.docx (517.53K)

Word count: 625

Character count: 4021

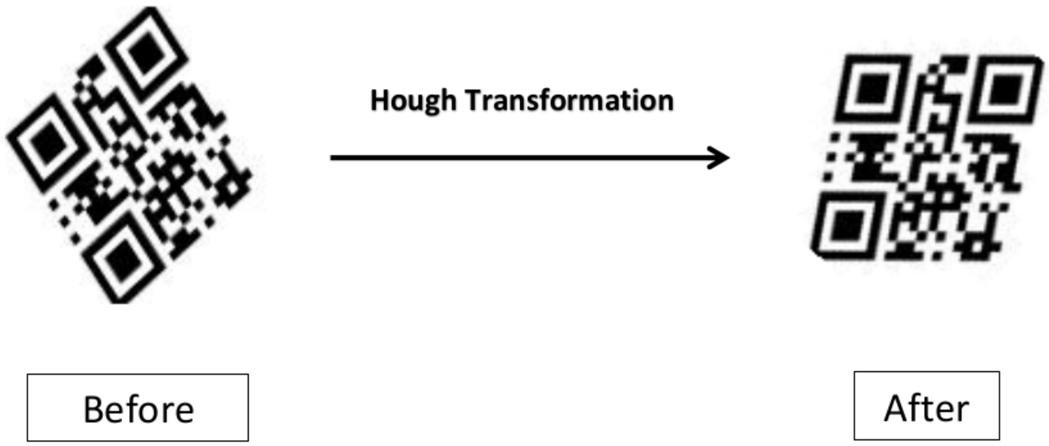
Objective:

- Able to identify a tilted QR/Bar code and straightens it by finding skew detection and correction using Hough Transform.

Problem Identification:

- Tilted QR/Bar code.

Example:



Methodology:

The methodology to find skew detection and correction using Hough Transform can be summarized as follows:

1. Pre-processing: The first step is to pre-process the image to enhance the text lines. This can include binarization, morphological operations, or filtering to remove noise and improve the edge quality.
2. Edge detection: The second stage is to find the image's edges. Several edge detection algorithms, including the Laplacian of Gaussian (LoG) filter, the Sobel operator, and the Canny edge detector, can be used to do this.
3. Line detection: After detecting edges, the next step is to detect lines in the image using the Hough Line Transform. This involves transforming the image space into the parameter space and counting the number of edge points that belong to each line.
4. Skew detection: Once the lines have been detected, the next step is to detect the skew angle of the text lines. This can be done by computing the orientation of the lines and averaging them to find the overall skew angle.
5. Skew correction: After detecting the skew angle, the next step is to correct for the skew. This can be done by rotating the image by the negative of the skew angle, or by transforming the image using a homography matrix that corrects for the skew.
6. Post-processing: The final step is to perform post-processing on the image to enhance the text lines and improve the overall quality of the image. This can include deskewing, binarization, morphological operations, or filtering to remove noise.

Codes (MATLAB)

Code for Images

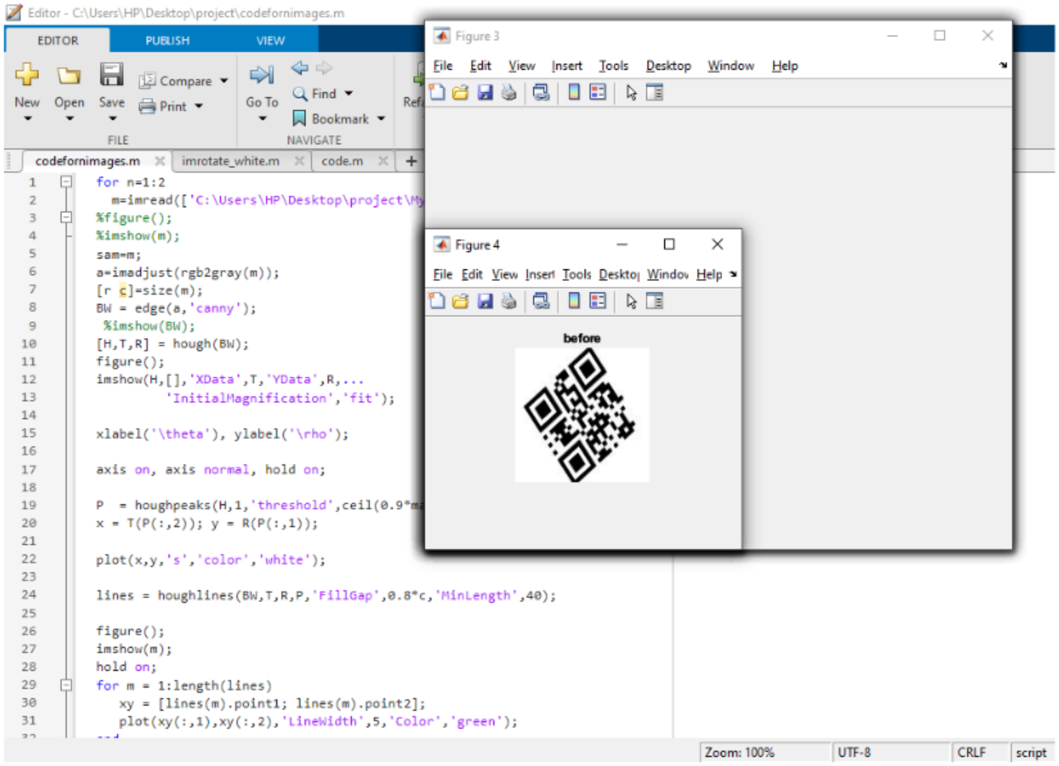
```
for n=20
m=imread(['C:\Users\HP\Desktop\project\My_Dataset\im' num2str(n) '.jpeg']);
%figure();
%imshow(m);
sam=m;
a=imadjust(rgb2gray(m));
[r c]=size(m);
BW = edge(a,'canny');
%imshow(BW);
[H,T,R] = hough(BW);
figure();
imshow(H,[],'XData',T,'YData',R,...
'InitialMagnification','fit');

xlabel('\theta'), ylabel('\rho');

axis on, axis normal, hold on;

P = houghpeaks(H,1,'threshold',ceil(0.9*max(H(:))));
x = T(P(:,2)); y = R(P(:,1));

plot(x,y,'s','color','white');

lines = houghlines(BW,T,R,P,'FillGap',0.8*c,'MinLength',40);

figure();
imshow(m);
```

```
hold on;
for m = 1:length(lines)
xy = [lines(m).point1; lines(m).point2];
plot(xy(:,1),xy(:,2),'LineWidth',5,'Color','green');
end
figure();
if(lines.theta<0)

g=imrotate_white(a,(90-abs(lines.theta)));

else

g=(imrotate_white(a,lines.theta-90));

end




figure();
imshow(sam);
title('before')
imwrite(sam,sprintf('Before_0%d.tif',n));
figure();imshow(g);
title('after')

imwrite(g,sprintf('After_0%d.tif',n));
end
```

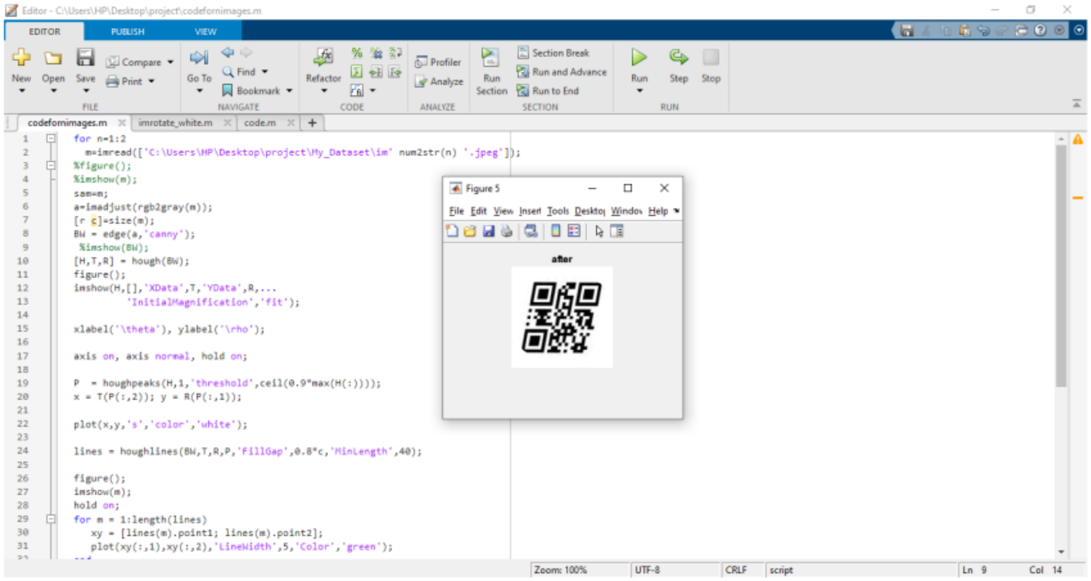
Code for image rotation

```matlab
function rotated_image = imrotate_white(image, rot_angle_degree)

RA = imref2d(size(image));

tform = affine2d([cosd(rot_angle_degree) -sind(rot_angle_degree) 0; ...

sind(rot_angle_degree) cosd(rot_angle_degree) 0; ...

0 0 1]);

Rout = images.spatialref.internal.applyGeometricTransformToSpatialRef(RA,tform);

Rout.ImageSize = RA.ImageSize;

xTrans = mean(Rout.XWorldLimits) - mean(RA.XWorldLimits);

yTrans = mean(Rout.YWorldLimits) - mean(RA.YWorldLimits);

Rout.XWorldLimits = RA.XWorldLimits+xTrans;

Rout.YWorldLimits = RA.YWorldLimits+yTrans;

rotated_image = imwarp(image, tform, 'OutputView', Rout, 'interp', 'cubic',
'fillvalues', 255);

end
```

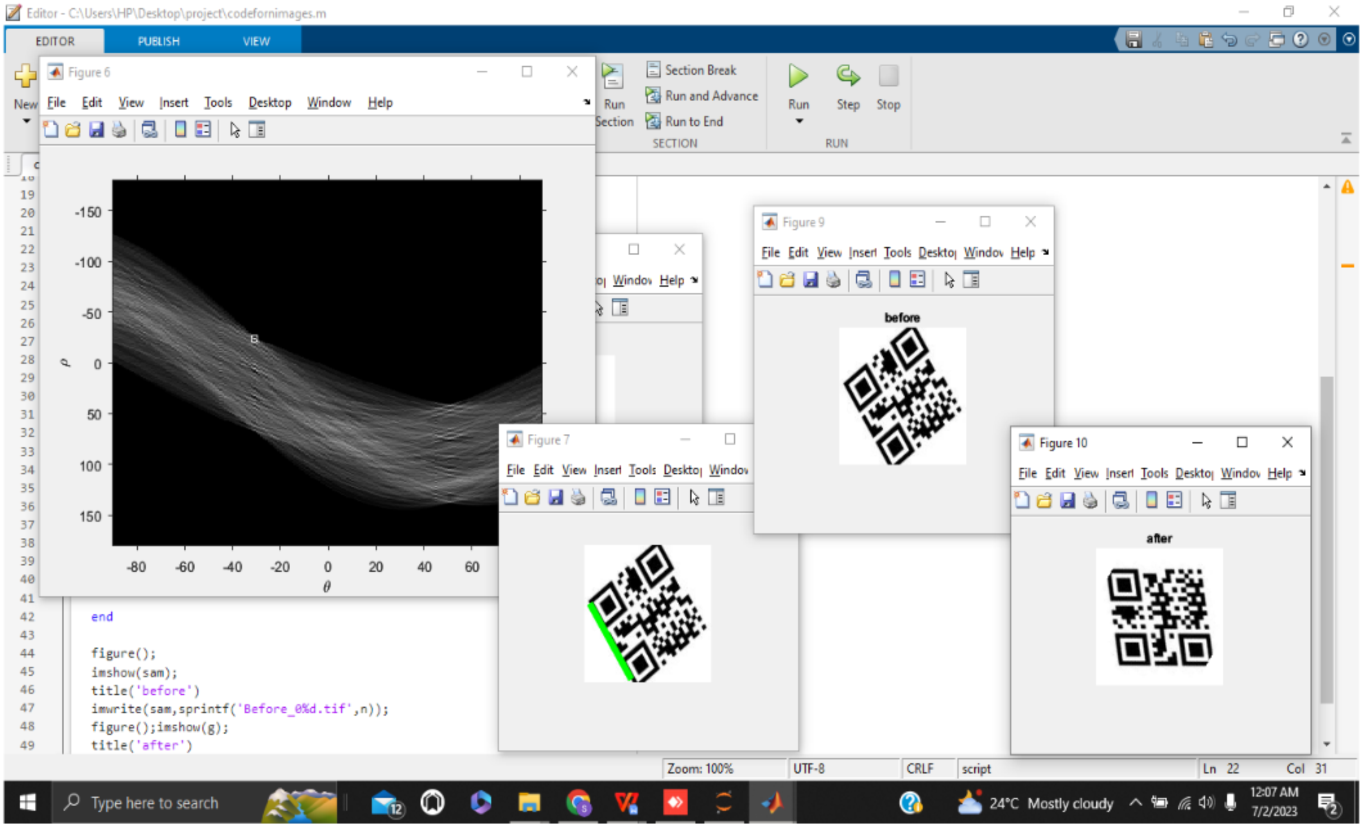
Results:



```
for n=1:2
  m=imread(['C:\Users\HP\Desktop\project\My
%figure();
%imshow(m);
sam=m;
a=imadjust(rgb2gray(m));
[r c]=size(m);
BW = edge(a,'canny');
 %imshow(BW);
[H,T,R] = hough(BW);
figure();
imshow(H,[],'XData',T,'YData',R,...
         'InitialMagnification','fit');

xlabel('\theta'), ylabel('\rho');

axis on, axis normal, hold on;

P  = houghpeaks(H,1,'threshold',ceil(0.9*ma
x = T(P(:,2)); y = R(P(:,1));

plot(x,y,'s','color','white');

lines = houghlines(BW,T,R,P,'FillGap',0.8*c,'MinLength',40);

figure();
imshow(m);
hold on;
for m = 1:length(lines)
   xy = [lines(m).point1; lines(m).point2];
   plot(xy(:,1),xy(:,2),'LineWidth',5,'Color','green');
```



```
for n=1:2
  m=imread(['C:\Users\HP\Desktop\project\My_Dataset\im' num2str(n) '.jpeg']);
%figure();
%imshow(m);
sam=m;
a=imadjust(rgb2gray(m));
[r c]=size(m);
BW = edge(a,'canny');
 %imshow(BW);
[H,T,R] = hough(BW);
figure();
imshow(H,[],'XData',T,'YData',R,...
         'InitialMagnification','fit');

xlabel('\theta'), ylabel('\rho');

axis on, axis normal, hold on;

P  = houghpeaks(H,1,'threshold',ceil(0.9*max(H(:))));
x = T(P(:,2)); y = R(P(:,1));

plot(x,y,'s','color','white');

lines = houghlines(BW,T,R,P,'FillGap',0.8*c,'MinLength',40);

figure();
imshow(m);
hold on;
for m = 1:length(lines)
   xy = [lines(m).point1; lines(m).point2];
   plot(xy(:,1),xy(:,2),'LineWidth',5,'Color','green');
```



```
end

figure();
imshow(sam);
title('before')
imwrite(sam,sprintf('Before_0%d.tif',n));
figure();imshow(g);
title('after')
```

Discussion:

The Hough Transformation works by transforming the image space into a parameter space where lines are represented as points. Each point in the image space is transformed into a sinusoidal curve in the parameter space. The intersection points of these curves correspond to the line in the image space.

The Hough Transformation is useful for detecting lines or circles in noisy or cluttered images, as it allows for robust detection even in the presence of gaps or missing line segments. It is also used for finding lines or circles in binary images, where edges have been detected and thresholded.

Overall, the Hough Transformation is an important tool in computer vision and image processing, and has been widely used for the detection of lines and circles in images.

Hence, that is why I have use Hough transformation as it is useful for detecting lines of tiled QR code and transform the barcode to it’s right position so the QR code function or else it is unable to be scanned by users.

Conclusion:

In summary, skew detection and correction involves identifying and correcting the tilt or inclination of an image or document, resulting in a straight and aligned image for improved processing and analysis with the help of Hough transformation technique.

Computer Vision_Savitha San Markan

ORIGINALITY REPORT

27%	25%	5%	19%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Cavite State University Student Paper	11%
2	in.mathworks.com Internet Source	11%
3	dpl.csrsr.ncu.edu.tw Internet Source	3%
4	Zhiwei Xu, Haoqian Wang, Yi Yang. "Semi-supervised self-growing generative adversarial networks for image recognition", Multimedia Tools and Applications, 2020 Publication	2%
5	Submitted to CSU, San Jose State University Student Paper	1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		